

dynatrace

AppMon Documentation

Please direct questions about your Dynatrace product or comments on this document to:

Dynatrace Customer Support
https://www.dynatrace.com/support/contact-support/

Copyright © 2017 Dynatrace LLC. All rights reserved. Unpublished rights reserved under the Copyright Laws of the United States.

U.S. GOVERNMENT RIGHTS-Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in Dynatrace LLC license agreement and as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (OCT 1988), FAR 12.212 (a) (1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as applicable.

This product contains confidential information and trade secrets of Dynatrace LLC. Disclosure is prohibited without the prior express written permission of Dynatrace LLC. Use of this product is subject to the terms and conditions of the user's License Agreement with Dynatrace LLC.

Documentation may only be reproduced by Licensee for internal use. The content of this document may not be altered, modified or changed without the express written consent of Dynatrace LLC. Dynatrace LLC may change the content specified herein at any time, with or without notice. All current Dynatrace LLC product documentation can be found at https://www.dynatrace.com/support/doc/.

Adobe® Reader® is a registered trademark of Adobe Systems Incorporated in the United States and/or other countries.

All other company and product names are trademarks or registered trademarks of their respective owners.

Dynatrace is the innovator behind the new generation of Application Performance Management. Our passion: helping customers, large and small, see their applications and digital channels through the lens of end users. Over 5,700 organizations use these insights to master complexity, gain operational agility, and grow revenue by delivering amazing user experiences.

AppMon Documentation 64
Knowledge base 64
Community Forums 64
Dynatrace University 64
Dynatrace Blog 64
More AppMon help resources 64
Integrations and extensions 64
Continuous delivery and test automation 64
User experience management 64
Database monitoring 65
Infrastructure monitoring 65
Application monitoring 65
Administration 65
Installation 65
Previous releases 65
Get to know AppMon 66
Getting started 66
Getting started 67
Step 1 - Explore the walkthroughs 67
Step 2 - Review defaults and adjust as necessary 67
Step 3 - Browse use cases and tasks for common roles 68
Step 4 - Fine-tune AppMon and capture more data 68
Step 5 - Explore advanced configuration topics 68
Developer use cases and tasks 70
PurePath and sensors 70
Memory, thread, and CPU diagnostics 70
Performance hotspot diagnostics 70
Error diagnostics 71
Operations use cases and tasks 72
Application Monitoring 72
System and availability monitoring 72
Performance hotspot diagnostics 73
Error diagnostics 73
Testers use cases and tasks 74
Load testing 74
Tool integrations 74
Performance hotspot diagnostics 74
Error diagnostics 75
Business use cases and tasks 76
Data capturing 76
Business intelligence 76
Business Transactions 76
Dashboards 77

Continuous delivery use cases and tasks 78
AppMon in continuous integration 78
Performance hotspot diagnostics 78
Quality metrics in continuous delivery 78
Automated deployments 79
Error diagnostics 79
Walkthroughs 80
Application performance monitoring 80
Proactive performance management 80
Diagnostics 80
Business Transaction management 81
Previous versions of AppMon documentation 82
Installation 83
Before you install 83
Component installation 83
Licensing 83
After you install 84
Install considerations 84
Install first time or upgrade? 84
Installation overview 86
Prepare for installation 87
Download and install AppMon 87
Activate licenses 88
Basic installation settings 88
Set up your System Profile and instrument applications 88
Verify data is being captured in your System Profile 89
Start using AppMon 89
System requirements 90
Dynatrace AppMon components 90
Application environments and technologies 94
Other supported application environments and technologies 97
CPU cores and memory requirements 114
Why AppMon has hardware recommendations? 114
I upgraded to the next size and you want me to add more cores. This doesn't make sense for me, because my current CPU is not fully utilized? 119
Why is the CPU clock speed also important? 119
Bare metals vs. virtualization 119
Conclusion 120
AppMon basics 121
Learn more about 121
Deployment Guide 122

Collect sizing data 124
Determine the number of agents 124
Special considerations for .NET Agents 125
Special considerations for web server processes 126
Collect PurePath numbers 127
UEM considerations 127
Session storage 127
Performance Warehouse 128
System boundaries 128
Use the Deployment Sizing calculator 130
What is the Deployment Sizing calculator? 130
Calculate sizing requirements 131
Resolve No Sizing Issues 132
Ensure sizing accuracy 133
Assess results 135
Default deployment sizes 138
Hardware requirements 138
VM support 139
Why are so many cores required? 140
Intel vs. AMD 140
Heap breakdown for frontend and backend process 140
Demo (POC) environment 142
Overview 142
Sizing suggestions 142
Small environment 144
Overview 144
Sizing suggestions 144
Medium environment 146
Overview 146
Sizing suggestions 146
Large environment 148
Overview 148
Sizing suggestions 148
XLarge environment 151
Overview 152
Sizing suggestions 152
Sizing questions or issues 155
Customer environment examples 157
Small environment 158
Medium environment 159
Large environment 161
XLarge environment 162
Verify results 169

Verify CPU requirements 171
Check power plan settings 171
Determine the physical cores 172
Additional deployment best practices 177
Server and Performance Warehouse best practices 178
AppMon Server best practices 178
Database Server / Performance Warehouse best practices 180
Memory Analysis Server best practices 186
Collector best practices 188
Distributed deployments 188
Collector sizing 189
Disk IO determination best practices 192
How to measure available disk I/O for session storage 193
Network bandwidth best practices 195
How to measure the available bandwidth 196
Download jPerf to run your own bandwidth test 197
Virtualization requirements and best practices 198
Dedicated resources 198
Virtual CPUs 198
Memory 199
I/O and network 199
Understanding VMware performance charts 199
Highly available installations 202
Availability of system under diagnosis versus AppMon availability 203
Availability of system components 203
Install the AppMon Server 207
What to know before installing 208
Download and install the server 208
Sizing and memory 211
Using dtserver.exe 211
AppMon Server services / daemons 216
Additional considerations and troubleshooting 218
Install collectors 221
What to know before installing collectors 221
Install collectors 223
Start and stop a collector 227
Automatic installation 230
Troubleshooting 230
Install Memory Analysis Server 232
What to know before you install 232
Install the Memory Analysis Server 233
Starting and stopping the Memory Analysis Server 235
Additional considerations 237

Install agents 239
Download and install agents 240
Automatic installation 241
Automation Downloads 243
General 243
Agents 243
Automatic installation 246
Compatibility table 246
Additional install considerations 248
Side-by-side component installation 249
Overview 249
Side-by-side installation 250
Component installation 250
Silent installation 253
Windows 253
*NIX 254
Access to web services 256
Update notification 256
Documentation and help 256
Agent download 256
Licensing 257
Breaking changes and potential pitfalls 258
AppMon 6.3 and later 258
Licensing 260
Overview 261
Licensing process 263
License activation and deactivation 267
License states 268
eServices 270
eServices profile management 272
View your eServices profile 272
Change your password 272
Manage newsletter subscriptions 272
eServices manage units, hours and visits 274
Create license 275
Search licenses 275
Edit license 275
Clone license 276
Terminate license 276
eServices view license information 277
eServices for partners 282

Online license activation and deactivation 286
License activation 287
License deactivation 287
Deactivate remaining UEM volume prior to license deactivation 288
License reactivation 288
License activation and deactivation via file 290
License activation 290
License deactivation 292
License reactivation via a file 293
Manage UEM visits volume 295
How visit-based UEM licensing is defined 296
How visits are deducted from your purchased volume 296
Check visits consumption 297
How to manage your visits volume 297
Add UEM visits volume 298
Manually add UEM visits volume via file 298
Deactivate UEM visits volume 299
Manually deactivate UEM visits volume 299
UEM volume reservation 301
Free UEM visits volume 302
Manage usage based agent hours volume 304
Overview 304
Agent hours 304
Usage-based volume vouchers 305
License assignments 310
Overview 310
Required permissions 313
Set up a license assignment 314
Overbooking 315
Licensing details 318
Configuring a proxy server to connect to eServices 318
Licensing models for supported technologies 318
Merging and splitting licenses 320
License upgrade 322
Recommended upgrade option 323
Upgrade option 1 - trial license 324
Upgrade option 2 - temporarily skip license deactivation 324
Upgrading a UEM license using self-service 325
Manually upgrade a license 325
Set up communication connections 327
Default ports used 327
Set up AppMon Server communication 329
Set up Server-embedded Collector connections 334
Set up SSL communication 336
Enable SSL encryption 336
Configure authentication strings 339

Set up proxies 341
Overview 341
HTTP tunneling 341
Connect using forward proxies 344
Example reverse proxy configuration (using apache2 on ubuntu) 346
Set up a forward or reverse proxy 351
Overview 351
Authentication 352
General proxy setup 352
Certificate management 355
Setting up SSL communication 355
Migration from customer-provided keystores 356
Currently used certificate 356
Deploy a new private key pair and certificate 356
Troubleshooting SSL communication 356
Certificate overview 358
Custom certificate requirements 360
Deploy certificate 365
Deploy a new key pair and certificate chain 365
Certificate wizard 367
Advanced features - certificates, private keys and keystore 370
Enable logging of keystore operations 370
Clear stored certificates to force usage of new ones 370
Clear stored private key to revert to default 371
Clear DC-RUM certificates 371
Reinitialize passwords 371
Disallow private key deployment to client and collector 372
Set up system components 374
Supported plattforms 374
Network communication 374
Set up the AppMon Server 376
Memory settings 376
Additional considerations and troubleshooting 376
Set up collectors 379
Overview 379
Configure memory 380
JRE on Solaris, HP-UX, AIX, z/Linux, z/OS 380
Configure a Collector other than localhost 381
Set up a Memory Analysis Server 383
Overview 383
Configure memory 383
Configure the port 384
Start and stop the Memory Analysis Server 384
Set up agents 387
Agent settings 388

Java Agent configuration 394
Java Agent configuration string / JVM arguments / parameters 395
Agent configuration for Java-based application servers 395
Agent Configuration for Java-based Windows Services 396
Parameters and options 396
Environment variables 398
Debugging and agent conflicts 399
OneAgent usage 399
.NET Agent configuration 401
.NET Agent configuration tool 401
Creating a .NET Agent configuration 404
Editing existing Agent configurations 409
Enable/disable Agent Configurations 409
Other settings 409
Upgrade 413
OneAgent usage 413
.NET Agent troubleshooting 415
Special behavior of IIS worker processes 415
.NET CLR load 420
PHP Agent configuration 421
Overview 421
Architecture 422
Install, configure and verify 423
Limitations 424
OneAgent usage 425
Standalone PHP Agent configuration 426
Prior to installation 426
Installation 427
Usage example - PHP and Auto sensors 430
Adding sensor rules for PHP methods picked up by Auto sensors 431
Usage example - user-defined PHP sensor 434
PHP function or Method Rule notes 434
Define a custom sensor 440
PHP API Breakdown 441
Apache Agent configuration 446
Architecture 446
Installing the agents 447
Configuring Apache 447
Deployment and file access permissions 449
Apache Agent troubleshooting 451
Apache starts but no Agent connects to the AppMon Collector 451
ModSecurity blocks the signal to the AppMon monitor (403 error) 451
Loading Apache Agent on SELinux 453

IIS Agent configuration 460
Classic Agent 460
OneAgent 461
Confgure the IIS module 461
Deployment and file access permissions 464
IIS Agent troubleshooting 465
The native modules are not listed in the Modules Dialog box 465
Simultaneous IIS restart scenario 466
NGINX Agent configuration 467
Prerequisites 467
Activate the AppMon Agent 468
NGINX Agent troubleshooting 469
Troubleshooting startup issues 469
Third party plugin issues and workarounds 473
Nginx distributions 475
URLs 475
Web Server Master Agent configuration 802
General settings 803
Compression of injected content and response overhead 804
IIS settings 804
*NIX settings 805
Restart master agents 805
Multiple instrumented web server instances on one machine 805
Node.js Agent configuration 808
Agent installation, update 808
Upgrade 810
Licensing 810
Node.js Agent troubleshooting 812
Basic Node.js testing setup 813
Resolve port conflicts 814
No Agent-loading feedback at application startup 814
Troubleshooting 814
Database Agent configuration 815
Grant necessary permissions for DB monitoring 816
Configuration 818
Database Agent troubleshooting 821
Frequently asked questions / issues 821
Dynatrace support 829
Database Agent configuration - DB2 830
DB2 9.7.0.11, 10.1, 10.5, 11.1 830
OneAgent configuration 833
Port Configuration 833
FAQ 834
OneAgent rollout 835
How to migrate from Classic Agent to OneAgent for AppMon 835

Set up the AppMon Client 837
What to know before setup 837
Starting the Client 838
Set up the Webstart Client 841
How does it work? 841
Requirements 842
Run the client using webstart 843
Run the client using JNLP file 843
Configuration 843
Working directory 844
Migration of stored sessions and dashboards 844
Frequently asked questions 846
Application integration 849
Application Environment Configuration 851
LAMPP Apache and PHP 855
XAMPP Apache and PHP 855
XAMPP Apache-only 855
IIS 855
Application server / Tomcat / Alfresco 855
Application server / IIS/ASP.NET / DotNetNuke 855
LAMPP Apache and PHP sample instrumentation 856
XAMPP Apache sample instrumentation 858
XAMPP Apache and PHP sample instrumentation 861
IIS sample instrumentation 864
Application server - Tomcat - Alfresco sample instrumentation 867
Application server - ASP.NET - DotNetNuke sample instrumentation 869
Cassandra 871
Installation 871
Usage 873
Supported client and server versions 885
Fujitsu Interstage Application Server 887
Interstage Application Server 9.0 887
Troubleshooting 889
GlassFish Server Open Source Edition 890
Installation 890
GlassFish 3.1 / 4.0 890
Java applets 894
Applet viewer 894
Web browser: Java control panel 896
Web browser: alternate agent deployment for JAVA_TOOL_OPTIONS 898

JBoss, Wildfly Application Server 900
Installation 900
Wildfly 8.x - 10.0.0 / JBoss 7 902
JBoss 6 905
JBoss 5 906
Microsoft SharePoint 909
Windows Services 910
Java 1.4 910
Java 1.5 or higher 911
Mule ESB 912
Mule ESB 1.x or later 912
Oracle Application Server 914
Oracle 10.1.3+ 914
Oracle 10g 916
Troubleshooting 917
SAP Netweaver 918
SAP Netweaver CE 7.3 (Java) 918
WebLogic Application Server 921
WebLogic 9 - WebLogic 12 921
Alternatives for all supported Operating Systems 924
WebLogic Agent naming 928
WebMethods Integration Server 929
WebMethods Integration Server 9.0+ 930
WebMethods Integration Server 8.2 931
WebSphere Application Server 932
Websphere 7.0 – 8.5.5 932
Troubleshooting 935
Agent naming for WebSphere 936
WebSphere Liberty Profile 937
WebSphere Liberty Profile 8.5.5.5 - 16.0.0.2 937
WebSphere Message Broker 939
Installation 939
WebSphere Message Broker 2017 May and higher / WebSphere integration bus 9 939
MongoDB 944
Installation 944
Usage 944
Configuration 953
Supported versions 956
Testing an installation 958
Updates 959
What is a product update? 959

Update Scenarios 962
Maintaining large enterprise environments 962
Reasons to keep an old version of an agent 963
Upgrade in multiple steps 963
Agent version pinning 963
FAQ 966
Automatic agent updates 967
Disable automatic updates 967
Requirements 968
Enforce selective agent update 968
Install an update 969
Requirements 969
Installation process overview 970
Install an update 971
REST API installation 972
Controlled update rollout 972
Troubleshooting 976
Roll back or remove an update 978
Roll back an update 978
Remove an update 980
User experience management data files 982
Update a standalone Memory Analysis Server 983
Upgrade and migration guide 984
Prepare to upgrade 987
Is this the right guide? 987
Get an overview 988
Check new requirements 988
Double-check on critical fixes 988
Update the source installation 989
Check Compatibility 989
Split server instances 989
Review custom-sized installations 989
Take notes 989
License 990
Download files 990
Running two versions 991
Component compatibility 992
Compatibility and Support 992
Agent-Collector Compatibility 992
Collector-Server Compatibility 993
Client-Server Compatibility 994
Running versions in parallel for migration 995

Upgrade server, collectors, and clients 997
1. Install the new Server, Collectors, and Clients 998
2. Deactivate and Upgrade License 999
3. Shut down the old version 999
4. Create backups 999
5. Services migration (Server, Collector and Memory Analysis Server) 999
6. Auto-start new services 1001
7. Start the new Server components 1001
8. Client migration 1002
9. Start the new Client 1002
10. Activate licenses for the new Server 1002
11. Select server sizing 1002
12. Apply update to new AppMon 1003
13. Connect the Performance Warehouse 1003
14. Configure session storage 1003
15. Start new Collectors 1003
Upgrade agents 1004
1. Bootstrapped Agents 1004
2. Non-bootstrapped Agents 1005
3. Bootstrap Agents 1005
4. z/OS Agents 1005
Upgrading Bootstrap Agents 1007
Preparation 1007
Bootstrap agent upgrade process 1008
Uninstall the outdated agents 1008
Upgrading zOS Agents 1010
Requirements 1010
Upgrade procedure 1011
Final migration checks 1014
1. Manual configuration migration 1014
2. Confirm operation of the new AppMon Server 1015
Frequently Asked Questions 1017
Troubleshooting migration 1021
Case 1: Collectors: Large class cache 1021
Case 2: Many agentres files 1021
AppMon support 1022
Collecting support information 1022
Performance warehouse migration troubleshooting 1024
Keystore Migration Troubleshooting 1026
Symptoms 1026
Solution 1 1027
Solution 2 1027
Migration Tool 1031

Migration tool details 1032
Overview 1032
Migration Tool requirements 1033
Use the Migration Tool 1033
Examples 1033
Migration tool syntax 1037
Migrating server instances to separate installations 1040
Overview 1040
Separation 1041
Manual steps 1042
Running multiple server instances on one system 1043
Server.config.xml changes for server Instance removal 1045
Administration 1046
Deployment dashboards 1049
Overview 1049
Working with Deployment dashboards 1051
AppMon Server Health dashboard 1054
AppMon Frontend Server Health dashboard 1058
AppMon Collector Sizing dashboard 1060
AppMon Session Storage Health dashboard 1062
AppMon Measure Health dashboard 1065
AppMon Performance Warehouse Overview - Write dashboard 1068
AppMon Performance Warehouse Overview - Read dashboard 1071
UEM Correlation Overview dashboard 1074
PureLytics Stream Connector Overview dashboard 1077
Server administration 1080
Access the AppMon Server settings dialog box 1081
Add or remove a Server 1081
Server configuration 1082
Connectivity 1084
AppMon Client to Server connectivity default values 1084
License 1086
Sizing settings 1087
Find the right sizing for your installation 1088
Available deployment sizes 1089
Why is the available physical memory less than expected? 1091
Warnings and errors 1091
Handling confidential strings 1093

Services 1096
General tab 1096
Management tab 1098
Webstart tab 1099
Dynatrace Web tab 1100
SSL certificates tab 1100
Email tab 1100
Dynatrace Memory Analysis Server tab 1100
Symbolication service tab 1100
Email 1102
Overview 1102
Configuring the system for sending email 1103
Importing the Email Server Certificate 1106
Collectors 1108
Collector groups for connection balancing and fail over 1110
Collector load balancing 1111
Clearing Collector caches 1111
Performance Warehouse 1113
Connection details 1114
Changing the AppMon Performance Warehouse database 1114
Manage System Profiles 1115
Manage storage 1115
Performance Warehouse database versus session store files 1115
Sensor packs 1116
Sensors for JavaScript frameworks 1118
AngularJs 1118
Backbone.js 1119
Users 1120
Plugins 1122
Installed plugins tab 1123
Plugin development tab 1124
Language packs 1125
Installing a language pack 1125
Selecting a language 1126
Storage 1127
Storage quota for sessions 1127
Stored sessions directory 1131
Infrastructure 1132
Overview 1132
Hostgroups tab 1138
Labels tab 1146
Sites tab 1147
Alerts tab 1147
Exclusions tab 1148
Troubleshooting 1151

Geographical locations - mapping of IP addresses 1153
Geographical locations database update 1153
Map IP address ranges to locations 1153
HTTP request headers 1156
Locations in the world map 1156
Defining custom locations in the world map 1157
Edit the Business Transaction corresponding to the app's System Profile 1157
Specify the IP address for a custom city 1161
Viewing visits and the world map 1162
Finding out the geographic location for a given point 1163
Synthetic Monitoring configuration 1165
Enabling Synthetic Test Integration for Application Monitoring 1165
server.config.xml 1167
Server Settings table 1168
Repository configuration 1176
Generating the default configuration file 1176
Backup of AppMon settings and data 1177
AppMon Server and Collector settings and data backup 1177
Optional session store and Performance Warehouse backup 1178
Performance Warehouse administration 1179
What's new and upgrade recommendations 1180
Data stored in the Performance Warehouse 1180
Supported databases 1180
Embedded Performance Warehouse usage restrictions 1181
Performance Warehouse resolution 1183
Performance Warehouse configuration 1184
Backups 1189
Usage of PostgreSQL 1190
FAQs / issues 1191
Application environment administration 1200
System Profile 1201
Operations on System Profiles 1203
System Profile storage 1204
Manual creation of a System Profile 1205
Guided creation of a System Profile 1207
Manage System Profile with the (Add Tier) wizard 1209
Edit a System Profile 1219
System Profile configuration 1221
Agent Groups 1221
System Profile - General 1224
System Profile - Sensors 1226
Sensor rule management 1227
Creating sensor packs 1227
System Profile - APIs 1229
Add PHP API rules 1230

System Profile - Measures 1232
List of measures with description 1233
System Profile - Business Transactions 1234
System Profile - Schedules 1235
Defining a schedule 1236
System Profile - Tasks 1237
System Profile - Monitors 1239
System, host and availability monitoring 1240
System Profile - Error Detection 1241
Alerts 1242
Rules 1242
Configure error detection rules 1243
System Profile - Applications 1251
How to define boundaries for logical applications 1252
Using logical applications 1254
System Profile - Incidents 1258
System Profile - Test Automation 1260
System Profile - User Experience 1262
Global settings 1263
Capturing settings 1264
General 1265
Web Applications 1271
CDN content analysis 1281
Overview 1281
Resource summary configuration 1288
Summary 1288
Settings 1290
Action groups advanced configuration 1291
Settings 1291
Dashlet metrics applicable to action groups 1293
System Profile - Integration 1295
Overview 1295
Add a new DC RUM appliance 1296
Edit an existing DC RUM appliance 1296
Drill down from DC RUM to AppMon 1297
Troubleshooting 1298
System Profile - Observed Tiers 1301
Agent Group - General 1302
Agent Group - Agent Mapping 1303
How agent mapping works 1304
Advanced settings 1304
Agent Group - Agent Mapping Advanced Settings 1306
Access the Agent Mapping Advanced Settings dialog box 1306
Agent Mapping advanced settings 1308

Agent Group - Sensor Placement 1311
Placement options 1311
Sensor placement 1312
Agent Group - Sensor Configuration 1313
Bulk edit of the sensor properties 1314
Configurations 1314
Options 1314
Sensor Configuration 1315
Sensor properties 1316
ADO.NET 1317
ASP.NET 1317
CICS Transaction Server 1317
EJB invocation 1317
Java exceptions 1317
Java Logging 1318
Java Web requests 1318
Java web services 1318
JDBC 1318
JMS Messaging 1318
MongoDB 1318
MSMQ 1318
.NET Exceptions 1318
.NET Logging 1319
.NET Web requests 1319
PHP Database 1319
PHP Exceptions 1319
PHP General 1319
PHP Logging 1319
Servlets 1319
Tibco Messaging 1319
User experience 1320
System Profile - Data Privacy 1321
Rename a System Profile 1323
Import a System Profile 1326
Export a System Profile 1328
Enable or disable a System Profile 1330
Disable a System Profile 1331
System Profile templates 1332
Creating a System Profile template 1334
System Profile - Business Hours 1335
Business Hours for Tasks and Monitors 1336
Business Hours for alert suppression 1337

Sensor Configuration 1338
Sensor terminology 1338
Sensor rules 1339
Memory sensor rules 1361
Class Browser 1363
Grouping options 1364
Rule generation 1364
Hot Sensor Placement 1366
Supported JVM versions 1367
Agents overview 1367
Applying Hot Sensor Placement 1368
Configuration change immediacy matrix 1370
Hot Sensor Placement 1373
Client administration 1376
Settings 1377
Services 1378
Directories 1378
External Tools 1378
Certificate Management 1379
AppMon Web administration 1380
Port configuration 1381
User account configuration 1381
Secure certificate setup 1381
Data display 1382
World map country drilldown 1382
User permissions and authentication 1384
Accounts 1384
Groups 1388
Roles 1390
LDAP 1393
Password security 1396
Permissions mapping 1398
Permissions reference tables 1399
SAML/SSO configuration 1419
Prerequisites and limitations 1419
Configuration 1420
Downloads 1422
Security and compliance whitepaper 1423
Understanding privacy regulation 1423
Payment card industry data security standard 1425
AppMon Application Monitoring implementation best practices 1427
User experience monitoring (UEM) security and privacy 1438
Cross-site scripting 1439
User experience monitoring (UEM) and security 1439
Privacy conclusions 1443

JMX management 1444
Configuration of the JMX interface 1444
Server control MBean 1446
System MBeans 1447
Measure MBeans 1448
Command line tool 1451
How to start and stop AppMon services or daemons 1452
Start and stop instructions 1452
Check if an AppMon Service is running 1453
Environment incidents 1454
VMware - hypervisor swapping 1454
VMware - memory ballooning 1454
VMware - memory limit too low 1454
VMware - reserved memory insufficient 1454
VMware - no memory reserved 1455
VMware - CPU MHz limit too low 1455
VMware - reserved CPU MHz insufficient 1455
VMware - no CPU MHz reserved 1455
VMware specific incidents 1455
General incidents 1457
Application Monitoring 1458
Smart Business Transactions 1459
Smart baselining and alerting 1460
Integrating host and system health monitoring 1461
Agents 1462
Java Agent 1463
Sensors 1464
OneAgent 1466
AMQP sensor 1468
AMQP Receive Entry Point sensor 1470
Define a custom method as receive entry point 1470
AWT sensor 1475
Sensor configuration 1475
Cassandra sensor 1476
Cassandra CQL binary protocol 1476
Thrift Protocol 1477
Captured information 1477
CQL binary protocol 1477
EJB Invocation sensor 1478
Sensor properties 1478
Sensor configuration 1478
CICS/IMS Gateways sensor 1479
Sensor configuration 1479
Executor Tagging sensor 1480

Hibernate sensor 1481
Sensor configuration 1481
Java Exception sensor 1482
Sensor Configuration 1482
Java Logging sensor 1484
Sensor Configuration 1484
Java Tests sensor 1485
Sensor configuration 1485
Java Web Requests sensor 1486
Sensor Configuration 1486
Example 1487
Java Web Service sensor 1488
Sensor properties 1488
Sensor Configuration 1488
JBoss ESB Tagging sensor 1490
Sensor configuration 1490
JDBC sensor 1491
Sensor configuration 1491
JMS sensor 1493
Sensor configuration 1493
Tagging 1493
JMX MBean Server sensor 1495
Sensor configuration 1495
JMS Receive Entry Point sensor 1496
Sensor configuration 1496
JNDI sensor 1497
Sensor configuration 1497
JTA sensor 1498
Sensor configuration 1498
MongoDB sensor 1499
Sensor configuration 1499
MQSeries Receive Entry Point sensor 1500
Mule Tagging sensor 1501
Sensor configuration 1501
Remote Method Invocation (RMI) sensor 1502
Sensor configuration 1502
Servlet sensor 1503
Sensor configuration 1504
Asynchronous servlet execution support 1505
Play Framework support 1506
Struts sensor 1507
Sensor configuration 1507
Thread Start Tagging sensor 1508

Thrift Tagging sensor 1509
Sensor configuration 1509
Tibco Messaging sensor 1510
Tagging 1510
Sensor configuration 1511
Java User Experience sensor 1512
Sensor configuration 1514
WebSphere MQ (MQSeries) sensor 1515
Captured information 1515
Visualization 1515
WebSphere MQ tagging 1517
Configuration 1517
WebSphere MQ mapping 1519
Using the MQ mappings tool 1520
XA sensor 1523
Sensor configuration 1523
.NET Agent 1524
Sensors 1525
OneAgent 1526
ADO.NET sensor 1529
Sensor configuration 1529
ASP.NET sensor 1531
Parameters and attributes 1531
URI filter 1531
Sensor properties 1531
Sensor configuration 1531
Supported technologies 1532
MSMQ sensor 1533
Sensor configuration 1533
MSMQ Receive Entry Point sensor 1534
Sensor configuration 1534
.NET AMQP sensor 1535
Sensor configuration 1535
.NET AMQP Receive Entry Point sensor 1536
Sensor configuration 1536
.NET Components sensor 1537
Sensor configuration 1537
.NET Exception sensor 1538
Overhead properties 1538
Configuration properties 1538
Aggregation 1538
Sensor Configuration 1539
.NET Logging sensor 1540
Sensor configuration 1540

ASP.NET Events sensor 1541
Sensor configuration 1541
.NET Remoting sensor 1542
Sensor configuration 1542
.NET Service Fabric sensor 1543
Known limitations 1543
.NET Test sensor 1544
Sensor configuration 1544
.NET Thread Tagging sensor 1545
Sensor configuration 1545
.NET Web Request sensor 1547
Sensor Configuration 1547
.NET WCF sensor 1549
.NET Windows Forms sensor 1550
Sensor configuration 1550
.NET WPF sensor 1551
Sensor configuration 1551
Node.js Agent 1552
OneAgent 1553
Apache Agent 1554
Classic Agent 1555
OneAgent 1555
Sensors 1556
FAQ 1556
Web Server sensor for Apache 1557
Sensor Configuration 1557
Apache Modules sensor 1559
Sensor configuration 1559
Apache User Experience sensor 1560
Sensor configuration 1560
IIS Agent 1564
Architecture 1565
Classic Agent 1565
OneAgent 1565
Sensors 1566
Web Server sensor for IIS 1567
Sensor Configuration 1567
IIS User Experience sensor 1569
Sensor configuration 1569

NGINX Agent 1573
Architecture 1574
Classic Agent 1574
OneAgent 1574
Sensors 1575
FAQ 1575
Web Server sensor for NGINX 1576
Sensor Configuration 1576
NGINX Modules sensor 1578
Properties 1578
Sensor configuration 1578
NGINX User Experience sensor 1579
Sensor configuration 1579
PHP Agent 1583
Sensors 1584
OneAgent 1584
PHP General sensor 1586
Properties 1586
Sensor configuration 1586
PHP Database sensors 1588
Sensor configuration 1588
Captured information 1589
PHP Exceptions sensor 1593
Sensor description 1593
Sensor configuration 1593
PHP Logging sensor 1595
Sensor configuration 1595
PHP Server sensor 1597
Sensor configuration 1597
Sensor results 1599
PHP Unit sensor 1601
Sensor configuration 1601
PHP Web Requests sensor 1602
Captured information 1602
Sensor configuration 1603
PurePaths with and without web requests sensor placed 1603
JavaScript Agent 1606
Agent injection 1607
Analysis performed by the JavaScript Agent 1609
Action name detection 1610
Bandwidth detection 1611

zOS Agent 1613
CICS Agent 1614
IMS Agent 1615
zDC (data collection subsystem) 1615
zLocal Agent 1615
zRemote Agent 1615
Java Agent(s) 1616
Installing the z/OS Agent 1617
z/OS Agent maintenance 1617
z/OS Agent installation summary 1617
Before you install the z/OS Agent 1617
Download and SMP/E install z/OS Agent components 1619
zDC 1626
Installing the zDC 1626
zDC reference information 1629
Running multiple zDC systems on a single z/OS image 1633
zDC messages 1636
Multiple TCP/IP stacks 1637
CICS Agent installation and configuration 1638
Adding the AppMon CICS programs to the CICS DFHRPL 1639
Installing the AppMon CICS programs and transaction 1639
Adding the AppMon CICS Agent to the CICS startup program list 1641
Adding the AppMon CICS Agent shutdown to the shutdown program list table 1642
Preparing your System Profile for CICS Agents 1642
Adding SIT override INITPARM for non-default zDC subsystems or FINE logging 1644
SOAP support 1644
CICS PurePath 1645
CICS Transaction Server sensor 1652
Sensor configuration 1652
Transaction measure 1655
Overview 1655
Charting CPU usage for all CICS transactions 1657
IMS Agent injection and configuration 1660
Injecting the AppMon IMS Agent 1662
IMS SOAP gateway instrumentation 1667
Preparing your System Profile for IMS Agents 1668
AppMon System Profile - IMS Agent Groups 1670
IMS sensor properties 1673
zLocal Agent 1677
zRemote Agent 1678
Installation and configuration 1678
Optional parameters to register the Windows service / start the *NIX daemon 1679
Windows service 1679
*NIX daemon 1680
Testing the installation 1682

DTAX transaction 1684
Change the log level 1685
Send a ping message 1686
Enable or disable the AppMon data capture 1686
DTAX messages 1688
zOS Agent messages 1689
zDC system messages 1689
DTAX system messages 1726
IMS Agent system messages 1746
zOS Agent logs 1749
CICS Agent logging 1749
IMS Agent logging 1750
zDC logging 1750
zLocal Agent logging 1750
zRemote Agent logging 1751
zOS Agent troubleshooting 1753
Component set up verification 1754
z/OS zLocal Agent could not be started 1760
System under diagnosis (CICS region) could not connect to the zDC 1761
Expected queues and/or transactions (including SOAP and CTG transactions) are not being monitored 1762
Disabled IMS Agent 1763
Migration troubleshooting 1764
zDC stats 1765
Host Monitoring Agent 1766
Overview 1766
Installation and configuration 1767
Agent bulk installation 1769
Database Agent 1771
OneAgent 1773
Agent Platform Overview 1773
OneAgent for Appmon 1774
Agent Groups 1778
What are Agent Groups? 1778
What are the benefits of Agent Groups? 1778
Configure Agent Groups 1779
Agent mapping workflow 1779
Disabling System Profiles 1782
Sensors 1784
What is a sensor? 1785
Hot Sensor Placement 1785
Entry points 1785
Auto sensors 1785
User-defined sensors 1786

Dashboards 1788
Overview 1788
Dashboards - AppMon Client vs web 1790
AppMon client dashboards 1790
Web dashboards 1790
AppMon client dashboards 1792
Create a dashboard 1793
Open a dashboard 1793
Save a dashboard 1793
Dashlets 1794
Adding dashlets 1795
Reuse and share dashboards 1795
Drill down from dashboards 1796
Filter dashboard and dashlet data 1796
Customize dashboard names & descriptions 1799
Link dashboards to a dashlet 1800
View in kiosk mode 1800
Dashboard vs dashlet 1802
Dashboard and dashlet properties 1804
Dashboard and dashlet common properties and inheritance 1804
Design settings 1805
Dashboard mode 1806
Dashlet toolbar 1808
Session comparison 1808
Dashlet-specific properties 1808
Cockpit dashlet 1809
Overview 1809
Offline sessions 1810
Getting data for a dashboard 1811
Quick vs. Full Analysis 1811
Analyzer progress visualization 1811
Analysis stopped due to PurePath limit reached 1812
Analysis cache 1812
Web dashboards 1814
Dashboards for every use 1815
Easy to collaborate 1817
Know and control data context at all times 1818
Understand the real meaning of the user’s experience 1818
Ensure infrastructure and environment health 1818
Responsive design 1818
Real-time data display 1818
Shared web dashboards 1819
Access AppMon web 1820
Technical requirements 1820
Permissions necessary 1821
Built in web dashboards 1824

Custom web dashboards 1826
Overview 1826
Before you begin 1827
Create a custom web dashboard 1829
Tile types 1831
Dual Axis Line/Bar Chart & Dual Axis Bar/Line Chart 1832
Stacked Percentage Bar Chart 1832
Pie chart 1832
Area Chart / Stacked Area Chart 1832
Meter chart 1832
Percentile Chart 1833
Traffic Light Chart 1833
Gauge Chart 1833
Measure 1834
Business Transaction 1834
Business Transaction List 1834
World Map 1835
Image 1835
External content 1835
Alerts 1835
Synthetic tests 1836
Text and measures 1836
Set up a Business Transaction List tile 1836
Add and edit measures 1840
Add and edit a measure 1840
Measure aggregation methods 1841
Measure splittings 1841
Web dashboard timeframe and resolution 1844
Timeframe 1844
Chart resolution 1845
Share web dashboards 1849
Capabilities 1849
Sharing a dashboard 1850
Mark a Web Dashboard as a Personal Favorite 1850
Publicly share a web dashboard 1850
Other reporting features 1851
Export and import web dashboards 1852
Export web dashboards 1852
Import web dashboards 1853
Delete custom web dashboards 1854
Embed custom content into a web dashboard 1856
Embed content using HTML 1856
Create and define the custom tile 1857
Define the custom content and save the tile 1857
Control access to AppMon Web dashboards 1859
Applications web view 1860

Transaction flow web view 1863
Edit topology layout 1865
Transaction flow print 1866
PurePath web view 1867
PurePath details view 1868
PurePath analytics web view 1870
Application incidents 1873
Incident statuses explained 1873
Operations web view 1876
Infrastructure monitoring with web dashboards 1877
Host health web view 1878
Host details view 1879
Process health web view 1882
Process details view 1883
Database health web view 1886
Database detailed view 1887
Act on infrastructure and application violations using alerts 1889
Create Incident Rules to set up alerts 1890
Alert tile on the operations dashboard 1891
Viewing additional incident details 1891
View and set the scope of alerts 1892
Focus on specific alerts for analysis 1893
Analyze an incident in the AppMon Client 1894
Drilldown to the AppMon Client 1896
Client setup and drilldown 1896
Share the open in client link 1899
AppMon Web FAQ & troubleshooting 1900
Status overview dashlets 1902
Agents Overview dashlet 1904
Dashlet details 1907
Incidents Overview dashlet 1917
Incident state and incident rule state 1919
Incident count 1919
History 1919
Drilldowns 1920
Table columns 1920
Context menu 1921
License Overview dashlet 1923
System Information dashlet 1925
Overview 1925
Create a support archive 1926

Tasks and Monitors dashlet 1927
Overview 1927
Current status icons 1928
Last status icons 1928
Monitoring Overview dashboard 1930
Welcome page - configure your application environment 1930
Monitoring overview dashlets 1935
Questions & answers 1936
Application Overview dashboard 1938
What is an application in the AppMon context? 1938
Overview 1938
Displaying the application overview 1939
Application Details 1948
General 1948
Walkthrough 1949
Business Transaction details 1960
Business Transaction splitting header 1961
Configurations 1961
Baseline chart 1966
Open in dashboard link / menu 1967
End User Experience dashboard 1975
Infrastructure monitoring dashboards 1976
Infrastructure overview - global / overall 1976
Infrastructure overview - bound to System Profile 1977
Working with views of the infrastructure in the AppMon Client 1979
Set up alerts 1981
Mobile Applications dashboard 1983
Performance 1984
Analytics 1984
User experience 1984
Event tracking 1984
Transaction Flow dashlet 1985
Transaction Flow icons 1990
View modes 1992
Zooming 1993
Transaction Flow zooming actions 1993
Differences in zooming view modes 1993
Health visualization 1993
Values visible in Transaction Flow 1994
Work with Transaction Flow 2008
Observed tiers 2015
Configure observed tier rules 2015
Observed tier appearance 2016

External API services 2018
Configuration in Method Rule dialog box 2018
Business Transactions 2020
PurePath dashlet 2024
Transaction Flow 2025
Leaf nodes 2026
Functional Health dashlet 2027
Selected time frame 2028
Other time frame 2028
Error HotSpots 2028
Available drilldowns 2028
Performance specific dashlets 2029
Methods dashlet 2033
Overview 2033
Metrics 2034
Context menu options 2036
Agent Breakdown dashlet 2037
Overview 2037
Metrics 2038
Business Transaction Hotspots dashlet 2039
Overview 2039
Usage 2040
Business Transactions dashlet 2046
Overview 2046
Business Transaction metrics 2049
Reanalyze all PurePaths 2050
Components dashlet 2051
Overview 2051
Metrics 2052
Context menu options 2053
Sensor Configuration 2053
Tuning options 2055
Database dashlet 2057
Overview 2057
Logical views 2057
History drilldown 2063
Sensor Configuration 2065
Analyze bind values 2065
Dashlet display options 2066
Connection pool information 2066
Tuning guidelines 2067
Database Hotspots dashlet 2069
Overview 2069
Database Hotspots table 2070
Caller breakdown tree 2070

Entry Points dashlet 2072
Overview 2072
Metrics 2073
Aggregation options 2074
Memory Allocations dashlet 2076
Overview 2076
Metrics 2077
Messaging dashlet 2078
Overview 2078
Metrics 2078
Context menu options 2081
Sensor Configuration 2081
Tuning options 2081
Method Hotspots dashlet 2083
Prerequisites 2083
Overview 2083
Caller breakdown 2084
Navigation 2085
API Breakdown dashlet 2087
Overview 2087
Metrics 2088
Context menu 2088
Mobile Workload dashlet 2090
Naming Services dashlet 2091
Overview 2091
Metrics 2092
Context menu options 2093
PurePath Comparison dashlet 2094
Overview 2094
General usage 2095
Comparison process 2098
Differential PurePath metrics 2099
Invoking the comparison from other dashlets 2099
PurePaths dashlet 2102
Overview 2102
PurePaths tab 2104
Contributors tab 2109
Errors tab 2112
PurePath Hotspots pane 2113
Tree tab 2115
Transaction Flow tab 2117
API Distribution tab 2117
Remoting dashlet 2119
Overview 2119
Remoting metrics 2119
Tuning guidelines 2122

Response Time Hotspots dashlet 2123
Overview 2123
Server transaction statistics 2124
HotSpots by tier 2125
HotSpots by API 2125
Sequence Diagram dashlet 2126
Overview 2126
Detail levels 2130
Tagged Web Requests dashlet 2134
Overview 2134
Grouping 2135
Metrics 2136
Aggregation options 2136
Web Requests dashlet 2139
Overview 2139
Performance metrics 2140
Analysis options 2143
Context menu options 2144
Sensor Configuration 2145
Performance tuning guidelines 2149
Web Services dashlet 2151
Overview 2151
Web services performance metrics 2151
Event specific dashlets 2154
Errors dashlet 2156
Error overview pane 2158
Error details pane 2158
Exceptions dashlet 2160
Overview 2160
Metrics 2161
Aggregation options 2161
Configuration 2163
Context menu options 2164
GUI Controls dashlet 2165
Overview 2165
Metrics 2166
Aggregation options 2166
Incidents dashlet 2168
Overview 2168
Dashlet filters and Incident Rules 2170
Incident metrics 2174
Context menu options 2174

Logging dashlet 2175
Overview 2175
Metrics 2176
Configuration 2176
Context menu options 2177
Runtime Suspensions dashlet 2178
Overview 2178
Metrics on the Runtime Suspension dashlet 2179
Runtime Specific dashlets 2180
CPU Sampling dashlet 2182
When to use sampling 2182
Sampling FYIs 2182
Limitations 2182
Wording 2183
Sampling sessions 2183
Sampling session interpretation 2183
Errors 2184
Selective Memory dashlet 2186
Overview 2186
Show content 2190
Compare selective memory snapshots 2191
Threads dashlet 2194
Overview 2194
Metrics 2194
Stack trace 2195
Owned monitors 2195
Tasks 2196
Total Memory dashlet 2197
Create a total memory snapshot 2204
Snapshot overview tabs 2205
Content of a total memory snapshot 2208
Comparing memory snapshots 2209
Analysis drill-downs 2209
Duplicate Strings Content dashlet 2210
Eliminate string duplicates 2210
Fields and values 2212
Overview 2213
Follow References dashlet 2214
Aggregated references 2215
Direct 2216
By keep alive 2217
Keep Alive Set dashlet 2218

Shortest Root Paths dashlet 2220
Simulate reference removal 2222
Reference types 2222
Direct 2225
By keep alive 2225
Total Memory Comparison dashlet 2226
Class comparison chart 2226
Class comparison 2227
Total Memory Content dashlet 2229
Analysis drilldown 2229
The GC size column 2230
Show instances 2230
Total memory content for leak analysis snapshots 2230
Total memory content for trending snapshots 2231
User Experience dashlets 2234
Client Errors dashlet 2236
Overview 2236
Client errors overview table 2237
Client Errors Detail table 2238
Custom filtering and splitting 2238
User Action PurePaths dashlet 2241
Overview 2242
PurePaths 2242
Contributors 2244
Errors 2245
Tree 2245
Transaction Flow 2245
API Distribution 2246
User Actions dashlet 2247
Overview 2247
User Actions list 2248
Action groupings 2249
Response time charts 2250
Web requests 2251
Custom filtering and splitting 2251
User Actions Health dashlet 2254
Selected time frame 2255
Previous time frame 2256
Error HotSpots 2256
Available drilldowns 2256
Visits dashlet 2257
Overview 2257
Visits list 2258
Visits details 2260
Custom filtering and splitting 2263
Example usage 2265

Other dashlets 2266
Session Browser dashlet 2267
Overview 2267
Stored session attributes 2267
Context menu options 2268
Grouping options 2269
Test Overview dashlet 2271
Bar chart section 2272
Pie charts section 2272
Drilldowns to Test Results dashlet 2273
Footer section 2273
Test Results dashlet 2274
Test case section 2275
Test status 2275
Viewing details 2276
Measures section 2277
Chart section 2277
Viewing markers 2277
Text and image dashlet 2279
Text and measurement dashlet 2282
Web tile 2282
Tile specific filters 2284
AppMon Client dashlet 2285
Web Page dashlet 2287
Charting 2288
Chart dashlet 2291
Add series 2292
Configure series 2293
Chart types 2293
Heat field 2301
Chart splitting modes 2304
Dual Axis Chart dashlet 2306
Dynamic Measure Matrix dashlet 2307
Configuration 2308
Column visualization 2308
Remarks 2309
Incident Chart dashlet 2310
Configuration 2310
Visualization 2312
Incidents table 2314
Layer Breakdown dashlet 2316
Time zones 2317
Example 2317
Possible pitfalls with time frames 2317
Charts time zone handling 2318

Load Test Overview dashboard 2319
Web Page Performance Analysis dashboard 2320
Web Requests 2320
Errors 2320
Database 2320
Response Time Hotspots 2321
Transaction Flow 2321
APIs 2323
Business Transactions 2324
Visualization of Business Transactions 2325
Business Transactions on stored sessions 2325
Examples 2326
Business Transaction configuration 2328
General 2329
Calculate Business Transaction performance 2329
Filter 2329
Results 2330
Splitting 2331
Base Line settings 2331
Measure configuration example 2331
Business Transaction page of the Monitoring dashboard 2334
Filter 2336
Actions 2336
Preventing excessive split results 2338
Causes of measure explosion 2339
Solutions 2339
Preventing excessive split results - FAQ 2342
My Business Transaction creates a bounded set of splitting values, but creating measures is disabled by the Measure Explosion Prevention 2342
I created my Business Transaction with a release earlier than AppMon 5. Is this Business Transaction affected by the splitting limit? 2344
My Business Transaction exceeded the default limit of splittings. I performed the steps to override the default limit but now the AppMon server does not start anymore. How do I fix this problem? 2345
Splitting options 2347
Restrictions 2348
Reset splittings 2350
Real time Business Transactions feed 2351
Message format 2352
Basic setup 2357
Business Transaction-specific setup 2359
Business Transaction management walkthroughs 2360

How to use Business Transactions 2361
The main features of Business Transactions 2361
Goal of this walkthrough 2362
Creating a basic Business Transaction for web requests 2362
Add additional return values 2365
Conclusion 2370
Follow your transactions with Transaction Flow 2372
Your application landscape from the bird’s eye view 2372
Goal of this walkthrough 2373
The topology view 2373
The transaction response time view 2378
Drill down from the Transaction Flow 2378
Conclusion 2378
Create a splitting Business Transaction 2380
Goal of this walkthrough 2380
Configure the Servlet Sensor 2381
Create the measure to use for splitting. 2383
Create a splitting Business Transaction 2384
See results 2386
Problem pattern detection 2387
Overview 2387
Problem pattern categories 2389
Threshold Adjustment 2394
Problem Pattern Detection Webinar 2394
Error analysis 2396
Overview 2396
Detecting errors 2397
Monitoring functional health 2398
Error root cause analysis 2399
Using sessions 2401
Sessions types 2401
Live sessions 2403
Stored sessions 2404
Store session context menu - explicit storage 2404
Make available offline 2405
Creating stored sessions 2406
Move session storage to a new disk or directory 2408
Troubleshooting 2408
If you have no existing data 2408
Instructions 2408
Stored session states 2410
Offline sessions 2411
Import and export sessions 2412

Session comparison 2414
Overview 2414
Usage 2416
Process configuration changes - re-analyze stored or offline session 2421
Session encryption 2423
Encryption options on OSes 2423
Baseline and smart alerting explained 2426
Significant measurement 2426
Baseline calculations 2427
Violations and alerting 2431
Incidents and alerting 2433
Configuring Incident Rules 2434
Conditions 2443
Actions 2444
Linked dashboard 2446
Server measures and server incidents 2448
Incident downtimes 2451
Creating a downtime 2452
Built-in incidents 2455
Tasks 2459
Overview 2459
Configuration 2459
Task types 2462
Runtime diagnostics 2470
Thread diagnostics 2471
Overview 2471
Grouping threads 2472
Comparing thread dumps 2474
Limitations 2476
Memory diagnostics 2477
Overview 2477
Memory diagnosis with AppMon 2479
Sampling based diagnostics 2496
When to use sampling 2496
Sampling FYIs 2497
Limitations 2497
Wording 2497
Sampling sessions 2497
Sampling session interpretation 2497
Errors 2498
Reporting 2500
Overview 2500
Types and formats 2501
Report content 2505

Client reporting 2507
Default report 2507
Customize reports for dashboards 2511
Scheduled reporting 2515
Scheduled reporting 2515
Automated reporting 2521
Overview 2521
Report automation options 2521
Sample use cases 2522
Walkthrough 2523
Reporting limits 2526
Limitations to client side reporting 2526
Size limits 2526
Non-latin scripts 2530
Trees in reports 2530
Application Monitoring walkthroughs 2531
Identify problems in your application 2532
Goal of this walkthrough 2532
Problem isolation overview 2532
Detailed steps 2533
Conclusion 2540
Regression analysis 2541
What is regression analysis? 2541
Goal of this walkthrough 2541
Detailed steps 2542
Conclusion 2551
Memory analysis 2552
What is memory analysis? 2552
Goal of this walkthrough 2552
Detailed steps 2553
Conclusion 2560
Analyzing out of memory errors 2561
What is an out of memory error? 2561
Goal of this walkthrough 2562
Detailed steps 2562
Conclusion 2564
Analyze exceptions and log messages 2565
Goal of this walkthrough 2565
Detailed steps 2566
Conclusion 2576
Analyzing a slow transaction 2577
What is a slow transaction? 2577
Goal of this walkthrough 2577
Requirements 2578
Detailed steps 2578

Tips and tricks 2585
FAQ - measures and charting 2586
Resolution mapping for stacked charts 2587
Resolution mapping for non-stacked charts 2588
Glossary - measures and charting 2595
Unit 2595
Aggregation 2595
Time frames 2596
Resolution 2596
Measure picker 2597
Client commandline arguments 2598
Arguments 2598
Troubleshooting 2600
RegEx - using regular expressions 2602
Test regular expressions 2602
Test regular group expressions 2603
String comparison options 2604
Useful regular expressions 2605
Pulling a value from a method argument (for example an XML field) 2605
Parsing cookie value (for example JSESSIONID) 2605
Parsing header/session/cookie/attrib 2606
Parsing EMail address from a cookie 2606
Extracting part of a URI 2606
Identify classes ending with a certain string (limited to [cls] or [bas]) 2607
Extract string of a constant length 2607
How to instrument an application 2608
Initial configuration 2608
Monitor specific applications 2609
Monitor Amazon applications 2611
Deploy AppMon agents in Amazon EC2 2612
Overview 2612
Adding AppMon to an EC2 image 2612
AppMon Server in EC2 2613
AppMon Collector in EC2 2614
The firewall 2615
Tying it together 2615
Next steps 2619
Monitor Amazon cloud environments 2620
How to deploy your application with AppMon monitoring in EC2 2620
End user monitoring for Amazon EC2 based applications 2621
Running the AppMon Server in EC2 2624

Analyze and monitor big data solutions 2625
Overview 2625
Get full end to end database visibility with Dynatrace Cassandra support 2625
Optimize your MapReduce jobs to achieve more with less easily 2626
Full end-to-end visibility into your SOLR searches 2626
Understand how Amazon SimpleDB impacts your response times 2627
Monitor Microsoft Azure 2628
Supported Azure Computer Services 2628
Deployment Guides 2628
Additional Resources 2628
Deploy AppMon agents in MS Azure cloud service 2630
Overview 2630
Prior to Deployment 2630
Add necessary files to your solution 2631
Configure cloud-service's ServiceDefinition file 2634
Deploy AppMon .NET Agent in MS Azure web apps 2637
Before you begin 2637
Instrumentation 2638
Update 2639
Uninstall 2640
Troubleshooting .NET Agent in MS Azure web apps 2641
Not receiving agent data 2641
503 Service Unavailable for Web App and Kudo 2642
Diagnose AWT and swing applications 2643
Download 2643
Overview 2644
AWT / swing demo application 2644
Agent injection 2646
Default AWT sensor pack 2647
Creating your own AWT sensor pack 2650
Catching PurePaths 2655
Business Transactions 2655
Dashboard 2687
Troubleshooting 2689
Gatling/AppMon 6.2 integration 2691
Use hooks to start/stop session recording and switch configuration 2692
Register test run 2694
Use header tags to convey context information with your requests 2696
Analyze results in AppMon 2698
Gatling/AppMon 7.0 integration 2700
WEB API test run 2701
LOAD test run 2704
Sample DSL scripts 2708
Monitor Cloud Foundry 2709

Monitor cloud applications 2710
Cloud monitoring 2710
Amazon AWS 2711
Microsoft Azure 2711
Docker 2711
Cloud Foundry 2711
VMware 2711
Cloud technology support 2711
Monitor Docker applications 2713
Monitor dockerized apps with AppMon 2714
Overview 2714
Option A: inheritance-based approach 2714
Option B: composition-based approach 2718
Q&A 2721
Monitor dockerized apps with AppMon - Kubernetes and OpenShift 2723
Option A: inheritance-based approach 2723
Option B: composition-based approach 2725
Monitor a WAS liberty profile server 2733
Overview 2733
Add classes 2733
Additional info 2735
Set up Performance Warehouse on Oracle Database (separate tablespace for indexes) 2737
Use AspectJ to connect PurePaths in Tibco BusinessWorks 2738
Example 1: Tibco BusinessWorks hosting a WebService 2739
Example 2: Tibco BusinessWorks queue receiver 2743
Example 3: connecting PurePaths for outbound SOAP requests 2744
Instrument BusinessWorks with AspectJ and ADK 2746
Infrastructure monitoring 2750
Process health monitoring 2752
Overview 2752
Installation and configuration 2755
Process types 2755
Process Health incidents 2757
Special scenarios 2758
Host health monitoring 2761
Data-center-aware infrastructure monitoring 2761
Installation and configuration 2762
What is monitored on what platform? 2762
Thresholds, alerts and inclusions or exclusions for host metrics 2764
User specific monitors 2764
Monitors 2765
Overview 2765
Configuration 2766
Monitor types 2767

Infrastructure monitoring - zero config with agents 2770
Overview 2770
Dashboards 2770
Plugin vs. agent-based measures 2772
SNMP monitor 2777
Overview 2777
Configuration 2777
Select host(s) that should be the target(s) of the monitor 2778
Customizing the execution schedule 2779
Centrally deploying a monitor to different global locations 2779
Measures 2779
Unix system monitoring 2807
Overview 2807
Setup and Basic Configuration 2808
Select host(s) that should be the target(s) of the monitor 2808
Customizing the execution schedule 2809
Centrally deploying a monitor to different global locations 2809
Executed commands 2811
Measures 2812
URL monitor 2815
Setup 2815
Select host(s) that should be the target(s) of the monitor 2816
VMware 2820
Overview 2820
About VMware monitoring 2820
Applications in a virtual machine 2829
Running AppMon in a virtual environment 2831
Web transaction monitor 2833
Windows Performance Monitor 2834
Overview 2834
Basic Configuration 2834
Select host(s) that should be the target(s) of the monitor 2835
Customizing the execution schedule 2836
Centrally deploying a monitor to different global locations 2836
Authentication and permissions 2837
Measure Configuration 2837
Infrastructure monitoring walkthroughs 2843
Setting up infrastructure monitoring 2844
Goal of this walkthrough 2844
Scenario 2845
Host monitoring of hosts - AppMon Agent installed 2845
Host monitoring of hosts - AppMon Agent not installed 2846
Monitor other system components 2848
Use hosts and hostgroups 2850
Analyze system monitoring data 2851

Setting up application availability monitoring 2852
What is availability monitoring? 2852
Goal of this walkthrough 2852
Detailed steps 2853
Availability monitoring for multiple Agent Groups 2859
Using host and process health monitoring 2862
What is health monitoring? 2862
Goal of this walkthrough 2862
Detailed steps 2863
Using host and hostgroups for large system monitoring 2867
What is multi-host monitoring? 2868
Goal of this walkthrough 2868
Detailed steps 2869
Conclusion 2877
Using custom JMX and PMI measures in charts 2878
What are JMX PMI measures? 2878
Goal of this walkthrough 2878
Detailed steps 2878
How to retrieve performance metrics with Windows Performance Monitor 2889
Goal of this walkthrough 2890
Prerequisites 2890
User authentication and permissions 2898
Hardening the monitored systems 2899
Testing performance monitor query 2900
Configuring the AppMon Collector 2901
Database monitoring 2902
Database Agent 2903
Database Agent configuration 2903
Auto-mapping of Database Agents into System Profiles 2904
Access database monitoring data 2904
Database health monitoring 2906
Oracle 2907
SQL Server 2908
MySQL 2909
DB2 2909
Database live view 2911
SQL execution plan 2913
Database monitoring measures 2917
SQL Server 2917
Oracle and Oracle RAC 2919
MySQL 2920
DB2 2922

User experience management 2923
Overview 2923
Getting started with UEM 2924
Read more about UEM 2926
User experience use cases and tasks 2927
Advanced UEM 2927
Getting started with UEM 2927
Mobile UEM 2928
How does UEM work? 2929
UEM for Web Applications 2931
UEM for mobile apps 2931
UEM only vs. Full-stack 2931
What is a User Action? 2932
What is a visit? 2936
What is Apdex? 2936
What is the User Experience Index? 2938
Geographical location 2939
Specific concepts for Web Applications 2939
Specific concepts for mobile apps 2943
User Action timings 2944
Load actions 2944
Source actions 2945
XMLHttpRequest actions 2946
Perceived render time 2948
Overview 2948
Setup 2948
Calculation of perceived render time 2949
User Action PurePath display 2951
Useable measures 2953
Limitations 2953
Third-party content analysis 2954
Overview 2954
Third-party content analysis measures 2955
Third Party Content dashboard 2956
Impact vs W3C Resource Timing 2957
W3C resource timing metrics 2960
Overview 2960
W3C Resource Timings 2961
User Action PurePaths 2961
Manual XHR framework instrumentation 2965
Support for IE 8 and lower 2966
Ajax detection modules requiring snippets 2966
FAQ 2970

Visually complete metric 2971
Overview 2971
Single page Applications 2972
Correlate user performance with business metrics 2973
End user analytics 2973
User experience management administration 2975
UEM for Web Applications 2975
Troubleshooting 2977
UEM for mobile applications 2988
Firewall constraints for UEM 2990
HTTP requests 2990
HTTP headers 2991
Cookies 2991
Troubleshooting 2991
UEM Health Check 2992
Basic checks 2992
Application checks 2995
Configure geographic location 3010
Overview 3010
Configuration - world map 3011
Configure streaming media monitoring 3013
Overview 3013
Monitor user experience 3019
World Map dashlet 3020
Overview 3020
Navigate the map 3021
Customize the map 3022
Drill down to dashlets 3024
Geographical granularity 3024
Monitor user visits by geographic region 3029
Overview 3029
Monitor User Actions by geographic region 3039
Overview 3039
Use top findings to analyze user visits 3047
Overview 3047
Monitor top landing and exit pages 3051
Overview 3051
Analyze visit details 3053
Overview 3053
Waterfall data 3053
Analyze PurePaths and drilldown 3056
Monitor user satisfaction 3058
Overview 3058

Monitor streaming media activity 3064
Display Streaming Media Data 3065
Streaming media measures 3066
Business transacton evaluation/filter/splitting values 3069
Streaming Media dashboard 3072
Troubleshooting 3074
Limitations 3075
Trace user activity through all channels 3077
Overview 3077
Find specific user visits 3080
Web performance optimization for UEM 3082
Overview 3082
How to instrument a web application 3089
Agent injection 3089
Agent Configuration 3091
Automatic injection Agent tag placement rules 3093
Manual injection details 3095
Monitor signal behavior 3097
Mobile UEM 3101
Supported platforms 3103
Configure AppMon 3103
Integrate with the application 3104
Link web requests 3104
Configure mobile data collection 3105
Define measures 3105
Analyze data 3106
Overhead 3111
How to instrument an Android app 3112
Auto-instrumentation for Android 3113
Auto-instrumentation from the command line 3114
Auto-Instrumentation features 3115
Mobile agent start-up 3115
Crash reporting 3116
Lifecycle monitoring 3116
User action monitoring 3117
Web request monitoring 3118
Additional customization 3120
Full property list 3121
Requirements 3124
Limitations 3124
Multi-dex applications 3125

Android Agent manual instrumentation API 3129
Getting Started 3130
Crash reporting 3132
Lifecycle instrumentation 3133
User action monitoring 3137
Web request monitoring 3140
Advanced settings 3150
ProGuard settings 3151
Limitations 3152
Overhead 3152
Gradle plugin for Android 3154
Supported Android Gradle plugin versions 3155
Configure your build script 3155
Signing 3158
Optimizations 3159
Limitations 3160
Migrating from previous Android Mobile Agent versions 3161
Changes introduced with AppMon 2017 May 3161
Changes introduced with AppMon 6.5 3162
Changes introduced with AppMon 6.2 3163
Android Agent troubleshooting and FAQ 3166
Mobile Agent 3166
Manual instrumentation 3168
Auto-Instrumentation 3170
Dynatrace Gradle plugin 3174
How to instrument an iOS app 3177
Auto-instrumentation for iOS 3178
Auto-Instrumentation Features 3178
Auto-Instrumentation Overview 3179
Instrumentation settings 3179
Limitations 3182
iOS Agent manual installation 3183
Manual iOS agent setup 3183
Instrument a manual setup 3185
iOS agent setup with CocoaPods 3186
iOS manual instrumenation 3188
Swift apps manual instrumentation 3188
Dynatrace 3189
DTXAction 3197
Logging 3204
Lifecycle instrumentation 3207
Overhead 3207
Advanced authentication 3207

Migrating from previous iOS agent versions 3210
Changes introduced with AppMon 2017 May 3210
Changes introduced with AppMon 6.5 3211
Changes introduced with AppMon 6.2 3212
Versions earlier than AppMon 5.6 3213
iOS Agent troubleshooting and FAQ 3214
How do I verifying that the mobile agent is connected 3214
Why is my button captured as "Touch on button"? 3214
Can I use auto-instrumentation on an application that is instrumented with an earlier release of the iOS mobile agent? 3215
Using -ObjC creates linker errors 3215
SQLite Databases 3215
How to instrument a hybrid app 3217
Instrumentation 3217
Debugging 3228
Kony One 3228
Known issues 3229
Mobile symbolication service 3230
Overview 3230
Enable the symbolication service and set storage 3230
Symbolication service REST API 3231
DSS client 3231
Integration with build systems 3234
Other modes 3236
User experience management - FAQ 3237
Continuous delivery and test automation 3241
Test automation explained 3243
Overview 3243
Step-by-Step Process to use Test Automation in AppMon 3245
Managing test runs 3251
Capture performance data from tests 3253
Test category 3254
Unit/integration and performance tests 3256
Java test framework 3256
.NET test framework 3257
PHP test framework 3260
Measures 3260
UI-driven browser test 3263
Requirements 3264
Configuration 3264
Mark a test as failed 3265
Integration 3266
Measures 3269
Samples 3270

Web API tests 3271
Usage 3272
Load tests 3275
External tests 3276
Overview 3276
Analyzing test automation data 3278
Test Overview dashlet 3278
Test Results dashlet 3280
Load Test Overview dashboard 3283
Jenkins plugin 3284
Automation and integration 3285
Build server 3286
Jenkins plugin 3287
Community provided plugins 3288
Continuous integration builds 3289
Integration process 3289
REST interfaces for test automation 3292
Test automation walkthroughs 3294
Integrate web API performance monitoring in JMeter 3295
Overview 3295
Requirements 3295
Goal 3296
Steps 3296
Download 3310
Integrate web API performance monitoring in SoapUI 3311
Overview 3311
Requirements 3311
Goal 3312
Create a SoapUI project that tests the functionality of a web service 3312
Monitor the functional test with AppMon 3315
Download 3322
Test automation FAQ 3323
FAQs 3323
Test automation terms 3324
Integrations and extensions 3326
IDE integration 3328
Eclipse integration plugin 3329
Launcher 3330
Test result 3330
Source code look-up 3331
IntelliJ IDEA plugin 3333
Launcher 3333
Test result 3334
Source code look-up 3335

Visual Studio extension 3337
Launcher 3337
Source code look-up 3338
Source lookup feature 3340
Lookup types 3340
Views in AppMon Client that offer source code lookup 3347
Source code viewer 3350
Load testing and monitoring tool integration 3351
DC RUM integration 3352
Supported releases 3352
Synthetic Monitoring integration 3354
Synthetic test integration for Application Monitoring 3354
UEM-PurePath integration for Synthetic Monitoring 3355
Synthetic test integration for Application Monitoring 3356
Overview 3356
UEM-PurePath integration for Synthetic Monitoring 3365
Viewing Data in AppMon 3366
AppMon drilldowns - landing dashboards 3368
Troubleshooting 3372
Enterprise Synthetic Integration 3374
SilkPerformer integration 3375
Segue/Borland SilkPerformer and SilkCentral Performance Manager integration 3375
LoadRunner integration 3379
LoadRunner script converter 3379
Modifying LoadRunner scripts for AppMon 3379
Conversion result 3383
Development kits 3385
Agent Development Kit (ADK) 3386
Purpose 3387
Components and platforms 3387
Concept 3387
ADK package contents 3390
Licensing 3391
Troubleshooting 3391
Native ADK 3393
Initialization 3393
Sensor placement and argument capturing 3394
Tagging 3396
Logging 3398
Configuration 3399
Sample applications 3403
Tagging ADK for .NET 3408
Using the Tagging ADK 3408

Tagging ADK for Java 3418
Linking client & server paths 3418
Using the ADK 3419
The tagging interface 3425
CICS and IMS Tagging ADK 3430
Link the stub 3431
Include API function declarations for C and PL/I 3431
API naming conventions 3431
API function examples for COBOL, PL/I and C 3432
API return codes 3443
Negative return codes 3444
API notes 3445
API include and sample files 3446
JavaScript ADK and JavaScript Ajax ADK 3447
Introduction 3447
Requirements 3447
Using the JavaScript ADK for Flash and Silverlight applications 3463
Collecting streaming media measures from Flash players 3467
Requirements 3467
Implementation 3469
Integration with web load testing and monitoring tools 3475
Overview 3475
Request header 3476
Response header 3477
Viewing Tagged Web Requests in AppMon 3478
Synchronizing session recording with load test runs 3478
WebAPI tests 3479
Usage 3480
Plugin architecture 3483
Plugin types 3484
Plugin development environment 3485
Building FastPacks 3486
Overview 3486
File structure 3487
Resource types 3490
Manage and develop user plugins 3491
Introduction 3491
Plugin types 3491
Library 3494
Plugin management 3496
Overview 3496
Manage plugins 3496
Plugin properties 3499

Developing a user plugin 3501
Overview 3501
Using the plugin development tab 3504
Using the New Plugin wizard 3506
Testing plugins 3520
Status codes 3521
Plugin files and directory structure 3523
API / PDK documentation download 3525
Plugin best practices and examples 3526
Working with a revision control system (RCS) 3526
Working with an IDE (Eclipse) 3526
Debugging a plugin 3527
Examples 3528
Dynamic monitor measures 3531
Plugin frequently asked questions 3533
Manual uninstallation of a system plugin 3535
Overview 3535
Removal procedure 3535
Realtime streaming 3538
Are there other ways to export AppMon data? 3538
Which one to choose? 3538
What is the Real Time Business Transactions Feed? 3539
What is the PureLytics Stream? 3539
PureLytics Stream 3540
Overview 3540
Using the PureLytics Stream with ElasticSearch and Kibana 3542
Technical details and limitations 3548
AppMon reference 3550
Client UI overview 3551
Menu bar and toolbar (1) 3552
Default Monitoring Perspective & dashboard tabs (2) 3553
Breadcrumb navigation buttons (3) 3553
Cockpit sidebar (4) 3555
Configuration menu (5) 3555
Dashboard content (6) 3556
Product and security notifications (7) 3558
Start Center 3560
Administration 3560
Getting started 3562
Take the AppMon Client wherever you go 3563
Get the Portable Client 3563
REST interfaces 3565

Server REST interfaces 3567
Overview 3567
HTTP response codes 3568
Authentication and authorization 3568
Legacy Server REST interfaces 3570
Overview 3571
HTTP response codes 3571
Authentication and authorization 3572
Plugins and updates (REST) 3573
Install plugin or update 3574
Plugin or update installation state 3575
Get update confirmation 3576
Create update confirmation 3577
List all update confirmations 3578
Agent Groups (REST) 3580
Agent Groups 3580
Agent mappings 3586
Dashboards and reporting (REST) 3593
Generate a dashboard report 3593
Dashboard information 3597
Types and formats 3598
Extended XML reporting 3600
XML reporting (REST) 3601
Overview 3601
Examples 3606
Incident downtime (REST) 3608
Overview 3608
List all incident downtimes 3609
GET incident downtimes 3610
Add or update incident downtimes 3611
Delete incident downtimes 3613
Questions and answers 3614
Incidents (REST) 3616
XML data description 3616
Get Incident 3617
Get all incidents 3618
Create incident 3620
Update incident 3621
Delete incident permanently 3623
Live sessions (REST) 3625
Store session 3625
Start session recording 3627
Stop session recording 3628
Deprecated 3629
Clear session 3629

Memory dumps (REST) 3631
XML data description 3631
Memory dump 3631
Post-processing 3640
Deprecated 3645
Performance Warehouse (REST) 3648
Connection status 3648
Connection configuration 3652
Agents and collectors (REST) 3657
Agents 3657
Collectors 3660
Deprecated 3662
PureLytics Stream (REST) 3663
Get PureLytics job 3664
Create PureLytics job 3668
On-demand export 3668
Resource dumps (REST) 3669
Create a thread dump 3669
Get thread dump status 3670
Create a sampling session 3671
Get sampling session status 3672
Server management (REST) 3674
Version 3674
License information 3675
License import 3676
License volume import 3678
Restart server 3679
Shut down server 3679
Support archives 3679
Deprecated 3681
Stored sessions (REST) 3684
List stored sessions 3684
Metadata 3687
Import and export 3689
Delete 3691
Reanalyze 3694
System Profile templates (REST) 3696
Create new System Profile based on template 3696
Copy agent group to existing system profile 3697
System Profiles (REST) 3699
Profile list 3699
Profile meta data 3702
Enable or disable System Profile 3703
System Profile configurations 3703

Tasks and monitors (REST) 3706
Tasks 3706
Monitors 3707
Test automation (REST) 3708
Server REST examples 3710
Alert suppression based on Business Hours (REST) 3711
Step 1: Create Business Hours definition 3711
Step 2: Create alert suppression 3712
Monitor creation (REST) 3714
Step 1: Create default Monitor for a specific type 3714
Step 2: View available Monitor task parameters 3715
Step 3: Update task parameters and execution target 3716
Step 4: Combine step 1 and 3 for automation scripts 3717
Step 5: Add target hosts 3717
User management REST 3719
Step 1: Query available permissions 3721
Step 2: Create a new role 3721
Step 3: Assign permissions to role 3723
Step 4: Create new user group 3724
Step 5: Add System Profile permissions to user group 3725
Step 6: Add dashboard permissions to user group 3727
Step 7: Add user 3728
Step 8: Add user to user groups 3729
Documentation downloads 3731
Download notes 3731
Keyboard shortcuts 3732
Windows & Linux shortcuts 3732
Mac shortcuts 3736
Glossary 3742
V 3742
U 3742
T 3743
S 3743
P 3744
M 3745
L 3745
K 3745
I 3746
H 3746
G 3746
E 3747
D 3747
B 3748
A 3748
* 3748
AppMon Vocabulary 3749

Measure 3750
Agent Group 3751
Auto Sensor 3752
Baseline 3753
Business Transaction 3754
Bytecode injection 3755
Discovery run 3756
DT_HOME 3757
Entry point 3759
Error 3760
Garbage collection size 3761
Host 3763
Incident 3764
Instrumentation 3765
Inter tier time 3766
Keep alive graph 3767
Logical application 3769
Agent 3770
Memory snapshot post-processing 3771
Monitor 3773
*NIX 3774
PureLytics 3775
PureModel technology 3776
PurePath technology 3777
PureStack technology 3778
Session 3779
Shallow size 3780
System under diagnosis (SUD) 3781
Task 3782
Transaction Flow 3783
Transaction splitting 3784
UEM 3785
Update 3786
Versioning scheme 3788
AppMon platform overview 3789

Features and capabilities 3790
Comprehensive & continuous application monitoring 3791
User experience management 3792
Real-time business transaction monitoring 3793
Host and system monitoring 3794
Visualize and share performance data and problem context 3795
Smart baselines and alert notifications 3797
Extensibility and systems integration 3798
Designed for continuous delivery processes 3799
Extensive technology and application environment support 3800
AppMon SaaS 3800
Architecture 3801
AppMon Server 3802
AppMon Frontend Server 3803
AppMon Memory Analysis Server 3803
AppMon Collector 3803
AppMon Performance Warehouse 3803
AppMon Client 3804
PurePath explained 3805
PurePath technology 3805
PureStack technology 3806
PureModel technology 3807
Editions 3809
Production edition 3810
Pre-Production edition 3810
Detailed feature comparison 3810

Dynatrace AppMon 2017 May Release Notes 3817
Platform support changes 3818
Support level and end of life overview 3818
End of Life policy 3818
Mainframe environments 3821
Supported OS for mobile 3821
Browser support for UEM 3823
AJAX frameworks for UEM 3825
Frameworks 3826
Web Service Stacks 3827
Messaging 3828
Remoting 3829
ESBs and SOA 3830
Platforms and operating systems 3831
Databases, connection pools, and big-data solutions 3832
Servlet containers & application servers 3834
Other supported application environments and technologies 3834
Application environments and technologies 3837
Dynatrace AppMon components 3842
2017 May resolved issues 3842
Fast and mobile friendly documentation! 3845
Why your feedback matters 3846
AngularJS, IBM DB2, React and much more 3846
Enterprise ready end-to-end encryption, SSO and Licensing 3848
Webserver module insights and BETA for .NET Core, powered by OneAgent 3849
Need more? Don’t worry, we scale with your business 3850
User experience management redefined 3852
The best metrics to optimize user experience 3853
Database insight for every team 3854
Amazing web dashboards 3856
Onboard DevOps teams 3858
Better apps with every deployment 3859
Sourcecode level insight for everyone 3861
Speed up your frontend teams 3863
Dynatrace AppMon 2017 May Mainframe Release Notes 3865
Resolved and known issues 3865

Update 2 3866
Nginx 3866
Mainframe 3866
Session Storage 3866
Charting 3866
Dashboard 3866
Java Agent 3866
WebUI 3867
Instrumentation 3867
Correlation and Analysis 3867
.NET Agent 3867
DB Agent 3867
Collector 3868
PWH 3868
Client 3868
Server Platform 3869
Server 3870
UEM 3870
All resolved issues 3870
Most critical issues since the last public updates 3870
7.0.2.1011 Resolved Issues 3872
Resolved Issues 3872
2017-07-04 3872

AppMon Documentation



Knowledge base



Community Forums



Dynatrace University



Dynatrace Blog

More AppMon help resources

Integrations and extensions

Overview
Plugins
Agent Development Kit (ADK)
REST interfaces
Load test & monitoring tools

Continuous delivery and test automation

Overview
Automation and integration
Capture performance data from tests

User experience management

Overview

Web
Android
iOS

Database monitoring

Overview
Database health monitoring
SQL execution plan

Infrastructure monitoring

Overview
Host health monitoring
Process health monitoring

Application monitoring

Overview
Agents
Sensors
Dashboards
Business transactions

Administration

Overview
Server administration
Client administration
User permissions and authentication

Installation

Overview
System requirements
Install agents
Documentation downloads

Previous releases

Dynatrace 6.5

Dynatrace 6.5
Dynatrace 6.3

Get to know AppMon

Free 30-day trial
Free analysis of your shared PurePath
Watch online performance clinics

Getting started

Getting started with AppMon
Architecture
Release notes
Introduction of the current release

Getting started

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Business use cases and tasks

Continuous delivery use cases and tasks

Developer use cases and tasks

Operations use cases and tasks

Previous versions of AppMon documentation

Testers use cases and tasks

Walkthroughs

Getting support

After you have AppMon installed and you have Agents gathering performance data, there are several steps that you can take enhance the default capabilities of AppMon.

Step 1 - Explore the walkthroughs

Read our walkthroughs to learn how to use AppMon to monitor and troubleshoot common user experience, application, and infrastructure performance issues.

Step 2 - Review defaults and adjust as necessary

AppMon comes with a plenty of default resources that provide insight into your monitored application. These default resources can be duplicated and customized to better match your application and meet your business needs.

- Inspect the Built In Dashboards and make customized versions as necessary.

- Take a look at the Business Analytics dashboard.
- Review the defaults for the built-in auto sensors and determine the need for installing custom sensors
- Configure auto sensor resolution settings to ensure acceptable overhead.
- Refine the default business transaction discovery.
- Review the default rules for error detection.

Step 3 - Browse use cases and tasks for common roles

- Developers
- Test Engineers
- Operations
- Business
- Continuous Delivery
- User Experience

Step 4 - Fine-tune AppMon and capture more data

Use the following features to monitor application and infrastructure performance:

- Configure baselines and use incidents and alerts to detect and troubleshoot problems before users are impacted.
- Set up monitors to capture additional performance data.
- Create and share built-in or custom dashboards – be it in the AppMon Client or the browser – to visualize key performance indicators (KPIs).
- Collaborate with other team members using stored sessions.
- Automate commonly performed tasks such as storing sessions.

Step 5 - Explore advanced configuration topics

Additional features that enhance the default capabilities of AppMon:

- Install plugins to unlock new product features for system monitoring, application monitoring, custom actions, and lifecycle automation.
- Use the Agent Development Kit, JavaScript ADK, and Mobile app ADK to extend the default AppMon Agent monitoring capabilities by integrating AppMon with technologies such as native C or C++ applications, custom communication protocols, custom JavaScript frameworks, and native mobile applications.

- Learn how to use AppMon to support continuous delivery processes.

Developer use cases and tasks

PurePath and sensors

- How to capture context data
- Advanced sensor rules
- PurePath explained
- System Profile - Sensors
- Configuring out-of-the-box Sensors

Memory, thread, and CPU diagnostics

- Memory and thread diagnostics
- Thread diagnostics
- Memory diagnostics
- Sampling based diagnostics

Performance hotspot diagnostics

How to identify performance hotspots

Response time and method hotspots

Solve database performance problems

Database and Database hotspots

Error diagnostics

Analyzing application errors

System Profile - Error Detection

Exceptions and Logging

Operations use cases and tasks

Application Monitoring

- Smart application monitoring (incidents, alerts und SLAs)
- Smart infrastructure monitoring
- Basic dashboarding
- Application monitoring
- Infrastructure overview
- Incidents

System and availability monitoring

- Synthetic monitoring integration
- Operations and notification plugins
- System monitoring plugins
- Windows performance monitor
- Unix system monitoring

Performance hotspot diagnostics

How to identify performance hotspots

Response time and method hotspots

Solve database performance problems

Database and database hotspots

Error diagnostics

Analyzing application errors

System Profile - Error Detection

Exceptions and logging

Testers use cases and tasks

Load testing

- Load testing made easy with AppMon
- Analyzing web requests
- Tagged web requests
- Load test overview dashboard
- Web page performance analysis dashboard

Tool integrations

- Integration with web load testing tools
- Synthetic test integration for Application Monitoring
- SilkPerformer integration
- LoadRunner integration
- JMeter integration

Performance hotspot diagnostics

How to identify performance hotspots

Response time and method hotspots

Solve database performance problems

Database and Database hotspots

Memory and thread diagnostics

Error diagnostics

Analyzing application errors

System Profile - Error Detection

Exceptions and Logging

Business use cases and tasks

Data capturing

- Capturing business context data
- Advanced sensor rules
- PurePath explained
- System Profile - Sensors
- Configuring out-of-the-box Sensors

Business intelligence

- Exporting real time data
- Real time Business Transactions feed explained
- Splunk application
- Unlocking insight - UEM into Splunk

Business Transactions

- Advanced Business Transactions
- Business Transactions best practices
- What are Business Transactions?
- Usage of Business Transactions

Dashboards

- Fancy dashboards in AppMon
- Client and Web dashboards
- Reporting
- Incidents and alerting

Continuous delivery use cases and tasks

AppMon in continuous integration

- Test automation with AppMon
- Continuous delivery and test automation
- Integrate AppMon in unit and integration tests
- Client and Server-Side REST Interfaces
- AppMon automation library
- Jenkins plugin for AppMon

Performance hotspot diagnostics

- Identify performance hotspots
- Response time and method hotspots
- Solve database performance problems
- Database and database hotspots

Quality metrics in continuous delivery

Fixing real problems of real users

UEM tips and tricks

Third party content analysis

UEM health check

Automated deployments

Continuous delivery 101: automated deployments

How to automate enterprise application monitoring with Ansible

How to automate enterprise application monitoring with Ansible - Part 2

Error diagnostics

Analyzing application errors

System Profile - Error Detection

Exceptions and logging

Walkthroughs

Application performance monitoring

Analyzing a slow transaction

Analyzing out of memory errors

How to retrieve performance metrics with Windows Performance Monitor

Identify problems in your application

Infrastructure monitoring walkthroughs

Memory analysis

Setting up application availability monitoring

Setting up infrastructure monitoring

Using host and hostgroups for large system monitoring

Using host and process health monitoring

Proactive performance management

Automating reporting tasks with server-side reporting

Regression Analysis

Diagnostics

Analyze exceptions and log messages

Analyzing a slow transaction

Analyzing out Of memory errors

Memory analysis

Regression analysis

Using custom JMX and PMI measures in charts

Using host and hostgroups for large system monitoring

Business Transaction management

Follow your transactions with Transaction Flow

How to use Business Transactions since AppMon 4.2

Walkthroughs are designed to provide detailed instructions and examples for using AppMon to monitor and troubleshoot common application performance problems. For many of the walkthroughs, a demo application is used to provide the necessary context for explaining AppMon feature usage. Several walkthroughs provide downloadable stored sessions files that you can import into the AppMon Client for demonstration purposes. This will not affect your environment or production settings.

Previous versions of AppMon documentation

> **Note**
>
> Documentation for AppMon versions that have reached end-of-life are no longer browseable but are still available in as a single PDF download.

- Dynatrace 6.5
- Dynatrace 6.3

You can find documentation for previous versions of the AppMon documentation in the following locations:

Installation

Before you install

System requirements

AppMon basics

Deployment Guide

Installation overview

Set up communication connections

Component installation

Server

Collector

Agents

Memory analysis server

Automatic Installation

Licensing

License activation

Manage UEM visits volume

License assignments

After you install

Set up the Server

Set up Collectors

Set up Agents

Set up Memory Analysis Server

Set up the AppMon Client

Set up the Webstart Client

Instrument Applications

Install considerations

Highly available installations

Migrate a Server to a new host

Silent installation

To upgrade your AppMon installation, refer to the Upgrade and migration guide.

For a standard installation in a live environment, see the instructions listed on the Installation overview page.

To evaluate AppMon or for small-scale demo installations, check out the Quick Start and Evaluation Guide.

Install first time or upgrade?

Installation overview

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Prepare for installation

Download and install AppMon

Activate licenses

Basic installation settings

Set up your System Profile and instrument applications

Verify data is being captured in your System Profile

Start using AppMon

Related pages

System requirements

AppMon basics

Deployment Guide

Licensing

Set up communication connections

Set up system components

System Profile

Application Environment Configuration

Monitoring Overview dashboard

Watch and learn

Installing Dynatrace in Windows

Installing Dynatrace in Unix

This page lists essential steps for planning, installing, and configuring your AppMon deployment in a live environment.

Prepare for installation

- Review the System requirements to determine that your environment has the necessary resources and compliant target systems on which to successfully install and run AppMon components.
- Learn about the AppMon + UEM basics:
  - Application Monitoring architecture.
  - AppMon features, and an overview of the Web and Client User Interfaces.
- Go through the steps in the Deployment Guide to properly identify the sizing needs of your environment and the AppMon components.
- Verify that network requirements are met and make ports available for the AppMon components to communicate to each other.
- Set up the Performance Warehouse database for long term storage of performance data.

Download and install AppMon

- Download software component installers for the AppMon Server, Collector, and Client based on your deployment plan. You can also download the a full installation package and (on Windows) select specific components.
- Install the AppMon Server (backend and frontend processes / Windows services / *NIX daemons). Important considerations include:
  - SSL Communication
  - User Authentication
  - High Availability Installations
- Install the AppMon Collector.

- Install the AppMon Client according to the following options:

  - **Local Client installation** - If you selected the basic installation option in the Setup wizard, a Client is installed on the same machine as the AppMon Server.
  - **Remote Client installation** - You can install the Client on a supported Windows, Linux, or Mac machine that is different from the machine on which the AppMon Server is installed.
  - **Set up the Webstart Client** - If your browser supports Java Webstart protocol, you can launch a fully functional AppMon Client directly from the AppMon Server via a URL.
- Install the Memory Analysis Server as required by your Deployment.

Activate licenses

- Set up or verify that the product licensing requirements are sufficient for the number of agents and UEM capacity (if applicable) required by your environment. (note: move to licensing)
- Start the Client and log in to the Server. Verify that the Collector is connected to the Server.
- Install and activate your licenses.

Basic installation settings

- Adjust your system memory sizing as needed and restart the Server.
- Set the session storage disk quota and location, and then restart the Server.

> **Note**
>
> Setting session disk quota only applies if disk allocation is required outside of the installation directory.

- Connect the Server to the Performance Warehouse database.
- Decide to run memory analysis by default or on demand. If your deployment doesn't include a Memory Analysis Server, disable the alert.
- Set up Collectors
- Set up an SMTP email server to enable alerting.
- Change the default administrative account password and set up user accounts.

Set up your System Profile and instrument applications

- Connect AppMon to your applications by creating one or more System Profiles, which are

workspaces in which you can model the tiers and nodes of your application environment and manage the initial setup and mapping of AppMon Agents.
- Use the Application Environment Configuration wizard to initially define the Agent Groups / tiers and Agent mappings required to configure your application processes and web servers to load the AppMon Agent and connect to the AppMon Collector / Server. During the wizard set up, you are prompted to download the Agent installer appropriate for your environment and configure the Agent connection string.

As an alternative, you can download the Agent installers from the AppMon Download Center and install and configure the Agents manually. See Agent Configuration for more details.

Verify data is being captured in your System Profile

Open the Monitoring Overview dashboard and verify that the transaction flow map visually models your application environment correctly. Also, review the following built-in analysis dashboards to make sure data is being captured: Active Visits, Application Overview Dashboard, Infrastructure Monitoring Dashboards, End User Experience, Mobile Applications, and Web Request Performance.

Start using AppMon

After completing these steps, follow the links on the Getting Started page to learn how to use AppMon for proactive and actionable insight into end-user experience, application performance and host and system performance.

System requirements

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Dynatrace AppMon components

Application environments and technologies

Other supported application environments and technologies

Dynatrace AppMon components

AppMon Agent

- Supported agent platforms and operating systems
- Oracle (Sun) JVM, IBM JVM, JRockit JVM, Azul JVM
- 10MB disk space
- Dynatrace AppMon bootstrap agent 6.x
- DB Agent supports Oracle Database 11g/12c, Microsoft SQL Server 2012/2014/2016 and MySQL 5.6/5.7

AppMon Server

The AppMon Server (backend process / Windows service / *NIX daemon) is only supported on operating systems that support Java 8. Only the included Java Runtime Edition is supported.

- **Windows x64**: 10/8.1/8/7/Vista or Server 2008/2012/2016 with latest service pack; installer (.msi) does not need a pre-installed JRE; package contains a private Java Runtime Edition v8.
- **Linux x64**: With a recent kernel version 2.6 or higher.
  Installation requires a pre-installed Java Runtime Edition v8.
- **LDAP**: For AppMon user authentication.
  - Active Directory
  - openLDAP

Hardware requirements

See the Deployment Guide for detailed hardware recommendations specific to your load.

Deployment size[1]	Physical cores required[2]	Recommended physical memory[3]	Also available memory options	Recommended disk space	A
Demo 5 agents | 150 transactions/sec	2	2GB	4GB, 8GB	1.5GB + 100GB Continuous session storage/week	N
Small 10 agents | 300 transactions/sec	4	4GB	8GB, 16GB	3GB + 200GB Continuous session storage/week	Y
Medium 250 agents | 1,500 transactions/sec	8	8GB	16GB, 32GB	5GB + 1TB Continuous session storage/week	Y
Large 750 agents | 3,500 transactions/sec	16	16GB	32GB, 64GB	10GB + 2TB Continuous session storage/week	Y
XLarge **32GB** | 1600 agents | 5,000 transactions/sec	24	32GB	-	20GB + >2TB Continuous session storage/week	N
64GB | 1800 agents | 5,800 transactions/sec	24	64GB	-	20GB + >2TB Continuous session storage/week	N
96GB | 2000 agents | 6,500 transactions/sec	24	96GB	-	20GB + >2TB Continuous session storage/week	N

[1] The maximum agent numbers consider a typical combination of different agents types (Java, .NET, host, WebServer, PHP, etc.).
In special circumstances, significant more agents are supported (e. g. mainly host agents). Please contact support for guidance and confirmation.

[2] No virtual cores (Hyper-Threading), for Medium and above a clock speed of 2.6 GHz or higher is required.

[3] Required memory for the AppMon Server and operating system. If there are other processes running, then additional heap is required.

Note

To improve performance, AppMon always uses the maximum available CPU and memory resources.
If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

VM support

All deployment sizes listed above may be deployed on a virtual machine. However, you must consider the following:

- No over-commitment of resources (CPU, memory, network, disk) and no virtual-only resources that are not backed by dedicated physical resources.
- Large and XLarge only. VM is equivalent to a physical box in all performance metrics. For more details see Guidelines for running AppMon in VMWare.
- XLarge only. Database partitioning is mandatory.

AppMon Collector

- **Windows** x86/x64 10/8.1/8/7/Vista or Server 2008/2012/2016 with latest service pack
- **Linux** x86/x64 with a recent kernel version 2.6 or higher
- **Solaris** SPARC or x86/x64 10/11/11.1/11.2/11.3
- **AIX** Power 7.1/7.2
- **zOS Unix** 2.1/2.2
- **zOS z/Linux** SLES 11/12 & RHEL 6.6/6.7/7.0/7.1/7.2

Compatible JVMs for Collector runtime: Oracle (Sun) JVM 1.8+ (included with Windows and Linux installers), IBM JVM 1.8+.

AppMon also provides different Collector sizes. For more information, see the Deployment Guide. For a proper configuration of the Collector(s) environment, see Set up Collectors.

Deployment Size	Recommended Disk Space	Required Memory[1]	Physical Cores[2]

Small[3] 100 Java / 25 .NET Agents	30GB	2GB	1
Medium 200 Java / 100 .NET Agents	50GB	4GB	2
Large 400 Java / 250 .NET Agents	50GB	8GB	4

[1] Required memory for the Collector. If there are other processes running, then additional heap is required.

[2] No virtual cores (Hyper-Threading).

[3] Small = Default shipped Collector size

AppMon Client

- **Windows x64** 10/8.1/8/7/Vista or Server 2008/2012 with latest service pack (certified)
- **Linux x64** SUSE / RedHat Linux with GTK (certified)
  The Linux Client package requires a JRE 1.6 or later for installation. (A private JRE 1.8 for runtime is included!)
  Mac: OS X 10.8/10.9/10.10/10.11 64bit (supported)
- 500MB disk space
- 2GB memory
- Minimum screen resolution: 1024x768px
- The Java **Webstart Client** can be started with any Webstart compatible browser.
  - Internet Explorer 8/9/10/11 and Firefox 38-48 are certified by AppMon.
  - The Java Webstart Client requires a Oracle JRE >= 1.8.0_45.

AppMon Web

AppMon Web is built for viewing on **tablet devices** and **desktop machines** that have a minimum browser resolution of 1024 x 768 pixels (e.g., iPad mini physical resolution 2048 x 1536 vs. browser resolution 1024 x 768). More details for your device can be found here.

AppMon Web is supported and tested with the following browsers:

- Edge 14+
- Firefox 50+

- Chrome 55+
- Safari 10+

Drilldowns to the **AppMon Client** require a running AppMon Client. See the AppMon Client requirements for more details.

AppMon Performance Warehouse

The AppMon Performance Warehouse uses a relational database to store long-term historical data. By default, AppMon installs and uses an embedded database for demo/testing purposes. Any production installation of AppMon **must use one of the following** database management systems **with the specified version** to host the Performance Warehouse database:

- Oracle 10g/11g/12c
- Microsoft SQL Server 2008 / 2012 / 2014 / 2016
- IBM DB2 Version 9.7 / 9.8 / 10.1 / 10.5 / 11.1
- PostgreSQL 9.2 / 9.3 / 9.4 / 9.5 / 9.6

AppMon supports partitioned tables for high-load scenarios for all supported databases. For all AppMon installations **greater than Large**, a partitioned database **is mandatory**. You can find more details in Performance Warehouse Partitioning for Dynatrace.

To determine disk-space requirements for the AppMon repository upfront, factor 15GB per 1,000 persistent measures (of all system profiles within the AppMon repository). You can find more details in the Deployment Guide.

AppMon zRemote Agent

The zRemote Agent is supported on the following operating systems:

- Windows x64 2008 R2 / 2012 R2
- Linux x64

Application environments and technologies

For third-party software, AppMon follows the End of Life (EOL) timelines announced by respective software vendors related to specific platform and version support. This means that AppMon announces EOL for support of specific platforms and versions once the respective software vendors announce EOL.

Supported Java virtual machines[1]

Operating system	Oracle JVM 8/7/6	IBM JVM 8/7/6	Oracle JRockit 6	SAP JVM 6.0/6.1[2,5] 64bit	OpenJDK
Windows	✓	✓	✓	✓	
Linux[3]	✓	✓[4]	✓	✓	✓
Solaris	✓		✓		
AIX		✓			
zLinux		✓			
zOS		✓			

[1] Only bytecode created by the Java Programming Language Compilers (javac) from supported JVM vendors.

[2] 64bit only.

[3] x86 and x64 architecture.

[4] PPC 32 and 64 bit.

[5] Only SAP Java Server VM 6.1.038 1.6-b04 on Windows and SAP Java Server VM 6.1.018 / 6.1.034 19.1-b02 on Linux.

Supported .NET Platforms

Operating system	.NET 3.5[1]	.NET 4.0[1]	.NET 4.5[1]	.NET 4.6[1]	.NET 4.7[1]	NET Core 1.0[1]	NET Core 1
Windows	✓	✓	✓	✓	✓	✓	✓

[1] *32 and 64bit.*

Supported web servers for UEM

Operating system	Apache Web Server 2.2/2.4	Microsoft IIS 7/7.5/8/8.5/10	IBM HTTP Server (IHS) 7/8/8.5	Or
Windows	✓	✓	✓	✓
Linux[1]	✓		✓	✓
Solaris[2]	✓		✓	✓
AIX	✓		✓	✓
zLinux	✓ 4			

[1] *x86 and x64 only.*

[2] *Solaris SPARC only.*

[3] *The NGINX agent is only available for builds where the debug symbols are available.*

[4] *Apache Web Server 2.2 only.*

Supported PHP versions

Operating system	PHP 5.3	PHP 5.4	PHP 5.5	PHP 5.6	PHP 7.0
PHP 7.1[5]	Windows (x86)	✓ 1,3	✓ 1,3	✓ 1,3	✓ 1,3
✗	✗	Linux (x86/x64)	✓ 1,2,3	✓ 1,2,3	✓ 1,2,3
✓ 1,2,3	✓ 1,2,3,4	✓ 1,2,3,4			

[1] mod_php (for Apache Web Server) and PHP CGI (for IIS).

[2] PHP FPM (fast process manager; for Nginx)

[3] PHP threaded MPM is not supported as it is not production setup safe. (reason: PHP_TS)

[4] Supported only via PHP OneAgent [5] As of 7.0.3

Supported Node.js environments

Operating system	Node.js 4.x	Node.js 5.x	Node.js 6.x	Node.js 7.x	Node.js 8.x
Linux	✓			✓	
Windows	(soon per update)		✓		✓
	✓	(soon per update)			

Supported frameworks:

- http, https, Connect, Express, Restify
- Errors
- Code-level visibility, memory metrics, CPU metrics

Supported Virtualization Monitoring

Operating system	vSphere 4.0	vSphere 4.1	vSphere 5.0	vSphere 5.1	vSphere 5.5	vSphere 6.0	vSph
Windows	✓	✓	✓	✓	✓	✗	✗
Linux	✓	✓	✓	✓	✓	✗	✗

Other supported application environments and technologies

Servlet containers & application servers

Servlet containers

Technology	Versions	Support level
Apache Tomcat	6/7/8	Certified
Jetty	8/9	Certified

J2EE 6/7/8 compliant app servers

Technology	Versions	Support level
WebSphere Application Server	7/8/8.5.5[1]	Certified
WebSphere Application Server	9[2]	Supported
WebSphere Liberty profile	8.5.5.5 - 16.0.0.2	Certified
WebLogic Application Server	11g/12	Certified
JBoss AS CE	6.0.0/7.1.1[3]	Certified
Wildfly	8.x/9.x/10.0	Certified
JBoss EAP	6.x/7.0.0	Supported
SAP NetWeaver	7.3 AS Java	Certified
Fujitsu Interstage	11[4]	Limited support
Glassfish	3.1/4 Open Source	Certified
Virgo/SpringSource dm Server	2.x	Supported

[1] only IBM JVM 7 on Windows, Linux & AIX

[2] only IBM JVM 8 on Windows, Linux & AIX

[3] not all JMX measures available

[4] only Fujitsu JVM 6/7 64bit

Others

Technology	Versions	Support level
WebMethods Broker & Integration Server	9/9.5 - 9.10	Certified
Sun Java Web Server	6.1	Certified
Microsoft IIS/ASP.NET	7/7.5/8	Certified
Microsoft IIS/ASP.NET	8.5/10	Supported
ASP.NET Core	1.0/1.1	Supported
Microsoft SharePoint		Certified
MS COM+ Component Services for .NET		Supported
Adobe LiveCycle		Supported
other/proprietary Java/.NET Servers		Limited support

Databases, connection pools, and big-data solutions

For DB Agents go to AppMon Agent section

Databases

Technology	Versions	Support level
IBM DB2	8/9[1]	Certified
Oracle DB[1,2]	9i/10g/11g/12c	Certified
Microsoft SQL Server[1,2]	2008/2012/2014/2016	Certified
PostgreSQL[1]	9.2/9.3/9.4/9.5/9.6	Certified
Derby[1]		Certified
generic support is available for ADO.NET data providers		Limited support
generic support is available for any Type4 JDBC driver		Limited support

[1] JDBC Type4 drivers

[2] ADO.NET data providers

Connection pools

Technology	Versions	Support level
WebSphere Connection Pool	7 - 8.5.5	Certified
WebSphere Connection Pool	9	Supported
WebLogic Connection Pool	10-12.1	Certified
JBoss Connection Pool	5.x-8.0	Certified
Apache DBCP	1.0-1.4/2	Certified

Tomcat Connection Pool	1.0.6 - 1.1.0	Certified
c3p0	0.8.5 - 0.9.5.0	Certified
GlassFish	3.x-4.x	Certified
Hybris		Certified
generic support is available for all J2EE connection pools		Limited support

NoSQL & big-data solutions

Technology	Versions	Support level
Cassandra	1.2.x[1], 2.0.1[1], 2.1.13[1], 3.0.3[1], 3.3.0[1], 3.4.0[1], 3.7.0[1]	Certified
MongoDB Server Versions	2.0.x, 2.2.x, 2.4.x, 2.6.x ,3.0[2]	Certified
Solr		Supported

[1] Java Thrift based Clients (Hector 1.0/1.1, Astyanax 1.x) and Datastax 1.0.2/2.1.0 - 2.1.3, 2.1.5-2.19, 3.0.0, 3.0.2

[2] Java drivers only, v. 2.3 - 2.13.x/3.0.0-3.0.2

Platforms and operating systems

On latest patch/service pack level

Technology	Versions	Support level
Windows[1]	10/8.1/8/7/Vista	Certified
Windows[1] Server	2008/2012/2016	Certified

Windows[1] Server	2008 R2 > 64 logical processors	Limited support
Suse Linux Enterprise Server[1]	11/12	Certified
RedHat Enterprise Linux AS[1]	6.7/7.0/7.1/7.2	Certified
RedHat Enterprise Linux AS Big Endian[2]	6.7/7.0/7.1/7.2	Certified
AIX[2]	7.1/7.2	Certified
Solaris[1, 3]	10/11/11.1/11.2/11.3	Certified
IBM zOS[4]	2.1/2.2	Certified
zLinux[5] SLES	11/12	Certified
zLinux[5] RHEL	6.6/6.7/7.0/7.1/7.2	Certified

[1] x86 and x64

[2] 7 and 8 architectures

[3] Solaris SPARC

[4] No host health information available

[5] OS390x

ESBs and SOA

Technology	Versions	Support level
IBM WebSphere Message Broker	8[1]	Certified
IBM Integration Bus[1]	9/10	Certified

Oracle Service Bus	10.3.4 / 12c (12.1.3)	Certified
JBoss ESB	4.12	Certified
WSO2	4.6[2]	Certified
Mule	3.0 - 3.8	Certified
TIBCO ActiveMatrix BusinessWorks Service Engine	5.8.2/5.9/5.10	Certified
Apache OpenEJB	3.1.4	Certified
SOA Software Service Manager 5.2/6.1		Limited support

[1] On AIX, Windows & Linux 64bit

[2] HTTP communication only

Remoting

Technology	Versions	Support level
RMI / IIOP		Certified
RMI / JRMP		Certified
RMI / T3		Certified
.NET Remoting		Certified
ORMI (OC4J 10g R3)		Certified

JBoss Remoting	Certified
WCF	Certified
HTTP Remoting	Certified
XML-RPC.NET	Certified
MS Service Fabric Remoting	Certified

Messaging

Technology	Versions	Support level
JMS	1.1/2.0[1]	Certified
IBM WebSphere MQ	7	Certified
Apache Active MQ	5.x	Certified
Microsoft MQ	3.0	Certified
RabbitMQ[2] Java & .NET	3.5.x/3.6.x	Certified
TIBCO EMS[3]		Certified
TIBCO Rendezvous	8.3.2 - 8.4.3	Certified

[1] On Glassfish 4 and WildFly 8

[2] Client only

[3] Only for JMS 1.1

- *Place JMS sensor on sender and receiver*
- *No agent needed on TIBCO EMS itself*

Web Service Stacks

Technology	Versions	Support level
IBM WebSphere	7/8/8.5.5	Certified
IBM WebSphere	9	Supported
Oracle WebLogic	8.1/9/9.1/9.2/10/11g/12	Certified
.NET WebServices		Certified
AXIS	1.4	Certified
AXIS2	1.3 / 1.4 / 1.5 / 1.6	Certified
Hessian	2.1/3/3.1/3.2/4.0	Certified
Oracle WebServices		Certified
JAX WebServices		Certified
Codehaus XFire	1.2	Certified
Apache CXF	2.2/2.3/2.4/2.5/2.6/3.0	Certified
Apache Wink	1.2.0 - 1.4.0	Certified
Restlet	2.1.0 - 2.3.7	Certified

RestEasy	2.3.5 - 3.0.19	Certified
Jersey	1.15/2.x	Certified
Spring Web Services	2.0	Certified
WebMethods Glue		Supported

Frameworks

Technology	Versions	Support level
Hibernate		Supported
Struts	1.x/2.x	Supported
AWT/Swing/RCP/WPF/WinForms		Supported
Spring		Supported
Applets		Supported
Play[1]	1.3/2.2/2.3/2.4/2.5	Certified

[1] Java only, no Scala support

HTTP Frameworks

Technology	Versions	Support level
JDK Http API		Certified

Apache HttpClient	3.1/4.x	Certified
Apache HttpAsyncClient	4.0.1	Certified
Jetty Http Client	8/9	Certified
NING Http Client	1.6.x / 1.7x / 2.x	Certified

Logging

Technology	Versions	Support level
Java Logging API		Certified
Log4J	2	Certified
Log4NET		Certified
Logback		Certified

AJAX frameworks for UEM

UEM is compatible with all JavaScript frameworks. Automatic click detection is available for:

Technology	Versions	Support level
dojo	1.6.1 – 1.12.2[1]	Supported
extjs	3.4.0 – 6.2.0[1 2]	Supported
icefaces	2 & 3[1]	Supported

jquery	1.3 – 1.12.4, 2.0 - 2.4 & 3.0-3.2[1]	Supported
prototype	1.6.0 – 1.7.3[1]	Supported
angular	1.0 – 1.5.8, 2.0.0[1]	Supported
mootools	1.4.5 – 1.6.0[1]	Supported

[1] Other versions of these frameworks still work, but certain measures might be missing. The JavaScript ADK can be used to capture the missing data.

[2] In certain cases the JS agent does not capture the correct end time for extjs 6 user actions if the promise API is used.

The following frameworks are covered with the basic XHR detection:

Technology	Versions	Support level
GWT		Supported
zk		Supported
RichFaces		Supported
ASP.net AJAX		Supported
YUI		Supported
Ember.js		Supported

The following frameworks are covered with the jQuery sensor:

Technology	Versions	Support level

backbone.js	Supported

Browser support for UEM

Highlighted entries are compatible with any JavaScript enabled browser.

Browsers

Technology	Versions	Support level
Internet Explorer	11	Supported
Internet Explorer	*8, 9, 10*	*Limited support*
Firefox[1]	*3.5 - 27*	*Limited support*
Firefox	28+	Supported
Safari	5+	Supported
Chrome	10+	Supported
Opera	*12*	*Limited support*
Opera	15+	Supported
Embedded Browser		*Limited support*
Microsoft Edge		Supported

[1] only load actions, no special frameworks.

Mobile browsers

Technology	Versions	Support level
Mobile Safari	4+	Supported
Android Webkit	*2.3 - 3.x*	*Limited support*
Android Webkit	4.0+	Supported
Internet Explorer Mobile		Supported

CORS support

Technology	Versions	Support level
Safari	4+	Supported
Internet Explorer	8+	Supported
Firefox	3.5+	Supported
Chrome	10+	Supported
Opera	15+	Supported

W3C resource timing support

Technology	Versions	Support level
Internet Explorer	10+	Supported
Chrome	25+	Supported
Firefox	35+	Supported

Opera	15+	Supported
Android Webkit	4.4+	Supported
Internet Explorer Mobile	10+	Supported

Supported OS for mobile

Technology	Versions	Support level
Apple iOS	6 - 10	Certified
Android	2.3 - 7	Certified

Frameworks

Technology	Versions	Support level
PhoneGap	2.9.1+	Supported
Apache Cordova	3.x	Supported
Custom WebViews		Supported
KonyOne on Android		Supported
OkHttp	3 (and OkHttp 3 based libraries like Retrofit 2)[1]	Supported

[1] *Supported for Android Auto Instrumentation.*

Mainframe environments

Technology	Versions	Support level
z/OS	2.1 / 2.2	Certified
WebSphere MQ	7.1/8.0	Certified
WebSphere MQ	9.0	Supported
DB2	v10 / v11	Certified
DB2	v12	Supported
CICS Transaction Server	4.1 / 4.2 / 5.1 / 5.2 / 5.3 / 5.4 open beta	Certified
CICS Transaction Gateway[1]	8.1 / 9.0 / 9.1 / 9.2	Certified
CICS SOAP over HTTP		Certified
CICS DLI		Certified
CICS MQ Bridge[2]		Certified
IMS	13 / 14 / 15 QPP	Certified
IMS MQ Bridge[3]		Certified
IMS MQ Trigger Monitor		Certified
IMS MSC		Certified
IMS Shared Queues		Certified

IMS Soap Gateway[3]	3.2	Certified
IMS Enterprise Suite[3]	3.2	Certified
IMS Connect API[3]	3.2	Certified
IMS TM Resource Adapter	13 / 14 / 15 QPP	Certified

[1] CTG for zOS: EXCI and IPIC CTG for Multiplatforms: IPIC (no support for TCPIP or SNA) CTG local mode: IPIC (no support for EXCI on z/OS)

[2] Requires CTS 4.2 or above.

[3] Inbound only

Releases that are expected to work but are no longer supported by IBM

Technology	Versions	Support level
z/OS	1.12 / 1.13	Limited support
IMS	11 / 12	Limited support
DB2v9		Limited support
CICS TransactionGateway[1]	8.0	Limited support
IBM WebSphere MQ	7.0.1	Limited support
IBM IMS Soap Gateway[2]	2.2 / 3.1	Limited support
IBM IMS Connect API[2]	2.2 / 3.1	Limited support

[1] CTG for zOS: EXCI and IPIC CTG for Multiplatforms: IPIC (no support for TCPIP or SNA) CTG local mode: IPIC (no support for EXCI on z/OS)

[2] Inbound only

CPU cores and memory requirements

On this page

Why AppMon has hardware recommendations?

I upgraded to the next size and you want me to add more cores. This doesn't make sense for me, because my current CPU is not fully utilized?

Why is the CPU clock speed also important?

Bare metals vs. virtualization

Conclusion

Related pages

Verify CPU requirements

Default deployment sizes

Server and Performance Warehouse best practices

Collector best practices

Network bandwidth best practices

Disk IO determination best practices

Customer environment examples

Dynatrace AppMon is a sophisticated application that requires resources for data processing, and user activity. In the Default Deployment Sizes we described certain hardware specifications in order to achieve a given level of throughput and user activity. If these recommendations are not followed, AppMon cannot achieve these throughput numbers and will not behave as desired.

Why AppMon has hardware recommendations?

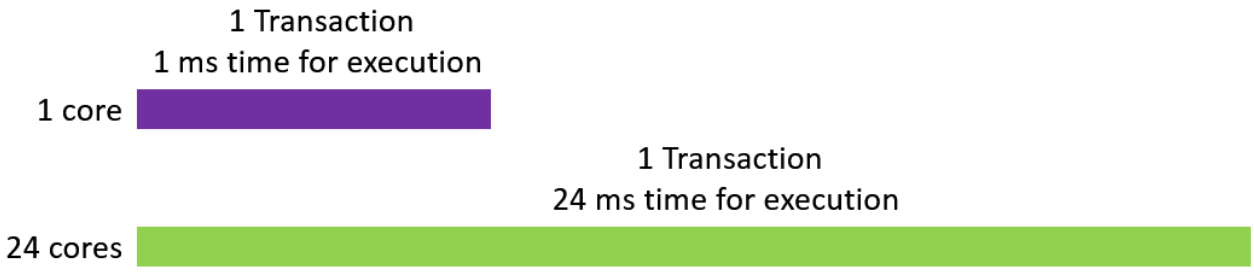
AppMon is a complex and powerful software solution which performs a huge amount of complex computation in the background. Remember, 24/7 every single transaction of your environment! Every sports-car needs horsepower and the right fuel to perform best, doesn't it?

Example 1

Here a simple example to explain better what AppMon is actually doing in the background:

Let's assume that the server has a load of 1000 transactions/sec. With 1 core the server has 1ms to complete all the required work for 1 transaction (1000ms x 1core/1000 transactions = 1ms/tx). In the case that the box has 24 cores, the server has 24ms to complete 1 transaction (1000ms x 24cores/1000 transactions = 24ms/tx). Differently formulated this could also mean **1 core = 1 transaction/ms or 24 cores = 24 transactions/ms**.

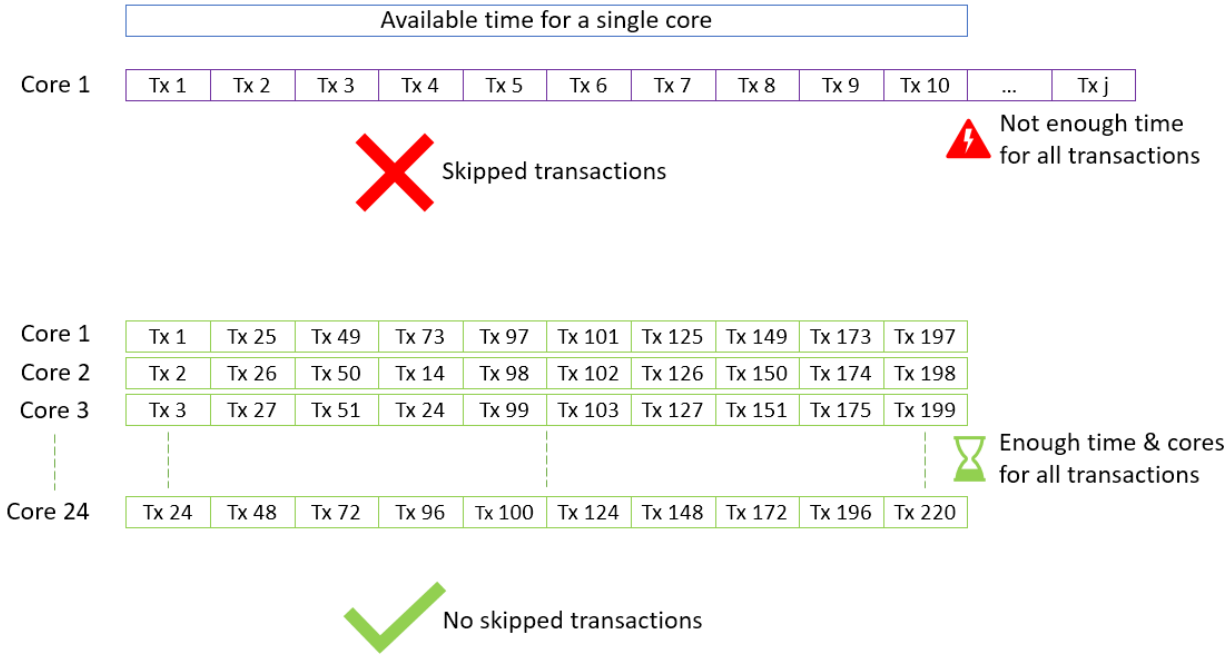
Formula:
1000ms x number of cores / transactions/second = max. time for transaction



Frankly, 24 transactions/ms sounds fairly good. But the situation can change immediately under a huge load.

Example 2

Let's look at another very simple example: the server has a total load of 50 transactions per second. This is really low, but this should not spoil the example. Every transaction needs 1ms to execute. With a total of 50 transactions we need a minimum of 50ms to perform all the computation. But a single core can only handle 10 transactions within this timeframe. The server is no longer able to compute all required steps to finalize a transaction and therefore the server starts to skip transactions. In the case of 24 cores, we can distribute the load over all available cores and as a result we gain much higher throughput. No more skipped transactions.



Example 3

Here is an example with AppMon running on a system with one physical core. In this example, we only focus on a few thread pools (out of 300+ possible "ready" threads) that use heavy amounts of CPU resources. In addition, to keep it simple, let's assume the scheduler allots time in figures of 50ms, and each thread has the same priority. Also, let's ignore the OS threads in this example as well.

1 Real-Time Analysis Thread(R) / 1 Correlation Thread(C) / 1 Session Writer (SW)

1 core

Time(ms)	0	50	100	150	200	250	300	350	400	450	500
CPU:1	R1	C1	SW	R1	R2	SW	C1	C2	R1	C1	SW

In the hypothetical example above, you can see that there are times were certain threads are idle waiting for the scheduler to assign them CPU time. Yes, the “wait time” is exaggerated, however we do have 3 threads, and one physical core, so there will be idle time for threads since CPU resources are limited.

And here is an AppMon server running on a system with 24 physical cores. Again, we only focus on a few thread pools (Out of 300+ possible “ready” threads) that use heavy amounts of CPU resources. In addition, to keep it simple, let’s assume the scheduler allots time in figures of 50ms, and each thread has the same priority. Also, let’s ignore the OS threads in this example as well.

12 Real-Time Analysis Threads(R) / 12 Correlation Threads(C)/ 3 Session Writer (SW)

24 cores

Time(ms)	0	50	100	150	200	250	300	350	400	450	500
CPU:1	R1	R11	SW2	R4	R2	SW1	C4	C2	R8	C9	C2
CPU:2	R4	C9	C7	SW3	R7	C4	R5	C10	R3	SW1	R9
...	..	..	..	..	..	..	..	..	..	..	..
...	..	..	..	..	..	..	..	..	..	..	..
CPU:23	SW2	C3	R3	R6	R12	C1	C8	SW3	C11	R12	C12
CPU:24	R3	C5	R8	R10	R7	C2	R1	C5	R11	C1	SW2

In the hypothetical example above, you can see two positive things happened.

- Our overall throughput increased since more threads can be assigned to a CPU in parallel.
- Time waiting for the CPU has decreased greatly. The only wait time that we can account for is programmatic wait time, or “parking”/“blocked” time due to locking.

Now let’s see what the server has to accomplish for every single transaction. Here a brief overview of all necessary components & computations:

1. Correlation & UEM correlation (if enabled)
    - Transaction stitching (find the correct paths for a node)
    - Garbage collection
    - Time calculation (elapsed time, duration, etc.)
    - Correlate Auto Sensor data
    - Calculate APIs (Web request, JDBC, etc.)

2. Postprocessing
    - PurePath categories
    - Application detection
    - …
3. Real time analysis
    - Measures: Computation, booking, aggregation, etc.
    - Business transactions: Computation, splittings, RegEx, etc.
    - Baselining
    - Error detection
4. PureLytics
    - Session storage lead
    - PWH access
    - Analyzers
    - Aggregation
    - …
5. Storage
    - Serialization
    - Disk write
6. Incident center
    - Timer tasks
    - Alert notifier
7. Communication
    - Collector & agent communication (Compress/Decompress, Encrypt/Decrypt)
    - Client(s)
8. Web servers
    - Service pages
    - WebUI
    - REST interfaces
    - …
9. Frontend
    - PWH access
    - Storage access
    - Triggered analyzers
    - Aggregation

0. Data export
    - Business transactions export
    - PureLytics stream
    - Tasks & Jobs
    - Clean up task
    - Reports
    - …

Please keep in mind, that the OS threads and the JVM G1 garbage collector threads (for Frontend and Backend) are also needed and are not covered here. Those threads scale with CPU cores too. Combining all threads from the Frontend, Backend, and G1 GC(for both), it is possible to have at least 300 or so threads running. In other words, a lot of fancy stuff is happening under the hood which you are not aware of.

I upgraded to the next size and you want me to add more cores. This doesn't make sense for me, because my current CPU is not fully utilized?

Basically, our requirements for CPU cores are threading related, or to be more precise concurrent threading execution of systems. The product is designed to use available resources as best as it can. Therefore, horsepower in terms of CPU cores is mandatory to guarantee the throughput. The following list shows the important parts of the product which requires cores:

- Sometimes server has to work under peak load and not only in an average scenario. This ensures to have enough headroom and resources for high load scenarios. See Collect Sizing data for more details.
- As stated in the previous section, AppMon has a lot of different components which require threads and resources. To optimize the overall throughput, a single component is not allowed to consume all CPU cores. If a component like the RTA or correlation is facing a resource limit, it cannot grab more cores to get rid of this bottleneck.

Why is the CPU clock speed also important?

First of all, more clock speed means yhat you can compute more transactions quicker. E.g. A transaction on a 2GHz CPU requires 10ms to be completed, it then takes ~6ms for a 2.6GHz CPU to complete it that same transaction.

Secondly, the product has of course a lot of synchronization points. In this case a faster CPU can dramatically reduce the execution time in a critical region. This leads to way less wait/sync times for other blocking threads.

Bare metals vs. virtualization

The above mentioned examples show the product running on a physical machine. Our AppMon product can also run on virtual machines as well. However, with virtual machines you can tweak settings relating to CPU's, Memory, Disks, and these resources can become "over-committed" and can be shared with other VM's. Even though over-committing resources is usually fine for most software (assuming cpu, memory, and disk metrics are healthy) our AppMon product requires dedicated resources simply because the product was designed to be as close to real-time as possible. In the examples above, we only touched on the systems that utilize the CPU the most. There are several other tasks that the AppMon Frontend process and Backend process must complete within milliseconds. Virtualization requirements & best practices can be found at the Virtualization requirements and best practices page.

Conclusion

To ensure a stable and well performing AppMon Server for certain sizes, our recommendations are important to follow. Otherwise the AppMon Server cannot perform under all workloads and scenarios. In addition, when running AppMon in a virtual machine, it is critical to dedicate virtual resources to the VM running the AppMon Server for optimal performance.

AppMon basics

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Architecture

PurePath explained

Features and capabilities

Editions

Watch and learn

Platform overview

Exploring core components

What is a PurePath?

This page provides links for learning the basic architecture, components, features, and capabilities of AppMon.

Learn more about

Architecture
PurePath technology
Main features and capabilities

Deployment Guide

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Installation overview

Install the AppMon Server

Install collectors

Install Memory Analysis Server

Install agents

Deployment steps

Additional deployment best practices

Assess results

Collect sizing data

Use the Deployment Sizing calculator

Verify results

The successful use and availability of your AppMon environment is the result of a well-planned and executed deployment. To achieve this, it is essential to accurately plan for the sizing needs of your environment and consider all other key deployment factors. This guide gives you the information and tools necessary to correctly size and deploy your AppMon environment.

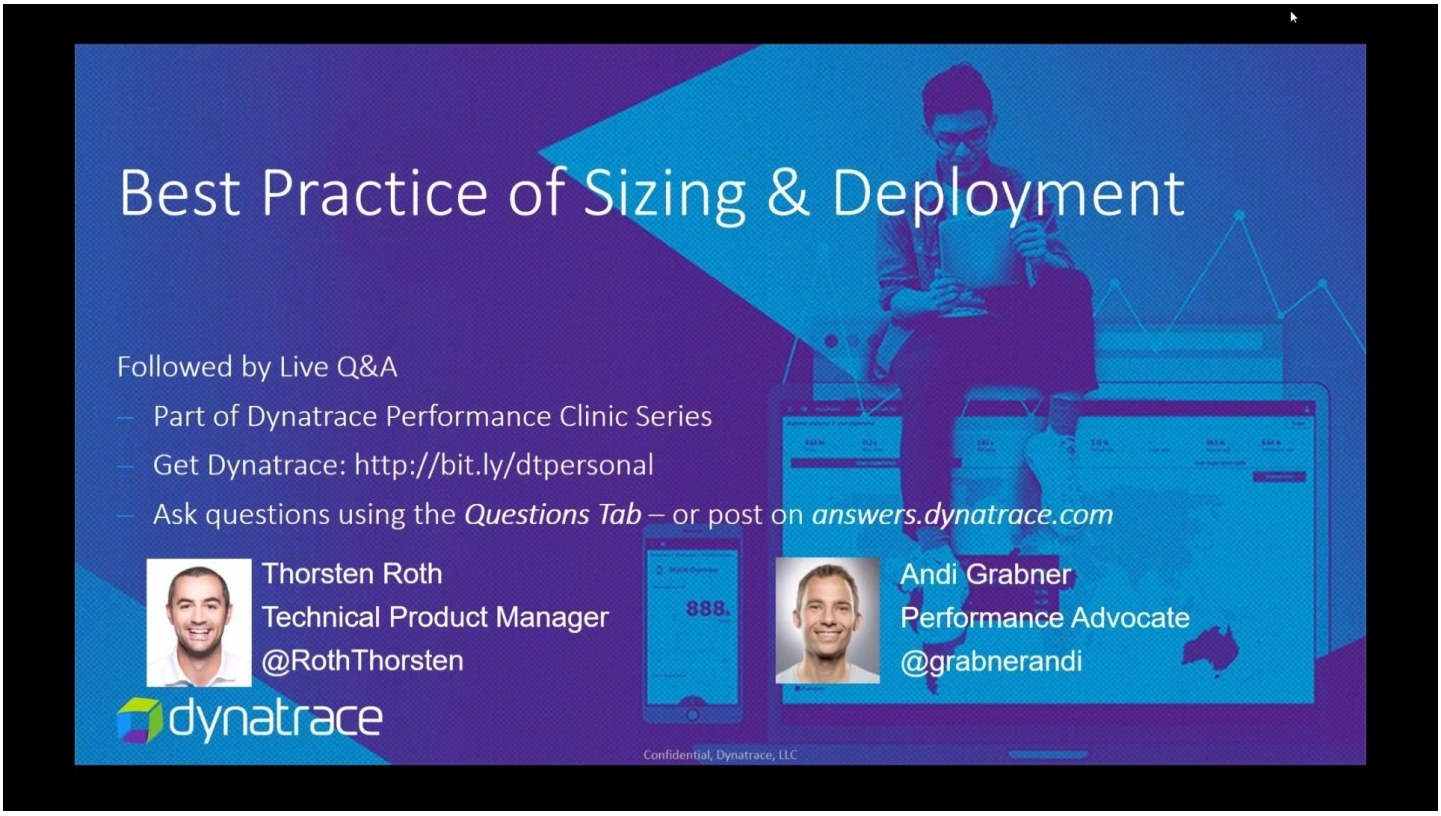
Deployment planning is a team effort that requires an iterative approach to ensure accurate results. This effort may take several days and typically requires working with Customer Support to set up a Proof of Concept (POC) engagement to accurately assess your deployment requirements and obtain relevant sizing data.

The AppMon deployment planning process consists of the following steps:

- Collect sizing data
- Use the Deployment Sizing Calculator
- Assess results
- Verify results
- Additional deployment best practices

If any questions or issues arise during the planning process, contact the Customer Support team for guidance.

This video shows details for Deployment and Sizing Best Practices for Dynatrace AppMon & UEM:



Collect sizing data

Looking for the 6.5 version of this page? Open in 6.5 documentation

Deployment steps

Use the Deployment Sizing calculator

Assess results

Verify results

Related pages

Set up SSL communication

Collect sizing data

The first step in planning an AppMon deployment is to collect the relevant sizing data required to accurately size your deployment. Although obtaining precise sizing data can be challenging, this step should be considered **essential** and not overlooked. The more accurate your sizing data is, the more likely your deployment successfully supports the current sizing requirements of your environment and scales to meet any future requirements.

To determine the data required to correctly size your AppMon deployment, answer the questions below. The answers to these questions provide the necessary input to enter in the Deployment Sizing Calculator. For help in answering a question, click the question to view information that explains the important considerations and details about the question.

1. How many Agents?

Determine the number of agents

Agents collect data and frequently send this data to the AppMon Collector, which in turn passes the data to the AppMon Server. More Agents means more data and more effect on your sizing requirements.

Determine the number of Agents you plan to instrument for:

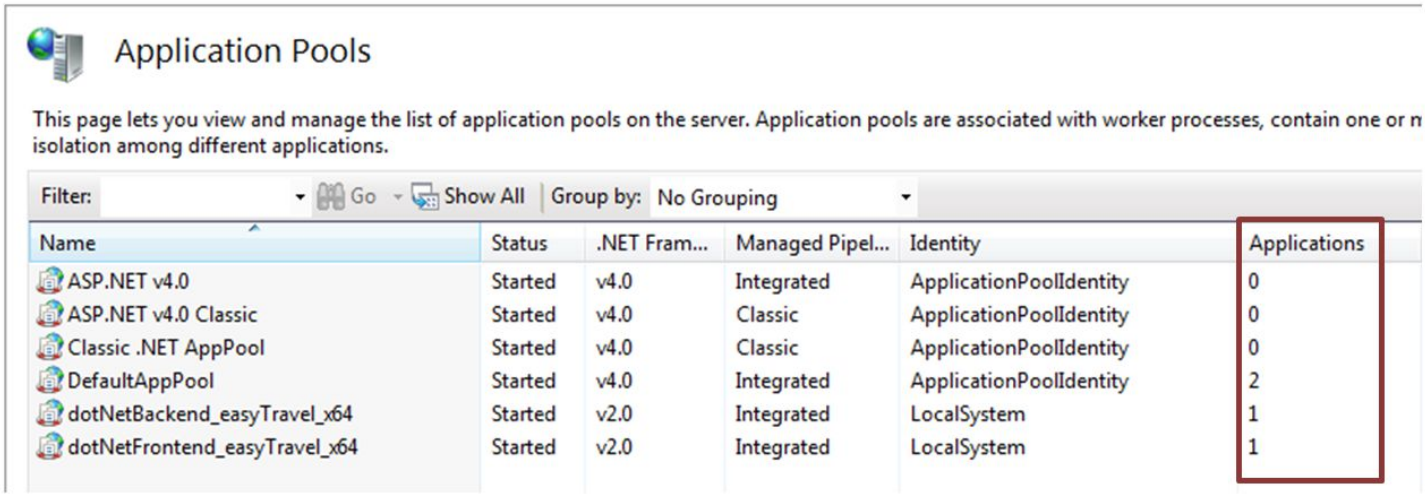
- Java virtual machines
- .NET Agents
- .NET servers
- PHP processes
- Web server processes
- Message Broker processes (execution groups)
- Native processes
- Host Monitoring Agents (servers)
- NoSQL processes
- CICS regions

Special considerations for .NET Agents

.NET Windows OS instances (WOSI) can support multiple Agents, but .NET is licensed per Windows OS instance (WOSI) and **not per Agent**. As a result, **do not use** the number of WOSI licenses as an accurate count of the number of .NET Agents.

To count the number of .NET Agents, add together the *total number of IIS w3wp worker processes*. This number will be equal to or greater than the number of app pools.

If you can not provide this number, count the total number of ASP.NET applications hosted in the w3wp processes (=webapps, app domains). The number is equal to or greater than the number of w3wp processes. ASP.NET webapps show up in IIS Manager (Applications column in the Application Pools page). Your system administrators should know how many ASP.NET webapps are deployed.



ASP.NET webapps displayed in IIS Manager

- Total number of non-IIS .NET instances. This is the sum of:
  - .NET-based Windows services that you want to instrument
  - .NET-based standalone executables, which could be user programs or worker processes of non-IIS .NET servers such as Cassini.

See http://www.iis.net/learn/get-started/getting-started-with-iis/getting-started-with-appcmdexe for more help to determine the number of required .NET Agents.

You must also determine the total number of app pools across all servers as part of this process.

> **Customer deployment - lessons learned**
>
> An AppMon customer purchased 20 .NET Agents. Filling in the number 20 in the Deployment Sizing Calculator created a recommendation for a small to medium deployment. However, when deploying this environment, the AppMon Server was completely undersized.
>
> What happened? Every one of the 20 .NET Windows OS instances (WOSIs) consisted of approximately 70 Agents. In the end, it was not a small environment but an xlarge environment with more than 1,400 Agents. Instead of deploying one Collector, 30 additional Collectors had to be deployed. Obtaining precise sizing data can make a very significant difference in a deployment.

Special considerations for web server processes

For Apache and IBM HTTP Servers, you can simply count the number of instrumented processes, which is identical to the number of licenses you need. Microsoft IIS Web Servers, however, require only one license per Windows OS (similar to .NET licensing), meaning that you need to count the total number of worker processes instead.

2. How many transactions per day?

Collect PurePath numbers

Estimate the number of transactions (this is the number of server-side requests) that are handled by the data center during one day. Normally, operators have a good understanding of how many transactions are produced. If you don't have this number, you can also estimate it with a POC in which you can see the current transactions per seconds in the Client, and then compute the daily transactions.

In addition, you may need to estimate the number of nodes per PurePath (or transaction). The number of nodes may vary. In some environments, the number is relatively low (about 50), but our experience leads us to recommend higher estimates (200 to 300). You can empirically derive the number of nodes with self-monitoring during a POC.

UEM considerations

If you use UEM, review the static resources.

Are static resources delivered by the web server? If yes, this leads to a higher number of transactions with only one node. A typical web request contains some 40 static resources, 80 percent of which are cached. As a result, the number of transactions per second is increased, but the average number of nodes per PurePath is lowered due to the caching. Within a UEM environment, the average **Nodes per PurePath** are estimated with the following formula:

(Transactions per Day * average Nodes per PurePath + Static Resources requested per Day) / (Transactions per Day + Static Resources requested per Day).

3. How long are you required to keep data?

Session storage

The number of transactions influences the Session Storage size. Session Storage is a data area controlled by the AppMon Server used to store all PurePath information in it. Consequently, the duration of how long you want to keep your PurePath sessions stored on the disk influence the Session Storage size.

In addition, the Session Storage size is influenced by the use of auto sensors, which increases the size requirements, and by UEM visits or PurePath percentages, which decrease the size requirements.

Sizing the Session Storage to less than 2 TB is recommended. Although it is technically possible to exceed this limit into the tens of terabytes, analyzing this amount of data is not a practical consideration.

Performance Warehouse

The Performance Warehouse size is primarily influenced by the amount of time required to retain high-resolution data. The medium- and high-resolution periods play a rather insignificant role.

More important than the Performance Warehouse size, which is typically quite reasonable, is the number of measurements it can handle. Too many measurements overload the database server. In this case, a reduction of cyclic measures or business transactions should be considered.

4. How many nodes?

System boundaries

In a production environment, more than just scalability must be considered. Even if an AppMon Server could accommodate more Agents (more load), the Server always has physical limits. To avoid reaching these limits, decrease the number of PurePath nodes. Usually there is not any reason to have an average node number of 5,000 in a production environment. A value of 10 or 50 is generally good for production. A successful POC is always a good source to get proper values. To reduce the average node length, you can switch off several sensors or use aggregation, such as for JDBC.

5. Do you use UEM?

User experience management (UEM) delivered with static resources has additional impact on the AppMon Server. Therefore, it's assumed that every transaction has an extra load and generates more total PurePaths. Learn more about UEM.

6. What is your Peak Factor?

It is crucial to figure out when your applications incur the heaviest traffic and transactions. For example, web shops need to prepare in advance for peak load during the holiday shopping. This means that the AppMon Server has to handle the entire load during the heavy load period calculated with the right Peak Factor.

7. What is your load distribution?

Our default value for load distribution is 24 hours, but this time period is dependent on your application or web shop and how the load is distributed. For example, a web shop that is available only in Europe will have a load distribution of about 8 hours. In this time frame, most of the business transactions occur. This means that the total number of transactions must be divided by 8 and not 24 hours, which significantly changes the number of total transactions per day.

Next: Use the Deployment Sizing Calculator

Use the Deployment Sizing calculator

Looking for the 6.5 version of this page? Open in 6.5 documentation

Downloads

Deployment sizing calculator v.1.8.6

Deployment steps

Collect sizing data

Assess results

Verify results

Note

Use the latest Deployment Sizing calculator spreadsheet, which is version **1.8.x**. Using a version other than the latest may result in a diminished experience.

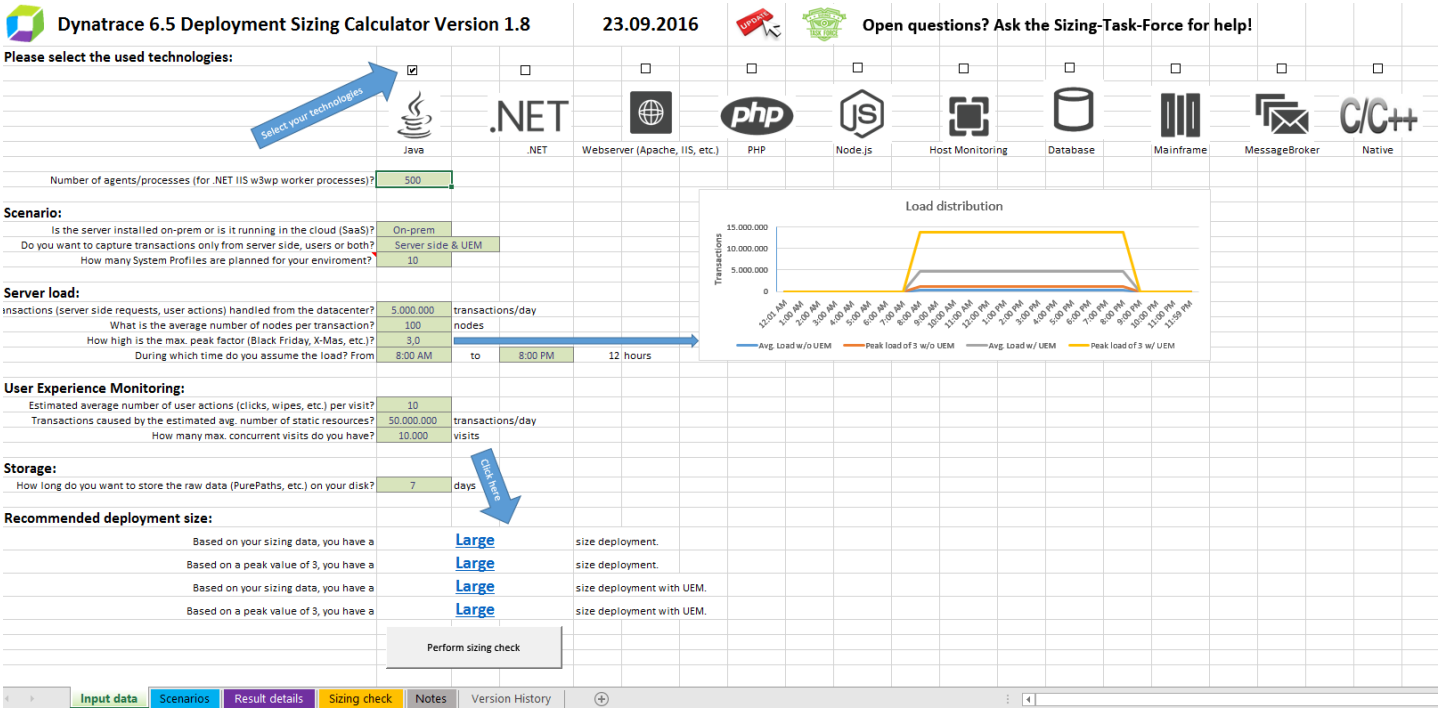
MAC users: You must use Microsoft Office version later than **15.19.1**, otherwise some macros will not work.

What is the Deployment Sizing calculator?

The Deployment Sizing calculator lets you calculate the recommended deployment size and categorize the result as a small, medium, large, or extra-large (XLarge) deployment. You provide the sizing, such as the number of JVMs, web servers, and daily transactions. The spreadsheet also calculates the recommended bandwidth, Session Storage, and Performance Warehouse sizing.

The data provides the necessary guidance to make an accurate sizing decision. For more clarity, assess the results of the Deployment Sizing Calculator against the component sizing guidelines and customer examples provided in Assess Results.

Deployment Sizing calculator



Deployment Sizing calculator

Calculate sizing requirements

To use the Deployment Sizing calculator, follow these steps:

1. Click here to save then open a copy of the Deployment Sizing calculator spreadsheet.

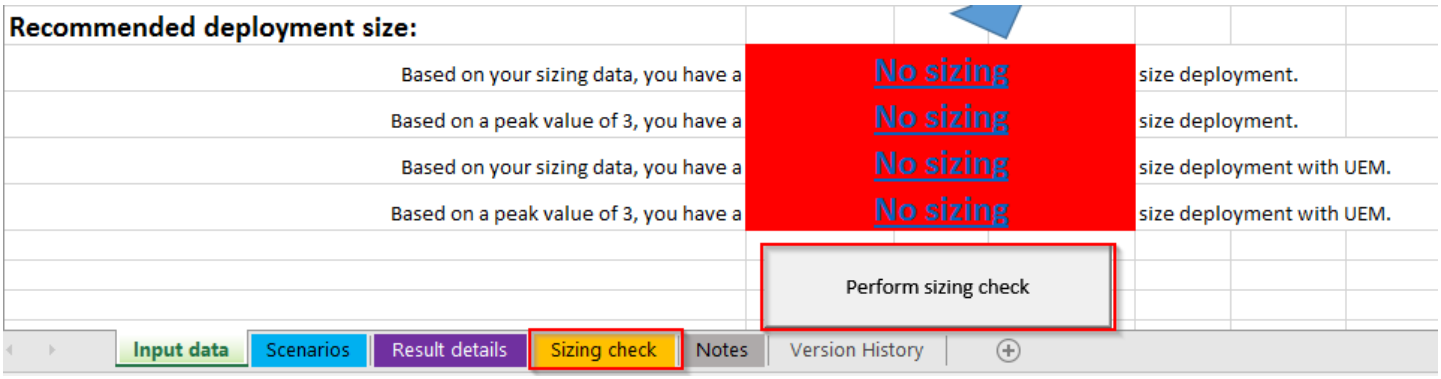
> **Tip**
>
> The spreadsheet may issue a warning that macros have been disabled. Click **Enable Content** before entering the requested sizing data.

2. Select the technologies to be used in deployment, then enter the requested sizing data for your environment in the **Input Data** worksheet. When entering data, the recommended deployment size and related sizing details automatically calculate on the spreadsheet.

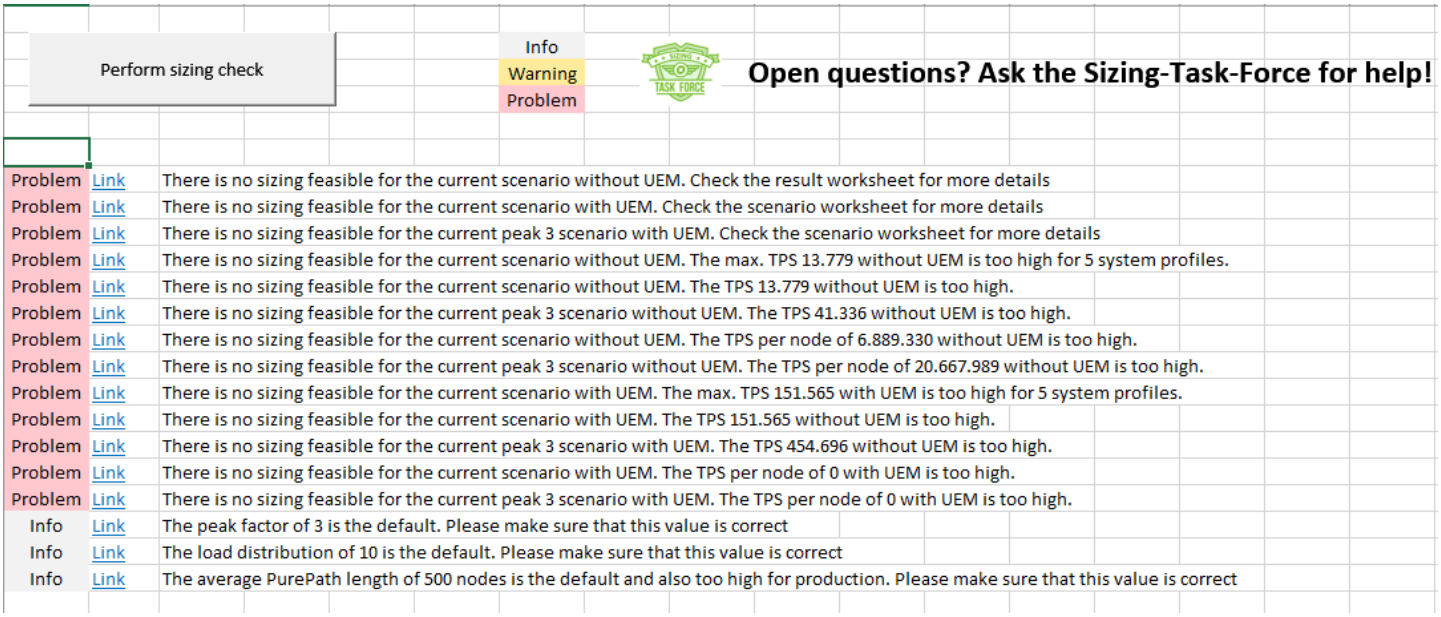
3. Under **Recommended deployment size**, click a deployment size link (for example, Small, Medium, Large) to go to the applicable sizing page in the help for details about your deployment sizing recommendation. See Assess Results for more information.

Resolve No Sizing Issues

In certain scenarios, sizing calculations may not be feasible. This may happen if you enter too much load or too many agents. If this occurs, **No sizing** appears in a red background in the **Recommended deployment size** cell for the applicable deployment category.



Version 1.8 and later of the Deployment Sizing calculator lets you check and solve sizing issues. Click **Perform sizing check** either under the **Recommended deployment size section** or in the **Sizing check worksheet** to see potential sizing issues. Problems are tagged with red. Click the related link to see where the issue occurs in the **Scenarios** worksheet.



Perform sizing check

Info
Warning
Problem

Open questions? Ask the Sizing-Task-Force for help!

Problem	Link	There is no sizing feasible for the current scenario without UEM. Check the result worksheet for more details
Problem	Link	There is no sizing feasible for the current scenario with UEM. Check the scenario worksheet for more details
Problem	Link	There is no sizing feasible for the current peak 3 scenario with UEM. Check the scenario worksheet for more details
Problem	Link	There is no sizing feasible for the current scenario without UEM. The max. TPS 13.779 without UEM is too high for 5 system profiles.
Problem	Link	There is no sizing feasible for the current scenario without UEM. The TPS 13.779 without UEM is too high.
Problem	Link	There is no sizing feasible for the current peak 3 scenario without UEM. The TPS 41.336 without UEM is too high.
Problem	Link	There is no sizing feasible for the current scenario without UEM. The TPS per node of 6.889.330 without UEM is too high.
Problem	Link	There is no sizing feasible for the current peak 3 scenario without UEM. The TPS per node of 20.667.989 without UEM is too high.
Problem	Link	There is no sizing feasible for the current scenario with UEM. The max. TPS 151.565 with UEM is too high for 5 system profiles.
Problem	Link	There is no sizing feasible for the current scenario with UEM. The TPS 151.565 without UEM is too high.
Problem	Link	There is no sizing feasible for the current peak 3 scenario with UEM. The TPS 454.696 without UEM is too high.
Problem	Link	There is no sizing feasible for the current scenario with UEM. The TPS per node of 0 with UEM is too high.
Problem	Link	There is no sizing feasible for the current peak 3 scenario with UEM. The TPS per node of 0 with UEM is too high.
Info	Link	The peak factor of 3 is the default. Please make sure that this value is correct
Info	Link	The load distribution of 10 is the default. Please make sure that this value is correct
Info	Link	The average PurePath length of 500 nodes is the default and also too high for production. Please make sure that this value is correct

The **Scenarios** worksheet covers all important KPIs and input data results. A red cell background color indicates that you should check the accuracy of your input values.

Input sizing						
Server side & UEM			w/o UEM	w/ UEM	w/o UEM	w/ UEM
		tps	13.779	151.565	No sizing	No sizing
		peak tps	41.336	454.696	No sizing	No sizing
		agents	100	100	Medium	Medium
		total agents	100	100	Medium	Medium
		system profiles	100	100	XLarge	XLarge
		max. tps for 100 profiles	5.267	5.267	No sizing	No sizing
		measures	16.000	16.000	Medium	Medium
		nps	6.889.330	6.454.154	No sizing	No sizing
		peak nps	20.667.989	19.362.461	No sizing	No sizing
		collectors	1	1	Demo	Demo
		concurrent visits		10.000		Medium
	Scenario				No sizing	No sizing
	Scenario Peak				No sizing	No sizing

Potential input sizing issues include:

- **Transactions per Second (tps)**: There are too many transactions for the assumed average node length. Check and adjust the server load.
- **Peak transactions per Second (peak tps)**: Transactions caused by the peak are too high.
- **Total agents**: You reached or exceeded the maximum allowed connected agents per server.
- **Max. tps for XY system profiles**: System Profiles require resources. Avoid having too many if possible. The more System Profiles, the greater impact on the server throughput.

- **Measures**: The Server has to handle too many cyclic measures generated by Agents.
- **Nodes per Second (nps)**: The limit of maximum nodes per second is reached or exceeded.
- **Peak nodes per Second (peak nps)**: The limit of maximum nodes per second during peak is reached or exceeded.
- **Collectors**: Your sizing reached or exceeded the maximum number of Collectors (60) for a single Server.
- **Concurrent visits**: UEM only. The maximum number of concurrent visits is reached or exceeded.

Ensure sizing accuracy

Use the sizing computations in the **Result Details** worksheet to ensure sizing estimates are in line with actual sizing data, and get to additional help on network bandwidth sizing and disk I/O guidelines.

Next : Assess Results

Assess results

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Default deployment sizes

Server and Performance Warehouse best practices

Collector best practices

Memory Analysis Server best practices

Network bandwidth best practices

Disk IO determination best practices

Customer environment examples

Deployment steps

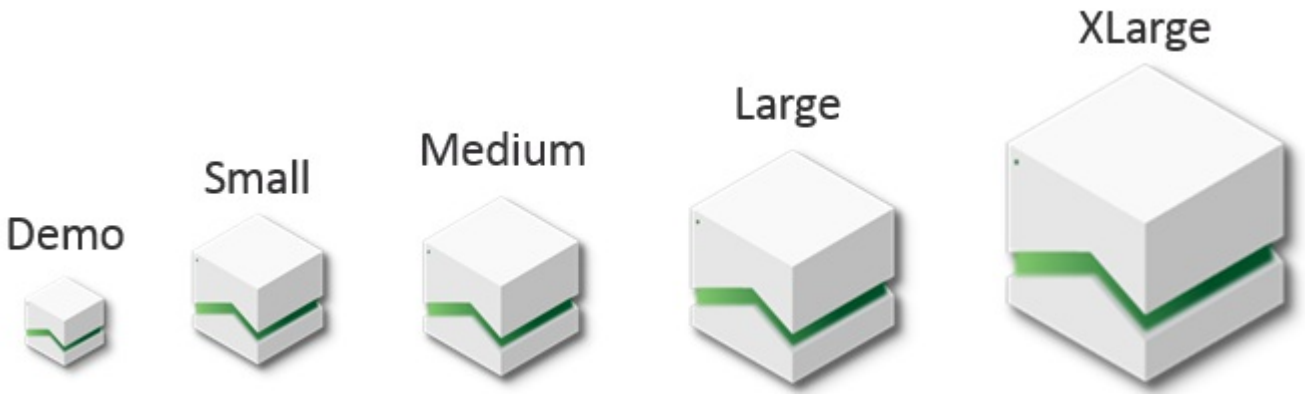
Collect sizing data

Use the Deployment Sizing calculator

Verify results

After entering your sizing data in the Deployment Sizing calculator spreadsheet, an appropriate deployment size (small, medium, large, etc.) is recommended and listed in the spreadsheet for your environment. The recommended deployment sizes specify the number of AppMon components required to support your environment and the system requirements (such as memory and processor type) for the machines on which the AppMon components are installed.

AppMon deployment sizes



To view the system requirements for a specific AppMon deployment size, refer to the deployment size page listed below for your environment:

- Demo (POC) environment
- Small environment
- Medium environment
- Large environment
- XLarge environment

> **Important**
>
> For small, medium, and large deployments, the results generated by the Deployment Sizing calculator provide an accurate assessment of the deployment sizing considerations.
> For **Xlarge deployments**, however, there are typically additional considerations that affect the ultimate sizing calculations. You should double-check your sizing input data for accuracy regardless of the deployment size, but this is especially true for an XLarge deployment. In addition, see the XLarge customer examples for a comparison before contacting Customer Support to finalize the sizing for XLarge deployments.

In addition to the deployment sizing pages, review to the following deployment guidelines for AppMon components and related information such as the customer environment examples:

- Server and Performance Warehouse best practices
- Collector best practices
- Memory Analysis Server best practices
- Network bandwidth configuration guidelines
- Disk IO determination best practices
- Customer environment examples

Next: Verify results

Default deployment sizes

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Demo (POC) environment

Small environment

Medium environment

Large environment

XLarge environment

The AppMon Server (backend process / Windows service / *NIX daemon) is only supported on operating systems that support Java 8. Only the included Java Runtime Edition is supported.

- **Windows x64**: 10/8.1/8/7/Vista or Server 2008/2012/2016 with latest service pack; installer (.msi) does not need a pre-installed JRE; package contains a private Java Runtime Edition v8.
- **Linux x64**: With a recent kernel version 2.6 or higher.
  Installation requires a pre-installed Java Runtime Edition v8.
- **LDAP**: For AppMon user authentication.
  - Active Directory
  - openLDAP

Hardware requirements

See the Deployment Guide for detailed hardware recommendations specific to your load.

Deployment size[1]	Physical cores required[2]	Recommended physical memory[3]	Also available memory options	Recommended disk space	A
Demo 5 agents | 150 transactions/sec	2	2GB	4GB, 8GB	1.5GB + 100GB Continuous session storage/week	N
Small 10 agents | 300 transactions/sec	4	4GB	8GB, 16GB	3GB + 200GB Continuous session storage/week	Y

Medium 250 agents | 1,500 transactions/sec	8	8GB	16GB, 32GB	5GB + 1TB Continuous session storage/week	Y
Large 750 agents | 3,500 transactions/sec	16	16GB	32GB, 64GB	10GB + 2TB Continuous session storage/week	Y
XLarge **32GB** | 1600 agents | 5,000 transactions/sec	24	32GB	-	20GB + >2TB Continuous session storage/week	N
64GB | 1800 agents | 5,800 transactions/sec	24	64GB	-	20GB + >2TB Continuous session storage/week	N
96GB | 2000 agents | 6,500 transactions/sec	24	96GB	-	20GB + >2TB Continuous session storage/week	N

[1] The maximum agent numbers consider a typical combination of different agents types (Java, .NET, host, WebServer, PHP, etc.).
In special circumstances, significant more agents are supported (e. g. mainly host agents). Please contact support for guidance and confirmation.

[2] No virtual cores (Hyper-Threading), for Medium and above a clock speed of 2.6 GHz or higher is required.

[3] Required memory for the AppMon Server and operating system. If there are other processes running, then additional heap is required.

> **Note**
>
> To improve performance, AppMon always uses the maximum available CPU and memory resources.
> If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

VM support

All deployment sizes listed above may be deployed on a virtual machine. However, you must consider the following:

- No over-commitment of resources (CPU, memory, network, disk) and no virtual-only resources that are not backed by dedicated physical resources.
- Large and XLarge only. VM is equivalent to a physical box in all performance metrics. For more details see Guidelines for running AppMon in VMWare.
- XLarge only. Database partitioning is mandatory.

Why are so many cores required?

AppMon performs a large amount of complex background computations. Therefore, horsepower in terms of CPU cores is mandatory to guarantee the throughput.

The following list shows the important parts of the product which requires cores:

- Correlation engine
- Real time analysis
- Communication (Clients, Collectors, etc.)
- Performance Warehouse
- Session storage
- Analysis cache
- Data exports (Synthetic, PureLytics, etc.)
- Web servers
- Job center for tasks and monitors
- Incident center
- Garbage collection
- Self monitoring and checks
- Operating system

Intel vs. AMD

The CPU is an integral part of your computer. To get the desired results, you are going to need a quality CPU. We did a wide variety of tests and the result shows, that AMD are not able to keep up with Intel CPUs. Therefore, and that's why – for now, at least – Intel's chips bring objectively better performance and is our suggested CPU vendor.

Heap breakdown for frontend and backend process

Deployment Size	Frontend heap GB	Backend heap GB	Total heap

Demo	0.5	1	1.5
Small	0.75	2.25	3
Medium	2	4	6
Large	4	8	12
XLarge	8	14	22
XLarge64	16	32	48
XLarge96	32	48	80

Demo (POC) environment

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Default deployment sizes

Small environment

Medium environment

Large environment

XLarge environment

Sizing questions or issues

Customer environment examples

Important

The Demo deployment size is for a POC only and should not be used in production deployments. Use the Small deployment size instead.

Overview

Agents	Collectors	Measures per second	Transactions per second	Avg. nodes length
5	internal	12,000	120	100

Sizing suggestions

Minimum process requirements

Server process
1.5 GB RAM 2 cores

Minimum hardware requirements

Machine/OS	Installed components[1]
x64 Linux or Windows 2 cores 2 GB RAM[2]	1 AppMon Server (1.5 GB heap) includes AppMon Collector internal includes InMemory database

[1] All components/processes are running on a single machine

[2] Also available memory options are 4GB and 8GB.

Note

To improve performance, AppMon always uses the maximum available CPU and memory resources.
If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

Small environment

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Default deployment sizes

Demo (POC) environment

Medium environment

Large environment

XLarge environment

Sizing questions or issues

Customer environment examples

Overview

Agents	Collectors	Measures per second	Transactions per second	Avg. nodes length
10	1	20,000	200	100

Sizing suggestions

Minimum process requirements

Server process	Collector process	Database process
3 GB RAM 4 cores	1 GB RAM 1 core	PostgreSQL 1 GB RAM 1 core

Minimum hardware requirements

Machine/OS	Installed components[1]
x64 Linux or Windows 4 cores 4 GB RAM[2]	1 AppMon Server (3 GB heap) 1 AppMon Collector (1 GB heap) 1 PostgreSQL database (1 GB heap)

[1] All components/processes are running on a single machine

[2] Also available memory options are 8GB and 16GB.

Note

To improve performance, AppMon always uses the maximum available CPU and memory resources.
If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

Medium environment

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Default deployment sizes

Demo (POC) environment

Small environment

Large environment

XLarge environment

Sizing questions or issues

Customer environment examples

Overview

Agents	Collectors	Measures per second	Transactions per second	Avg. nodes length
100	1	120,000	1,200	100

Sizing suggestions

Minimum process requirements

Server process	Collector process	Collector for monitoring Plugins process	Memory Analysis Server process
6 GB RAM 8 cores[1]	2 GB RAM 1 core	2 GB RAM 1 core	4 GB RAM 2 cores

[1] No virtual cores (Hyper-Threading) for Server size larger that Medium. Minimum 2.6 GHz required.

[2] PostgreSQL is supported, but Oracle, DB2, or SQL Server database is recommended (especially if you want to extend the environment later). For DB hardware requirements, consult the DB provider.

Minimum hardware requirements

Machine/OS	Installed components[3]
x64 Linux or Windows 8 cores 8 GB RAM[4]	1 AppMon Server (6 GB heap) 2 AppMon Collector (2 GB heap each) 1 AppMon Collector for monitoring Plugins (2 GB heap) 1 AppMon Memory Analysis Server (4 GB heap) 1 PostgreSQL database (1 GB heap)

[3] All components/processes are running on a single machine.

[4] Also available memory options are 16GB and 32 GB.

Note

To improve performance, AppMon always uses the maximum available CPU and memory resources.
If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

Large environment

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Default deployment sizes

Demo (POC) environment

Small environment

Medium environment

XLarge environment

Sizing questions or issues

Customer environment examples

Overview

Agents	Collectors	Measures per second	Transactions per second	Avg. nodes length
750	6	300,000	3,000	100

Sizing suggestions

Minimum process requirements

Server process	Collector process	Memory Analysis Server process	Database process
16 GB RAM 16 cores[1]	4 GB RAM 2 cores	16 GB RAM 2 cores	Oracle[2] 4 GB RAM 4 cores

[1]No virtual cores (Hyper-Threading) for Server size larger that Medium. Minimum 2.6 GHz required.

[2] Oracle, DB2 or SQL Server database is highly recommended especially if you want to extend the environment later. For DB hardware requirements, check with the DB provider.

Minimum hardware requirements

Machine/OS	Installed components
x64 Linux or Windows 16 cores 16 GB RAM[3]	1 AppMon Server (16 GB heap) 1 AppMon Memory Analysis Server (16 GB heap)
x64 Linux or Windows 16 cores 16 GB RAM[3]	1 Oracle database (4GB heap)

[3] Also available memory options are 32GB and 64GB.

> **Note**
>
> To improve performance, AppMon always uses the maximum available CPU and memory resources.
> If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

Collector box with multiple instances

Hardware	Installed components
x64 Linux or Windows 12 cores 24 GB RAM	5 AppMon Collectors (4 GB heap each) 1 AppMon Collector for monitoring Plugins (1 GB heap)

Collector box with single instances -

Multiple boxes required for this option.

Hardware	Installed components
x64 Linux or Windows 2 cores 4 GB RAM	1 AppMon Collectors (4 GB heap)

Note

To ensure the optimal performance of the Collector, a fast hard disk is recommended. As a best practice, use a solid-state disk (SSD) or other fast hard disk (small HD such as 32 GB is sufficient).

XLarge environment

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Default deployment sizes

Demo (POC) environment

Small environment

Medium environment

Large environment

Sizing questions or issues

Customer environment examples

XLarge Environment? Contact us

Based on our experience, an XLarge deployment is easier to design with AppMon Support technicians. Please open a support ticket with the **following information** :

- Completed Deployment Sizing -> download here
- Information about the system architecture -> see examples

Failure to provide the requested information will result in unnecessary delays in addressing your support call.

Please be aware that the final sizing recommendation usually takes longer than 1 to 2 business days.

After determining that you have an XLarge environment, contact Customer Support by creating a support case.



Overview

Agents	Collectors	Measures per second	Transactions per second	Avg. nodes length
1600	4	400,000	4,500	100

Sizing suggestions

Minimum process requirements

Server process	Collector process	Memory Analysis server process	Database process
22 GB RAM 24 cores[1]	8 GB RAM 4 cores	16 GB RAM 4 cores	Oracle[2] 8 GB RAM 4 cores

[1] No virtual cores (Hyper-Threading) for Server size larger than Medium. Minimum 2.6 GHz required.

[2] Oracle, DB2 or SQL Server database is highly recommended especially if you want to extend the environment later. For DB hardware requirements, check with the DB provider.

Minimum hardware requirements

Machine/OS	Installed components
x64 Linux or Windows 24 cores 32 GB RAM[3]	1 AppMon Server (22 GB heap)
x64 Linux or Windows 24 cores 32 GB RAM[3]	1 AppMon Memory Analysis server (16 GB heap)
x64 Linux or Windows 24 cores 32 GB RAM[3]	1 Oracle database (8GB heap)

[3] Also available memory options are 64GB and 96GB. With them you'll get 5,000 and 5,500 transactions per second respectevely.

Note

To improve performance, AppMon always uses the maximum available CPU and memory resources.
If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

Collector box with multiple instances

Hardware	Installed components
x64 Linux or Windows 12 cores 24 GB RAM	2 AppMon Collectors (8 GB heap each) 1 AppMon Collector for monitoring Plugins (1 GB heap)

Collector box with single instances

Multiple boxes required for this option.

Hardware	Installed components
x64 Linux or Windows 4+ cores 12 GB RAM	1 AppMon Collectors (8 GB heap)

Note

To ensure the optimal performance of the Collector, a fast hard disk is recommended. As a best practice, use a solid-state disk (SSD) or other fast hard disk (small HD such as 32 GB is sufficient).

Sizing questions or issues

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Default deployment sizes

Demo (POC) environment

Small environment

Medium environment

Large environment

XLarge environment

Customer environment examples

Do you have sizing questions or issues? contact us

If you have sizing questions or you are facing any issues, please contact us. We kindly ask you to open a support ticket with the following information:

- Completed Deployment Sizing -> download here
- Information about the system architecture -> see examples
- Sizing proposal -> click here to submit a proposal

Failure to provide the requested information will result in unnecessary delays in addressing your support call.

Please be aware that the final sizing recommendation usually takes longer than 1 to 2 business days.

Customer environment examples

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Small environment

Medium environment

Large environment

XLarge environment

Related pages

Default deployment sizes

Demo (POC) environment

Small environment

Medium environment

Large environment

XLarge environment

Sizing questions or issues

This page contains recommended sizing requirements and related information for typical customer environments.

- **Network** - Although proper hardware is one key factor for a smooth working environment, always consider the network of your environment. For more information, see the network bandwidth best practices.

- **RegEx** - Regular Expressions (RegEx) can be very expensive. Certain unintended and intended strings may cause RegEx to exhibit exponential behavior. RegEx is very useful, but developers need to understand how it works and to always be careful with it. For more details, see RegEx - Using Regular Expressions.

> **Note**
>
> In the following examples, the CPU core listings imply the number of physical cores and not virtual cores. Server sizes larger than Medium require a minimum 2.6 GHz.
> The suggested memory for the hardware where the server is running is a recommendation. For instance, in a XLarge scenario, 48GB is recommended, because caches and other running processes, and future proof for upcoming releases also need resources.

Small environment

Example

Environment overview

Environment	Agents	Data centers	Transactions	Measures	Peak	Comments
Production Java	5	1	17 million/day 197 tps	2,000		without UEM

Sizing suggestion

Minimal process requirements

Environment	Server process	Collector process	Database process	Comments
Production Java	3 GB RAM 4 cores 100 GB local file storage (depends on how long to keep the session store)	2 GB RAM 1 cores 30 GB storage	PostgreSQL 1 GB RAM 1 core 100 GB storage	

Minimal hardware requirements

Machine/OS	Installed components*

x64 Linux or Windows 8 cores 12 GB RAM	1 PostgreSQL database (1 GB heap) 1 AppMon Server (8 GB heap) 1 Small AppMon Collector (2 GB heap)

** All components/processes are running on a single machine.*

Minimal storage requirements

	Storage	Comments
Application and configuration	5 GB	
Performance Warehouse storage	5 GB	PostgreSQL database is sufficient
Session storage for 1 day	89 GB	
Collector application and configuration	30 GB	Including class cache

Minimal network requirements

	Bandwidth	Comments
Network Bandwidth	11 MB	Agent <-> Collector

Other considerations:

- The Memory Analysis Server should be installed to a separate machine if the memory snapshots processed are larger than the remaining physical RAM.
- A minimum one Collector per physical location (data center); 200 JVMs per collector process.
- Collectors must be very close to their connected Agents (same LAN).

Medium environment

Example

Environment overview

Environment	Agents	Data centers	Transactions	Measures	Peak	Comments
Production Java	80	1	17 million/day 410 tps	17,000	820 tps	without UEM

Sizing suggestion

Minimal process requirements

Environment	Server process	Collector process	Database process	Comments
Production Java	6 GB RAM 8 cores 100 GB local file storage (depends on how long to keep the session store)	4 GB RAM 2 cores 50 GB storage	PostgreSQL 2 GB RAM 2 core 100 GB storage	

Minimal hardware requirements

Machine/OS	Installed components*
x64 Linux or Windows 12 cores 16 GB RAM	1 PostgreSQL database (2 GB heap) 1 AppMon Server (6 GB heap) 1 Small AppMon Collector (2 GB heap)

* *All components/processes are running on a single machine.*

Minimal storage requirements

	Storage	Comments
Application and configuration	5 GB	
Performance Warehouse storage	40 GB	PostgreSQL seems basically sufficient. However, Oracle, DB2, or SQLServer database is recommended (especially if you want to extend the environment later). For DB hardware requirements, please consult the DB provider.
Session storage for 1 day	235 GB	
Collector application and configuration	50 GB	Including class cache

Minimal network requirements

	Bandwidth	Comments
Network bandwidth	28 MB	Agent <-> Collector

Other considerations:

- The Memory Analysis Server should be on a separate machine if the memory snapshots processed are larger than the remaining physical RAM.
- A minimum of one Medium Collector per physical location (data center); 200 JVMs per Collector process.
- Collectors must be very close to their connected Agents (same LAN).

Large environment

Example

Environment overview

Environment	Agents	Data centers	Transactions	Measures	Peak	Comments
Production Java	400	2	52 million/day 1,200 tps		2,400 tps	with UEM

Sizing suggestion

Minimal process requirements

Environment	Server process	Collector process	Memory Analysis Server process	Collector for monitoring plug
Production Java	12 GB RAM 16 cores 500 GB local file storage (depends on how long to keep the session store)	8 GB RAM 4 cores 50 GB storage	8 GB RAM 2 cores	1 GB RAM 2 cores

Minimal hardware requirements

Machine/OS	Installed components
x64 Linux or Windows 16 cores 32 GB RAM	1 AppMon Server (12 GB heap) 1 AppMon Memory Analysis Server (8 GB heap) More heap for analysis server, OS and to be future proof Analysis server is optional. If required, then more cores are needed.
x64 Linux or Windows 6 cores 8 GB RAM	1 Oracle database (4 GB heap)

x64 Linux or Windows 12 cores 24 GB RAM	2 Large AppMon Collectors (8 GB heap each) 1 AppMon Collector for monitoring plugins (1 GB heap)

Minimal storage requirements

	Storage	Comments
Application and configuration	5 GB	
Performance Warehouse storage	191 GB	Oracle, DB2 or MS SQL database is needed. For DB hardware requirements, please consult the DB provider.
Session storage for 1 day	349 GB	RAID 10 or SAN
Collector application and configuration	50 GB	Including class cache

Minimal network requirements

	Bandwidth	Comments
Network bandwidth	42 MB	Agent <-> Collector
Network bndwidth	42 MB	Collector <-> Server
Network bandwidth compressed	10.4 MB	Collector <-> Server

Other considerations:

- The Memory Analysis Server should be installed to a separate machine if the memory snapshots processed are larger than the remaining physical RAM.
- A minimum of one large Collector per physical location (data center); 400 JVMs per collector process
- Collectors must be very close to their connected Agents (same LAN)

XLarge environment

Example 1

Environment overview

Environment	Agents	Data Centers	Transactions	Measures	Peak	Comments
Production Java	1,000	8	10 million/hour 2,777 tps	200,000	3000 tps	100 nodes per transaction, without UEM

Sizing suggestion

Minimal process requirements

Environment	Server process	Collector process	Memory Analysis Server process	Database process	Comr
Production Java	22 GB RAM 24 cores >2 TB local file storage (depends on how long to keep the session store)	8 GB RAM 4 cores 50 GB storage	8 GB RAM 2 cores	Oracle 8 GB RAM 8 cores 1 TB storage	Sess stora depe how r days the s 3-4 T be su for or The load so SS (RAID and Perfo Ware datat partit are recor

Minimal hardware requirements

Machine/OS	Installed components
x64 Linux or Windows 24 cores 48 GB RAM	1 AppMon Server (30 GB heap) More heap for OS and to be future proof
x64 Linux or Windows 10 cores 24 GB RAM	1 AppMon Memory Analysis Server (8 GB heap) 1 Oracle database (8 GB heap)
x64 Linux or Windows 16 cores 32 GB RAM	3 Large AppMon Collectors (8 GB heap each)
x64 Linux or Windows 10 cores 24 GB RAM	2 Large AppMon Collectors (8 GB heap each)

Minimal storage requirements

	Storage	Comments
Application and configuration	5 GB	
Performance Warehouse storage	473 GB	Oracle database is needed. For DB hardware requirements, please consult the DB provider.
Session storage for 1 day	1,250 GB	RAID 10 or SAN
Collector application and configuration	50 GB	Including class cache

Minimal network requirements

	Bandwidth	Comments
Network bandwidth	150 MB	Agent <-> Collector
Network bandwidth	150 MB	Collector <-> Server
Network bandwidth compressed	40 MB	Collector <-> Server

Other considerations:

- The Memory Analysis Server should be installed on a separate machine if processed memory snapshots are larger than the remaining physical RAM.
- Collectors must be very close to their connected Agents (same LAN)
- Installation of fast hard disks such as SSDs
- Usage of partitioning for database

Example 2

Environment overview

Environment	Agents	Data centers	Transactions	Measures	Peak	Comments

Production Java	1,000	6	800 million/day 9,259 tps (many short UEM paths)	201,000	The 800 million transactions per day for the production environment are the worst case estimate. During the POC, the customer measured the number of transactions for a few high throughput agents, which was then used to compute the expected upper limit.

Sizing suggestion

Minimal process requirements

Environment	Server process	Collector process	Memory Analysis Server process	Database process	Comr
Production Java	22 GB RAM 24 cores >4 TB local file storage (depends on how long to keep the session store)	8 GB RAM 4 cores 50 GB storage	16 GB RAM 2 cores	Oracle 8 GB RAM 8 cores 2 TB storage	Do n with a bang deplo but ra slowl expe that r the appli have requi than meas ones the P Starti one s recor perha time a secor The c load so SS (RAID and Perfo Ware datab partit recor

Minimal hardware requirements

Machine/OS	Installed components
x64 Linux or Windows 24 cores 48 GB RAM	1 AppMon Server (30 GB heap) More heap for OS and to be future proof

x64 Linux or Windows 10 cores 24 GB RAM	1 AppMon Memory Analysis Server (16 GB heap) 1 Oracle database (8 GB heap)
x64 Linux or Windows 16 cores 32 GB RAM	3 Large AppMon Collectors (8 GB heap each)
x64 Linux or Windows 16 cores 32 GB RAM	3 Large AppMon Collectors (8 GB heap each)

Minimal storage requirements

	Storage	Comments
Application and configuration	5 GB	
Performance Warehouse storage	314 GB	Oracle database is needed. For DB hardware requirements, please consult the DB provider.
Session storage for 1 day	888 GB	RAID 10 or SAN
Collector application and configuration	50 GB	Including Class Cache

Minimal network requirements

	Bandwidth	Comments
Network bandwidth	110 MB	Agent <-> Collector
Network bandwidth	110 MB	Collector <-> Server
Network bandwidth compressed	27 MB	Collector <-> Server

Other considerations:

- The Memory Analysis Server should be installed on separate machine if the memory snapshots processed are larger than the remaining physical RAM.
- Collectors must be very close to their connected Agents (same LAN).
- Installation of fast hard disks such as SSDs.
- Usage of partitioning for database.

Example 3

Environment overview

Environment	Agents	Data centers	Transactions	Measures	Peak	Comments
Production .NET	500 (w3wp processes) 10 App Pools 1 TB storage	5	80 million/day 926 tps	30,430		

Sizing suggestion

Minimal process requirements

Environment	Server process	Collector processes	Database process	Comments
Production Java	22 GB RAM 24 cores	8 GB RAM 4 cores 50 GB storage	Oracle 8 GB RAM 8 cores 2 TB storage	SSD disks (RAID 1-0) and Performance Warehouse database partitioning are recommended. .NET Collectors needs SSD hard disks.

Minimal hardware requirements

Machine/OS	Installed components
x64 Linux or Windows 24 cores 48 GB RAM	1 AppMon Server (48 GB heap) More heap for OS and to be future proof
x64 Linux or Windows 8 cores 16 GB RAM	1 Oracle database (8 GB heap)
x64 Linux or Windows 10 cores 24 GB RAM	2 Large AppMon Collectors (8 GB heap each)
x64 Linux or Windows 6 cores 10 GB RAM	1 Large AppMon Collectors (8 GB heap each)

Minimal storage requirements

	Storage	Comments
Application and configuration	5 GB	
Performance Warehouse Storage	48 GB	Oracle database is needed. For DB hardware requirements, consult the DB provider.

Session storage for 1 day	89 GB	RAID 10 or SAN
Collector application and configuration	50 GB	Including class cache

Minimal network requirements

	Bandwidth	Comments
Network bandwidth	11 MB	Agent <-> Collector
Network bandwidth	11 MB	Collector <-> Server
Network bandwidth compressed	2.7 MB	Collector <-> Server

Other considerations:

- Collectors must be very close to their connected Agents (same LAN)
- Max. 250 .NET Agents per Large Collector
- Installation of fast hard disks such as SSDs
- Usage of partitioning for database

Verify results

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Deployment dashboards

Default deployment sizes

Deployment steps

Collect sizing data

Use the Deployment Sizing calculator

Assess results

If AppMon is already running in a staging environment or as a POC, be sure to double-check your results and compare them with your requirements. To help you with this, AppMon provides several system performance metrics that you can use to verify your deployment. These metrics, which you can display in specialized AppMon self-monitoring dashboards, are listed and described in the table below. For more information about self-monitoring dashboards, see Deployment Dashboards.

Deployment Verification Metrics

Metric	Comments
Skipped Events per Interval	Must be 0 or you receive incomplete PurePaths. If this happens, check the **MPS** metric, the send queue buffer size of your Collectors, and the **Bytes Received per Interval** metric.
MPS (Measures or Nodes per Second)	Compare the values at Agent/Collector and Server level with your estimate. If they are much higher than expected, the instrumentation rules should be justified or you underestimated the transaction load. You can use the **Completed PurePaths per Interval** or the **PurePath Length** measures to decide what is causing the problem.
Bytes Received per Interval	Compare these values at the Collector level with the physical bandwidth available to make sure that this will not be a bottleneck.

Send Buffer Queue Used/Committed Size	Check whether the used size consistently falls below the committed size. Whenever the used buffer reaches the committed size, skipped events occur. In some instances, the solution is as simple as increasing send queue buffer size.
PurePath Buffer Duration	Check this value to determine the retention period of PurePath analysis data. This value is computed as the difference between the start time of the oldest PurePath and the end time of the latest PurePath in memory. As a result, it increases after the Server starts and stabilizes after the Server's buffers are filled.

Verify CPU requirements

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Check power plan settings

Determine the physical cores

Related pages

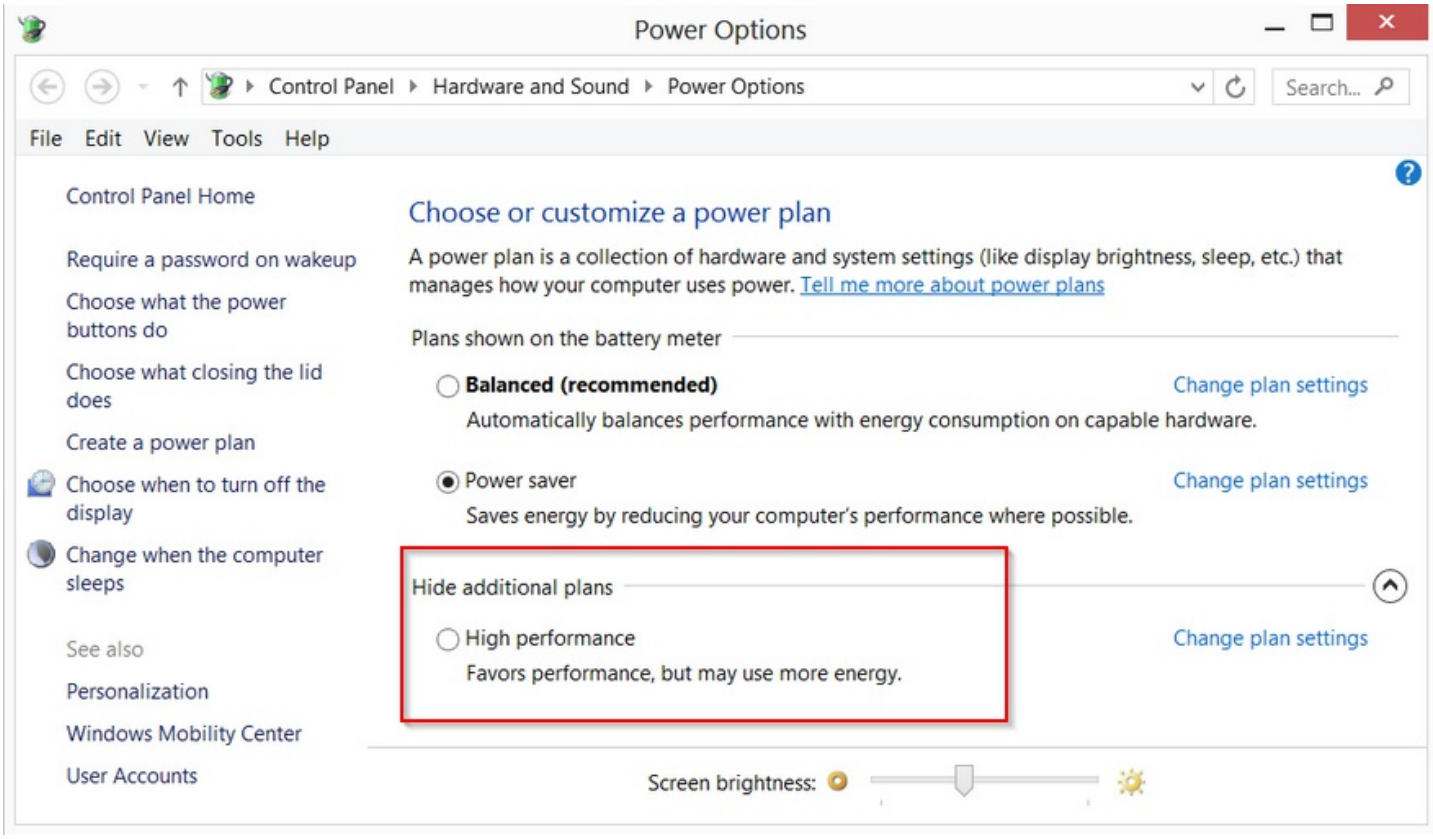
Verify results

This page describes what you can do to ensure that power plan settings are optimized for your deployment to avoid any performance degradation.

Check power plan settings

Windows

The default power plan for a Windows Server 2008 R2 is Balanced, so the CPU can be throttled to save on power consumption. Set High Performance for the AppMon Server to have consistent and predictable performance. Depending on the hardware vendor you can also modify the setting in the server BIOS (see below).



Linux

To get details on a Linux OS, you should use /proc/cpuinfo command. More information how to get all the CPU details like frequency, cores, etc. can be found here

Note that in the model name the clock frequency is different than in the CPU MHz row. This is a clear indicator that the power plan is not set to high performance.

```
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 23
model name      : Intel(R) Core(TM)2 Quad CPU    Q8400  @ 2.66GHz
stepping        : 10
microcode       : 0xa07
cpu MHz         : 1998.000
cache size      : 2048 KB
```

Check CPU clock speed

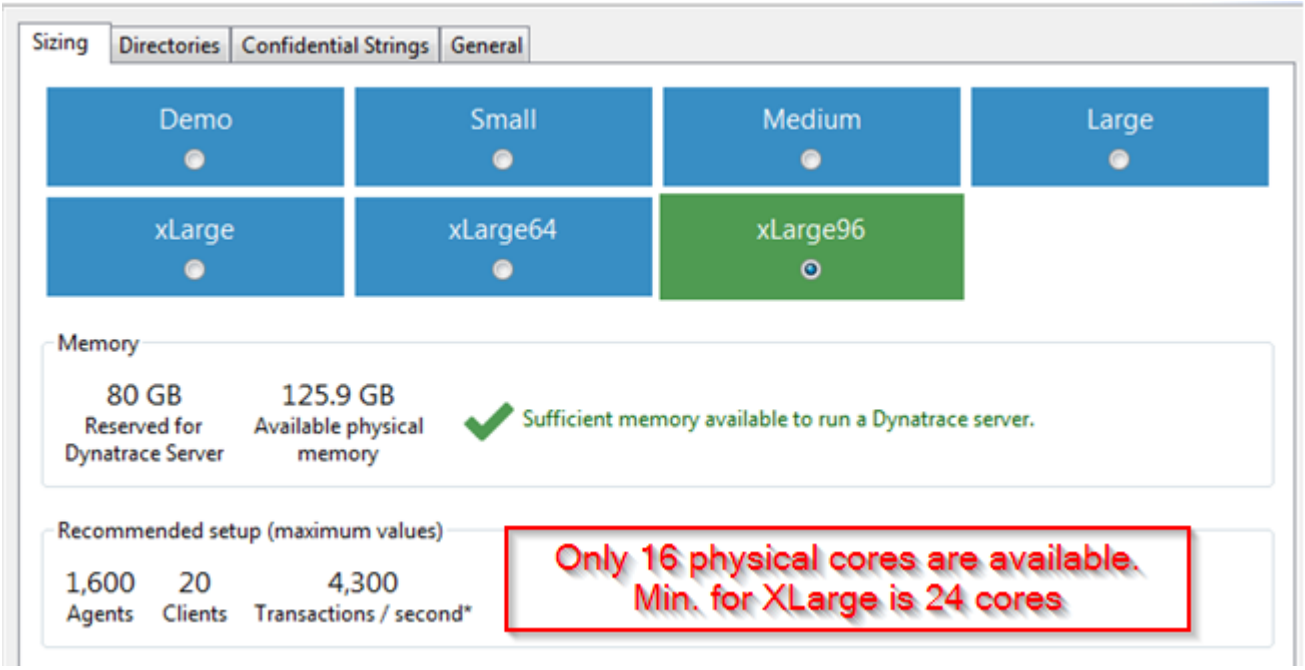
With the above mentioned command, you can also double-check the CPU MHz and verify if this meets the server requirements (see Default Deployment Sizes). When it comes to CPUs, you have one of two firms to choose from: AMD and Intel. There are plenty of options available, but we highly recommend to use **Intel CPUs** , because from a performance and efficiency standpoint there is nothing comparable at the moment.

Determine the physical cores

There are a certain number of **physical** cores required to run an AppMon Server of specific size and that cores must not be hyperthreaded or virtual / logical ones.

Hyperthreading is used to improve the parallel computations on CPUs to increase the number of independent instructions in the pipeline, so for each physical processor core the operation system addresses two virtual or logical cores instead. Depending on the applications, hyperthreading can improve the. Unfortunately, the AppMon Server cannot benefit from hyperthreading. The server requires full attention of the available processors and hyperthreading slows down the whole system, including the OS and sharing processor resources between applications. Hence, make sure that the min. requirement of physical cores is fulfilled.

In the following example shows a server with 16 physical cores, with hyperthreading enabled. The Sizing dialog box allows Xlarge to be selected, but warns that an insufficient number of cores are available.



How to detect Hyperthreading?

Rule of thumb:

	disabled	enabled
Hyperthreading	Number of processing units = number of cores	Number of processing units = number of cores * 2

Check for Hyperthreading on Windows

The Powershell script below from this source can help you to identify if hyperthreading is enabled on your machine.

```
# This Sample Code is provided for the purpose of illustration only and is not intended t
# Author: Amit Banerjee

# Purpose: Helps identify the number of physical processors, logical processors and hyper

# Provide the computer information
$vComputerName = "."
$vLogicalCPUs = 0
$vPhysicalCPUs = 0
$vCPUCores = 0
$vSocketDesignation = 0
$vIsHyperThreaded = -1

# Get the Processor information from the WMI object
$vProcessors = [object[]]$(get-WMIObject Win32_Processor -ComputerName $vComputerName)

# To account for older machines
if ($vProcessors[0].NumberOfCores -eq $null)
{
$vSocketDesignation = new-object hashtable
$vProcessors |%{$vSocketDesignation[$_.SocketDesignation] = 1}
$vPhysicalCPUs = $vSocketDesignation.count
$vLogicalCPUs = $vProcessors.count
}
# If the necessary hotfixes are installed as mentioned below, then the NumberOfCores and
else
{
# For any machine of Windows Server 2008 or above
# For Windows Server 2003, KB932370 needs to be installed
```

```
# For Windows Server 2003, KB932370 needs to be installed
# For Windows XP, KB936235 needs to be installed

$vCores = $vProcessors.count
$vLogicalCPUs = $($vProcessors|measure-object NumberOfLogicalProcessors -sum).Sum
$vPhysicalCPUs = $($vProcessors|measure-object NumberOfCores -sum).Sum
}

# Additional code can be written here to input the data below into a database
"Logical CPUs: {0}; Physical CPUs: {1}; Number of Cores: {2}" -f $vLogicalCPUs,$vPhysical

if ($vLogicalCPUs -gt $vPhysicalCPUs)
{
"Hyperthreading: Active"
}
else
{
"Hyperthreading: Inactive"
}
```

Output:

```
}
Logical CPUs: 4; Physical CPUs: 2; Number of Cores: 1
Hyperthreading: Active

PS C:\Windows\system32>
```

Check for Hyperthreading on Linux

To get details on a Linux OS, you should use the `/proc/cpuinfo` command. More on how to get all the CPU details like frequency and cores can be found here

> **Tip**
>
> “Thread(s) per core: 2” indicates that hyperthreading is enabled.

Output:

```
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
CPU(s):                8
Thread(s) per core:    2
Core(s) per socket:    4
CPU socket(s):         1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 30
Stepping:              5
CPU MHz:               1199.000
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              8192K
```

Additional deployment best practices

Server and Performance Warehouse best practices
Memory Analysis Server best practices
Collector best practices
Disk I/O Determination best practices
Network Bandwidth best practices
Virtualization best practices
Highly available installations
CPU Cores/Memory Requirements

Achieving a well-planned and executed deployment not only depends on accurately gathering and assessing sizing data. There are many key deployment factors that play into effective AppMon deployment. Pages in this section provide other critical information to consider to achieve the most effective deployment for your organization. These include:

Server and Performance Warehouse best practices

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

AppMon Server best practices

Database Server / Performance Warehouse best practices

Related pages

Default deployment sizes

Collector best practices

Memory Analysis Server best practices

Network bandwidth best practices

Disk IO determination best practices

Customer environment examples

CPU cores and memory requirements

This page explains some of the additional considerations for the AppMon Server and the Performance Warehouse to help you get the most out of your AppMon installation.

AppMon Server best practices

Use the most recent operating system possible for your AppMon Server.

Server heap size

Garbage collector (GC) activity has an important impact on AppMon Server performance and for the performance of any Java application. High correlation load on the AppMon Server typically increases the load for the GC in the following ways:

- More transactions per second directly causes higher churn rates.
- Using a bigger heap for more transactions per second keeps PurePath buffer duration constant, but this can lead to more objects on the heap. GC options set for the AppMon Server cover a broad range of deployment scenarios. You should not change these settings unless you have carefully tested them in pre-production and are sure that your settings are a reasonable improvement over the default settings in your specific scenario.

Session storage

To look up the details of a specific transaction after a period of time, you can store all transactions to a consecutive series of stored sessions using the continuous session recording feature. This needs careful planning due to high data volume in productive systems. The Session Store should normally be configured for less than 2 TB. If the Session Store estimation grows too large, change configuration settings to keep it within reasonable boundaries.

If you have a continuous load of 500 transactions per second (in a non-UEM environment), then approximately 1 TB is required to store the transactions of a single day.

Base your storage capacity estimates on the assumption of continuous load. If you know the transaction load on your system is discontinuous (for example, if there is no load during the night), then you can reduce this estimate.

Here important information regarding the session storage:

- Don’t use FTP to store session related data. FTP has pretty high latency and is designed for transfer of full files (potentially including resuming). Therefore FTP is **not suitable at all** for the random access needed for the AppMon session storage.
- Don’t use NFS, SMB or CIFS to store session related data. Those are designed for NAS environments and provide a file-level access, which is – again – not very well suited for random file access

Bottom line:

- Install AppMon on a local disk
- If the session storage has to be on the network, use a SAN

- If SAN is not possible, then try a NAS. Don’t expect very good storage performance!
- **Never ever** use FTP!

Example

In a medium deployment scenario with 250 transactions per second and a back office application only used during office hours (8h/day), the given number is 1 TB/day (for 500 transactions per second and 24h/day). A more realistic value would be 1 TB / 500 tps * 250 tps / 24h * 8h= 170 GB / day. Therefore, a storage capacity of 1 TB should be sufficient to store 6 days of data.

Internal disks such as a RAID array are usually sufficient, because the Session Store is a continuous recording of all transactions into big data blocks. In most cases, a backup solution is not required since the data is often kept only for a few days or even hours.

Database Server / Performance Warehouse best practices

AppMon Server to PWH DBMS to DB relationship

Be careful with semantics. (**DBMSes** / Database Management Systems / database servers using SQL as query language like Oracle, MS SQL Server, IBM DB2, PostgreSQL,… each supporting many *databases* in one instance).

You **may** connect each AppMon Server to a separate / differently named database, where all databases may be located on the same DB server instance, as long as your DB server can take the load. Though you may want to consolidate data or reports from AppMon Servers covering different geographical regions, you **must not** connect more than one AppMon Server to the same (named) database on the same DB server instance. This causes unintended and possibly grave results with your data. Current consolidation options are:

- Connect the stand-alone AppMon Client to your Servers, report and put the reports side-by-side.
- Consolidate the data from different databases using your DBMS.
- Export your data to a central analysis server.
- Stream the data to ElasticSearch and then chart.

Consult your Dynatrace sales representative for further details.

Performance Warehouse sizing

Embedded Derby DB restrictions

The embedded Derby database is only suitable for demonstration and evaluation purposes. For a production AppMon deployment, use a supported RDBMS for the Performance Warehouse. For details, see the Release notes.

To keep Performance Warehouse size in reasonable boundaries, AppMon uses an automated data-aging algorithm that aggregates measures in 3 steps from a very high live resolution of 10 seconds down to 1 minute, 1 hour, and 1 day.

The following table shows the recommended settings for the duration of these resolutions and the disk space necessary to store 1,000 measures in a PostgreSQL database (Oracle, DB2, SQLServer will be slightly less):

Database Sizing for 1000 Measures

Resolution	Resolution interval	Recommended duration	Used space in million rows	Disk size estimation
high	1 minute	2 weeks	~ 20 million rows	~ 2 GB
mid	1 hour	2 months	1.4 million rows measures + 0.1 million rows percentiles	~ 0.5 GB
low	1 day	1 year	0.37 million rows + 0.1 million rows percentiles	~ 0.2 GB

If you follow the recommended duration for the three resolutions, you can roughly calculate with a total space of 2 GB needed per 1,000 Measures. You should ask your database administrator about the growth rate of these tables and check the Deployment Sizing Calculator spreadsheet for the database transactions per second to determine the actual disk space needed and the hardware requirements for the database.

To estimate the number of measures you need to store in the Performance Warehouse, you must add the following major contributors:

Number of typical measures per contributor

Component	Number of typical measures	Subscribed measures
Agent	200	JVM metrics, PurePath metrics
Self-Monitoring	1,000	Performance counters, PMI values

For a deployment with 20 Agents, calculate with 1,000+20*200 = 5,000 measures.

Database suggestions

For Measures <= 30,000 a PostgreSQL database is sufficient.

For Measures > 30,000, we recommend using an Oracle, DB2 or SQLServer database.

The best way is to perform a test run with your System Profile and get the number of measures from the chart. This gives you the real number and you will not need to estimate.

Performance Warehouse database

AppMon stores very large amounts of measure data points in the Performance Warehouse. Being highly optimized, it bursts data every minute to the database. Involving a database administrator (DBA) for database sizing is recommended. Your DBA knows how many transactions a database can allow without conflicts to other applications.

There is an upper limit of 300,000 measurements per minute per instance. This is an upper limit for database sizing - it does not make much sense to invest into ultra-high-end server hardware beyond this limit for a single AppMon Server instance.

SQL Server recommendations

- The SQL Server OS should reside on its own hard disk. Data files, transaction log files, and temporary tables should also reside on a separate disk system.
- Data files (.mdf) have many random read/write operations. For high-end systems, a disk system with a RAID 1+0 configuration is recommended.
- Transaction log files (.ldf) mainly write sequentially. The transaction log heavily influences the write performance. A RAID 1+0 disk system configuration is recommended.
- Temporary tables (tempdb). For high-end systems, using an SSD for tempdb, is recommended. Otherwise, use RAID 1+0.

Other considerations

- The more IOPS the better, but not using separate disk systems for data files, transaction log, temporary tables and OS has more impact than the number of IOPS.

- Disks rated at 10,000 RPM or higher are recommended, but not using separate disk systems has more impact than lower RPM.
- When creating a new database using SQL Server Management Studio, the default size and growth increment rate for both data and transaction log files is too small. Pre-allocating both with a much higher value is recomended. For example, the data file with the estimated size for the performance warehouse, derived from the Deployment Sizing Calculator spreadsheet.
- A standard Dell database server usually is sufficient. In our lab, we use a machine for high-end purposes that costs less than $10,000.
- For a small system, you can put together the data files and the temporary tables on one disk system. The transaction log files shall remain on a separate disk system.

Oracle server recommendations

- Evenly distribute database I/O requests to multiple physical disks. A larger amount of smaller disks should be preferred over less large disks.
- Ideally, put redo logs, archive logs, table data, index data, temp data, and control files on their own dedicated physical disks or disk groups.
- Use different table spaces for persisting table data and for index data.
- A RAID 1+0 disk system configuration is recommended.

Other considerations:

- Generally, the hardware investments should follow an I/O >> RAM >> CPU priority chain. Scaling I/O and more RAM is more important than doubling the CPUs, although a good balance between the three resource types is still important.
- Disks rated at 10,000 RPM disks or higher is recommended.
- The more RAM, the better. Also, use the opportunity to store the temp data in a RAM disk.
- For XLarge deployments, partitioning the high-data-volume tables MEASUREMENT_HIGH and PERCENTILES_HIGH must be taken into account.

> **Note**
>
> Partitioning in Oracle must be licensed separately as an option on top of the Oracle Enterprise Edition.

Memory Analysis Server best practices

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Default deployment sizes

Server and Performance Warehouse best practices

Collector best practices

Network bandwidth best practices

Disk IO determination best practices

Customer environment examples

CPU cores and memory requirements

This page explains some of the additional considerations for deployments that include the AppMon Memory Analysis Server. Preprocessing memory dumps from the AppMon Server is very memory-intensive, and AppMon Memory Analysis Server component makes it possible to offload this preprocessing.

- Assign a larger heap to the Memory Analysis Server than you have assigned to the largest JVM you want to analyze.
- As long as your JVMs are reasonably sized, you can run the Memory Analysis Server on the same machine as the AppMon Server, which reduces network traffic and post-processing time.
- Use a second machine in the following situations:
  - You need to analyze huge memory dumps (16 GB or greater) and do not have enough physical memory for both processes.
  - The AppMon Server already has high constant CPU usage (less than one core free during high load hours). In this case, post processing memory dumps can lead to data loss, such as skipped events or unanalyzed PurePaths.
- If you plan a standalone Memory Analysis Server, you should consider four cores for it, and RAM should be around 25 percent larger than the largest memory dump you want to analyze.

Collector best practices

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Distributed deployments

Collector sizing

Related pages

Default deployment sizes

Server and Performance Warehouse best practices

Memory Analysis Server best practices

Network bandwidth best practices

Disk IO determination best practices

Customer environment examples

CPU cores and memory requirements

Distributed deployments

The AppMon Collector performs the byte-code modifications when instrumented applications start up. Therefore, the AppMon Agents need to send the byte code of your applications to the Collector. To keep start-up times low, it is necessary to deploy at least one Collector in every location/data center that is connected to an AppMon Server.

Hardware recommendations for a Collector box (for up to 100 Java / 25 .Net Agents):

Small AppMon Collectors x86 Linux or Windows 1 Cores+ [1] 1 GB RAM+ [2] (or comparable Solaris/AIX hardware) Used network bandwidth: 50 MB (12 MB compressed) Storage: Install: 5 GB Run-time data: 25 GB	1 AppMon Collectors (1 GB heap)

Hardware recommendations for a Collector box (for up to 150 Java / 50 .NET Agents):

Default AppMon Collectors [3] x64 Linux or Windows 2 Cores+ [1] 2 GB RAM+ [2] (or comparable Solaris/AIX hardware) Used network bandwidth: 100 MB (25 MB compressed) Storage: Install: 5 GB Run-time data: 45 GB	1 AppMon Collectors (2 GB heap) 1 AppMon Collector for monitoring plugins (1 GB heap)

Hardware recommendations for a Collector box (for up to 200 Java / 100 .Net Agents):

Medium AppMon Collectors x64 Linux or Windows 2 Cores+ [1] 4 GB RAM+ [2] (or comparable Solaris/AIX hardware) Used network bandwidth: 100 MB (25 MB compressed) Storage: Install: 5 GB Run-time data: 45 GB	1 AppMon Collectors (4 GB heap) 1 AppMon Collector for monitoring plugins (1 GB heap)

Hardware recommendations for a Collector box (for up to 400 Java / 250 .Net Agents):

Large AppMon Collectors x64 Linux or Windows 4 Cores+ [1] 8 GB RAM+ [2] (or comparable Solaris/AIX hardware) Used network bandwidth: 100 MB (25 MB compressed) Storage: Install: 5 GB Run-time data: 45 GB	1 AppMon Collectors (8 GB heap) 1 AppMon Collector for monitoring plugins (1 GB heap)

[1]) No virtual cores (Hyper-Threading), for Large Collector min. 2.2 GHz required.

[2]) Required memory for the Collector. If there are other processes running, then additional heap is required.

[3]) The Default Collector size for up to 150 Java / 50 .NET Agents is the shipped Collector size.

Collector sizing

The following deployment scenarios provide reasonable figures to plan Collector usage. You still must define Agent distribution and Infrastructure Monitoring Plugins over your Collectors.

Separate collector(s) from monitoring plugins

Plugins can affect the Collector's performance in unexpected ways and they do need memory. In addition, many Plugins are not officially supported by AppMon. To avoid any problems, you should never deploy Plugins on a Collector that serves Agents.

Connect Agents that run the same application to the same Collector to reduce memory usage on the Collector, but distribute the load over all of the Collectors that you are using. This ensures that you are not overloading a single Collector.

No firewall between agent and collectors

A firewall introduces latency in the calls between the Agent and Collector. This is often the reason for slow application start-up. The Agent needs to do several 10,000 round trips to the Collector at application start up. Even 1 ms firewall latency adds up to a noticeable time. Therefore, either use a real fast (in latency time) firewall or put the Collector into the same subnet as the Agents.

Note

The heap recommendations for the Collector (1 GB for a maximum of 100 Agents, 4 GB for a maximum of 200 Agents and 8 GB for a maximum of 400 Agents) in the deployment scenarios described here are based on the following assumptions:

- Application Size: 25,000 loaded classes
- Instrumentation Depth: 10,000 instrumented
- Clustering: High Number of “similar” applications connected to a Collector.

Although these assumptions are satisfied in typical scenarios, there are situations in which the Collector needs more memory than is expected. In such situations, the Collector rejects a certain number of Agents due to insufficient heap size. In addition to assigning more heap, you could try to optimize Agent clustering by assigning identical applications to the same Agent Group and connecting them to the same Collector or by reducing instrumentation to resolve this problem.

Disk IO determination best practices

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

How to measure available disk I/O for session storage

Related pages

Virtualization requirements and best practices

Verify CPU requirements

Default deployment sizes

Server and Performance Warehouse best practices

Collector best practices

Memory Analysis Server best practices

Network bandwidth best practices

Customer environment examples

CPU cores and memory requirements

Downloads

IOzone for Linux

IOzone for Windows

The AppMon Server always reads and writes using the following block sizes:

- Read block size: min. 1 K, max 1024 K (dynamic)
- Write block size: 64 K

There are various types of storage types available. Keep in mind that fast onboard disks (SSD, RPM 15k, etc.) typically satisfy the I/O requirements. Often a bottleneck in terms of I/O write and read speed occurs if a SAN-based storage is used. Therefore, it is important to test the I/O performance of the available hardware. Follow the steps below to verify the I/O write and read speed.

> **Important**
>
> Do not install the AppMon Server on SAN-based storage. Use the internal fast onboard disk to avoid unusual performance behavior.

How to measure available disk I/O for session storage

These tests write 32 GB of data to your disk and may take some time to complete. They generate results for write, rewrite, read, reread, random read, and random write.

1. Download the latest free IOzone tool for your operating system:

   - Download IOzone for Linux
   - Download IOzone for Windows

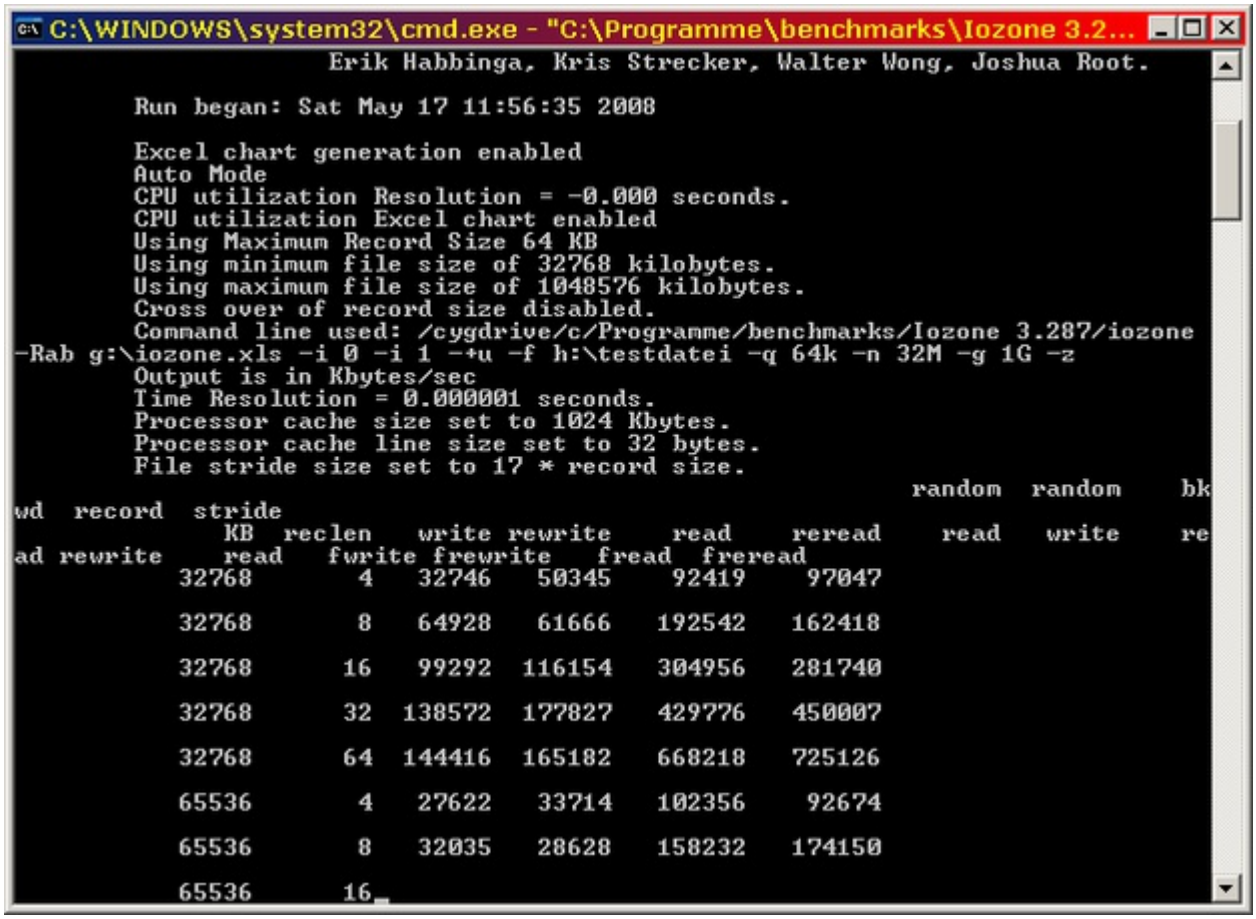
2. Execute the following commands:

	For Linux	For Windows
Test for WriteSpeed	/opt/iozone/bin/iozone -s 32000m -r 64k -i 0 -i 1 -w -f /<installation-directory>/testwritespeed	<installation-directory>/iozone -s 32000m -r64k -i 0 -i 1 -w -f testwritespeed
Test for RandomRead	/opt/iozone/bin/iozone -s 32000m -r 16k -i 2 -w -f /<installation-directory>/testrandomread	<installation-directory>/iozone -s 32000m -r16k -i 2 -w -f testrandomread

   For RandomRead try different sizes, like 1 K or 1024 K to see how the disk behaves.

3. Verify available disk I/O.

   The results from these tests should match the disk I/O requirements specified in the results of the Deployment Sizing Calculator.


C:\WINDOWS\system32\cmd.exe - "C:\Programme\benchmarks\Iozone 3.2...
Erik Habbinga, Kris Strecker, Walter Wong, Joshua Root.
Run began: Sat May 17 11:56:35 2008
Excel chart generation enabled
Auto Mode
CPU utilization Resolution = -0.000 seconds.
CPU utilization Excel chart enabled
Using Maximum Record Size 64 KB
Using minimum file size of 32768 kilobytes.
Using maximum file size of 1048576 kilobytes.
Cross over of record size disabled.
Command line used: /cygdrive/c/Programme/benchmarks/Iozone 3.287/iozone
-Rab g:\iozone.xls -i 0 -i 1 -+u -f h:\testdatei -q 64k -n 32M -g 1G -z
Output is in Kbytes/sec
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 Kbytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.
random random bk
wd record stride
KB reclen write rewrite read reread read write re
ad rewrite read fwrite frewrite fread freread
32768 4 32746 50345 92419 97047
32768 8 64928 61666 192542 162418
32768 16 99292 116154 304956 281740
32768 32 138572 177827 429776 450007
32768 64 144416 165182 668218 725126
65536 4 27622 33714 102356 92674
65536 8 32035 28628 158232 174150
65536 16

Network bandwidth best practices

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

How to measure the available bandwidth

Download jPerf to run your own bandwidth test

Related pages

Default deployment sizes

Server and Performance Warehouse best practices

Collector best practices

Memory Analysis Server best practices

Disk IO determination best practices

Customer environment examples

CPU cores and memory requirements

Network bandwidth is difficult to estimate because it heavily depends on the type of instrumentation. For example, large attachments such as stack traces collected with exceptions or SQL statements cause much more traffic than simple PurePath nodes.

In typical scenarios, every 500 tps sent from a Collector to an AppMon Server contributes 8 MBit/sec (1 MByte/sec) of upstream bandwidth. The downstream is negligible in comparison. Using compression reduces this value by a factor of approximately 4, but be aware of increased CPU demand on the Collector and AppMon Server when compression is used.

For example:

Network bandwidth from AppMon Collector to AppMon Server

Transactions	Network Bandwidth	Network Bandwidth (compressed)
500 tps	8 MB/sec	2,000 kBit/sec
1,000 tps	16 MB/sec	4,000 kB/sec
2,000 tps	32 MB/sec	8 MB/sec

If using a WAN connection, you should always verify these estimates after deployment.

Also, consider that for your configuration:

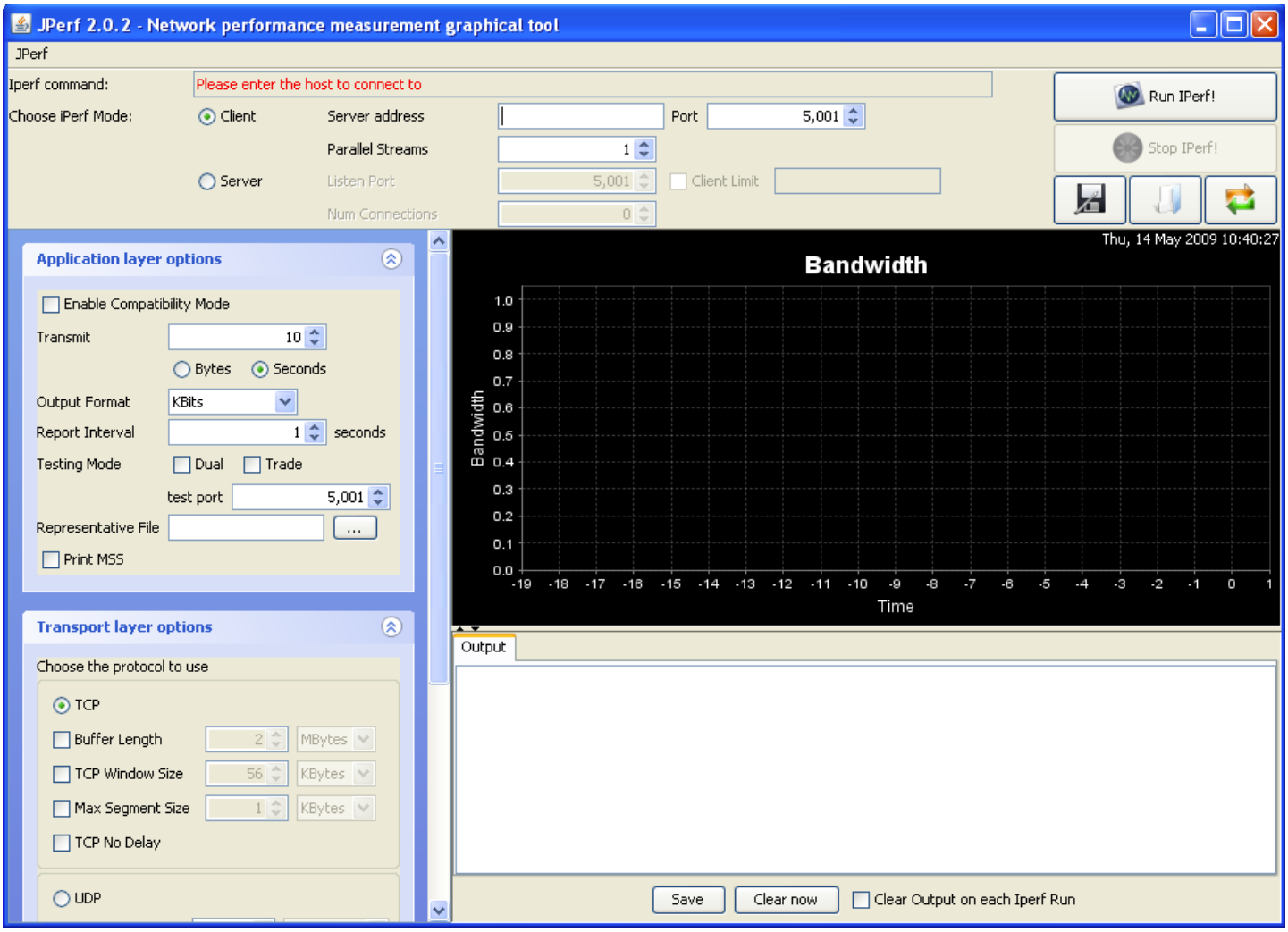
- The network between the Agent and Collector needs low latency.
- The network between the Server and Collector needs high bandwidth.

> **Important**
>
> If you need to put an AppMon Collector on the application hardware itself (not recommended), make sure you have enough bandwidth to share, or, perhaps better, have a separate network interface for the AppMon Collector.

How to measure the available bandwidth

The question now is “how can I find out the available bandwidth for my current environment?” Use jPerf to determine the available bandwidth between components. jPerf is a modern alternative for measuring TCP and UDP bandwidth performance. You can use the jPerf Java-based client, which has a simplistic but easy to use UI. The results display in the graph on the right side.

To measure bandwidth between all components, you must install the server part of jPerf. For example, to measure the bandwidth between the Collector and Server, install the jPerf server on the Collector or on the AppMon Server. For the component on which you do not install the jPerf server, use the graphical jPerf application to run the test. It is recommended to install the iPerf server on the AppMon Server, so you can run the test without installing iPerf on every component.



jPerf Network Performance Measurement Tool

Download jPerf to run your own bandwidth test

iPerf Server: Download jPerf

jPerf Graphical UI for iPerf : Download jPerf

Virtualization requirements and best practices

Related pages

Verify CPU requirements

Default deployment sizes

Server and Performance Warehouse best practices

Collector best practices

Network bandwidth best practices

Disk IO determination best practices

Customer environment examples

CPU cores and memory requirements

This page provides a requirements and best practice guide for running an AppMon server virtualized. AppMon can run virtualized, but there are some requirements to guarantee a smooth running server.

Dedicated resources

Dynatrace recommends dedicated\entitled cores, memory and disk for the server via a VMWare resource pool. Overcommitment of cores or memory can cause performance problems during data collection and force the server to skip some data such as PurePaths.

Virtual CPUs

For most processes, VMWare recommends a single virtual CPU for a virtual machine (VM). This reduces the management overhead on both the guest system and VMWare. A single virtual CPU however has the same impact as a single physical CPU. Only a single thread at a time can be executed. The AppMon Server has a lot of threads and it must therefore be an exception to this rule. In addition, VMWare cannot parallelize something that is not executed in parallel by the VM. That means a VM with two virtual CPUs can never use more than two physical CPUs at the same time. In that regard we are depending on the power of the physical hardware. As a general rule, the number of vCPUs depends on the size of the setup. You should take the number of cores from the Default Deployment Sizes as a base recommendation. If the numbers do not fulfill the requirements, then increase the cores on the VM.

Memory

VMWare does resource pooling on available physical memory if not otherwise instructed. That means you can have four VMs with each having 4 GB memory and physical hardware with only 12GB. Also consider that the hypervisor needs memory (approx. 500 MB).

The vHost can instruct the VM not to use a portion of the assigned memory. This is known as ballooning, and it lets VMWare dynamically assign available memory from one VM to the other. This can cause the VM to swap out running processes or parts of them.

Swapping in general can harm application performance. It is particularly harmful to java applications because memory is cycled through all the time, which is harmful to the AppMon Server. Thus, the whole of a java program must always be in physical memory. As a rule, the VM running the AppMon Server must have enough reserved memory to accommodate the AppMon Server plus the operating system. See Default Deployment Sizes for proper memory sizing.

I/O and network

The AppMon Server writes a lot of data to disk for session recording, receives a lot of data, and also sends a lot data to the database. Disk and network are still shared resources in a VM, so ensure there are enough resources reserved to the AppMon Server VM.

Guidelines

- Separate Network Switch for incoming traffic to the AppMon Server. See the bandwidth requirements in Network Bandwidth Best Practices to learn how.
- Make sure that all other traffic (database, client, disk) is taking a different physical network switch.

- Disks are mostly connected via SAN to a virtual machine. Make sure the network switch can take the load and, if possible, is dedicated to the usage of the SAN.

See Disk IO Determination Best Practices for more guidance regarding disk I/O for deployment, See Network Bandwidth Best Practices for more guidance regarding network load.

Understanding VMware performance charts

As an VMWare hypervisor administrator, you must understand and know your infrastructure. The following table shows performance chartsidentify, troubleshoot, and resolve issues.

Key CPU metrics

Counter name in API	Measurement	Unit	Description	Desired value
cpu.ready,summation	Ready	ms	Ready time is the time spend waiting for any CPUs to become available in the past update interval. Be aware that this is a counter and is dependent on a time interval, such as 20 seconds.	**Should be close to 0**.
cpu.idle.summation	Idle	ms	CPU Idle	
cpu.used.summation	Used	ms	CPU Used	
cpu.swapwait.summation	Swap Wait	ms	Swap wait time is time spent waiting for memory to be swapped in. When the VM is waiting for memory, it is not doing work.	**Should be 0**.

Key memory metrics

Counter name in API	Measurement	Unit	Description	
mem.vmmemctl.average	Balloon	kB	The amount of memory currently claimed by the balloon driver. This does not represent a performance problem, but indicates the host starting to take memory from less needful VMs for those with large amounts of active memory. But if the host is ballooning, check swap rates (swapin and swapout) which would be indicative of performance problems.	**Should be 0**.
mem.swapin.average	Swap in	kB	The rate at which memory is being swapped in from disk. A larger number indicates lack of memory and is a clear indication that performance is suffering as a result.	**Should be 0**.

mem.swapout.average	Swap out	kB	The rate at which memory is being swapped out to disk. A larger number indicates lack of memory and is a clear indication that performance is suffering as a result.	**Should be 0**.
mem.swapinrate.average	Swap in rate	kB/s	The swap in rate reports the rate at which a VM's memory is swapped in from disk.	**Should be 0**.
mem.swapoutrate.average	Swap out rate	kB/s	The swap out rate reports the rate at which a VM's memory is swapped out to disk.	**Should be 0**.
mem.swapped.average	Swapped	kB	Memory Swapped (Average).	**Should be 0**.
cpu.swapwait.summation	Swap Wait	ms	Swap wait time is time spent waiting for memory to be swapped in. When the VM is waiting for memory, it is not doing work.	**Should be 0**.

See VMWare performance counters for more information.

Highly available installations

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Availability of system under diagnosis versus AppMon availability

Availability of system components

Related pages

Verify CPU requirements

Default deployment sizes

Server and Performance Warehouse best practices

Collector best practices

Network bandwidth best practices

Disk IO determination best practices

Customer environment examples

CPU cores and memory requirements

This page describes how AppMon handles availability and describes measures you can take to increase the availability of a deployment. This page also describes high-availability strategies to protect AppMon against data loss due to hardware or software faults.

Availability is an important consideration in application deployment, especially in production environments. What you can do to increase availablity ranges from very simple and inexpensive backup and disaster recovery strategies to sophisticated usage of Collector fail-over and / or fail-over clusters.

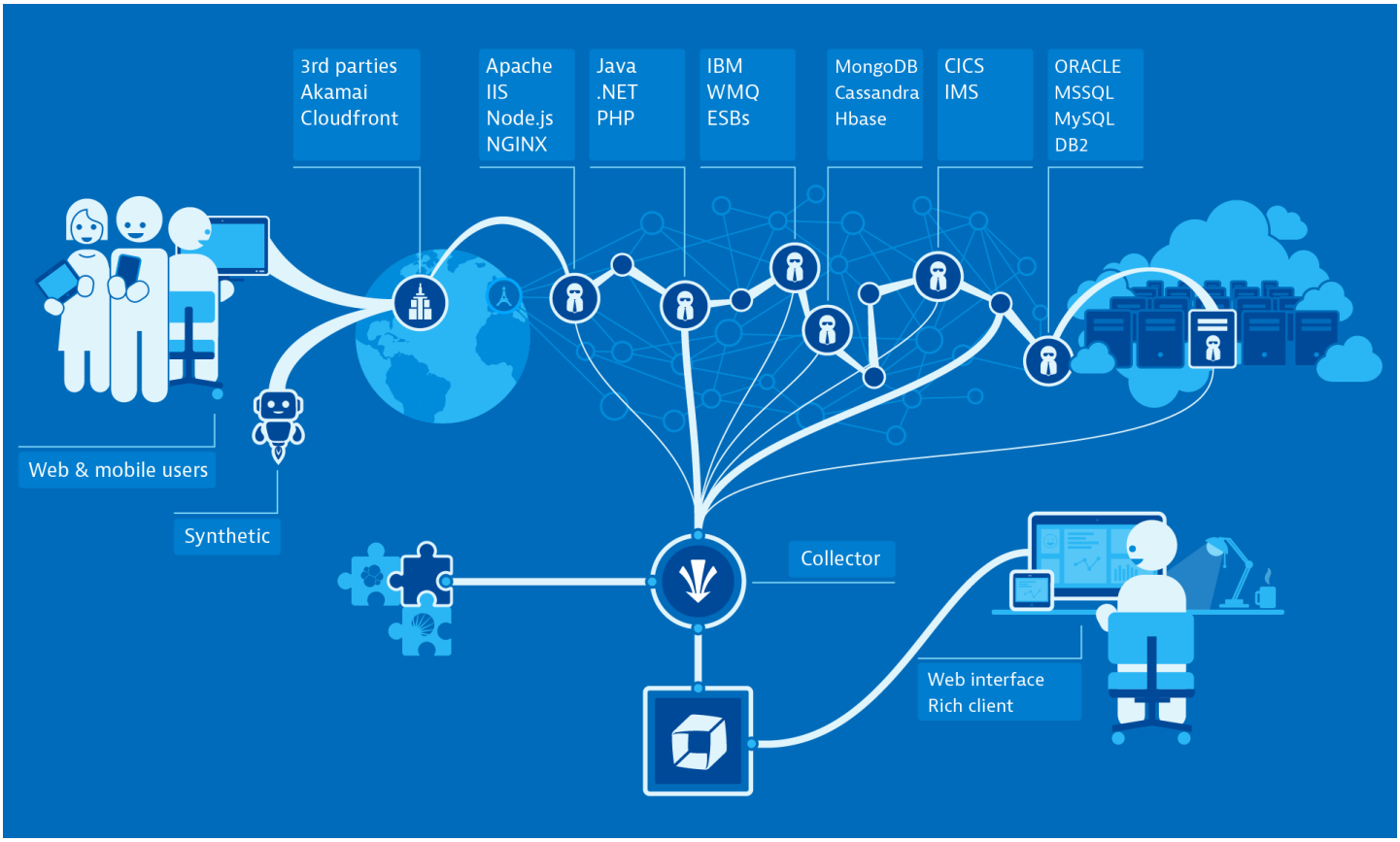
Note

Implementing a high-availability AppMon deployment is not required for all customers. In a typical QA environment and most production environments, it is acceptable to lose monitoring data in the rare case of a hardware fault. In addition, AppMon has a built-in watchdog mechanism that automatically brings up the system again after a software fault and Agents can fail-over to a redundant Collector in a designated group.

Availability of system under diagnosis versus AppMon availability

The availability of the AppMon solution and the availability of the System Under Diagnosis (SUD) are not intrinsically related, which means that an AppMon failure does not typically cause the SUD to fail and vice versa. This is an important benefit of loosely coupled AppMon Agents.

Availability of system components



The **AppMon** system architecture implements loose coupling between the components (see diagram above). This guarantees that a failing AppMon Agent does not take down the AppMon Collector or the AppMon Server. Similarly, a failing AppMon Client does not cause an AppMon Server failure. All components automatically reconnect after a failure. Therefore, it is important to focus on the availability of individual AppMon components.

SUD and AppMon Agent

Because the AppMon Agent runs in the same process as the SUD, an Agent failure may cause the SUD to stop responding. Note that this is the only AppMon component that cause the SUD to stop responding.

To minimize this risk as much as possible, the following measures are taken:

- The Agent is a very thin software layer. As much work as possible is done by the Collector and Server.
- The Agent usually fails gracefully. For example, if the connection to the Collector / Server fails, the Agent simply skips application events or at worst fails to instrument the SUD. In either case, this will not cause failure of the SUD.
- Significant testing is done by the AppMon QA Team to ensure the reliability of this component.

If you need high availability (HA), such as fail-over support (FO) for the SUD, you can take any measure supported by the SUD to ensure its health (beyond the scope of this page).

Summary: SUD and Agent Availability

Risk	Consequences	Result
Hardware Failure	SUD is unavailable	AppMon monitoring alerts you
Software Failure of SUD	SUD is unavailable Agent may be unavailable	At least missing AppMon alerts you
Software Failure of Agent	Agent is unavailable SUD may be unavailable	Even in this one case that AppMon has an adverse effect on the SUD, missing AppMon data alerts you

AppMon Collector

The Collector collects data such as measurements and PurePath-related events from the Agents and sends these data to the Server. If a Collector fails due to hardware or software failure, the Agents buffer data from a couple of seconds to up to a minute, depending on load. As a result, no data is lost if the Collector is started again within this time.

You should use more than one Collector for Agents of the same type (Agent Group / tier) and configure Collector groups in a production environment.

If the Collector comes up within a minute again, the Agents will automatically reconnect to the Collector and the latter to the Server.

If not, the Agents can fail over to a different Collector in the Collector group.

The Collector uses self monitoring with an integrated software watchdog to detect fatal software problems. For issues such as out-of-memory or hanging threads, the Collector process is restarted automatically. This is a very important feature that increases availability whether or not you plan to use clustering techniques for high availability, because the chances for cluster software to detect such problems are very limited.

The time necessary for the watchdog to detect software problems ranges from immediate (for example, out-of-memory) to a couple of minutes (hanging threads).

Summary: Collector Availability

Risk	Consequences	Precautions
Hardware Failure Collector Software Failure Collector	Connected Agents and Collector plugin's PurePaths and measurements are missing from the time the Agents try to reconnect to the Collector and could not buffer.	Make the Collector highly available by creating Collector groups and having head-room and redundancy in Collectors

AppMon Server

If the Server fails due to a hardware or software failure, the Collectors buffer data for a period of time, ranging from 30 seconds to a couple of minutes depending on load and Collector heap configuration. As a result, no PurePaths or measurements are lost if the Server is started again within this period.

The Server uses self monitoring with an integrated software watchdog to detect fatal software problems. For issues such as out-of-memory or a hanging threads, the Server process is restarted automatically. This is a very important feature that increases availability if you plan to use clustering techniques for high availability or not, because the chances for cluster software to detect such problems are very limited.

The time necessary for the watchdog to detect software problems ranges from immediate (for example, out-of-memory) to a couple of minutes (hanging threads).

Summary: Server Availability

Risk	Consequences	Precautions
Hardware Failure Server Software Failure	Connected Collector PurePaths and measurements that could not be buffered are missing.	Make the Server highly available

AppMon Frontend Server

For the Frontend Server it is not necessary to implement special availability precautions, but it plays an important role in the component puzzle. It frees the Server from having to provide analysis data to the Clients, giving it more headroom for PurePaths correlation and protecting it from potentially harmful queries.

The AppMon Client is a much less critical component than the Agent, Collector and Server. It is not necessary to implement high availability for the Client.

Performance Warehouse

All Measures and Incidents are stored in the Performance Warehouse RDBMS. Therefore, availability is extremely important. A backup and disaster recovery strategy if high availability of the AppMon solution is a priority.

> **Note**
>
> The AppMon Server buffers data for up to one hour (memory permitting) if the Performance Warehouse is not available. After this period of time has elapsed, the data is removed from the buffer.

Install the AppMon Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What to know before installing

Download and install the server

Sizing and memory

Using dtserver.exe

AppMon Server services / daemons

Additional considerations and troubleshooting

Related pages

System requirements

Server administration

Set up collectors

Install collectors

Install Memory Analysis Server

Install agents

Watch and learn

Installing Application Monitoring (Windows)

Installing Application Monitoring (*NIX)

This page details installing the AppMon Server and covers critical information needed before

installation. It also covers starting and stopping AppMon Server services and daemons, and notes additional considerations for installing the AppMon Server.

What to know before installing

All platforms

- Before installing AppMon Server (backend and frontend processes / Windows services / *NIX daemons), see the Deployment Guide to calculate the size of your AppMon deployment and for detailed hardware recommendations specific to your load.
- You must have a valid AppMon license to start the AppMon Client and begin using AppMon.
- You must have access to a database of adequate size for the AppMon Performance Warehouse. You can only connect to the Performance Warehouse with a valid license.

*NIX

- Some UNIX and Linux versions set the limit for file handles / sockets as low as 1024. The AppMon Server checks that the OS allows at least for 2048. However, you can quickly outgrow this, with several active System Profiles.

As there is no reservation overhead involved, increase the limit generously by adding 50 * 3 * 2 * System Profiles.
Estimate 50 files per session * maximum 3 writer threads * 2 for (backend) Server and Frontend Server * number of active System Profiles.

Check the hard and soft limits with `ulimit -Hn` and `ulimit -n` respectively before making any settings, and increase them if needed. For example, `ulimit -n <numberOfFileDescriptors>` raises the soft limit of the current user. Add entries for other users / groups to `/etc/security/limits.conf`. See the section Number of open files limit on Linux for more information).

- Installation requires Java Runtime Environment (JRE) 1.8 or later.
- For AppMon 6.3 and later, `init.d` shell scripts allow the Server (and Collectors) to start as non-root under the desired user account. If your Linux uses `systemd` for startup, please use the attached systemd script samples, which are the wrappers for `init.d`.

Download and install the server

Windows

Download the Windows 64-bit installer package for your architecture and execute it.

The installation creates Start menu entries and registers the AppMon Server to automatically start the Windows service. To get information for debugging a Windows installer problem, run the installer with the following command:

```
dynatrace-full-7.0.0.1289-x86-64.msi /lx c:\debug_log.txt
```

The default installation directory is `C:\Program Files\dynaTrace\Dynatrace <version>`.

The installation directory is referred to as `<DT_HOME>`. You can optionally set the installation directory in an environment variable with the following command:

```
setx DT_HOME "C:\Program Files (x86)\dynaTrace\Dynatrace 7.0"
```

Reference the variable with `%DT_HOME%`.

Linux, AIX and Solaris

> **Do not run the AppMon Server as root.**
>
> Use a designated account and run the installation in that account. For AppMon 6.3 and later, you can define `DT_RUNASUSER` in an install script to prevent running the AppMon Server as root.

Download the Linux/UNIX installation package (JAR file).

The installation directory defaults to `<currentDir>/dynatrace-7.0` (suggested parent: `/opt`).

For example, if you issue `cd /opt` and the Linux full installation file is in your home's `Downloads` directory, then executing
`sudo java -jar ~/Downloads/dynatrace-full-7.0.0.1289-linux-x86-64.jar`
installs in `/opt/dynatrace-7.0` if you accept the default. Then change `user:group` by entering the following:

```
sudo chown -R <DT_RUNASUSER>:<DT_USER'SGROUP> /opt/dynatrace-6.3
```

Copy the `dynaTraceServer` shell script to `/etc/init.d`. Define `DT_RUNASUSER` in the script (and other variables like DT_OPTARGS and DT_FE_OPTARGS if necessary). Add this script to the system boot by using `chkconfig --add dynaTraceServer` or, depending on your distribution, such as if you are using Debian, use `sudo update-rc.d dynaTraceServerNonRoot defaults`.

> **Note**
>
> If not specifying `DT_RUNASUSER` and registering the script and restarting, then the Server starts as root. Locations such as `<DT_HOME>/server/osgi` and `temp` then belong to root. To correct this, rerun the above `chown` command.

Check if `dtserver` runs - and under what user with `ps -eo pid,fname,user|grep dtserver` (lists launcher). If this doesn't work on your Linux flavor, try `ps aux|grep dtserver.` The first, short output is for the launcher, the second, long one is for the Server. The third one is for the Frontend Server, as some libs contain `dtserver` and the fourth one is for `grep`. The user the Server runs under is at the very beginning, before the PID.

On Linux or UNIX systems, the Java installer suggests installing in a subdirectory named `dynatrace-7.0` of the current directory. The installation directory is referred to as `<DT_HOME>`. You can optionally set the installation directory in an environment variable with the following command:

`export DT_HOME=/opt/dynatrace-7.0` in `~/.bash_profile` **or** `~/.bashrc`, depending on your UNIX/Linux version.

Reference the variable with `$DT_HOME`.

The `DT_HOME` directory structure includes these subdirectories.

Relative Path	Contents
`<DT_HOME>`	Executables and the main configuration (`.ini`) files, such as `dtserver.ini`.

`<DT_HOME>/server/conf`	Configuration files for dashboards, System Profiles and plugins.
`<DT_HOME>/server/sessions`	Session storage directories for stored or recorded sessions. You may want to change that path in any non-testing scenario due to it's potential large size.

Sizing and memory

You can set deployment sizing and the associated memory using the AppMon Client at startup. After activating a license, you can choose from a list of predefined deployment sizes.

You can change the selected deployment size at any time later by selecting **Settings > Dynatrace Server > Settings > Sizing** in the Client, or directly under `-memory` in `DT_HOME/dtserver.ini`. See Sizing Settings for more information.

Using dtserver.exe

`dtserver.exe` lets you interface with the AppMon Server from a command line.

Windows options for dtserver.exe

Execute the following commands to list all the available options for `dtserver.exe`:

```
cd %DT_HOME%
dtserver.exe -?
```

The default `DT_HOME` is `C:\Program Files\dynaTrace\dynaTrace 7.0`.

Sample Windows output:

```
Dynatrace Server
Version: <version> - built <date>
Copyright (C) 2004-201x, Dynatrace

Usage: dtserver [options]

Available options:
   -?, -h, --help Print this help message
```

```
-?, -h, --help Print this help message
-v             Display version information
-loglevel [0..8]
               Set loglevel, valid values are: [0..8] or
               [finest|finer|fine|config|info|warning|severe|debug|none]
               Default setting: 4 (info)
-instance <name>
               Instance name for this Dynatrace Server instance
-portoffset <port>
               Adds the specified port offset to each port in the
               configuration of Dynatrace Server, allowing to start
               multiple instances based on the default configuration.
-listen [<ip>:]<port>
               Listen for dynaTrace Agents on [<ip>:]<port>
-memory [demo|small|medium|large|xlarge|xlarge64|xlarge96]
               Define memory model/memory settings for Dynatrace Server.
               Default memory model: demo
-service [install | uninstall | start | stop | list(all)]
               Install/uninstall Windows Service
               (uninstall might require you to log off)
-basedir <dir> Set working directory for application
-logdir <dir>  Override default log directory (command line argument only)
-ini <config>  ini file to use, default is dtserver.ini
               A value of 'ignore' will skip reading the ini file.
-timeout <sec> Timeout value for watchdog feature, 0 turns the watchdog off.
               Default value: 180, valid value range [30..86400] sec
-watchdogaddress 127.0.0.1
               IP address to be used for watchdog feature
-portrange <n>[:<m>]
               Port range to be used by watchdog feature
               By default a port within range 50000:65535 is used
-vm <javaVM>   The Java VM to be used for launching the application
-startup <startup.jar>
               The startup jar to execute. The default is to find the
               plugins/org.eclipse.equinox.launcher jar with the highest
               version number.
-ea[:<package>"..."|:<class>]
               Enable assertions (assertions are disabled by default).
               A <package> or <class> name can be specified (cf. Java docs)
-Dosgi.*=<value>
               Set arbitrary OSGI configuration parameters
```

```
   -D*=<value>    Set properties for Java application
   -X*=<value>    Set (non standard) options for Java VM
```

Linux and Unix options for dtserver

Execute the following commands to list all the available options:

```
cd $DT_HOME
./dtserver -?
./dtfrontendserver -? //Frontend Server specific options
```

The options are the same as for Windows, with one exception. Instead of `-service start`, the option is:

```
cd $DT_HOME
//Interactively start the process in daemon mode / in the background
./dtserver -bg //Generally or at system boot use the init.d/dynaTraceServer shell script
./dtfrontendserver -bg //Shell script starts both processes
```

The Linux/UNIX commands `./dtserver -bg` is equal to `dtserver.exe -service start` on Windows.
The *NIX shell script `init.d/dynaTraceServer` starts both processes. On Windows you must start the Frontend Server separately with `dtfrontendserver.exe -service start` or through the Services control panel.

Comparing dtserver.exe options for Windows Services and Linux/Unix daemons

Commands and switches differ based on whether you are using Windows or Linux/UNIX.

Install

Linux/UNIX: `dtserver -service install`

`dtserver [-instance <instance_name>] -service install` `dtserver -service install`

In the above command, `<service_name>` is the file name of the shell script copied to or sym-linked in `/etc/init.d`, for example `sudo chkconfig --add dynaTraceServer`.

Windows: The service installs automatically.

Execute `"<DT_HOME>\dtserver" -service install` to install the default instance manually, where `<DT_HOME>` is `C:\Program Files\dynaTrace\dynaTrace 7.0` in a full default installation.

Use the following example to change from auto-start to manual start: `sc config "dynaTrace Server 7.0" start= demand`

Uninstall

Linux: `dtserver -service uninstall` **Windows**: `/sbin/chkconfig --del <service_name>`

`/sbin/chkconfig --del <service_name>`

List

```
dtserver -service list
dtserver -service listall
```

Linux/UNIX: `list` daemons.
Windows: `/sbin/chkconfig --list`. On Windows, `list` Server details or `listall` installed AppMon services: Servers, Analysis Servers, Collectors, Host Agents and Web Server Agents.

Start

```
dtserver -bg
```
or `init.d/dynaTraceServer start`

Stop

Linux: `dtserver.exe -service stop` **Windows**: `init.d/dynaTraceServer stop`

Status

Linux: `dtserver -service list` **Windows**: `init.d/dynaTraceServer status`

Version

`dtserver -v` `dtserver -v`

`init.d/dynaTraceServer version`

On Windows, the AppMon Server service is set to start automatically by default. To check whether the default has been changed, go to **Control Panel > Administration > Services** or execute `sc query "<Service Name>"`. To change from automatic to manual or vice versa, execute `sc config "Service Name" start= auto| demand| ...`. Note that there is no space between `start` and `=`, but there must be a space after `start=`.

For example:
`sc config "dynaTrace Server 7.0" start= auto`

On Linux or UNIX, copy the `$DT_HOME/init.d/dynaTraceServer` script to `/etc/init.d` and execute the following command to autostart the Server at boot time:

`/sbin/chkconfig --add dynaTraceServer`

On other *NIX flavors use `update-rc.d`.

On Ubuntu you may have to execute `sudo apt-get install sysv-rc-conf` .
Then `sudo sysv-rc-conf dynaTraceServer on`. Check with `sudo sysv-rc-conf --list`.

See the documentation of your distribution for details. Be sure to do this in the right user context to have the necessary Write and Execute permissions.

For the default #, you do not need to modify the `dynaTraceServer` script. It was generated during installation to match the installation details. See Linux/UNIX documentation for `/sbin/chkconfig` options.
For GUI tools to administer daemons, see, for example, `ntsysv` on Red Hat or the YaST > System > System Services (Runlevel) editor on SUSE.

AppMon Server services / daemons

Starting and stopping server services / daemons

> **Note**
>
> Make sure the AppMon Server service has sufficient privileges if changing any output directories such as the session directory.

Using the services web page

To start or stop AppMon Servers, Collectors, Agents, etc. through a web browser on all operating systems:

1. Enter `http://<hostname>:8020` in the browser's address bar. For example, http://localhost:8020. The **<Server> Services** page appears.
2. Click **Administration**.
3. Click **RESTful Services**.
4. Enter the AppMon administrator user name and password when prompted.
5. Click the desired selections on the **RESTful HTML Services** and subsequent pages. For example, to shut down Collector@60test, click **Collectors** on the **RESTful HTML Services** page, **dynaTrace Collector@60test** on the **Collectors** page, and finally **Shutdown** on the **Collector Details** page.

Using the Windows GUI

The Windows **Start** menu contains shortcuts to start and stop all services running locally. In **Windows Services**, use **dynaTrace Server 7.0** and **dynaTrace Frontend Server 7.0** to start and stop AppMon services.

Using the Windows command line

To start or stop the AppMon Server backend and frontend processes / services through the command line:

```
cd %DT_HOME% dtserver.exe -service start
dtfrontendserver.exe -service start
```

or

```
dtserver.exe -service stop
dtfrontendserver.exe -service stop
```

Or use `sc` :

```
sc start "dynaTrace Server 7.0"

sc start "dynaTrace Frontend Server 7.0"
```

or

```
sc stop "dynaTrace Server 7.0"

sc stop "dynaTrace Frontend Server 7.0"
```

Using the Linux or Unix command line

To start or stop the AppMon Server backend and frontend processes / daemons through the command line:

Interactively (for testing):

```
cd $DT_HOME
./dtserver -bg

./dtfrontendserver -bg
```

Or use the shell script (add to system boot with `chkconfig` or `update-rc.d`, depending on *NIX flavor) to start both, the backend and frontend processes as daemons in the background:

```
<DT_HOME>/init.d/dynaTraceServer start
```

Kill both processes (same as `kill -2 <process ID> for each`): `<DT_HOME>/init.d/dynaTraceServer stop`

Additional considerations and troubleshooting

Automatic installation

You can automate the installation with Ansible, Chef, or Puppet. Click the link below for more detailed information.

- Dynatrace-Ansible
- Dynatrace-Chef
- Dynatrace-Puppet

Performance considerations

The AppMon Frontend Server provides the data to the connected Clients for analysis. This way the original AppMon Server has more headroom to gather and correlate data from the Collectors.

The AppMon Server requires disk storage for recorded sessions. By default, the directory used for stored sessions is in the installation directory (see above). You may need to relocate this directory to a high-performance I/O device or a high-volume device to increase the Server's performance when working intensively with stored sessions. You can:

- Mount a designated volume to that directory or create a symbolic link to another volume.
- Change configuration settings to point to a different directory in the client by selecting **Settings > Dynatrace Server > Storage**.

Memory and CPU requirements are covered in detail in the Deployment Guide.

Versioned profiles

All System Profiles are stored in the `profiles` folder as XML files (ending with `.profile.xml`). Those files should not only be backed up on a regular basis but can also be archived with common version control systems. Doing so provides easy history control and rollback over the course of time.

Parallel startup of server and analysis server

If you have the AppMon Server and AppMon Analysis Server installed on the same machine (sharing the same core executables), it is not possible to start both servers at the same time, for example by using a UNIX shell script.

In this case, possible solutions are:

- Install the Analysis Server standalone on the machine, or
- In your shell script, first start the Analysis Server, wait a short time (typically 10 seconds suffices) and then start the AppMon Server.

Number of open files limit on Linux

Some UNIX-based systems impose a rather low limit on the number of files or file descriptors a user or process can have open. This limit includes open socket connections.

Because the AppMon Server needs to read / write many files from / to the session store and have many socket connections open, the system default limits for the number of open files may be too low.
Some Linux distributions use a default of 1024 file handles, which is most likely too low for medium to large AppMoninstallations. The AppMon Server enforces a minimum of 2048 file handles. You can verify this by entering a query with `ulimit -n`.

> **Note**
>
> Don´t confuse the `ulimit` (thousands) with the system-wide limit (hundreds of thousands) that you can check with `more /proc/sys/fs/file-max`.

You may learn of an insufficient limit through an error message like `IOException: Too many open files` or `java.net.SocketException: Too many open files`. Before changing anything, check the current (soft) limit with `ulimit -n` or `ulimit -Sn`. Use `ulimit -Hn` to determine how much you can raise it as non-root.

Use `ulimit -n <numberOfFileDescriptors>` to raise the (soft) limit temporarily for the current user. Add entries for other users / groups to `/etc/security/limits.conf`. Use `ulimit -n` to check that the change was successful.

Install collectors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What to know before installing collectors

Install collectors

Start and stop a collector

Automatic installation

Troubleshooting

Related pages

Install the AppMon Server

Install Memory Analysis Server

Install agents

Server administration

Set up the AppMon Server

Collectors

This page details installing Collector instances and covers critical information needed before installation. It also covers starting and stopping a Collector, embedded Collector restrictions, and troubleshooting.

What to know before installing collectors

*NIX

- For AppMon 6.3 and later, reworked `init.d` shell scripts allow Collectors (and the Server) to start as non-root or under the desired user account.
- If your Linux uses `systemd` for startup, see the attached `systemd` script samples (wrappers for `init.d`).

Some Linux flavors come with low limits for the maximum number of files, sockets and processes or threads per user. (1024). This normally poses no problem for the AppMon Collector. However, if running several of them within one user context and a couple of dozen Agents are connected to each Collector, raise both settings. Some Linux Operating Systems have limits of 64k, so increase low limits generously to (500 + 3 * numberOfAgentsConnected) * Collectors and be on the safe side for all eventualities, such as a high number of Agent Groups.

- Check limits before setting them too low or lower than they were, which could harm your system.
- Check hard limits (and set them) to account for desired soft limits, as soft limits cannot exceed hard limits. Permissions are required.

Set limits permanently for daemons in `/etc/security/limits.conf`. See an explanation in that file for more information.

Per User	Check hard limit	Check soft limit	Set soft limit on the fly
files / sockets	`ulimit -Hn`	`ulimit -n`	`ulimit -n <numberToSet>`
processes / threads	`ulimit -Hu`	`ulimit -u`	`ulimit -u <numberToSet>`

You can force a check for 2048 at Collector-startup with the `-openfiles` parameter, but be aware that several run under the same user.

- The installation requires Java Runtime Environment (JRE) 1.6 or later. Because the UNIX installer does not include the private JRE for Collector run time (Windows and Linux do), you should update your system's public JRE to 1.8 first, as that is required for the Collector run time.

All platforms

- A Collector should always be placed close (in terms of latency) to the connected Agents. There are thousands of start-up round-trips between them to instrument application classes. A Collector should be placed in the same sub-net, or at least in the same data center as the Agents and instrumented applications. This allows connecting to the AppMon Server over the WAN if needed.

- The number of Collectors is not the exclusive sizing factor, but also JVM heap memory and the number of cores (and naturally disk space) available to a Collector process determines the number of Agents it can take. Also, Collectors need a lot more heap and disk space (for class cache and inheritance map) if the instrumented applications are different, so applications should be hooked to Collector groups according to their similarity. Sensible grouping of apps and equally sized Collectors in a group are also prerequisite for Collector - Agent load balancing. See Collector Best Practices to adapt settings when your requirements change or ask sales or support.

Install collectors

Windows

Download the full Windows installation package for your architecture. Do a custom installation if you only need the Collector.

The setup defaults to an installation path (%DT_HOME%) of `%ProgramFiles%\dynaTrace\dynaTrace 7.0`.

Installation creates Start menu entries and registers an auto-starting Windows service, configured to connect to an AppMon Server on the same machine (localhost) on the default port (6698 or 6699 (SSL)) and name (AppMon Collector 7.0).

*NIX

> **Do not run the AppMon Collector as root**
>
> Run the installation using a designated account. For AppMon 6.3 and later, you can define `DT_RUNASUSER` in the script.

Download the AppMon Collector installation JAR file.
The Linux Collector installation file is named `dynatrace-collector-<version>-linux-x86.jar`, implying 32-bit on the x86 platform if you look at the other namings, but contains both bitnesses.

On Linux, the installer prompts for bitness (32-bit or 64-bit) and for the installation directory, which defaults to `<currentDirectory>/dynatrace-collector-7.0`. This path is referred to as `<DT_HOME>`.

If installing on Solaris, HP-UX, AIX, z/Linux, or z/OS, the installer prompts you for `JAVA_HOME`, which specifies the Java installation used to run the AppMon Collector. The Linux version comes with a private JVM.

The recommended installation is in `/opt`. Start the installation by issuing:

```
cd /opt
sudo java -jar <path>/dynatrace-collector-<version>-<platform>.jar
sudo chown -R <DT_RUNASUSER>:<DT_USER'SGROUP> /opt/dynatrace-<version>
```

Copy the `dynaTraceCollector` shell script to `/etc/init.d`.
Define DT_RUNASUSER in the script. You can also define other variables like DT_OPTARGS and DT_INSTANCE if necessary.
Add this script to the system boot by using `chkconfig --add dynaTraceCollector` or, depending on your distribution, such as on Debian or Ubuntu, use `sudo update-rc.d dynaTraceCollectorNonRoot defaults`.

> **Note**
>
> If you do not specify a `DT_RUNASUSER`, then register the script and restart, the Collector starts in the root. This means `<DT_HOME>/collector/osgi` and `temp` belong to the root. To correct this, rerun the above `chown` command.

Check if `dtcollector` not work on your Linux flavor, try `ps aux|grep dtcollector`. The first, short output is for the launcher and the second, long one is for the Collector. The third one is for `grep`. The user the Collector runs under is at the very beginning, before the PID.

Install extra collector instances

All operating systems

The Collector names (Windows: service names) and the port each Collector listens for Agents must be unique.

If you want your Collector instance to connect to a specific Server other than `localhost`, add `-server <dynaTraceServerName>` to the registration (Windows) or shell script (Linux/UNIX).

The first start of the new Collector instance creates the directory structure `<DT_HOME>\collector\instances\<CollectorInstanceName>` that replicates the structure of the default instance and copies the files from there.

Windows

This sample code shows how to register and start a new service called `Dynatrace Collector 7.0 (MyCollector)` that listens for Agent connections on port 9997:

```
CD %DT_HOME%
dtcollector -service install -instance MyCollector -listen :9997
sc start "Dynatrace Collector  (MyCollector)"
```

To register an extra Collector service instance that listens for Agents on port 9997 and connects to `myAppMonServer` (on a different machine):

```
dtcollector -service install -instance MyCollector -listen :9997 -server myAppMonServer
```

To un-register a Collector instance, for example (to possibly re-register it with new arguments):

```
sc stop "Dynatrace Collector 7.0 (MyCollector)"
sc delete "Dynatrace Collector 7.0 (MyCollector)"
```

or

```
%DT_HOME%/dtcollector -service uninstall -instance MyCollector
```

Linux and Unix

The following sample code prepares and starts a Collector instance `collector02` that listens for Agent connections on port 10001 and connects to a Server on the same machine on the default port (6698).

```
cd $DT_HOME
./dtcollector -instance collector02 -listen :10001
```

To start the `collector02` instance at system startup, copy the `$DT_HOME/init.d/dynaTraceCollector` shell script, for example using **Save As** to create the file `/etc/init.d/dynaTraceCollector02`. Make the following minimum changes:

```
DT_OPTARGS="-instance collector02"
DT_INSTANCE=collector02
```

Be sure the `<instanceName>` in `DT_INSTANCE` and `DT_OPTARGS="-instance <instanceName>"` are the same.
Add the script to the startup items by using `/sbin/chkconfig --add dynaTraceCollector02` (or the chkconfig equivalent on your distribution).
Check for running Collectors by using `ps ax | grep dtcollector`.

Installed directories

This table lists directories of special interest created when installing a Collector.

Relative path	Description
`DT_HOME`	The executables and the main configuration file `dtcollector.ini`.
`DT_HOME\log\collector`	Log files.
`DT_HOME\collector\cache`	Class cache directory.
`DT_HOME\collector\conf`	Main configuration directory.

Collector fail-Over / Agent connection balancing

See Collectors > Collector Groups for Connection-Balancing and Fail-Over for details.

Embedded collector usage restrictions

The AppMon Server includes an embedded (as opposed to stand-alone) Collector that should only be used for testing purposes. The embedded Collector is started in-process with the Server, a stand-alone Collector service / daemon must be explicitly started.
Enable the embedded Collector in the Client by selecting **Settings > Dynatrace Server > Services** and selecting the **Allow Agent Connections to Dynatrace Server** check box under Embedded Collector Settings. You must restart the AppMon Server after changing this setting.

The table indicates whether the embedded Collector can (

) be used by different AppMon editions in different (license) scenarios.

Edition / Installation type	Production	Pre-production (aka Test Center)	Dev team (discontinued)
POC			
Evaluation			
Perpetual			
Term			

Start and stop a collector

Windows

On Windows, you can control the AppMon Collector via the **Start** menu, command line, or Services control panel. The registered service name is **Dynatrace Collector <version> [(<instanceName>)]**.

For example, to start the Collector, execute:

```
sc start "Dynatrace Collector 7.0"
```

To configure the service to start automatically on system startup, execute:

```
sc config "Dynatrace Collector 7.0" start= auto
```

There is no space between `start` and `=`, but there must be a space after `start=`.

Alternatively, you can start the Collector from the command line:

```
cd %DT_HOME%
dtcollector.exe -service start
```

See Extra Instances on Windows to start or stop more than one Collector instance on one machine.

*NIX

For testing purposes, you can interactively start the AppMon Collector in the background (`-bg`) by using the launcher `dtcollector`. The following commands start a Collector that listens on `localhost` for Agents on default port 9998 and connects to an AppMon Server via default port 6698:

```
cd $DT_HOME
./dtcollector -bg
```

You can use `<DT_HOME>/init.d/dynaTraceCollector start | stop | status` to stop or check the Collector without caring about its pid.

For normal operations, you should use the `<DT_HOME>/init.d/dynaTraceCollector` script that starts the Collector on the desired run levels at boot time. For Linux, you can use the script without changes: Copy it to `/etc/init.d` and add it to boot time with `chkconfig --add dynaTraceCollector`.

To detect any running AppMon Collector instance, the `init` script checks for a file `dtcollector.pid` located in `/var/run`.

If an instance name is being used, the corresponding `.pid` file is called: `/var/run/dtcollector.<instance>.pid`. If the `.pid` file is not present for some reason, such as missing write permissions to `/var/run`, the script tries to find the process ID by grepping the `ps` output for running `dtcollector` instances. You may need to change `ps` and `grep` arguments to get the right Collector instance, which might be tricky when an instance name is being used. Therefore, it is strongly recommended to configure the system so that `.pid` files can be used.

dTCollector arguments

`dtcollector -?` renders the supported arguments:

```
dynaTrace Collector
Version: ... - built ...
Copyright (C) 2004-20.., dynaTrace software GmbH

Usage: dtcollector [options]

Available options:
   -?, -h, --help Print this help message
   -v             Display version information
   -loglevel [0..8]
                  Set loglevel, valid values are: [0..8] or
                  [finest|finer|fine|config|info|warning|severe|debug|none]
                  Default setting: 4 (info)
   -basedir <dir> Set working directory for application
   -ini <config>  ini file to use, default is dtcollector.ini
                  A value of 'ignore' will skip reading the ini file.
   -instance <name>
                  Instance name for this dynaTrace Collector instance
   -listen [<ip>:]<port>
                  Listen for dynaTrace Agents on [<ip>:]<port>
   -server <host>[:<port>]
                  Connect to dynaTrace Server at <host>[:<port>]
   -service [install | uninstall | start | stop | list(all)]
                  Install/uninstall Windows Service
                  (uninstall might require you to log off)
   -timeout <sec> Timeout value for watchdog feature, 0 turns the watchdog off.
                  Default value: 180, valid value range [30..86400] sec
   -portrange <n>[:<m>]
```

```
                Port range to be used by watchdog feature
                By default a port within range 50000:65535 is used
-vm <javaVM>    The Java VM to be used for launching the application
-startup <startup.jar>
                The startup jar to execute. The default is to find the
                plugins/org.eclipse.equinox.launcher jar with the highest
                version number.
-ea[:<package>"..."|:<class>]
                Enable assertions (assertions are disabled by default).
                A <package> or <class> name can be specified (cf. Java docs)
-osgicleanonrestart
                Clean OSGI cache on restart (set system property
                osgi.clean=true if application gets restarted)
-Dosgi.\*=<value>
                Set arbitrary OSGI configuration parameters
-D\*=<value>     Set properties for Java application
-X\*=<value>     Set (nonstandard) options for Java VM
```

Automatic installation

You can automate the installation with Ansible, Chef, or Puppet. Click the link below for more detailed information.

- Dynatrace-Ansible
- Dynatrace-Chef
- Dynatrace-Puppet

Troubleshooting

Collector load

One particular scenario might overload the Collector: If you collectively restart your IIS instances, you should not have more than 50 Web Server Agents (depending on configuration) connected to one Collector. With normal operations, you may be able to connect more Web Server Agents to the same Collector.

Slow application startup

If you experience high application startup time, chances are that the network latency between the Agents and the Collector they connect to is too high and the number of round-trips add up. Move the Collector closer to its Agents, avoiding firewalls in between if possible, or at least use a very fast one.

Collector class cache and inheritance map sizes

Do not underestimate the impact of Collector class cache and inheritance map sizes. Despite recurrent automatic cleanup (by default every week), they can get quite big quickly. You can set cleanup recurrence with (maximum age in milliseconds; below: one day):

```
-Dcom.dynatrace.diagnostics.imap.cleanupItemAge=86400000
```

For details see KB-484 - How to fix a class cache explosion.

Install Memory Analysis Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What to know before you install

Install the Memory Analysis Server

Starting and stopping the Memory Analysis Server

Additional considerations

This page describes how to optionally install a Memory Analysis Server if it is part of your deployment.

The AppMon Memory Analysis Server (formerly Analysis Server) is used solely for post-processing memory snapshots. The name of this component has been changed to accurately reflect its purpose and prevent confusion. It should not be confused with the AppMon Frontend Server, which serves other analysis data to the AppMon Client.

What to know before you install

- The Memory Analysis Server can be installed with all the other components through the full installation Windows `.msi` file, or separately with a custom installation on Windows or the `dynatrace-analysisserver-<version><platform>.jar` installers for the Memory Analysis Server on Linux and UNIX. See the Deployment Guide for recommendations for the type of installation you should use.
- One Memory Analysis Server can be shared by multiple AppMon Server instances. If the Servers are installed on different machines, a high-speed network connection between the Memory Analysis Server and AppMon Server is required.

Note

The *NIX AppMon **installer** requires an Oracle / Sun or IBM JVM 1.6 or later on the target machine to run the installer process. On Linux (and Windows) AppMon components use a private JVM 1.7 at runtime.

The Linux Memory Analysis Server installation file is named `dynatrace-analysisserver-<version>-linux-x86.jar`, implying 32-bit on the x86 platform if you look at the other namings, but contains both bitnesses.

- Because memory snapshot analysis can require a considerable amount of memory, it should be done using this component. The Memory Analysis Server is automatically used for post-processing if it is running. If you are not using a production installation and the Memory Analysis Server is not running, the snapshot is post-processed by the AppMon Server. For production installations, a Memory Analysis Server is mandatory.

The default configuration expects the Memory Analysis Server to run on the same machine as the AppMon Server. If another machine is used, you must configure the location in the Services pane of the AppMon Server Settings dialog box. If you do not post-process memory snapshots regularly, you can start the AppMon Memory Analysis Server on demand.

Install the Memory Analysis Server

Directory layout

By default, the Memory Analysis Server installs into the `dynaTrace <version>` directory on Windows or `dynatrace-<version>` on Linux or UNIX. This directory is referred to as `DT_HOME`. The following table shows the location of relevant files.

Relative path	Description
`DT_HOME`	The executables and the main configuration file `dtanalysisserver.ini`.
`DT_HOME\log\analysisserver`	Log files.

Memory requirements

The amount of memory required for post-processing leak analysis memory snapshots mainly depends on the number of objects on the heap. For fully cached post-processing, required memory equals the heap size of the application for which the snapshot was taken. Although post-processing with less memory is possible, running with less than full-cache memory is not recommended.

Windows

If the Memory Analysis Server is not already installed, download the full installation package and run a custom installation.

The default location is `%ProgramFiles%\dynatrace\dynaTrace <version>`. The installer creates Windows **Start** menu entries and registers a Windows service for the Memory Analysis Server process.

Linux

If the Memory Analysis Server was not installed with the full package, download the specialized Linux installation JAR file. The `x86` in the file name refers to the platform. You will be prompted to specify 32-bit or 64-bit for the installation.

To install, you need a current Java Runtime Environment (JRE 1.6 or later) installed on your system. The Memory Analysis Server runs with a private JRE 1.7 by default.
It is recommended that you run the AppMon Memory Analysis Server not as root user, but with a dedicated account. To avoid directory permission issues, create the dedicated user and the target directory before beginning the installation, and run the installation in the user context. The installer prompts for 32-bit or 64-bit and for the installation directory. The default location is the current directory.

Execute the following command:
`java -jar dynatrace-analysisserver-7.0.0-linux-x86.jar`

Configure memory

Before starting the AppMon Memory Analysis Server, adjust the memory settings in the main configuration file `dtanalysisserver.ini`. Set the option `-Xmx` to the maximum heap size of applications + 25% to analyze.

The following is an example of the code to add to `dtanalysisserver.ini`.

```
...
-Xmx2048M
-Xms128M
-XX:MaxPermSize=128M
-XX:PermSize=128M
...
```

The `-XX:+UseCompressedOops` automatically appends when you use the 64-bit installer.

Set up Memory Analysis Server communication

The Memory Analysis Server listens on a specified port for incoming connections from the AppMon Server. To change the default port number, add `-Dcom.dynatrace.diagnostics.listen=[<ip>]:<port>` to the `dtanalysisserver.ini` file.

Starting and stopping the Memory Analysis Server

Windows

On Windows, you can start the Memory Analysis Server service from the **Start** menu, the command line, or the **Services** control panel. The registered service name is `Dynatrace Memory Analysis Server <version>`. The service runs with restricted permissions as Local Service.

> **Note**
>
> If you change any output directories (such as the session directory), make sure the AppMon Server service has sufficient privileges in the new directory.

To start the Memory Analysis Server from the command line, change to the `DT_HOME` and execute:

```
dtanalysisserver.exe -service start
```

To list all the available options, execute `dtanalysisserver.exe -?`. This displays the following:

```
dynaTrace Analysis Server
Version: 6.3.0.1305 - built 2016-02-22 19:07:10 +0100
Copyright (C) 2004-2017, Dynatrace
Usage: dtanalysisserver [options]

Available options:
 -?, -h, --help Print this help message
 -v Display version information
 -loglevel [0..8]
 Set loglevel, valid values are: [0..8] or
 [finest|finer|fine|config|info|warning|severe|debug|none]
 Default setting: 4 (info)
 -listen [<ip>:]<port>
 Listen for dynaTrace Server instances on [<ip>:]<port>
 -service [install | uninstall | start | stop | list(all)]
 Install/uninstall Windows Service
 (uninstall might require you to log off)
 -basedir <dir> Set working directory for application
 -logdir <dir> Override default log directory (command line argument only)
 -ini <config> ini file to use, default is dtanalysisserver.ini
 A value of 'ignore' will skip reading the ini file.
 -timeout <sec> Timeout value for watchdog feature, 0 turns the watchdog off.
 Default value: 180, valid value range [30..86400] sec
 -watchdogaddress 127.0.0.1
 IP address to be used for watchdog feature
 -portrange <n>[:<m>]
 Port range to be used by watchdog feature
 By default a port within range 50000:65535 is used
 -vm <javaVM> The Java VM to be used for launching the application
 -startup <startup.jar>
 The startup jar to execute. The default is to find the
 plugins/org.eclipse.equinox.launcher jar with the highest
 version number.
 -ea[:<package>"..."|:<class>]
 Enable assertions (assertions are disabled by default).
 A <package> or <class> name can be specified (cf. Java docs)
 -Dosgi.\*=<value>
 Set arbitrary OSGI configuration parameters
 -D*=<value> Set properties for Java application
 -X*=<value> Set (non standard) options for Java VM
```

Linux Daemon

The syntax for the Memory Analysis Server is:

```
<DT_HOME>/init.d/dynaTraceAnalysis start | stop | version | status
```

> **Note**
>
> Do not log-rotate the Memory Analysis Server's log files. This will be done automatically by the Server.

Additional considerations

Automatic installation

You can automate the installation with Ansible, Chef, or Puppet. Click the link below for more detailed information.

- Dynatrace-Ansible
- Dynatrace-Chef
- Dynatrace-Puppet

Parallel startup of Server and Analysis Server

If you have the AppMon Server and Memory Analysis Server installed on the same machine (sharing the same core executables), it is not possible to start both servers at the same time, for example using a Linux shell script.

In this case, there are these possible solutions:

- Install the Memory Analysis Server standalone on the machine.

- In your shell script, first start the Memory Analysis Server, wait a short time (10 seconds usually suffices), and then start the AppMon Server.

Install agents

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Download and install agents

Automatic installation

Related pages

Agents

OneAgent

Java Agent

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

Watch and learn

Managing agents

This page provides a generic description of common items in Agent installation and configuration.

See Related Pages for information on configuring specific Agents.

The AppMon Agent is a shared library that needs to be injected into the application process.
For most Agents there is a bootstrap part that takes care of versioning.

Download and install agents

Notes

- Documentation typically refers to the AppMon installation directory as DT_HOME (or <DT_HOME>) see more detailed description.

- With a full install, `DT_HOME` defaults to `%ProgramFiles%\dynaTrace\Dynatrace <major.minorVersion>` on Windows and `/opt/dynatrace-<major.minorVersion>` on *NIX if you `cd /opt` before installation as recommended.
- With an Agent-only installer on Windows, `DT_HOME` defaults to `%ProgramFiles%\dynaTrace\dynaTrace Agent <major.minorVersion>`.
- With a *NIX shell script or `.tar`, you need to create a directory by entering `md` then enter `cd` to go the desired directory before running / unpacking it.

The actual bootstrap and core Agent parts are installed to `DT_HOME/agent/lib[64]` and updates to the core are in `DT_HOME/agent/downloads/<version>`. `.ini` files and shared memory are in `DT_HOME/agent/conf`. `DT_HOME/log` is common to all components, which is one reason that it is important to keep the `DT_HOME` folder structure and permissions.

Windows

Download the AppMon Agent Windows installation package of choice for your architecture (x86 or x64) and run it. In the installer select the applicable package for your application environment.

Linux

Download the appropriate AppMon Agent Linux installation package (`.jar` or `.tar` file) for your architecture (x86, x64, PPC) and instrumentation needs.

The `.jar`-based AppMon Agent installer requires a JRE 1.4 or later. Some Agent installers such as those for Web Server and PHP Agents come as `.tar` shell scripts and do not require a JRE, as the `.sh` has the Agents embedded. The `.jar` file asks for the install directory, but the `.sh` does not. Both default to the current directory.

> If you use SELinux, see SELinux and Dynatrace Product Issues on Redhat.

Use the following to Install `.jar`-based Agents:

```
cd /opt
sudo java -jar <pathToJar>/dynatrace-agent-<version>-unix.jar
```

Automation Downloads

Details on Downloads for automated agent integration.

Other platforms

Download the AppMon Agent installation package (`.jar` file or shell script) for your platform and install as described above.

OneAgent

To use OneAgent in AppMon 2017 May you need to install Classic Agents first, and then roll them out to the OneAgent.

Automatic installation

You can automate the installation with Ansible, Chef, or Puppet. Click the link below for more detailed information.

- Dynatrace-Ansible
- Dynatrace-Chef
- Dynatrace-Puppet

Automation Downloads

On this page

General

Agents

Related pages

Agent

OneAgent

Install agents

Download services for automated AppMon Agent deployment.

General

- These services will be available for the support lifetime of the respective product version.
- Use only secure URLs (https) and don't disable certificate verification

Agents

The base URL format is: https://downloads.dynatrace.com/clientservices/agent?version=APPMON_VERSION&os=OS&arch=ARCH&techtype=TECHTYPE

Certain combinations will not work because they are unsupported.

Example

https://downloads.dynatrace.com/clientservices/agent?version=6.5.0.1289&os=windows&arch=x64&techtype=java_net

Parameter ‘Version’

The full four component build version number of the desired **installer** containing the **Agent bootstrapper**. Agents will auto-update to newer versions of the same release (e.g. 6.5.1.1013) when that update is rolled out on the AppMon Server. So this version should always stay the same unless you upgrade to a new release.

Parameter ‘os’

- windows
- linux
- solaris
- aix
- zlinux
- zos

Parameter ‘arch’

- x64 (Windows)
- x32 (Windows)
- x86 (Unix-based platforms)
- ppc32
- ppc64
- SPARC32
- SPARC64
- p5_32
- p5_64
- s390 (z/Linux, z/OS)
- s390x (z/Linux, z/OS)

Parameter ‘techtype’

- java
- java_net (IIS)
- nodejs
- webserver (PHP)
- cics (CICS, IMS)

Automatic installation

Related pages

Install the AppMon Server

Install agents

Install collectors

Install Memory Analysis Server

Additional install considerations

You can automate the installation with Ansible, Chef, or Puppet. Click the link below for more detailed information.

- Dynatrace-Ansible
- Dynatrace-Chef
- Dynatrace-Puppet

Compatibility table

Component	Ansible	Chef	Puppet
Server	✓	✓	✓
Agents	✓	✓	✓
Collector	✓	✓	✓
Memory Analysis Server	✗	✓	✓
AppMon Client	✓	✗	✗
License	✗	✓	✓

Additional install considerations

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Access to web services

Breaking changes and potential pitfalls

Side-by-side component installation

Silent installation

This page contains links to other topics and optional scenarios to consider when planning your AppMon deployment.

.

Side-by-side component installation

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Side-by-side installation

Component installation

Related pages

Silent installation

Breaking changes and potential pitfalls

Access to web services

Overview

Side-by-side installation usually means having different versions of the same application on the same operating system or virtual machine coexisting in different directory structures. See below for the exact rules applying to AppMon.

Similar:
Many applications, such as text editors and spreadsheets, consist only of one component, or at least appear as such. However, database management systems (DBMS) already consist of at least two perceptible components – back and front end.

The AppMon Server, Collector, Agent, and Clients components need to work together. It is recommended that you use the very same version and build number for all components. Dynatrace puts great effort into ensuring compatibility across versions in upgrade situations. Build- and revision-level update distribution is handled automatically by default. AppMon build-level updates are automatically propagated from the Server to the Clients and Collectors and from the Collectors to all Agents except non-bootstrap Agents which are denied update at Client startup. Other updates apply after component and application restart.

Different:
Most applications need to be shut down to be updated or patched in the same directory structure through an installer. However, AppMon updates are applied through the Client to a *running Server*. For the changes to be applied you must restart components and any applications with an injected Agent with a pending update. See Updates for more information.

*NIX

You can install whole AppMon suites or single components as needed using `java -jar <path>/<installer>.jar`.

Windows

Because the Windows registry requires GUIDs for installations, there are limitations to what can be done. This is also why there are instances, so you can run several Collectors on the same Windows machine if configured with different ports.

Side-by-side installation

You can install different versions of AppMon (major and minor version) side-by-side in different directory structures because they have different GUIDs.

Any configurations and data you want to keep must be migrated, Careful configuration is needed. See the Upgrade and Migration Guide to learn more.

Component installation

- Use the 64-bit full installation package. This determines if the Server or Collector run with a 32-bit or 64-bit JRE, but you still get the other-bitness Agent bootstrappers and DLLs to instrument any-bitness applications.
- If you already installed a component with a special installer such as the Agent, and you want to add a component where there is no special installer for Windows, then uninstall the special installation and install all the desired components with a custom install of the full Windows `.msi`. If you then want to add or remove components, just run the full installer again and modify the installation.

> **Example:**
>
> You installed a Java Agent with the All AppMon Agents for Windows package. To add a Collector on the same machine, uninstall the Agent package and do a custom install with the full installer for the Java Agent and the Collector.
>
> To add a .NET Agent, run the full installer again and modify the installation.

The following matrix shows which installer packages of the same release can and cannot be installed when the other is already on the Windows machine.

	Full x-64	Collector	Collector x-64	Agent	Client	Client x-64
Full x-64		s	x	a	x	x
Collector	s		s	✓	✓	✓
Collector x-64	x	s		✓	✓	✓
Agent	a	✓	✓		✓	✓
Client	x	✓	✓	✓		x
Client x-64	x	✓	✓	✓	x	

Legend:
s = conflicts with service
a = conflicts with agents
x = not applicable

Silent installation

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Windows

*NIX

Related pages

Side-by-side component installation

Breaking changes and potential pitfalls

Access to web services

Windows

64-bit client- and JRE-only installation to the c:\Dynatrace directory: `msiexec /passive /i C:\tmp\dynatrace-full-7.0.0.1289-x86-64.msi APPDIR="c:\dynaTrace" ADDLOCAL="DiagnosticsClient,JRE"`

Generic Command Syntax:
`msiexec /passive /i <path_to_msi\msi_file_name> APPDIR="<target_folder>" ADDLOCAL="<feature>"`

Substitute appropriate values for fields in angle brackets. `<feature>` is a comma-separated list of AppMon features to be installed. Useful combinations are:

Type of installation	Feature
AppMon Server	`DiagnosticsServer,WindowsService,JRE`

AppMon Collector	`DiagnosticsCollector,CollectorService,JRE`
AppMon Analysis Server	`DiagnosticsServer,AnalysisService,JRE`
AppMon Client	`DiagnosticsClient,JRE`
32-bit and 64-bit AppMon Agents using 32-bit installer or Agent installer	`DiagnosticsAgent,JavaAgent,JavaAgentx64,DotNetAgent,DotNet FFAgent,IEAgent,WebServerAgent,IIS7Agent, IIS7Agentx64,Apache20Agent,Apache20Agentx64,Apache22Agent, Apache22Agentx64,Apache24Agent,Apache24Agentx64,WSMB,HostA zOSAgent`
32-bit and 64-bit AppMon Agents using 64-bit installer	`DiagnosticsAgent,JavaAgent,JavaAgentx86,DotNetAgent,DotNet FFAgent,IEAgent,WebServerAgent,IIS7Agent, IIS7Agentx86,Apache20Agent,Apache20Agentx86,Apache22Agent, Apache22Agentx86,Apache24Agent,Apache24Agentx86,WSMB,HostA zOSAgent`
`JRE only`	`JRE`

> **Note**
>
> The AppMon Server, Collector and Client cannot be installed without an AppMon-private JRE.

For more information about Windows installer properties, see ADDLOCAL Property and Property Reference on MSDN.

*NIX

To perform an unattended installation, use the command line parameters of the install command.

Passing `-h` provides a list of available command line parameters.

Linux

Install the full 64-bit AppMon to `/opt/dynatrace/7.0` (not to the default, but to a separate version-directory):

```
java -jar dynatrace-7.0.0.1289-linux-x64.jar -t /opt/dynatrace/7.0 -y
```

Install the 32-bit AppMon Collector into the default directory:

```
java -jar dynatrace-collector-7.0.0.1289-linux-x86.jar -b 32 -y
```

Unix

Install the 32-bit AppMon Collector into the default directory and use the already installed Java to run AppMon:

```
java -jar dynatrace-collector-7.0.0.1289-unix.jar -b 32 -y
```

Access to web services

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Licensing

Client administration

Updates

The AppMon Client uses a set of web services that can be granted access through firewall and network configuration. All mentioned services use HTTPS with default ports.

Update notification

Shows push notification in the client for new Updates. Provides access to the following hosts:

- downloads.dynatrace.com
- files.dynatrace.com

Documentation and help

The help buttons in the AppMon Client direct to:

- www.dynatrace.com

Agent download

Download the Add Tier wizard for remote Agent installation references:

- downloads.dynatrace.com

Licensing

To activate and import licenses without the need to access files:

- eservices.dynatrace.com

Breaking changes and potential pitfalls

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

AppMon 6.3 and later

This page lists breaking changes and different defaults or behavior between versions that could pose issues.
Please use the doc feedback link on this page to make us aware of other issues you may encounter.

AppMon 6.3 and later

Accept authentication data only via HTTPS

Selecting this AppMon Client configuration check box in **Settings > Dynatrace Server > Services > Management** is a potential pitfall, especially for REST requests. See Services > Management Tab for more information.

Challenges with authentication, certificates and keys

An easy-to-use interface in the AppMon Client (**Settings > Dynatrace Server > Certificate Management > Manage**, then click **Start Wizard** under **Import and Deploy**]) lets you easily import self-signed or well-trusted certificates. However, dealing with secure communications has some potential pitfalls and means additional work.

- If you use a self-signed certificate, you must accept it somewhere. Having a UI to alert you is acceptable, but otherwise you must tell the component about it, even if it is just “ignore it”.
- If it is a well-trusted one, issued by a CA, acceptance should be automatic, however most certificated may be self-signed.

Example in case: the Jenkins plugin

The options for the Jenkins plugin may serve as template for other components.

- The most convenient, but not the best practice is to continue using insecure HTTP.
- Allowing authentication data via unsecured communication lines (by selecting the **Accept authentication data only via HTTPS** check box in **Settings > Dynatrace Server > Services > Management**) is not recommended.
- Ignore the certificate by using the Jenkins Skip Certificate Check Plugin.
- Import the self-signed certificate into the Jenkins key store.
- Use a well-trusted certificate.

Licensing

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Licensing process

License activation and deactivation

License states

Related pages

eServices

License activation and deactivation via file

Online license activation and deactivation

License assignments

License upgrade

Licensing details

Manage UEM visits volume

Manage usage based agent hours volume

Access to web services

License Overview dashlet

Watch and learn

Distributing licenses

Managing licenses

Managing UEM volume

External links

eServices

AppMon not only offers a **full product functionality**[1] **free 30-day trial** for AppMon Application Monitoring and UEM, but also a **free-for-life personal**[2] **license** as a follow-up.

[1] *With full product functionality on five machines.*
[2] *Full product functionality, but Agent connections are restricted to localhost.*

Overview

AppMon requires a license for full product functionality as defined by the license terms. Each AppMon Server requires its own license. License properties include:

- **License Number**
- **Edition**: Production, or Pre-Production. See Editions for details.
- **Type**: POC, Evaluation, Term, Perpetual, or Online.
- **Version**: Licenses are issued with major and minor version identifiers (for example, 6.3, 6.5, 7.0).
- **Issue Date**
- **Expiration Date**
- **Status**: A license is assigned a status to indicate its current state in the licensing workflow.
- **License details**: A license specifies the maximum amount of Agents per Agent type that can connect to the AppMon Server on which the license is installed (for example, 8 Java Agents, 2 Web Server, 4 PHP, 2 Hadoop).
  - **Technology**: The type of the agent.
  - **Licensed Units**: Amount of available license units for the agent type.
  - **Unit**
- **UEM**: Available volume of UEM visits.
- **Comsumption**: Available volume of Agent hours.

License types

Type	Description
POC	Proof of concept license.
Evaluation	Free trial license.
Perpetual	Fully operational license. Perpetual license contains separate agent units for each technology. Once the corresponding Agent is connected, the license unit is consumed. The consumed license units are calculated based on weights. For example one Java agent consumes 1 Java agent unit, one Web server agent consumes 0.6 Web server agent units.
Term	Same as Perpetual type, but with defined expiration date.
Online	Fully operational license. Online license contains overall amount of agent units, regardless of the technology. The consumed license units are calculated based on weights. For example, one Java Agent consumes 1 license unit, and one Node.js Agent consumes 0.1 license unit. With an online license, AppMon connects to eServices daily to verify that the license is still valid and report license units usage. To avoid down times or data loss, the license terminates only if the verifications fail for seven consecutive days. A terminated license can not be reactivated. Unused license units go to your company's pool. You can use them to create a new license.

License statuses

Status	Description
Issued	Only assigned for POC licenses. Disabled on the expiration date.
Open	When a license is created, it's assigned an open status until activated.
Activated	A license is activated when you install it on a Server and apply an activation key. An activated license is bound to the machine on which it was installed.
Deactivated	A license that is deactivated on a machine can be reactivated on a different machine.
Terminated	A terminated license reactivated. It is removed from available licenses list on the eServices My Licensing page. Licenses terminate when they exceed their expiration date. Terminate perpetual licenses manually.

UEM visits volume is created and managed separately from an AppMon license. See Manage UEM Visits Volume for more information.

Licensing process

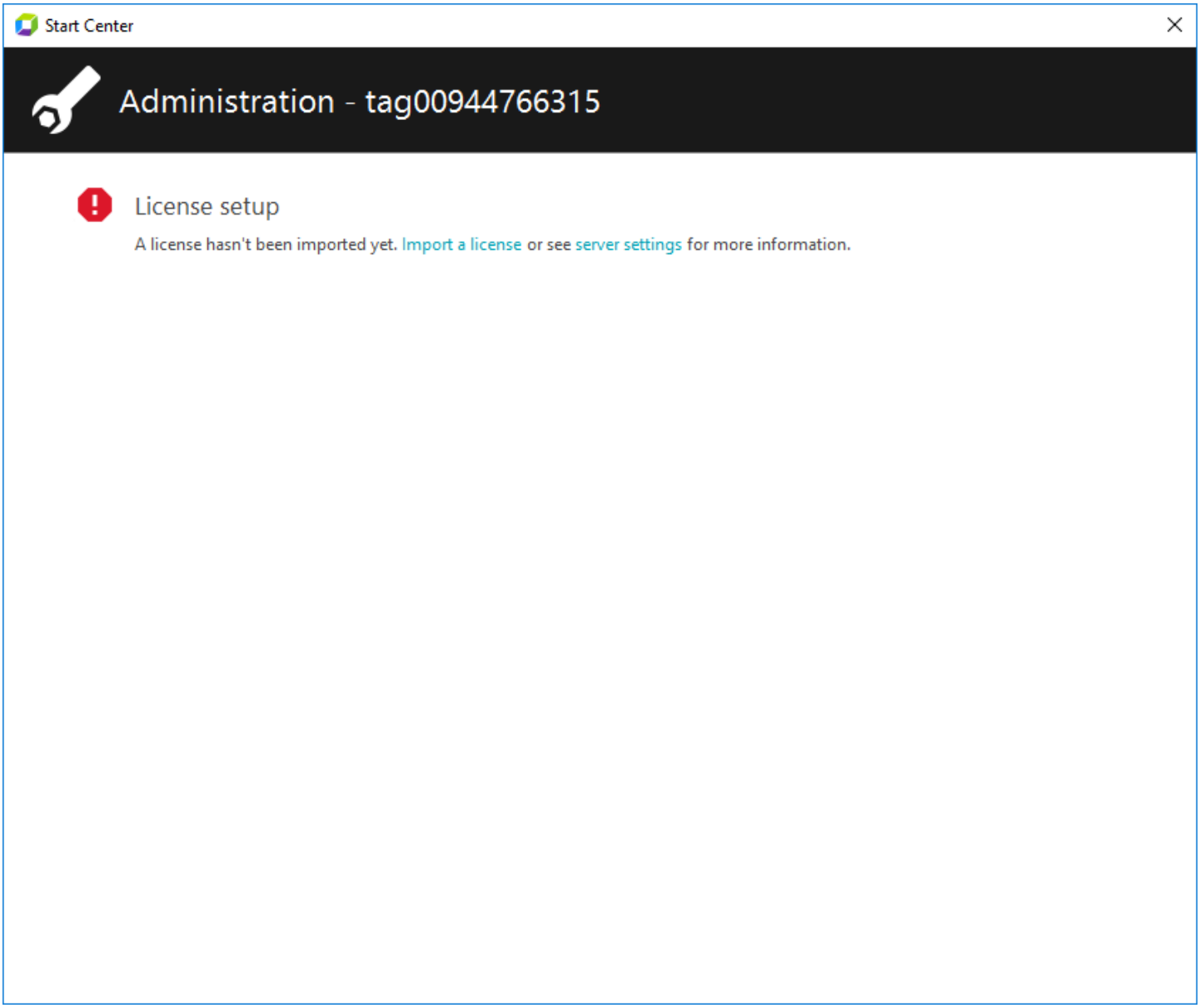
After purchasing AppMon, a welcome email is sent to the account owners. The accounts are associated with the corporate email addresses as specified in the purchase agreement. The email includes the Community user account credentials required to access the eServices licensing website, product installation downloads, and related Community product support content.

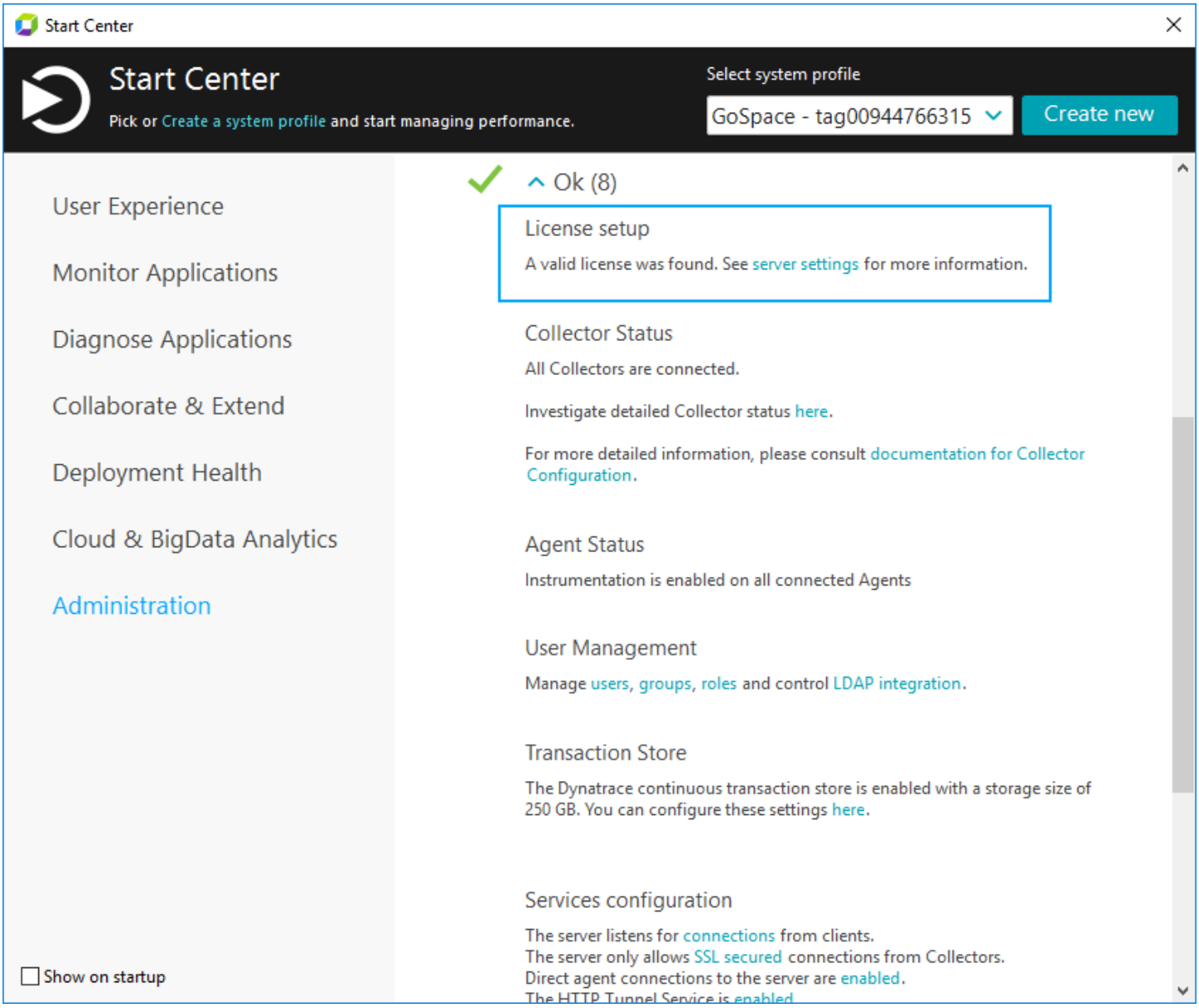
The license management team performs the following tasks:

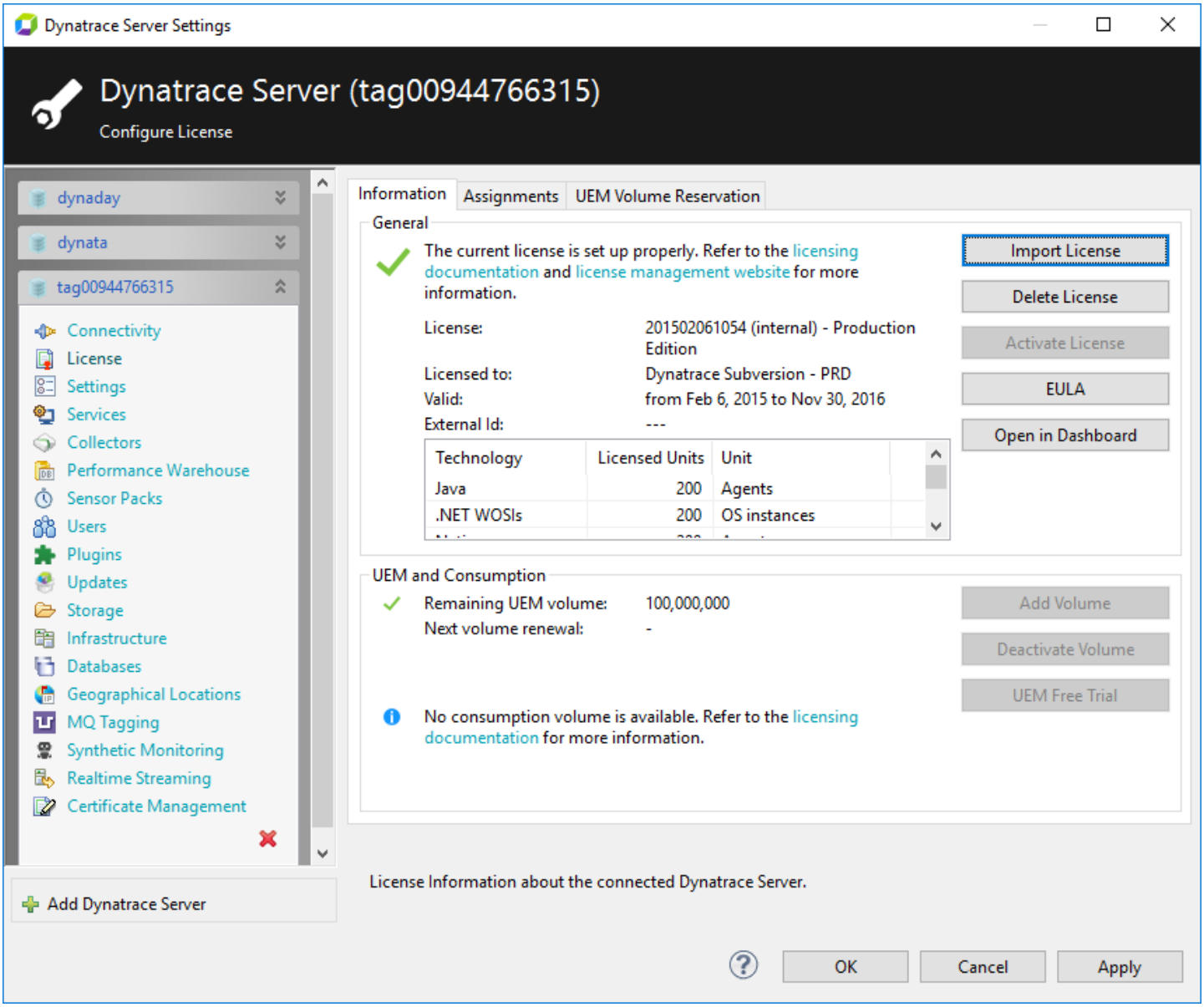
- Creates a license and sends a notification email to account owners with an unactivated license key file. Only license owners can perform license management tasks such as activating, deactivating, or reactivating a license on the eServices website.
- Purchased UEM volume creates a UEM visits volume voucher. Email is sent to account owners to notify that UEM volume can be redeemed and applied to the license. You can then install and activate the license.

  Most eServices website licensing features closely integrate with the AppMon Client. For example, you can activate and install a license directly from the Client to automate downloading and importing a license key file to your AppMon Server. Using the Client simplifies installation and activation and eliminates the need to use the license key file sent in your notification email. Use the AppMon Client to initiate a licensing task whenever possible. See eServices for details about licensing tasks you can perform in the Client and those that you can only perform on the eServices website.

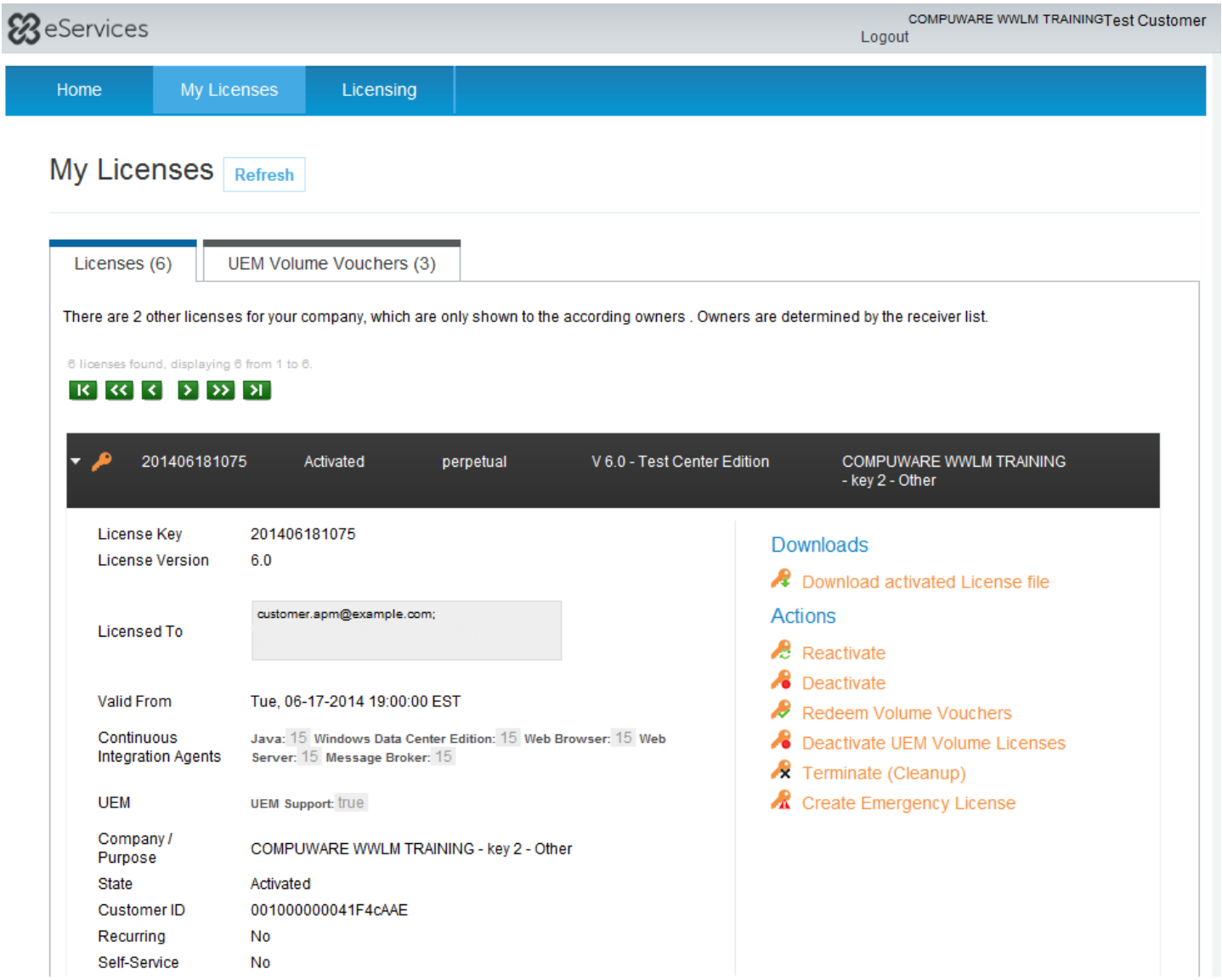
License management in the AppMon Client


Start Center
Administration - tag00944766315
License setup
A license hasn't been imported yet. Import a license or see server settings for more information.


Start Center
Start Center
Pick or Create a system profile and start managing performance.
Select system profile
GoSpace - tag00944766315
Create new
User Experience
Monitor Applications
Diagnose Applications
Collaborate & Extend
Deployment Health
Cloud & BigData Analytics
Administration
Ok (8)
License setup
A valid license was found. See server settings for more information.
Collector Status
All Collectors are connected.
Investigate detailed Collector status here.
For more detailed information, please consult documentation for Collector Configuration.
Agent Status
Instrumentation is enabled on all connected Agents
User Management
Manage users, groups, roles and control LDAP integration.
Transaction Store
The Dynatrace continuous transaction store is enabled with a storage size of 250 GB. You can configure these settings here.
Services configuration
The server listens for connections from clients.
The server only allows SSL secured connections from Collectors.
Direct agent connections to the server are enabled.
Show on startup


Dynatrace Server Settings
Dynatrace Server (tag00944766315)
Configure License
dynaday
dynata
tag00944766315
Connectivity
License
Settings
Services
Collectors
Performance Warehouse
Sensor Packs
Users
Plugins
Updates
Storage
Infrastructure
Databases
Geographical Locations
MQ Tagging
Synthetic Monitoring
Realtime Streaming
Certificate Management
Add Dynatrace Server
Information
Assignments
UEM Volume Reservation
General
The current license is set up properly. Refer to the licensing documentation and license management website for more information.
License: 201502061054 (internal) - Production Edition
Licensed to: Dynatrace Subversion - PRD
Valid: from Feb 6, 2015 to Nov 30, 2016
External Id: ---
Technology
Licensed Units
Unit
Java 200 Agents
.NET WOSIs 200 OS instances
Import License
Delete License
Activate License
EULA
Open in Dashboard
UEM and Consumption
Remaining UEM volume: 100,000,000
Next volume renewal: -
Add Volume
Deactivate Volume
UEM Free Trial
No consumption volume is available. Refer to the licensing documentation for more information.
License Information about the connected Dynatrace Server.
OK
Cancel
Apply

License Management in eServices



License activation and deactivation

An activated license is the fingerprint that uniquely identifies and links a license to a single AppMon Server. The activated license only works on the machine that matches the fingerprint. There are two different methods for activating, deactivating, or reactivating a license:

- **Online method**: The licensing task initiates in the AppMon Client. For example, activating a license using the Client automates downloading and importing the license file from eServices. An Internet connection is required to access eServices from the Client. See Online license activation and deactivation for details for more information.

- **Manual method**: Requires obtaining a license key file or accessing the eServices website directly to perform a licensing task. For example, activating a license manually requires downloading the license key file from the eServices website or using the emailed license key file, and then manually importing the license file using the Client. See License activation and deactivation via file for more information.

License states

No license: No license has been imported. This is the initial state of the license at the AppMon Server.

Not activated: The current license hasn't been activated yet. Depending on the overall validity period and license type, a new license works for up to 3 days without activation. After that period the license must be activated.

Deactivated: The license has been deactivated. Any associated UEM volume has been deactivated as well. The license can now be activated on another server. Depending on the overall validity period and license type, a deactivated license continues to work for up to 3 days. After that period another valid license is required.

Expired: The current license has expired and cannot be used. A new license must be imported to continue using AppMon.

Issued future: The validity period of the current license has not yet begun.

Lock violation: The license has been activated on another machine or some activation-specific information has changed. Usually a license should be deactivated before updating a server or moving to another machine. It's possible to re-activate a license in eServices. If an UEM volume has been added to the license please contact Dynatrace support to get a new volume voucher.

Version conflict: The version of the current license isn't compatible with the version of the AppMon Server. The license can be upgraded in eServices or another valid license can be imported.

Invalid license: The current license is not valid. A valid license needs to be imported to work with AppMon. Please contact Dynatrace support.

Free trial local only: Currently a free-trial license is in use. The license has expired and will continue to work for local agent connections.

Consumption low/exhausted: The consumption volume (agent hours) is running low, or has already been exhausted.

eServices

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

eServices for partners

eServices profile management

eServices manage units, hours and visits

eServices view license information

The eServices license management website is the central platform for viewing AppMon licensing information, performing licensing management tasks, and managing your eServices account profile. Most eServices website licensing features are closely integrated and accessible from the AppMon Client. For example, you can activate a license directly from the Client to automate the process of downloading and importing a license file. As a result, the recommended method for performing any supported licensing task is to use the AppMon Client to initiate the licensing task whenever possible.

The following features are available on eServices:

- Manage your eService profile.
- View licenses assigned to you.
- Manage licenses for your company.

Note

The licensing features of the Client require Internet access. If you are unable to connect to eServices from the Client, you may need to configure a proxy server. See Configuring a Proxy Server for more information.

Not all licensing tasks can be initiated from the Client. For example, the following tasks must performed on the eServices website:

- Upgrading a license.
- Creating an emergency license.
- Reactivating a license in the event of an unexpected hardware failure or forced move of the AppMon Server.
- Viewing information about your AppMon licenses.
- Managing your eServices and Dynatrace Community Account profile.

eServices profile management

On this page

View your eServices profile

Change your password

Manage newsletter subscriptions

Related pages

eServices

eServices for partners

Licensing details

You can perform the following license management tasks on the eServices website:

View your eServices profile

Choose **Home > Profile** to view roles currently assigned to your eServices profile. You can also change the website's default time zone to match your current time zone.

Change your password

Choose **Home > Change Password** to change your Dynatrace Community user account password. Changing your password on eServices changes the password you use to log into the Dynatrace Community to access the product downloads page and Customer Support.

Manage newsletter subscriptions

Choose **Home > Manage Subscriptions** to subscribe to the Dynatrace Community Newsletter, which contains information on Dynatrace product best practices, tips & tricks, product updates, and news from Dynatrace labs.

eServices manage units, hours and visits

On this page

Create license

Search licenses

Edit license

Clone license

Terminate license

Related pages

eServices

eServices for partners

Licensing details

If your company has a pool of unused Agent units, Agent hours or UEM visits, you can:

- Create a new license
- Search for existing licenses
- Edit an existing license
- Clone an existing license
- Terminate an existing license

Note

If your company purchased AppMon before May 2017, a contract change might be required. Please contact your sales representative for clarification. An additional prerequisite is that all of your AppMon servers need to be at least on version 2017 May (7) and your eServices account has to have special permissions to create licenses. Please check under **Home -> Manage Profile** if you have **appmon-licensing** listed in **Your Roles**.

Create license

Ensure you have available license units. Click **My Licenses -> Manage Licenses**. The **Manage Licenses** page displays the available license units. If these are sufficient, you can create a new license.

1. Click **My Licenses -> Create License**. The **Create New License** page displays.
2. Select the required AppMon version.
3. Select either Perpetual or Online license type. Read more about license types.
4. Select the edition of the new license. Read more about editions.
5. Specify the purpose of the new license. Optional text field for your own use, which should help to identify this license later.
6. Enter the license holder's eServices credentials.
7. Select the start day of the new license.
8. Specify the license units:
   - For Perpetual license, specify the amount of agents for each technology, and volumes of Agent hours and UEM visits.
   - For Online license, specify overall amount of agents, and volumes of Agent hours and UEM visits.
9. Click **Create**. The license account owner receives email notification including the license file and the pool balances are debited according to the license values for units, hours and visits.

Search licenses

The license search lets you list all licenses which meet the search criteria. Licenses not created based on pools by Dynatrace license management can't be listed/managed here.

1. Click **My Licenses -> Manage Licenses**.

2. Search for licenses by specifying the search criteria license key, email, purpose or server by enter at least 3 characters and click **Search**.
3. All licenses which meet the search criteria display.

Edit license

Edit an existing license to change the receivers and the purpose. This is an optional text field you can use to identify a license.

1. Search for the license you want to edit.
2. Select the **Action Edit** in the license table.
3. The **Edit license attributes** dialog box opens, where you can edit recipients and purpose.
4. Click **Edit** and **Finish and Close** when you are done. Changes appear in the license table.

Clone license

Cloning a license let you to create a new license based on an excising on (type, edition and units are identical).

1. Search for the license you want to clone.
2. Select the **Action Clone** in the license table.
3. The **Clone a license** dialog box opens, where you can edit recipients and purpose.
4. Click **Clone** and **Finish and Close** when you are done. Your cloned license appears in the license table.

Terminate license

When a license is terminated, it's not available any more for use and the amount of units, hours and visits within this license gets booked back to the pools and are available to create new licenses.

Terminate a license in eService

1. Search for the license you want to terminate.
2. Select the **Action Terminate** in the license table.
3. The **Terminate License** dialog box opens and asks for the confirmation to terminate the license.

4. Click **Terminate** and **Finish and Close** when you want to terminate the license. The terminated license is removed from license table.

eServices view license information

Related pages

eServices

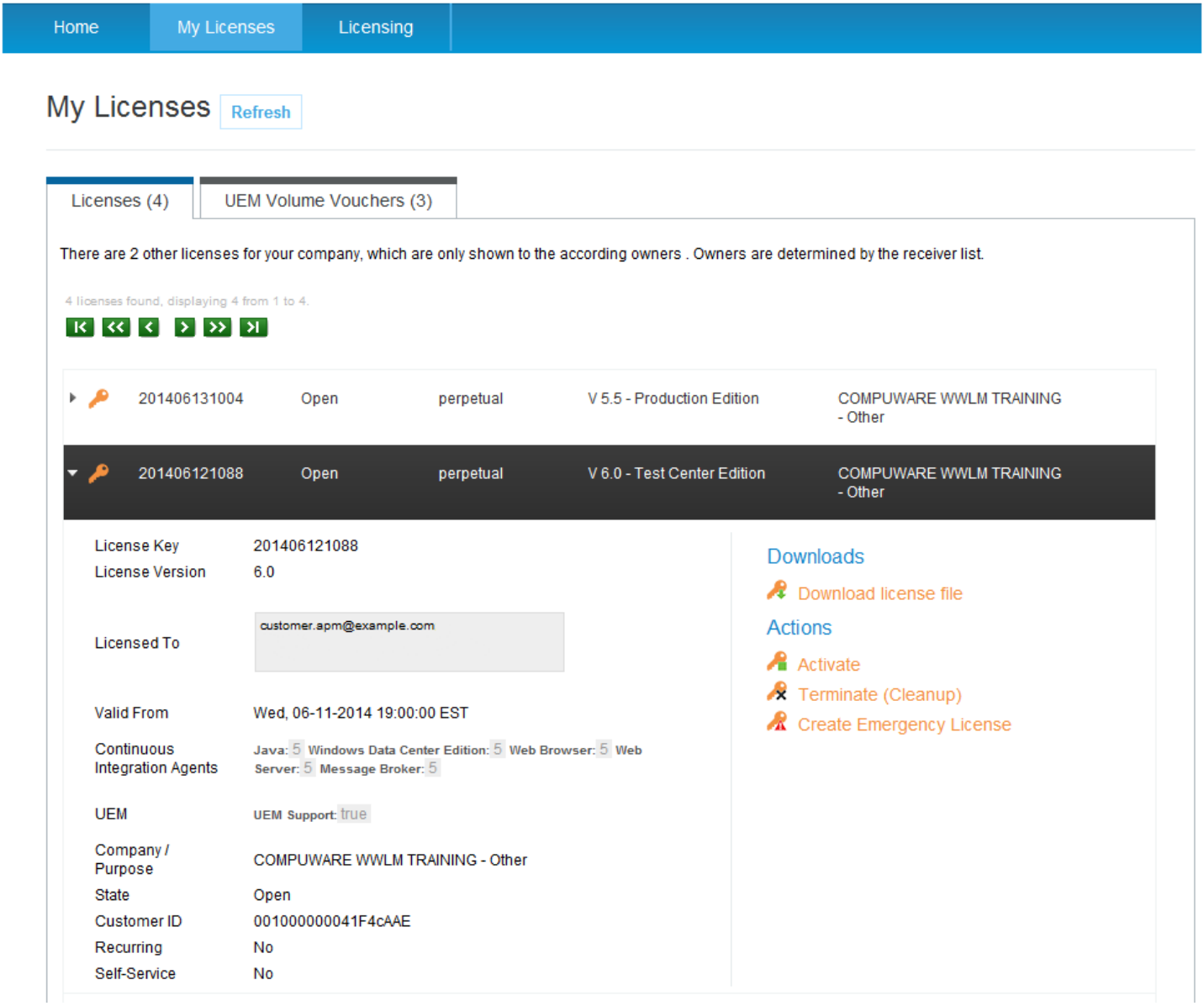
eServices for partners

Licensing details

Once the License Management team creates your license, you can view the license details by choosing **My Licenses** from the eServices menu. The **My Licenses** page contains two tabs: **Licenses** and **UEM Volume Vouchers**. On these two tabs, you can view license details and UEM volume. You can also select actions to perform, such as downloading a license file, creating an emergency file, and redeeming or upgrading UEM volume.

Note

If you cannot view licensing information or have access to the license management features, contact the eServices team to have your user account added to the license.

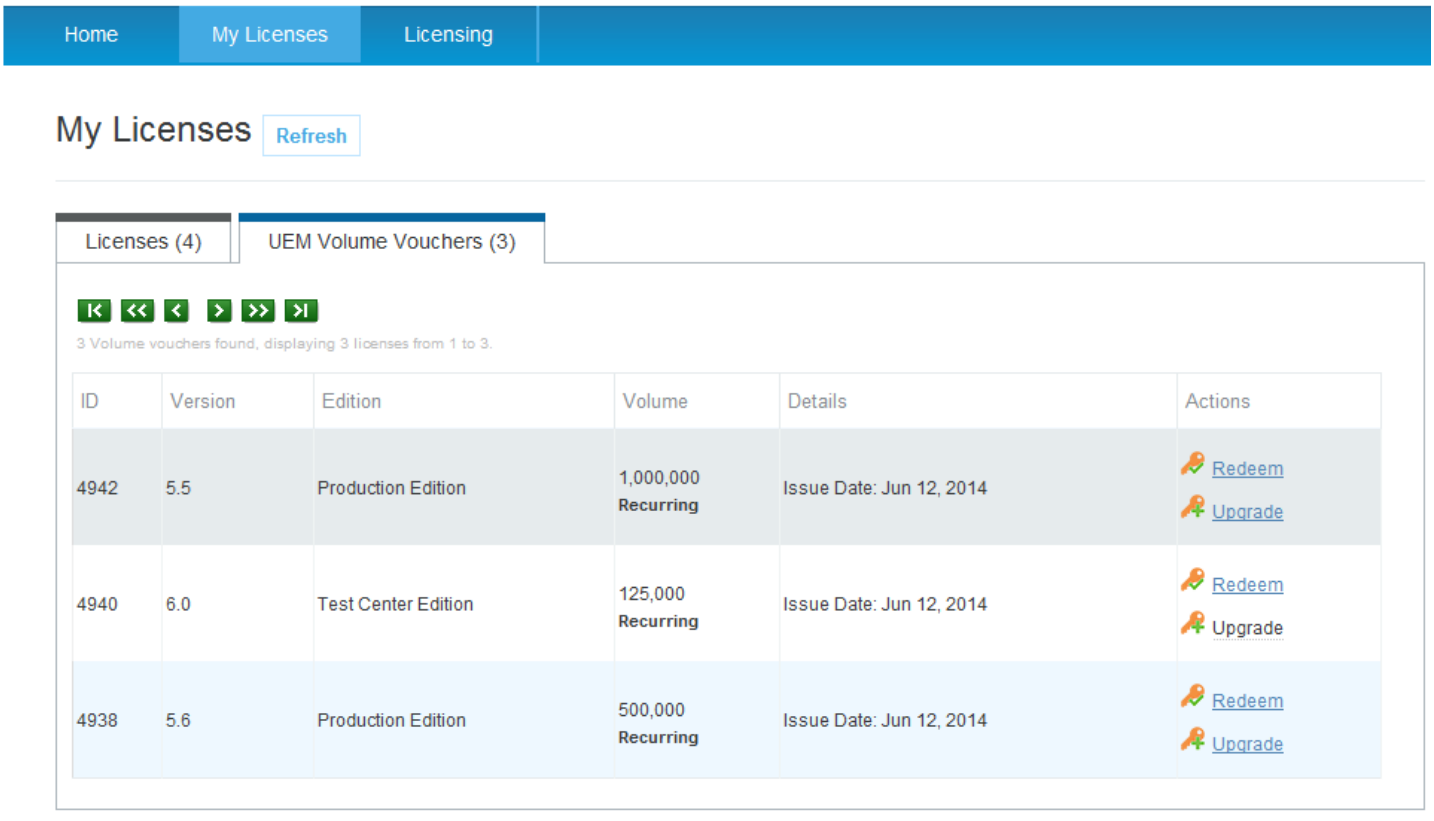


My License Page - Licenses Tab

Item name	Possible values	Value (shown in screenshot above)	Description
License no.		201406121088	
State	Open, issued, activated, deactivated, terminated	Open	• POC and Demo licenses are in an sta Issued and don't need to be activated. • When a license has been deactivated specific hardware fingerprint, it can on reactivated for a different one. • Terminated licenses can be filtered fro certain lists for clarity.

Type of license	POC, evaluation, term, perpetual, ISV	Perpetual	POC - Proof of Concept; Term - time-limited; Perpetual - not limited in time
For Product version #		V 7.0	This license was issued for AppMon versio
For Product edition	• Test Center edition • Production edition	Test Center edition	Editions expose different features and functionality in the product, such as applic scenarios, Agent configurations, and dashboards.
Licensed to	Mail address	customer.apm@example.com	Can contain multiple email addresses for t license owners. A license may have more one owner.
Valid from		Wed, 06-14-2017 19:00:00 EST	Issue date of the license.
Valid until			Does not apply to perpetual licenses.
Available Agents	**All Editions** **Agents**: Java, WOSI (Windows OS Instance) Enterprise or Data center (=.Net), ADK, browser, Web Server **Other:** UEM visits (volume)	Continuous Integration Automation vs. Developer Advanced Profiling Agents	Available Agent types and numbers for dif editions. **On z/OS:** A Licensing Management Syste must be installed on z/OS in addition to th AppMon Server license. See Upgrading z Agents.
Downloads	Download license file, download *activated* license file	Download license file	If you must perform licensing tasks manua file), you can download the *dynaTrace_license_20YYMMTTHHMMNN* file you initially received by email. After yo imported it in the Client, choose **Server Settings > License** to see your license d including the activation key.
Activation action	Activate, Deactivate	Activate	With the Activation Key you can manually Activate the license here and then import *_activated.lic* pressing the same button in same Client dialog box as you did with the unactivated key before. Deactivate manua pressing the button in the Client licensing box and copying the Activation and Deacti Keys to the eServices Deactivate dialog b

Reactivation action	Reactivate		The Reactivate link only appears after a li has been activated. Reactivation lets you continue operation without assistance fror License Management team when an emer happens, such as: • hardware defect • AppMon Server move • network configuration change
Upgrade action	Upgrade		Upgrade a license in Trial mode and issue temporary license or upgrade a license in mode and convert it to a new license.
Redeem UEM volume	Redeem volume vouchers		Click this link to redeem a UEM volume vo for the currently selected license. After redeeming the voucher, you receive a volu license file (.vol file) to import into your Ap Server.
Deactivate UEM volume	Deactivate UEM volume licenses		Click this link to deactivate any UEM volun the currently selected license. A voucher i created for the volume not consumed prio deactivation.
Terminate (Cleanup) action	Terminate	Terminate (Cleanup)	Delete deactivated or expired licenses fro profile.
Create emergency license	Create emergency license	Create emergency license	Click this link to create a one-time emerge license.



My License Page - UEM Volume Vouchers Tab

Item name	Possible values	Item (shown in screenshot above)	Description
Redeem (volume vouchers)	Redeem	Redeem	To redeem a voucher for a license, select the UEM Volume Vouchers tab on the My Licenses page, and then click the Redeem link.
Upgrade (volume vouchers)	Upgrade	Upgrade	After deactivating a license, a voucher is created for any unused UEM volume. You can upgrade this voucher to the current version by clicking the Upgrade link.

If the information on the eServices website does not update after making any changes, **Refresh** on the **My Licenses** page.



Refresh eServices Content

eServices for partners

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

eServices

In eServices, partners can quickly create one-month evaluation licenses using the provided license templates. Each template allocates a set number of Agents and UEM volume for a Demo license.

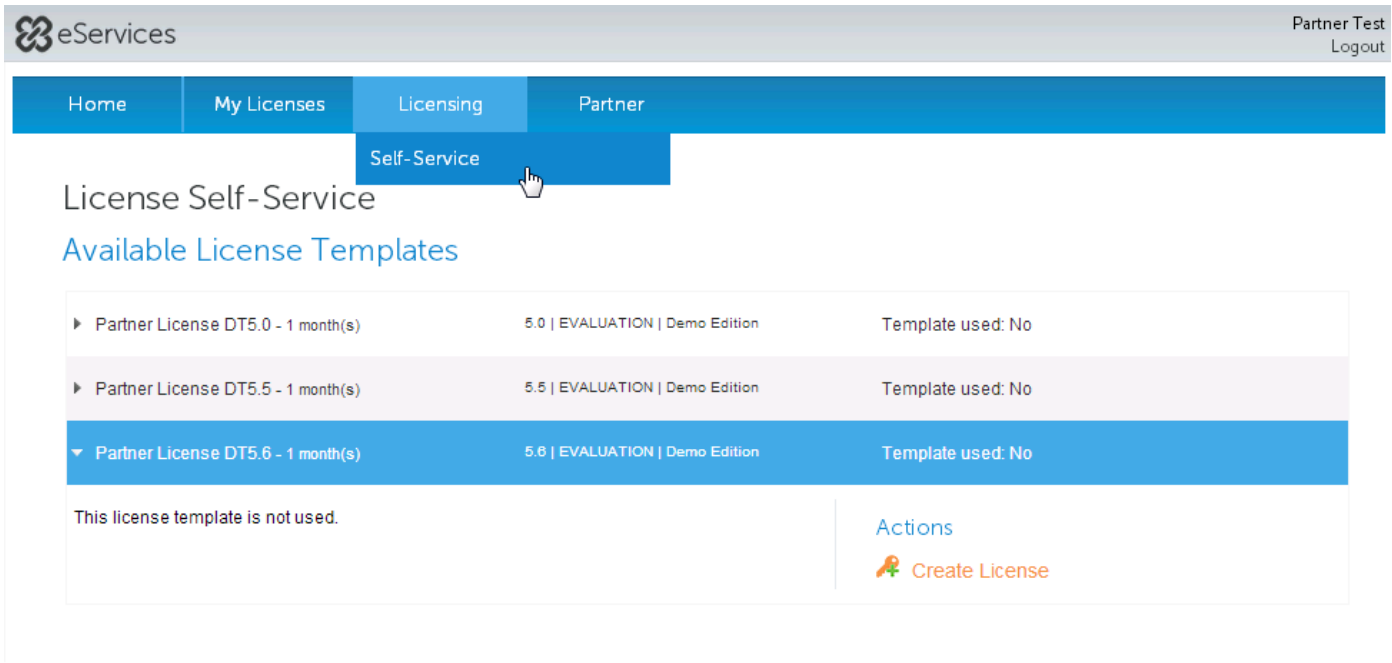
Note

The Demo edition does not support remote access. Contact the license management team for an license other than the Demo license.

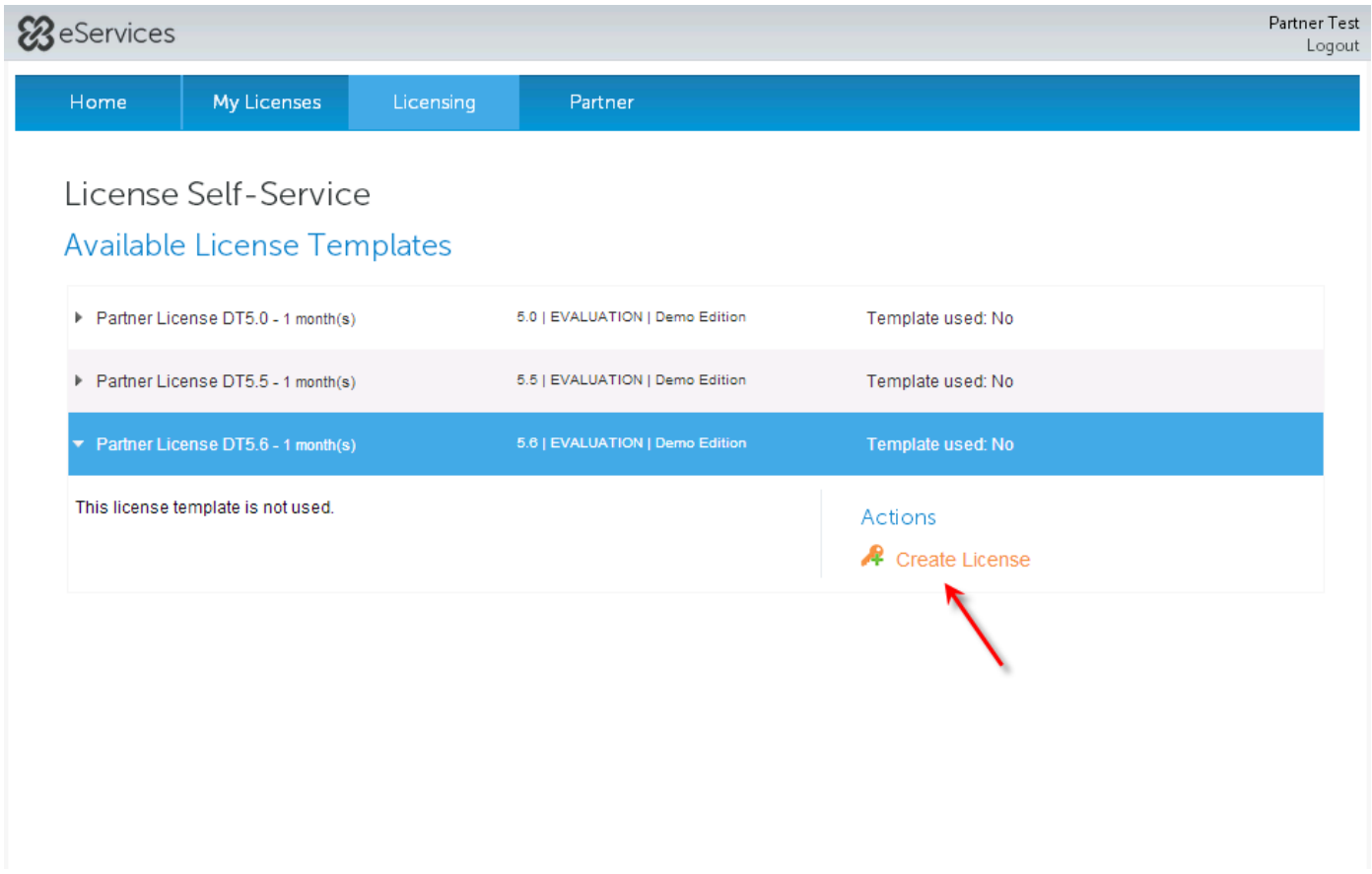
To create a license:

1. Log into eServices using your Dynatrace Community partner account.

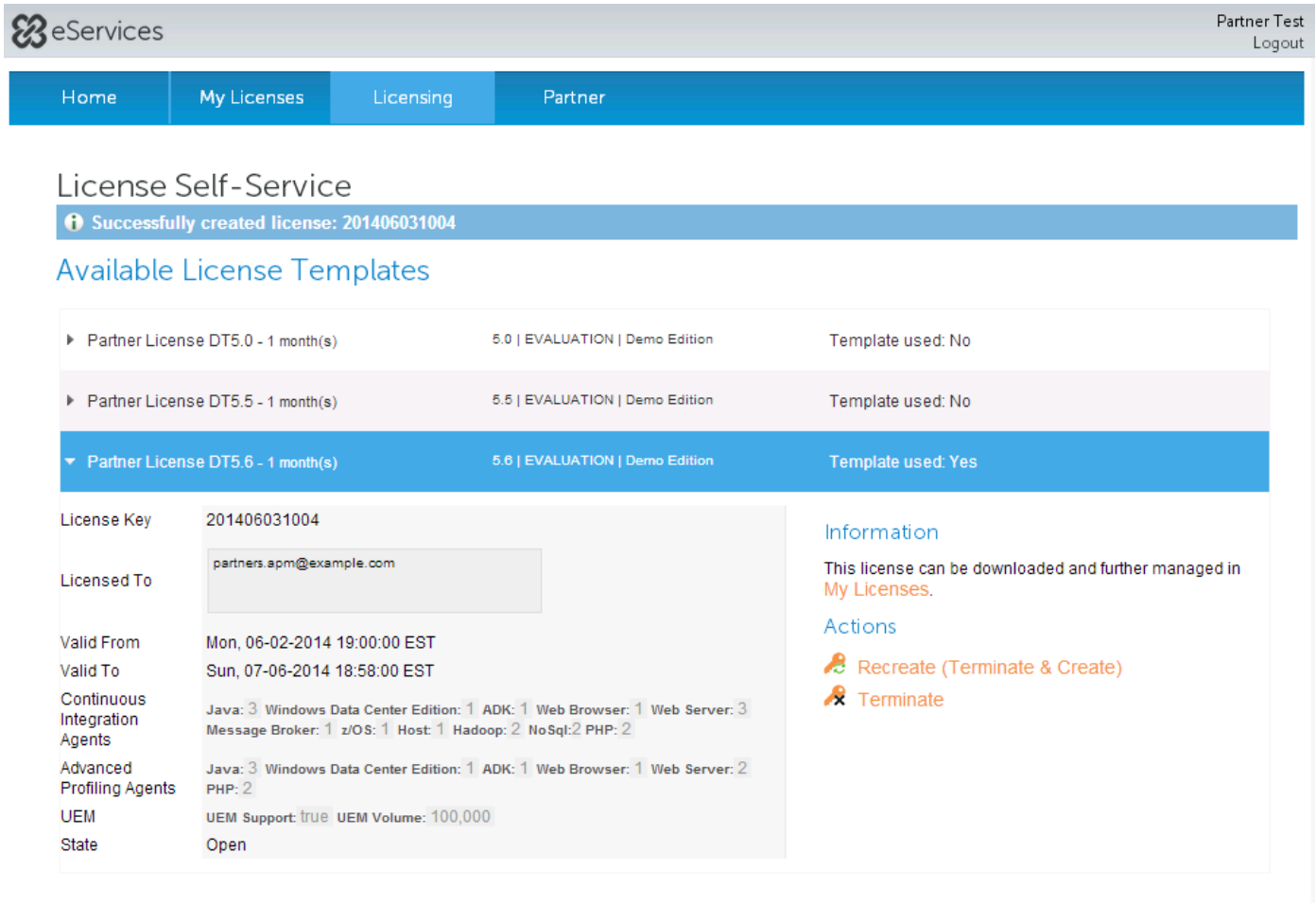
2. Choose **Licensing > Self-Service** from the menu.



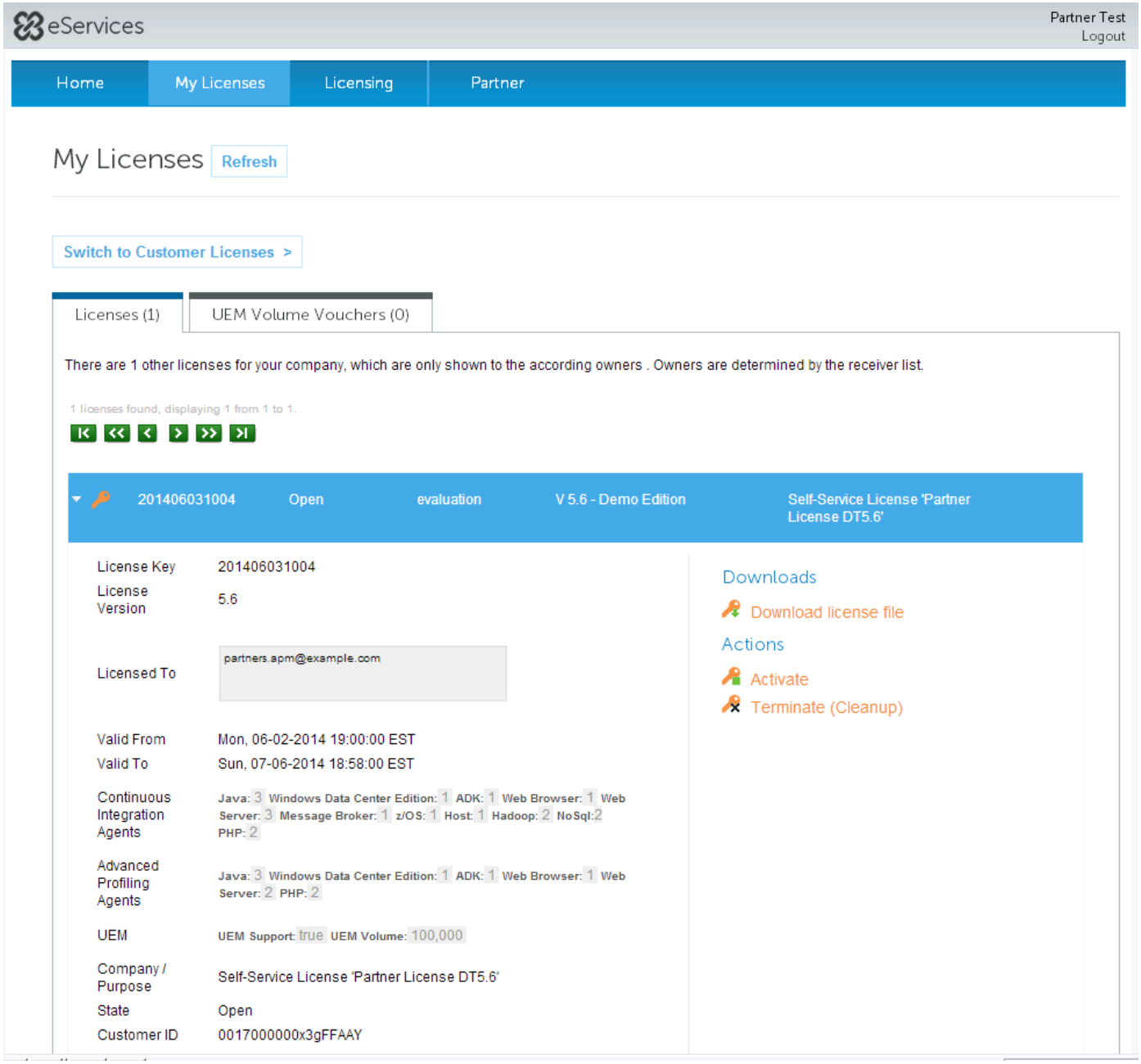
3. Select from the available templates to expand the template information.
4. Under **Actions**, click **Create License**.



License information, such as license key, number of Agents, and UEM volume, displays.


eServices
Partner Test
Logout
Home
My Licenses
Licensing
Partner
License Self-Service
Successfully created license: 201406031004
Available License Templates
Partner License DT5.0 - 1 month(s)
5.0 | EVALUATION | Demo Edition
Template used: No
Partner License DT5.5 - 1 month(s)
5.5 | EVALUATION | Demo Edition
Template used: No
Partner License DT5.6 - 1 month(s)
5.6 | EVALUATION | Demo Edition
Template used: Yes
License Key
201406031004
Licensed To
partners.apm@example.com
Valid From
Mon, 06-02-2014 19:00:00 EST
Valid To
Sun, 07-06-2014 18:58:00 EST
Continuous Integration Agents
Java: 3 Windows Data Center Edition: 1 ADK: 1 Web Browser: 1 Web Server: 3 Message Broker: 1 z/OS: 1 Host: 1 Hadoop: 2 NoSql:2 PHP: 2
Advanced Profiling Agents
Java: 3 Windows Data Center Edition: 1 ADK: 1 Web Browser: 1 Web Server: 2 PHP: 2
UEM
UEM Support: true UEM Volume: 100,000
State
Open
Information
This license can be downloaded and further managed in My Licenses.
Actions
Recreate (Terminate & Create)
Terminate

5. To activate the license or download the license file, access the **My License** page and choose from the available options.



Online license activation and deactivation

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

License activation

License deactivation

Deactivate remaining UEM volume prior to license deactivation

License reactivation

Related pages

License activation and deactivation via file

Manage UEM visits volume

Manage usage based agent hours volume

License upgrade

License assignments

Licensing details

eServices

This page describes how to use the online method to activate, deactivate, and reactivate a license. For instructions on using the manual method, see License Activation and Deactivation via File.

Note

If you are unable to connect to eServices from the Client, you may need to configure a proxy server. For more details, see Configuring a Proxy Server to Connect to eServices.

To activate a license, you must have the necessary permissions enabled. Contact the License management team if you need elevated licensing permissions.

License activation

An activated license is the fingerprint that uniquely identifies and links a license to a single AppMon Server. The activated license only works on the machine that matches the fingerprint.

For online license activation, follow these steps:

1. In the Client, access the **License** pane of the **Server Settings** dialog box from either the **Start Center > Administration** pane or from the Client menu: select **Settings > Dynatrace Server** and click **License**.
2. Click **Import License**. The **Select location of a license** dialog box appears.
3. Select **Retrieve license online**, type your Dynatrace Community account user name and password, and click **Next**.
4. At the **Available Licenses** page, select the license to activate, and click **Next**. The license file is downloaded from eServices, imported into the AppMon Server, and activated online.
5. The **Summary** page displays the status of the activation process. Click **Next**.
6. At the **Memory Settings** page, choose the appropriate memory configuration for your environment, and click **Finish** to complete the wizard.
   After completing the activation process, you must add any purchased UEM visits volume to the license. Optionally, you can set up a license reservation to help manage the distribution of Agents allocated for use by the license.

License deactivation

You must deactivate your license in the following scenarios:

- Your AppMon Server machine hardware changes.

- Your AppMon Server machine network setting changes.
- You plan to move the AppMon Server to a different machine.
- You plan to merge or split a license.

Hardware or network configuration changes can lead to automatic license deactivation. If this happens, the AppMon environment runs normally until the AppMon Server reboots. The AppMon administrator receives an incident notification if the license has been deactivated automatically. If it's not possible to deactivate the license, for example if the machine is not available, contact the License management team.

To avoid down times or data loss, a license continues to work for 1 day (evaluation license) or 3 days (perpetual, term, or online license) after it is deactivated.

For online license deactivation, follow these steps:

1. In the Client, access the **License** pane of the **Server Settings** dialog box from either the **Start Center > Administration** pane or from the Client menu: select **Settings > Dynatrace Server** and click **License**.
2. Click the **Deactivate** button. The **Deactivate current license** dialog box appears.
3. Select **Deactivate current license** online, and then type your Dynatrace Community account user name and password, and click **Finish**. The license deactivates along with any remaining UEM volume.

Deactivate remaining UEM volume prior to license deactivation

Prior to deactivation, you must release any unused UEM volume from the current license, you must deactivate remaining UEM volume from the current license before deactivating that license so you can then apply those UEM volumes to an upgraded or updated license. Deactivating the unused UEM volume using eServices ensures those the remaining volumes can be applied to the new license. Access eServices and enter the deactivation information you receive after you manually deactivate the license. See Manage UEM Visits Volume to learn how.

License reactivation

You can reactivate a license that was previously deactivated, provided the machine on which the license is reactivated is **different from the original machine** . If you are unable to deactivate a license due to a hardware issue, for example, then open a License support ticket to obtain a replacement license.

For online license reactivation, follow these steps:

1. In the Client, access the **License** pane of the **Server Settings** dialog box from either the **Start Center > Administration** pane or from the Client menu: select **Settings > Dynatrace Server** and click **License**.
2. Click the **Import** button.
3. Perform the same steps as you would to activate a license.

License activation and deactivation via file

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

License activation

License deactivation

License reactivation via a file

Related pages

Configuring a Proxy Server

Online license activation and deactivation

eServices

This page describes how to use the manual method to activate, deactivate, and reactivate a license via file. For instructions on using the online method, see Online license activation and deactivation.

The manual method for activating or deactivating a license via a file should only be performed if you are unable to connect to eServices from the Client, in which case, you may need to configure a proxy server. See Configuring a Proxy Server for more information.

> **Note**
>
> To activate a license, you must have the necessary permissions enabled. Contact the License management team if you need elevated licensing permissions.

License activation

An activated license is the fingerprint that uniquely identifies and links a license to a single AppMon Server. The activated license only works on the machine that matches the fingerprint.

The process for activating a license via a file is a multi-step process that includes the following general steps:

1. In the Client, import the original **unactivated** license key file sent in the license notification email or downloaded from eServices.
2. After importing the unactivated license key file, make a copy of the license number and generated **activation key**, and then log into eServices on a machine with Internet access.
3. Activate the license and download the newly created **activated** license key file.
4. In the Client, import the activated license key file to complete the process. For step-by-step instructions, see below.

To activate a license via a file, follow these steps:

1. In the Client, access the **License** pane of the **Server Settings** dialog box from either the **Start Center > Administration** pane or from the Client menu by choosing **Settings > Dynatrace Server**, and clicking **License**.

2. Click **Import License**. The **Select location of a license** dialog box appears.

3. Select **Import license from a file**, and then browse and select the *unactivated* license key file with the file name: `dynaTrace_license_20yymmddnnnn.key`. The unactivated file generates an *activation key*, which uniquely identifies the Server machine with the license. Click **Next**.

4. On the **Activate License** page, select **Activate license manually** and click **Next**.

5. On the **Summary** page, click the **Show Details** link, and then manually copy and save the license number and the generated activation key.

6. From a machine with Internet access, log into eServices and select **My Licenses**.

> **Note**
>
> If you are unable to log into eServices, contact the license management team and forward the activation information from step 5.

7. Expand the license information for the license that matches the number you copied in step 5, and then click **Activate**.

8. Type in the generated activation key, and then click **Activate**.

9. If the license was activated successfully, click the Download Activated File link and save your activated license key file, which now has the word activated appended to the end of the file name (for example, dynaTrace_license_20yymmddnnnn_activated.key)

0. Start the Client again and import the activated license key file as described in steps 1 - 3.

License deactivation

You must deactivate your license in the following scenarios:

- Your AppMon Server machine hardware changes.
- Your AppMon Server machine network setting changes.
- The AppMon Server moves to a different machine.
- You merge or split a license.

Hardware or network configuration changes can lead to an automatic deactivation of the license. If this happens, the AppMon environment will continue to run normally until the AppMon Server is rebooted. The AppMon administrator will receive an incident notification if the license has been deactivated automatically. If it's not possible to deactivate the license (for example, the machine has gone down), contact the license management team.

To avoid down times or data loss, a license continues to work for one day (evaluation license) or three days (perpetual or term licenses) after it's deactivated. If the license has been deactivated, a reactivation is required.

The process for deactivating a license via a file is a multi-step process that includes the following general steps:

1. In the Client, deactivate the license on the Server and obtain the license number, activation key, and deactivation key.
2. Log into eServices on a machine with Internet access and deactivate the license. For step-by-step instructions, see below.

To deactivate a license via a file, follow these steps:

1. In the Client, access the **License** pane of the **Server Settings** dialog box from either the **Start Center > Administration** pane or from the Client menu: select **Settings > Dynatrace Server** and click **License**.

2. Click **Deactivate**. The **License Deactivation** page appears.

3. Select **Deactivate current license manually** and click **Finish**. The license is now deactivated on the Server, however, there is a 1 or 3 day grace period depending on the license type.

4. On the **Summary** page, click the **Show Details** link and manually copy and save the license number, activation key, and deactivation key.
   From a machine with Internet access, log into eServices and select **My Licenses**.

> **Note**
>
> If you are unable to log into eServices, contact the license management team and forward the deactivation information from step 4.

5. Expand the license information for the license that matches the number you copied in step 4 and click **Deactivate**.

6. Enter the activation key and deactivation key and click **Deactivate**. An email is sent to the account owners.

7. Click **Finish and Close** to complete the deactivation process.

License reactivation via a file

Manually reactivating a license on another machine is necessary when:

- A hardware defect occurs.
- The AppMon Server moves to a different machine.
- A network configuration change occurs.

Manage UEM visits volume

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

How visit-based UEM licensing is defined

How visits are deducted from your purchased volume

Check visits consumption

How to manage your visits volume

Add UEM visits volume

Manually add UEM visits volume via file

Deactivate UEM visits volume

Manually deactivate UEM visits volume

UEM volume reservation

Free UEM visits volume

Related pages

License upgrade

Manage usage based agent hours volume

Online license activation and deactivation

eServices

License Overview dashlet

Generally, a UEM visit is defined as the actions performed by the same user within a certain time period. The visit duration is calculated from the start of the first user action to the end of the last user action.

This page shows how visit-based pricing is defined, what counts as a visit and gives instructions how to monitor consumption and add and deactivate UEM visits volumes to / from your license using both, the online method and the manual method.

How visit-based UEM licensing is defined

The UEM license is based on an annual visit volume that is renewed on a yearly basis automatically.

The annual consumption start date for all the blocks is the same as the first block purchased and resets to the full available amount of visits at the beginning of a period. Informational email notifications are sent to registered administrators when 75%, 90%, 95% and 100% of annual visits are consumed.

How visits are deducted from your purchased volume

Here is a sincere attempt to describe a complex computer algorithm in plain words. In doubt observed product behavior applies.

- A visit is deducted from the license volume the second after a user action occurs.
- One user action is considered a bounce and does not incur a deduction from the license volume.
- Long running visits reduces your license volume by one every sixty minutes. In other words, an additional visit is deducted from the license volume for every completed hour the visit lasts.
  For example, if a visit lasts for 25 minutes, the volume decreases by one. If it lasts for 6 hours and 25 minutes, the volume decreases by 7 (7th hour). This shows as one visit (user action) in the Visits dashlet, but the visits volume reduces by seven. (license terms)
- A visit ends when the browser or application either closes or is inactive for 30 minutes.
- If the same user accesses the application again, it creates a new visit and consume one from the volume.
- A visit can't cross applications. Applications and visits are a one to one mapping.
  So if you have two applications defined by URL-pattern and your customer accesses both, it results in two visits, even if it was without a timeout.
- For licensing, there is no difference between native mobile and browser / web mobile visits.
- Visits created by Synthetic Monitoring tests are not deducted from the volume.

For the best and most effective answer, contact your sales representative. Your representative should know what your contract looks like and, depending on your version, can advise for the best scenario for you.

Please check with sales for currency and actual agreements:

- The UEM license is based on an annual visit volume. This volume automatically renews every year.
- Annual visits are sold in blocks. You can start with a small block and add additional blocks if needed. This helps if you don't have a good estimate of visits per year to expect.
- After a 100% of purchased annual visits are consumed, an additional visits buffer is provided. The buffer is the number of annual visits purchased divided by 12.
- The annual consumption start date for all the blocks is the same as the first block purchased and automatically resets to the full available visits amount at the beginning of each maintenance / consumption year.
- Informational email notifications are sent to registered administrators when 75, 90, 95 and a 100% of annual visits are consumed during the maintenance / consumption year.

Check visits consumption

To check consumption, open the **License Overview** dashlet to quickly check UEM visit consumption. In the AppMon Client sidebar, simply double-click **License Overview**. See License Overview dashlet for more information.

Also, you can check out the UEM Correlation Overview dashboard. In the AppMon Client cockpit, click **Deployment Health > UEM Correlation**.

How to manage your visits volume

The online method is initiated in the Client and requires the Internet connection to access the eServices website. The online method is recommended as it simplifies the process for adding and deactivating UEM volume from your license.

> **Note**
>
> If you are unable to connect to eServices from the Client, you may need to configure a proxy server. See Configuring a Proxy Server to Connect to eServices for more information.

For instructions on upgrading UEM volume, see License Upgrade.

Once activated, create a UEM license reservation to precisely manage the consumption of UEM volume across multiple System Profiles and Applications.

Add UEM visits volume

Activate UEM volume in the Client. This includes logging into eServices and activating the volume.

Follow these steps to add UEM visits volume

1. Click **Settings > Dynatrace Server > License** to access the License pane of the **Server Settings** dialog box.
2. In the **UEM and Consumption** pane, click **Add Volume**. The **Add UEM Volume** wizard appears.
3. On the **Select location of volume** page, select **Retrieve license online** and enter your Dynatrace Community account user name and password. Click **Next**.
4. On the **Available volume vouchers** page, select the desired UEM visits volume in the table and click **Finish**.

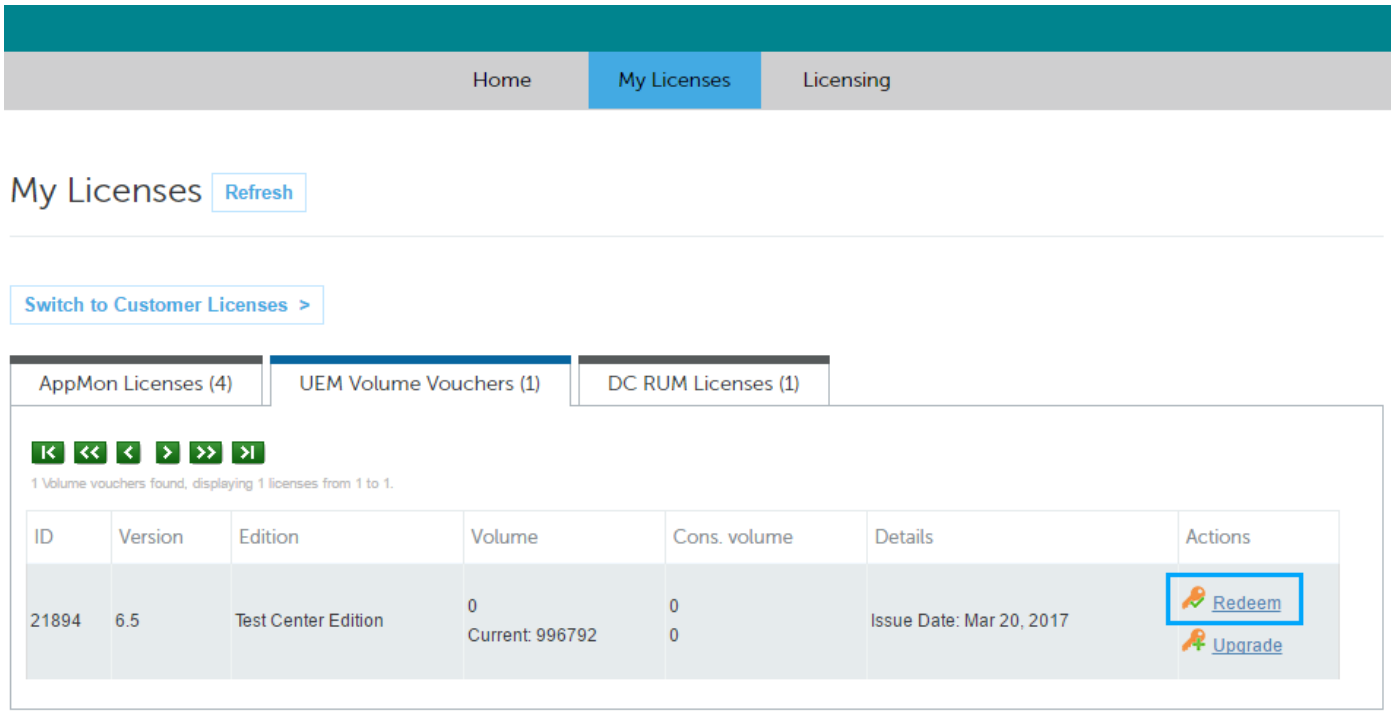
Manually add UEM visits volume via file

Manually adding UEM visits volume requires downloading a voucher file from eServices and importing the file in the Client. The license must be activated to enable the Redeem button in eServices.

Follow these steps to manually add UEM visits volume via a file

1. Access eServices and navigate to the **My Licenses** page.
2. Select the **UEM Volume Vouchers** tab. Available vouchers to redeem display.

3. Click **Redeem** for the UEM voucher you want to download.



4. Click **Choose for Redeeming**.
5. Click **Download Volume License** and save the voucher file.
6. Start the Client, and access the **License** pane of the **Server Settings** dialog by clicking **Settings > Dynatrace Server > License**.
7. In the **UEM and Consumption** pane, click **Add Volume** to start the **Add UEM Volume** wizard.
8. Select **Import volume from a file** then browse and open the download voucher file.
9. Click **Finish** to close the wizard and add the UEM volume to your license.

Deactivate UEM visits volume

Deactivate UEM volume using the Client. This includes logging into eServices and deactivating the volume.

Follow these steps to deactivate UEM visits

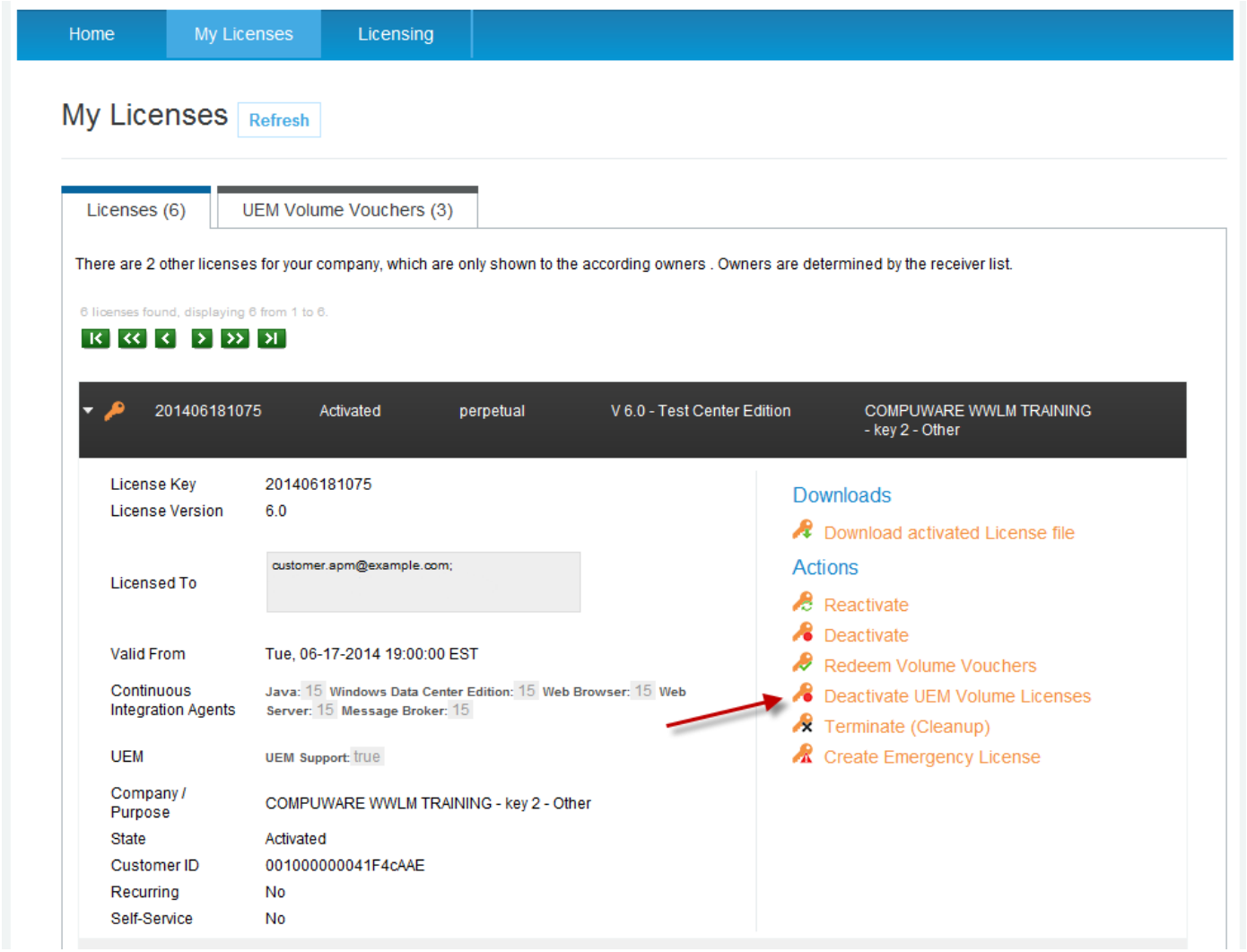
1. In the Client click **Settings > Dynatrace Server > License > Information**.
2. In the **UEM and Consumption** pane, click **Deactivate Volume**. The **Deactivate current volume** wizard appears.
3. Select **Deactivate current volume online** and click **Finish** to close the wizard.

Manually deactivate UEM visits volume

Manually deactivate UEM visits volume by generating a volume deactivation key in the Client, then accessing eServices and manually entering the key.

Follow these steps to manually deactivate UEM visits volume

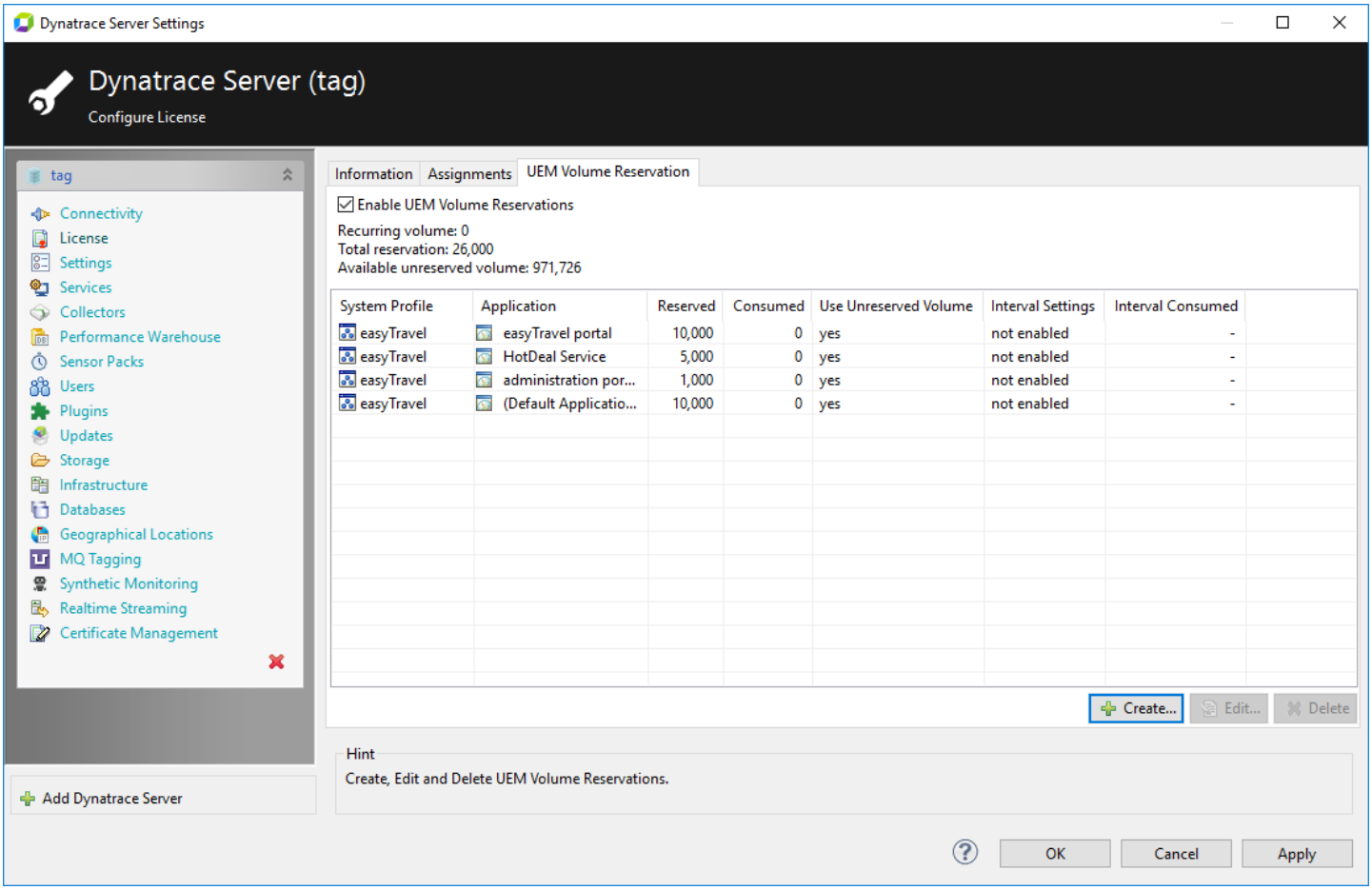
1. In the Client click **Settings > Dynatrace Server > License > Information**.
2. In the **UEM and Consumption** pane, click **Deactivate Volume**. The **Deactivate current volume** wizard appears.
3. Select **Deactivate current volume manually** and click **Finish**. The **Summary** page appears.
4. Click the **Show Details** link, then manually copy the license number and volume deactivation key.
5. Go to eServices and click **My Licenses**.
6. Expand the license from which to deactivate the UEM volume.
7. Under **Actions**, click **Deactivate UEM Volume Licenses**. The **License Volume deactivation** dialog box appears.



8. Copy the volume deactivation key.
9. In the **Deactivation Code** box, enter the volume deactivation key you previously copied. Click **Next**. A message appears stating the UEM volume was deactivated successfully and any remaining volume was stored in a voucher.

UEM volume reservation

To precisely manage the consumption of UEM volume, you can reserve a specified amount of total UEM volume per System Profile and Application. For example, if your license is provisioned for 1,000,000 UEM visits, you could reserve half of this volume for one System Profile and half for another System Profile. In addition, you can reserve UEM volume per Application of an individual System Profile, as shown in the following screenshot.



Additionally to reserving a UEM volume it's possible to limit the usage of UEM volume per interval. For example you could enter a limit of 10,000 UEM visits per day.

Follow these steps to create a UEM volume reservation

1. Choose **Settings > Dynatrace Server > License** to access the License pane of the **Server Settings** dialog box.
2. Select the **UEM Volume Reservation** tab, then select the **Enable UEM Volume Reservation** check box.
3. Click **Create**. The **UEM Volume Reservation Configuration** dialog box appears.

4. Select the System Profile and Application for which to reserve UEM volume and enter the number of UEM volume to reserve.

If you select **(Default Application)**, only auto-detected applications without a user-defined UEM configuration are included. See System Profile - User Experience for more information.

1. Optionally select the **Allow consumption of unreserved volume** check box to enable the consumption of available UEM volume beyond what is specified in the reservation.
2. If you want limit the UEM volume usage per interval you can do so by checking **Configure Interval**. The interval can be configured in days or months. The **Reference date** specifies the start of the first interval. If the specified number of UEM visits has been used up no more visits will be captured until the end of the interval. When the end of an interval is reached the next interval will be started automatically.
3. Click **OK** to create the reservation.

Free UEM visits volume

The **UEM Free Trial** button activates if your license contains free UEM visits volume. After clicking this button, you can use the free UEM visits volume associated with your license.

Manage usage based agent hours volume

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

License upgrade

Manage UEM visits volume

Online license activation and deactivation

eServices

Overview

AppMon supports usage-based licensing. Instead of a static number of agents that are allowed to connect usage-based licensing offers a more flexible model. In times with high load a usage-based volume can be used to connect extra agents in addition to a static baseline. The unit used to count the consumption is “Agent Hour” and can be consumed by any supported technology (except the z/OS based agents: CICS & IMS).

Agent hours

An agent running for a 60 minute period, counted from the beginning to the end of the hour (AppMon server time) is measured as one agent hour. Partial hours are counted as a full hour, but only to the maximum of concurrently attached agents of the same technology type:

- Two Java agents operating in sequence during an hour counts as one Java agent hour.
- Five Java agents operating concurrently for 10 minutes within one hour counts as five Java agent hours.

Usage-based volume vouchers

To use usage-based licensing a volume voucher needs to be added to the license. It contains a number of agent hours as purchased, where the unit volume is usually Agent Years. One agent year is the equivalent of 9,000 agent hours (~ 24h * 365). Each usage-based volume has a start and an end date. Multiple volumes are treated separately by the AppMon server. The volume that expires first will be used first. Incidents are created when the total used agent hours volume reaches certain thresholds (75%, 90%, 95% and 100%). If the start date of the volume has not been reached yet it will not be used.

Usage-based volume consumption

Usage-based volume is used whenever there aren't enough perpetual agent licenses to satisfy the current demand. For example, if there are five perpertual Java agents in the license but six are connected, the 6th consumes one agent hour per hour of the usage-based volume. License Assignments can be used for limiting consumption of usage-based volume and for configuring the usage of agent hours.

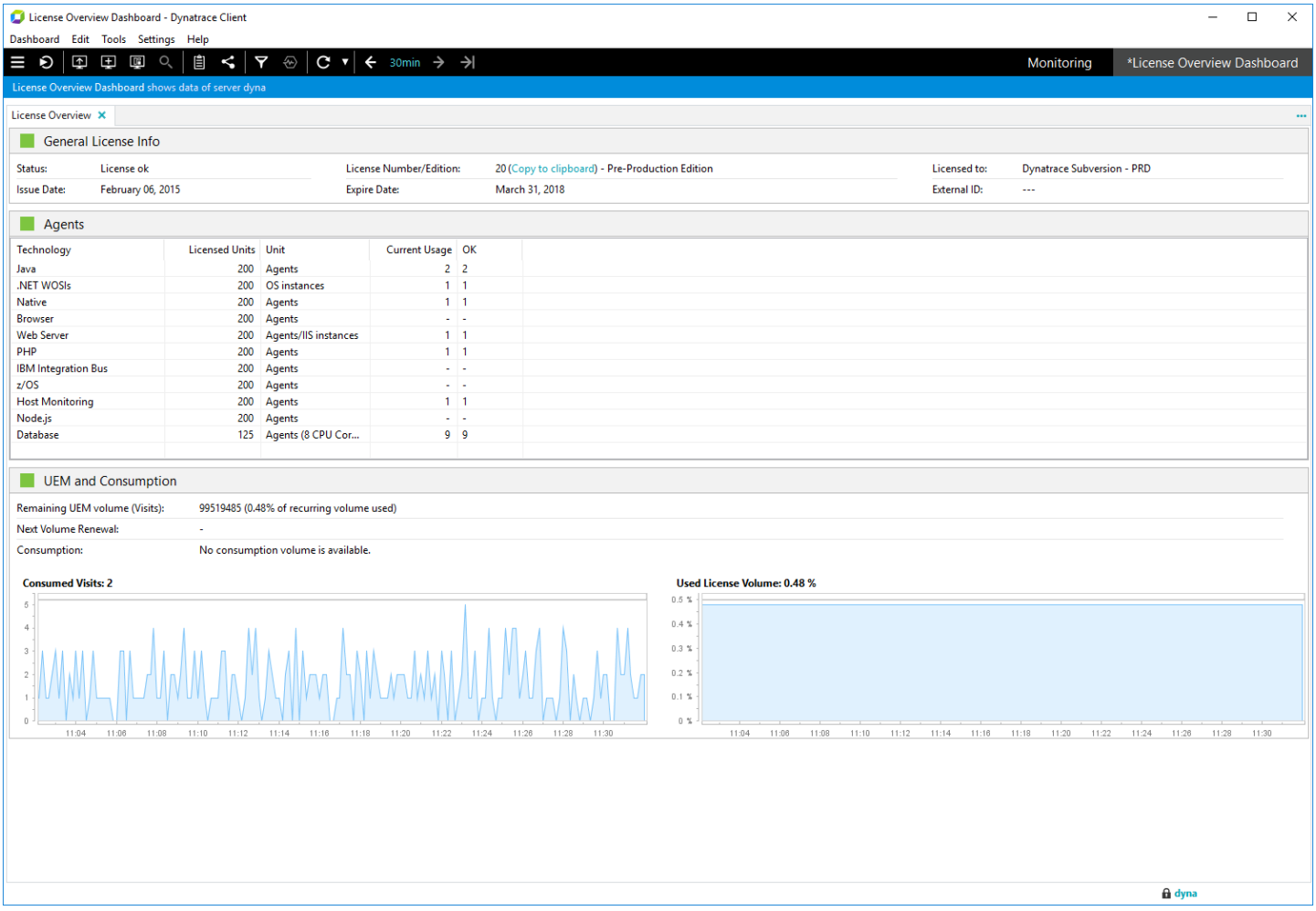
For the consumption of usage-based volume the maximum number of concurrently connected agents per technology type and per hour is taken into consideration. This value is multiplied with a technology-dependent weight to get the number of agent hours deducted from the usage-based volume. The calculation is performed every full hour.

Agent technology types available for usage-based licensing	Unit volume	Weight
Java Agent (JVM) - Windows/Unix/Linux/zOS	Per JVM	1
.NET Windows OS Instance (WOSI)	Per WOSI	1
PHP Agent	Per parent process ID (PID)	1
Node.js Agent	Per process	0.1
IBM integration bus agent	Per process	0.2
Native Agent (ADK)	Per process	1
Database Agent	Per instance (8 Cores)	1

WebServer Agent	Per Agent	0.6
NoSQL Agent	Per JVM	0.6
Host Monitoring Agent	Per host	0.04

> For example, 20 Java agents plus five WebServer agents monitoring for one hour = (20 Java agent hours * 1) + (5 WebServer agent hours * 0.6) = 23 agent hours consumed from the usage-based volume.

The License Overview dashboard provides information about the agent hours volume usage. The column “Current Usage” lists the total count of agents, the column “OK” the agents that are covered by the perpetual license, “Consumption Based” lists how many agents are currently consuming agent hours.



How to manage your usage-based volume

Use the AppMon Client to manage your usage-based volume. The Client must have an Internet connection to access the eServices website. This is the recommended way to manage usage-based volume because it simplifies the process for adding and deactivating consumption volume from your license.

If you are unable to connect to eServices from the Client, you may need to configure a proxy server. For more details, see Configuring a Proxy Server to Connect to eServices.

For instructions on upgrading usage-based volume, see License Upgrade.

Once activated, create License Assignments to precisely manage usage-based volume across multiple System Profiles and Agent Groups.

Add usage-based volume

The online method for adding usage-based volume is initiated in the Client and includes logging into eServices and activating the volume. See below for step-by-step instructions.

Follow these steps to add usage-based volume

1. Choose **Settings > Dynatrace Server > License** to access the License pane of the **Server Settings** dialog box.
2. Click **Add Volume**. The **Add Volume** wizard appears.
3. On the **Select location of volume** page, select **Retrieve license online** and enter your Dynatrace Community account user name and password. Click **Next**.
4. On the **Available volume vouchers** page, select the desired consumption volume in the table and click **Finish**.

Manually add usage-based volume via file

To manually add usage-based volume, you must download a volume file from eServices and import the file in the Client. The license must be activated to enable the Redeem button in eServices.

Follow these steps to manually add usage-based volume

1. Access eServices and browse to the **My Licenses** page.

2. Select the **UEM Volume Vouchers** tab. The vouchers available to redeem display.
3. Click **Redeem** for the voucher you want to redeem.
4. Click **Choose for Redeeming**.
5. Click **Download Volume License** and save the volume file.
6. Start the Client and choose **Settings > Dynatrace Server > License** to access the License pane of the **Server Settings** dialog box.
7. Click **Add Volume**. The **Add Volume** wizard appears.
8. Select **Import volume from a file**, and select the downloaded volume file.
9. Click **Finish** to close the wizard and add the volume to your license.

Deactivate usage-based volume

Use the AppMon Client to deactivate usage-based volume. This includes logging into eServices and deactivating the volume.

Follow these steps to deactivate usage-based volume

1. Choose **Settings > Dynatrace Server > License** to access the License pane of the **Server Settings** dialog box.
2. Click **Deactivate Volume**. The **Deactivate current volume** wizard appears.
3. Choose **Deactivate agent hours**.
4. Select **Deactivate current volume** online, and then click **Finish** to close the wizard.

Manually deactivate usage-based volume

Deactivating Consumption volume requires generating a volume deactivation key in the Client, and then accessing eServices and manually entering the key. See below for step-by-step instructions.

Follow these steps to manually deactivate usage-based volume

1. Choose **Settings > Dynatrace Server > License** to access the License pane of the **Server Settings** dialog box.
2. Click **Deactivate Volume**. The **Deactivate current volume** wizard appears.
3. Choose **Deactivate agent hours**.
4. Select **Deactivate current volume manually** and click **Finish**. The **Summary** page appears.
5. Click the **Show Details** link and manually copy the license number and volume deactivation key.
6. Go to eServices and click **My Licenses**.
7. Expand the license from which to deactivate the volume.

8. Under **Actions**, click **Deactivate UEM Volume Licenses**.
9. In the **Deactivation Code** box, enter the volume deactivation key you previously copied. Click **Next**. A message appears stating the volume was deactivated successfully and any remaining volume was stored in a voucher.

License assignments

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Required permissions

Set up a license assignment

Overbooking

Related pages

eServices

Web Server Master Agent configuration

Prepare to upgrade

Manage UEM visits volume

Manage usage based agent hours volume

Online license activation and deactivation

Licensing details

Overview

Use the **license assignment** feature to precisely manage the availability of Agent licenses per System Profile and Agent Group.

There are two types of license assignments:

- **Reservation**: a user-defined number that guarantees Agent license availability for a given System Profile and Agent Group.

- **Maximum Allowed**: a user-defined number that limits Agent license availability for a given System Profile and Agent Group. This is also useful for Usage Based Licensing to limit the consumption of usage based volume.

When you create a license assignment, you can use either a reservation or maximum allowed limit by itself or both types in conjunction with each other.

In addition to the license assignments, you can define whether technology types may consume Agent Hours to fulfill its reservations or limits. If no reservation or limit exists for a system profile, its agents can use Agent Hours.

License assignment example

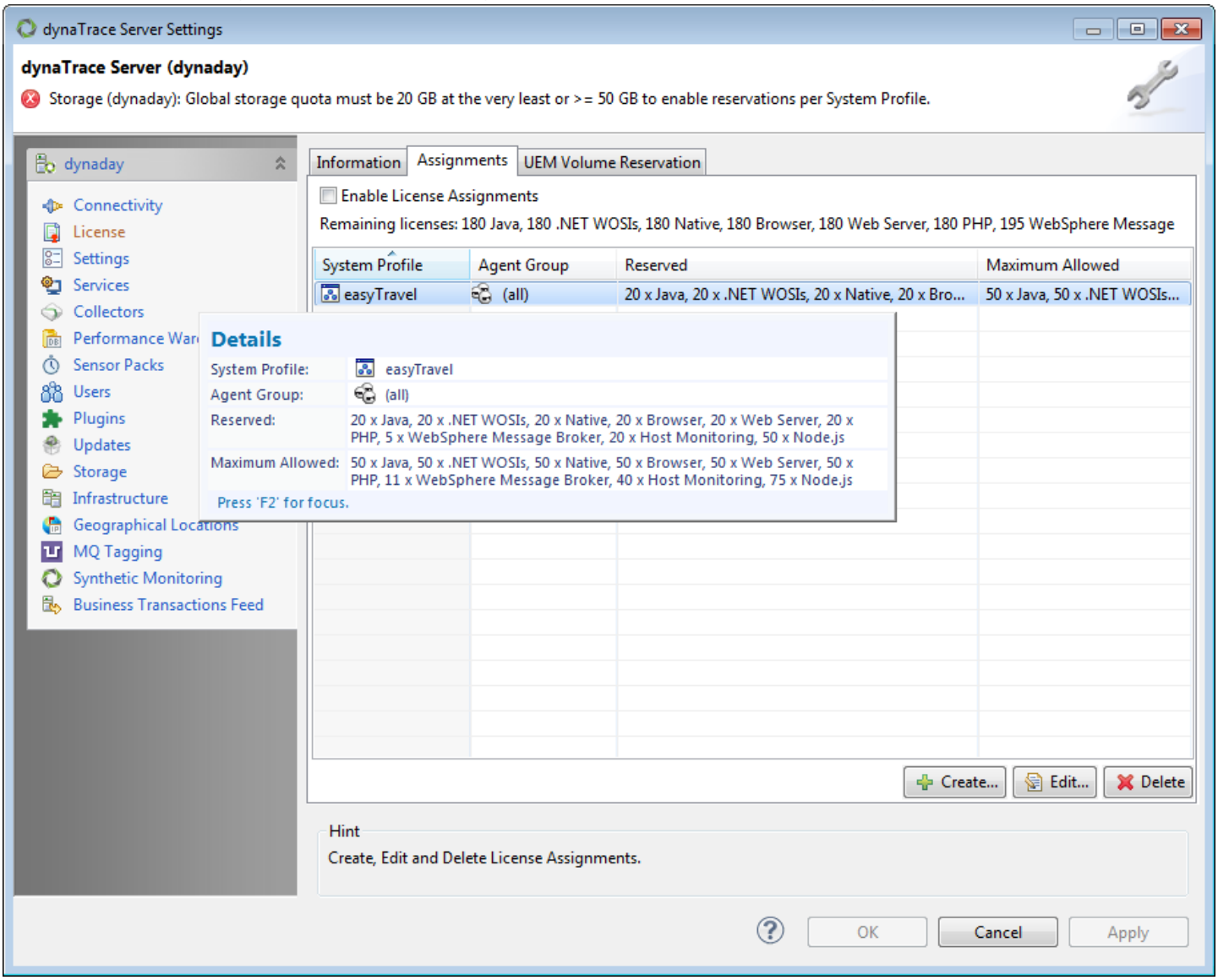
In this example, assume that you have purchased 100 Java Agent licenses. In your deployment, you have the following three System Profiles:

- Test
- Production A
- Production B

To ensure that the Java Agent connections made in the Test System Profile don't consume all of the available licenses, you can set the maximum allowed limit to 40. In addition, you can guarantee that 20 licenses for Java Agents of the Test System Profile will always be available by creating a reservation for 20 Java Agents.

Similarly, to safeguard against the Production A or Production B System Profile consuming too many of the remaining 60 Agent licenses, you can create a reservation for 30 Java Agents in each of these System Profiles.

Create license assignments on the Assignments tab of the Server Settings License pane



Inheritance

System Profile Assignments apply to all agent groups for which no Assignments have been specified. Assignments for agent groups don't inherit settings from assignments for the corresponding system profile.

For example:

- System Profile "A" has a reservation for 10 Java Agents and 10 .NET WOSIs and a maximum of 10 ADK agents allowed.
- AgentGroup "A_1" belonging to system profile "A" has as reservation for five Java Agents.

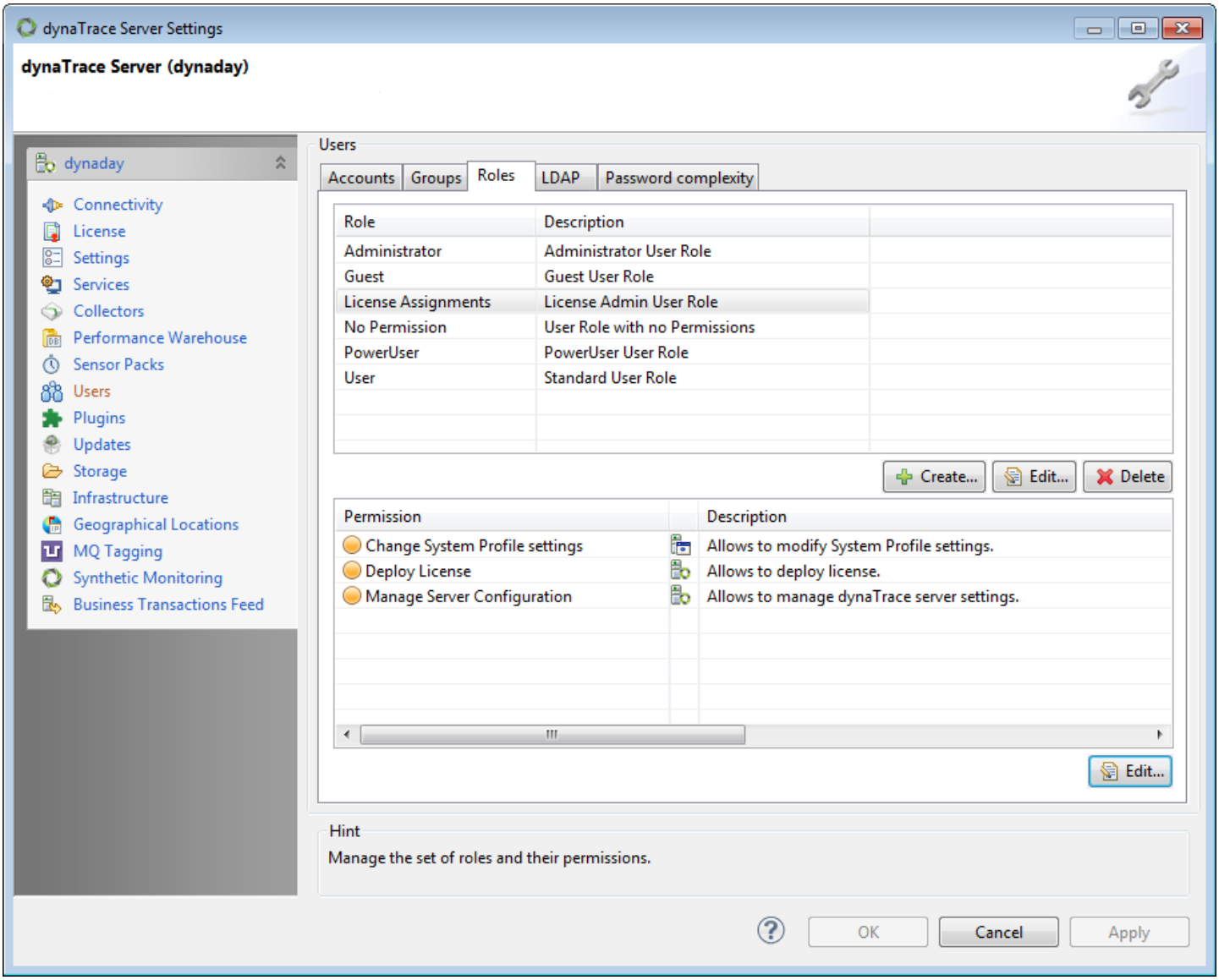
The five java agents for “A_1” are not deducted from “A”, meaning that for system profile “A” without “A_1” 10 Java Agents are reserved, 15 in total for “A” and “A_1”. .net WOSIs connected to agent group “A_1” will not use agent licenses which are reserved for system profile “A”. The number of ADK agents that are allowed to connect to agent group “A_1” is not limited.

Required permissions

To create, edit, or delete license assignments, your user account must have the following permissions enabled:

- Change System Profile Settings
- Deploy License
- Manage Server Configuration

Enabling license reservations requires specific permissions



Set up a license assignment

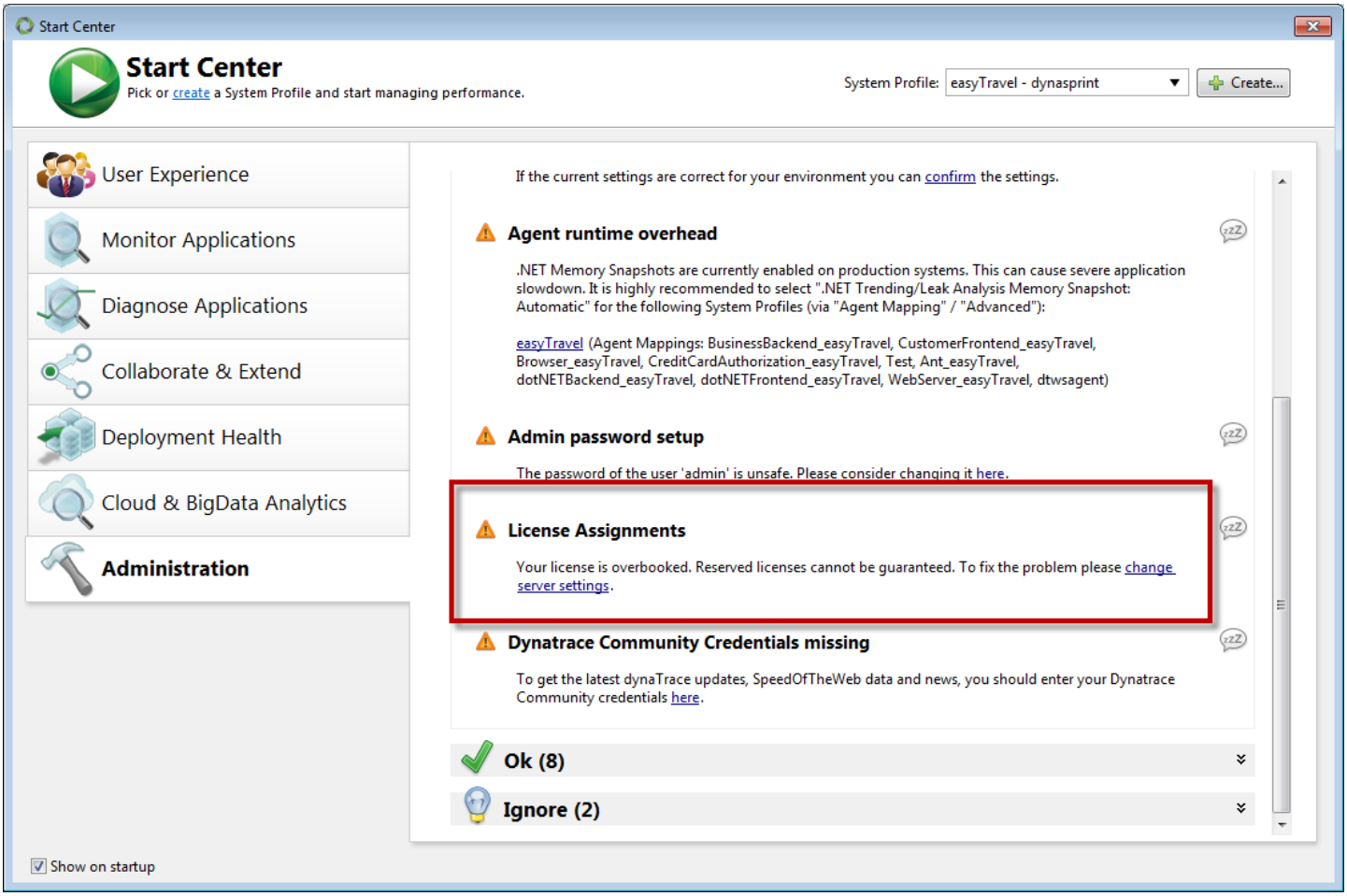
1. Choose **Settings > Dynatrace Server > License** to access the License pane of the **Server Settings** dialog box.
2. Click the **Assignments** tab in the right pane. On this tab, you can create, edit, and delete license assignments using the buttons at the bottom of the table. The number of remaining available Agent licenses are shown above the assignments table.
3. Select the **Enable License Assignments** check box.
4. Click **Create** to open the **Create New License Assignment** dialog box.

5. Select a System Profile and optionally an Agent Group. If an Agent Group is not selected, the assignment applies to all Agents of that System Profile that don't belong to an Agent Group with its own assignment. Remaining (unassigned) Agent counts are available for all Agents in the order in which they are connected.
6. Assign the number of Agents you want to reserve and/or the maximum number of Agents allowed for a given technology type. The maximum number depends on the number of Agents allowed by your license. Also define whether your reservations or limits may consume agent hours.

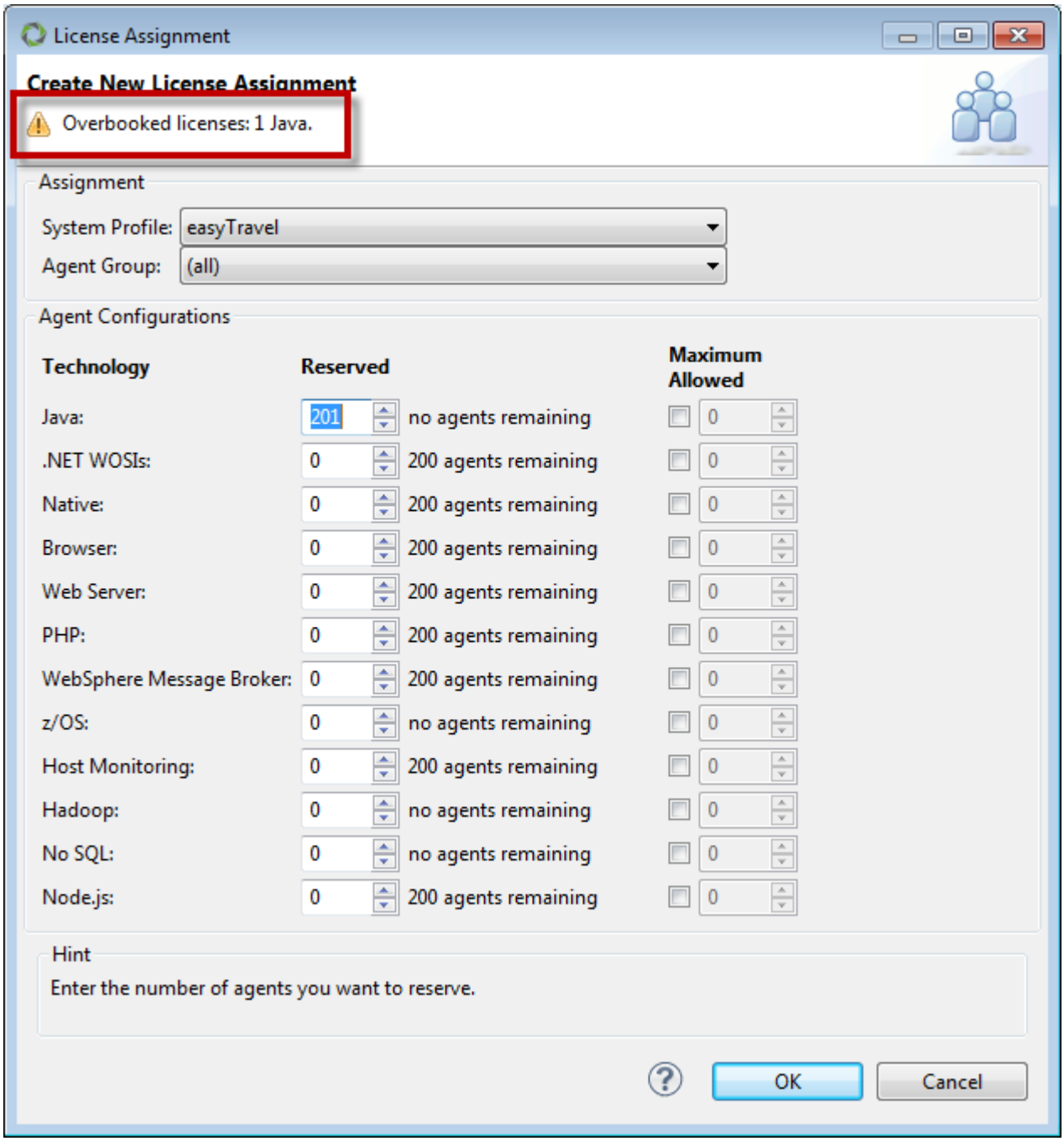
Overbooking

Overbooking occurs when the number of reservations exceeds the number permitted by your license. Overbooking can occur during the initial reservation assignment or when you change your license and decrease the number of available agents to fewer than what you originally reserved.

If you overbook a reservation, the number of agents allowed by your license still connects, however, you receive a notification on the **Start Screen** and **Server Settings** dialog box, as shown below:



Overbooked license warning on administration tab of Start Center



Overbooked license warning on License Assignment dialog box

Licensing details

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Configuring a proxy server to connect to eServices

Licensing models for supported technologies

Merging and splitting licenses

Related pages

Online license activation and deactivation

License assignments

Configuring a proxy server to connect to eServices

If you cannot connect to eServices from the Client, you may need to configure a proxy server.

1. From the Client, choose **Settings > Dynatrace Client**.
2. Click **Services** in the left pane of the Client Preferences dialog box.
3. Under **Proxy for access to external services**, select **Manually configure proxy** and enter the host, port, and authentication method.

Licensing models for supported technologies

This section outlines how Agents are licensed for different technologies and how the allowance per Agent type and UEM visits volume per AppMon Server is enforced by the Server.

For finer legal details, actual pricing of Agents, bundles and packs, and information currency, the price book precedes this documentation. For information on other details such as what AppMon edition and version is licensed, the type of license, and the validity time frame, see the Licensing page, the **Client Settings menu > Dynatrace Server… > License** vertical tab, eServices or talk with sales.

There is a three day licensing grace period and you can generate an emergency license in eServices.

See System requirements and the Deployment Guide for technical limit details.

Most technologies are licensed on Agent-per-process bases, with a certain number of each Agent type licensed per AppMon Server.

Restrictions such as the maximum number of cores a DB Agent is licensed for may apply, and there are notable exceptions like UEM (visits-based), DB (per DBMS instance) and mainframe CICS and IMS (region-based).

Technology	License	Comment
Java	Per Java Virtual Machine (JVM)	Each JVM with an injected Agent consumes one Java license on the AppMon Server.
.NET	Per Windows operating system instance (WOSI)	Even when more than one .NET Agent runs on a WOSI, only one .NET license is consumed.
Web Server	Per Agent in Apache, IIS, NGINX or IBM HTTP server	No license is necessary for Web Server (master) Agent services / daemons, but this is also the reason that slave / core Agents deliver host monitoring data.
PHP	Per (parent) process ID (PID)	Considered as children: module worker processes and threads, CGI and FastCGI If PHP-FPM (FastCGI Process Manager) is used and configured as one pool of FPM workers and another virtual host and pool (parent 2), then two licenses are used.
Node.js	Per process	Node.js spawns one process per CPU core to handle requests.
UEM	Consumes a UEM visit	See Manage UEM Visits Volume for a discussion of UEM visits-based licensing and which visits reduce the licensed visits volume.
Mobile / Android and iOS	Consumes UEM visits	The Agent generates events which become part of a visit, all licensed based on UEM visits. No other Agent license is required.
Native (ADK)	Per process	Each instance of a C++ program requires a license. No further license is required for a native part, if an app already has a license.

Rich / fat clients	Per instrumented client	Rich / fat clients based on Java, .NET or native require licenses for the respective technology, but pricing is different and technical limits apply.
Database	Per DBMS instance	Oracle/SQL Server: A license is consumed for each DB instance with a maximum of eight CPU cores per license. Every other instance or eight cores consume another license. MySQL: A license is consumed for each DB instance - no limitation on the number of CPU cores. See Database Agent Troubleshooting for more details on how the CPU cores are retrieved from the Database.
Infrastructure / Host Monitoring	Per host (Agent)	Hosts explicitly instrumented with a Host Monitoring Agent consume such a license. Java, .NET, Web Server, PHP and native (ADK) Agents deliver monitoring data and need no additional license.
IBM WebSphere Message Borker (WMB) / IBM Integration Bus (IIB)	Per process / execution group	
Mainframe / z/OS CICS and IMS	Per region (CICS / IMS) Per JVM if Java is instrumented on z/OS	Each CICS or IMS region with an injected Agent consumes one z/OS license. If a JVM is instrumented on z/OS it consumes a Java license. No zLocal or zRemote Agent is necessary for Java.
NoSQL	Per (Agent in) Java Virtual Machine (JVM)	Special NoSQL Agents are consumed first for Cassandra JVMs. More expensive Java Agents can optionally jump in. As of 2017 May, HBase instrumentation has been discontinued.
Hadoop		As of 2017 May, instrumentation has been discontinued.
Browser	Per browser instance Only maintenance licenses are issued.	Browser Agents are end of life. They can be used if compatible with the browser version in question.

Merging and splitting licenses

You can merge multiple licenses into a single license to increase the number of Agents allocated for the single license. Conversely, you can split a single license into multiple licenses and redistribute the number of allocated Agents across the multiple licenses. In either case, you must contact the license management beforehand. The Licensing Team will then instruct you to deactivate the original license or licenses, and then perform a standard license activation.

License upgrade

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Upgrading a UEM license using self-service

Manually upgrade a license

Related pages

eServices

Web Server Master Agent configuration

Prepare to upgrade

Manage UEM visits volume

Manage usage based agent hours volume

Online license activation and deactivation

License assignments

Licensing details

There are several options for license upgrade. To choose the right option, decide on the appropriate period of time required to test your upgraded installation before activating the new license. The recommended option is a three days grace period after deactivation, which provides the fastest and most convenient results. The recommended option is the quickest and most convenient and results in a three day grace period after deactivation. Two alternative methods provide a longer grace period, but are more involved. This page describes all options below.

The instructions on this page assume an Internet connection is available for the Client to access eServices. See Manually Upgrade a License to perform a license upgrade without internet access.

Before performing a licensing upgrade, be aware:

- An active maintenance contract is required to enable the appearance of the **Upgrade** button on the eServices website.
- Due to the extensive list of new features and improvements in AppMon 2017 May, you must obtain a new license to migrate from a previous version. Additionally unused UEM volume requires an upgraded UEM volume voucher.
- For AppMon 5.0 and later, the exact number of Web Server Agents is counted against the number of licensed Agents. Be sure to account for an increase in the number of Agents used when upgrading to a new license. For more information, see Web Server Agent Configuration or contact your sales representative.

Recommended upgrade option

The general steps for performing the recommended upgrade option are:

1. Deactivate the license using online deactivation initiated in the Client.
2. In eServices, upgrade the license.
3. Activate the license using online activation initiated in the Client.

Follow these steps to upgrade your license using the recommended method

1. Click **Settings > Dynatrace Server > License** to access the **License** pane of the **Server Settings** dialog box.
2. Click **Deactivate**. The **Deactivate current license** dialog box appears. Be sure to select the correct Server if you have more than one.
3. Select **Deactivate current license online** and enter your Dynatrace Community account user name and password. Click **Finish**. The license deactivates along with the UEM volume. A voucher is created for any unused UEM volume. Although deactivated, the Server is still operational for one or three days (depending on your license type).
4. Access eServices and select **My Licenses**.
5. Expand the information for the deactivated license and click **Upgrade**.
6. Select the target version and the **Direct** mode. Click **Next**.
7. If applicable, upgrade any UEM volume vouchers. See Upgrade a UEM volume voucher to learn how.
8. In the Client, click **Import** on the **License** pane of the **Server Settings** dialog box.
9. Select **Retrieve license online** and enter your Dynatrace Community account user name and password. Click **Next**.

0. On the **Available Licenses** page, select the license to activate and click **Next**. The license file is downloaded from eServices, imported into the AppMon Server, and activated online.
1. If applicable, add UEM visits volume.

Upgrade option 1 - trial license

Use this upgrade option if you need a grace period of more than three days to test your upgraded installation. This option creates a three-weeks trial license. After completing upgraded installation testing, you then create a full license and deactivate the previous version license.

Follow these steps to upgrade your license using option 1

1. Access eServices and then select **My Licenses**.
2. Expand the information for the license to upgrade and click **Upgrade**.
3. Choose the target version and **Trial** mode. Click **Next**.
4. An informational message appears stating that you are creating a three-week trial (evaluation) license. Click **Next**.
5. If you must perform a manual activation, download the trial license key file. Otherwise, click **Finish and Close**.
6. In the Client, click **Import** on the **License** pane of the **Server Settings** dialog box.
7. Select **Retrieve license online** and enter your Dynatrace Community account user name and password. Click **Next**.
8. On the **Available Licenses** page, select the trial (evaluation) license and click **Next**. The license file is downloaded from eServices, imported into the AppMon Server, and activated online.

After completing the testing of your upgraded installation, deactivate and install a full license using the recommended upgrade alternative described above.

Upgrade option 2 - temporarily skip license deactivation

The second upgrade option also provides a time extension to test your upgraded installation. Use this option to create a full license but skip the deactivation step until you have completed your testing.

> **Important**
>
> After creating and activating an upgraded license using this method, you are legally restricted from using the previous version license.

Follow these steps to upgrade your license using option 2

1. Access eServices and select **My Licenses**.
2. Expand the information for the license to upgrade and click **Upgrade**.
3. Choose the target version and select **Direct** mode. Click **Next**.
4. Choose **Skip Deactivation and Upgrade** and click **Next**.
5. If you must perform a manual activation, download the license key file. Otherwise, click **Finish and Close**.
6. In the Client, click **Import** at the **License** pane of the **Server Settings** dialog box.
7. Select **Retrieve license online** and enter your Dynatrace Community account user name and password. Click **Next**.
8. On the **Available Licenses** page, select the new upgraded license and click **Next**. The license file is downloaded from eServices, imported into the AppMon Server, and activated online.
   After completing the testing of your upgraded installation, the previous version license must be deactivated.

Upgrading a UEM license using self-service

If you have remaining UEM volumes in your current license, you must deactivate them before you deactivate that license to have those remaining volumes available for the new license. See Manage UEM Visits Volume to learn how.

Upgrade a UEM volume voucher

When applicable, upgrade any UEM volume vouchers.

1. Click the **UEM Volume Vouchers** tab.
2. Click **Upgrade**. The **Upgrade UEM Volume Voucher** dialog appears.
3. Select the target version and click **Upgrade**.

Manually upgrade a license

If the Internet connection is not available for the Client, you can upgrade a license manually.

If you are unable to connect to eServices from the Client, you may need to configure a proxy server. For more details, see Configuring a Proxy Server to Connect to eServices.

Perform the following steps to manually upgrade a license.

1. Deactivate the current license. See License Deactivation via File for more information. If you're using a License File, see License Activation and Deactivation.
2. Upgrade any UEM volume vouchers. See Upgrade a UEM volume voucher to learn how.
3. Access eServices and upgrade the license.
4. Activate the upgraded license. See License Activation via File for instructions.

Set up communication connections

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Default ports used

Set up AppMon Server communication

Set up Server-embedded Collector connections

Related pages

Certificate management

Configure authentication strings

Set up proxies

Set up SSL communication

This page introduces the default communication layer used by AppMon components, and provides access to information about configuring the settings needed to deploy AppMon into a networked environment.

The communication layer handles data transmission between AppMon components, which includes the Agent, Collector, Server, Frontend Server, and Client. The communication layer compresses, encrypts and authenticates and provides optimal performance and high throughput.

Default ports used

This table shows the default communication ports used for communication between AppMon components that are set up during installation.

Components to connect	Port non-SSL	Port SSL or e. HTTP	Protocol	Comment

Client to frontend server		2021	TCP	For AppMon 6.5 and later, non-SSL connections are no longer supported.
Client to Frontend Server		8023	HTTP	Tunnel via HTTP. Sent data is by default encrypted.
Frontend Server to Server		2031	TCP	The communication port between the Frontend Server and the backend server.
Collector to Server	(6698)	6699	TCP	For AppMon 6.5 and later, non-SSL connections are no longer supported. For compatibility reasons with pre-6.5 collectors the 6698 plain port is still available and must be activated using the Client running in debug mode.
Collector to Server		8033	HTTP	Tunnel via HTTP. Sent data is by default encrypted.
Collector to Server	8040	8041	HTTP(S)	For HTTP-based Agent connections[1]
Agent to Collector	9998		TCP	Or Agent to Server-embedded Collector if enabled for demo scenarios
Agent to Collector	8042	8043	HTTP(S)	The node.js Agent uses this connection.[1]
Memory Analysis Server to Server		7788	TCP	For AppMon 6.5 and later, non-SSL connections are no longer supported.
Browser to Server	8020	8021	HTTP(S)	Web interface for RESTful Server administration, to start the Webstart Client.
AppMon Web to Frontend Server		9911	HTTPS	Port on which browser-based dashboards get their data.
Slave Web Server agent to master Web Server agent	8001		UDP	The UDP port on which the master Web Server Agent should listen for data packets of the slave Web Server Agent

[1] These ports are used for the Node.js Agent (Linux only) to connect the Collector to the Server. There is no UI to change them. Edit communication port in `<DT_HOME>/collector/conf/collector.config.xml` *and* `<DT_HOME>/server/conf/`

Ensure that your firewall configuration allows communication between the components.

Note

For performance reasons, do not pass communication between the AppMon Agent and the Collector through firewalls. Place a Collector as close as possible to the Agents.

Set up AppMon Server communication

You can configure most AppMon Server connections using the AppMon Client. Select **Settings > Dynatrace Server > Services**.

Bind address

For machines with multiple network interface cards, the Server can be bound to a specific network adapter. This is useful when you want to disallow connections from other connected networks. To do this, enter the optional bind address next to the service port you want to restrict. Both host name and IP address work. You can specify bind address for both Client connections and Collector connections to the AppMon Server.

Note

By default, AppMon Server also uses the machine's DNS hostname, which may be ambiguous or unacceptable. To manually specify a name for the AppMon Server, stop the AppMon Server service, then edit the `DT_SERVER_HOME/server/conf/server.config.xml` file. Modify the `name` attribute of the `server` tag (XPath: `/dynatrace/serverconfig/settings/server/@name`). Save your changes and start the AppMon Server.

For example:

```
<settings>
...
<server
tunnelenabled="true" tunneladdress="" tunnelport="8023"
upperdeletiontriggerlimitenabled="false" upperdeletiontriggerlimit="51200" lowe
eventstatistics="false"
id="12345678"
jmxexportmanagement="false" jmxexportmonitoring="false" jmxport="1099"
name="MyNewHostName"
watchdogtimeout="10"
selfmonitoringenabled="true"
hascollector="true" />
...
</settings>
```

Make a backup copy of this file before you edit.

Note

If a configured port is already in use, or cannot be bound for any other reason, the Server restart fails. You must manually change the port to a free port. To do this, edit the corresponding attribute in the file `server.config.xml` in `<DT_HOME>/server/conf`. The `settings` tag (for example, XPath) specifies the ports for the single components:

- AppMon Client: `/dynatrace/serverconfig/settings/client/@clientsslport`
- AppMon Collector: `/dynatrace/serverconfig/settings/collector/@collectorport`
- AppMon Memory Analysis Server: `/dynatrace/serverconfig/settings/oopanalyzer/@port`

Set up AppMon Client connections

To connect the Client to the Server you must configure host / IP port. Use the Client Connection strings area of the General Tab (services vertical tab) if you need to make changes to the default Client connection settings.

The connection types are as follows:

- **Default:** The default connection is SSL encrypted and compressed.
- **HTTP Tunnel:** The HTTP Tunnel uses only HTTP traffic between the AppMon Client and the AppMon Server. Use this setting if restrictive firewalls are in use and/or if reverse proxies must forward the traffic.
- **Via Proxy:** You can specify a forward proxy with this setting. Use this setting if a forward proxy (HTTP Connect) must forward the traffic.
- **HTTP Tunnel Via Proxy:** This connection style combines the HTTP tunnel with the ability to specify a forward proxy. Use this setting if the traffic must be routed through a proxy chain that consists of forward and reverse proxies.

> **Note**
>
> Non-SSL connections for AppMon Client connections are not allowed.

For details on how to configure the HTTP Tunnel and forward proxies, see Set up Proxies tunneling and forward proxy.

> **Note**
>
> For changes to take effect, the Server must be restarted.

Set Up AppMon Collector communication

Once a Collector is connected to the AppMon Server, you can change Collector connection settings such as the Server port to connect to or listen for in the General tab of the Services vertical tab.

Secure connections (SSL)

For AppMon 6.5 and later, the Server only supports SSL connections. For compatibility, any pre 6.5 Collector wanting to connect to the Server can use the “plain” Server port. To enable that port, select the **Allow non-SSL connections** check box. This starts the plain Collector port on the next restart of the Server.

> **Note**
>
> Non-SSL communication is not available in AppMon 6.5 and later.

Specify communications settings in the configuration file

You can also specify communication settings between theCollector and the Server in the Collector configuration file, which is located at `<CollectorHome>/conf/collector.config.xml`. The most important settings are also available as startup parameters and override `collector.config.xml`:

```
dynaTrace Collector - Command Line Options

Usage:
  dtcollector.exe [-service (install|uninstall)] [-instance <instance>]
                  [-listen <ip>[:port]] [-server <host>[:port]]

Options:
  -service (install|uninstall)     (Un)install Windows Service
                                   (uninstall might require you to logoff)
  -instance <instance>             Instance name
  -listen <ip>[:<port>]            Listen for Agents on <ip>/<port>
                                   (overrides config file)
  -server <host>[:<port>]          Use Server at <host>/<port>
                                   (overrides config file)
```

This table explains connection-related XML attributes, values and defaults for the Collector:

XML attribute	Description	Value type	Default
serveraddress	The server host to connect this collector to.	String	‘localhost’
serverport	The server port to connect this collector to.	Integer	6699
useproxy	Whether or not to connect through a proxy.	true, false	false
proxyhost	The HTTP proxy host name to connect to.	String	*empty*
proxyport	The HTTP proxy port to connect to.	Integer	8080
useproxyauthentication	Whether or not to authenticate to the configured proxy.	true, false	false

proxyusername	The username to authenticate with at the configured HTTP proxy.	String	*empty*
proxypassword	The password to authenticate with at the configured HTTP proxy.	String	*empty*
usepreemtiveproxyauth	Enables and disables preemptive proxy authentication.	true, false	true
compress	Whether or not to compress traffic.	true, false	false
usetunnel	Whether or not to use HTTP Tunneling.	true, false	false
tunnel	Server tunnel address. Only change host name or port on demand.	String	'http://localhost:8023/tunnel'

Set up Server-embedded Collector connections

Only demo or POC environments can use the Server-embedded Collector and are allowed to bypass the stand-alone Collector and directly connect Agents to the AppMon Server.

So by default, the **Allow Agent Connections to Dynatrace Server** check box is not selected, which closes this port on the Server.

Set up built-in webserver connections

The AppMon Server has a built-in web server that offers SOAP/REST web services for automation and the remote installation of AppMon Clients via Java Web Start. You can also use JMX to monitor AppMon Server and AppMon Client internals. This is similar to the SOAP/REST web services. All related settings are available in the **Management** tab of the **Services** item.

> **Note**
>
> The AppMon Server requires sufficient user privileges, depending on the automation operation.

Set up Webstart Client communication

Use settings in the **Webstart** tab of the **Services** item to set up the connection between the Webstart Client client and the AppMon Server. For more information, see Configure Webstart Client.

Set up E-mail communication

For more information, see Email configuration.

Set up AppMon Memory Analysis Server communication

Use the **Dynatrace Memory Analysis Server** tab of the **Services** item to configure the connection settings to the AppMon Memory Analysis Server. For more information, see Set up a Memory Analysis Server.

Setup AppMon Web communication

See server.config.xml to configure communication ports for AppMon Web.

Set up SSL communication

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Enable SSL encryption

Related pages

Set up communication connections

Certificate management

Configure authentication strings

Advanced features - certificates, private keys and keystore

This page describes how to set up SSL / encrypted communication between AppMon tiers. Use SSL encrypted communication.

Data sent between browsers and web servers is typically in plain text. SSL encryption is a protocol that provides a layer of data security in situations where a network may have unsecured elements such as non-WPA2 wireless data transfer.

Enable SSL encryption

AppMon Server - Collector connection

By default Collectors connect via SSL (default port 6699) to the Server and the latter accepts both, SSL (default SSL port 6699) and non-SSL (default non-SSL port 6698) connections.

If you want to disallow non-SSL and you got the Collectors running, restart the Server before you switch off non-SSL. Make sure you change the Collectors to the SSL port used to connect to the Server.

You can change the port for connected Collectors in the Client by selecting **Settings > Dynatrace Server > Collectors**, then selecting a Collector in the list and changing configuration in lower pane.

For disconnected Collectors, please change the port to SSL in `collector.config.xml` manually as shown below.

AppMon Client - AppMon Server connection

Both SSL encryption (default port 2021) and compression are enabled by default. It is no longer possible to enable the non-SSL communication port (2020) of the AppMon Server via the UI. Also see Client Communication Setup.

AppMon Agent - Collector connection

To enable SSL connections between Collector and AppMon Agent, you need to set the `DT_USESSL` environmental variable, or the `usessl` parameter as `true`. See the configuration topic for a particular Agent to learn how. Encrypted communication uses the same port as non-encrypted.

Web Server agents

The Apache, IIS, and NGINX Agents don't communicate with the Collector directly.

- On the Classic Agent platform, the Web Server master Agent communicates with the Collector. See Web Server Master Agent configuration for the configuration instructions.
- On the OneAgent platform, the OneAgent communicates with the Collector. The OneAgent uses SSL by default, you just need to make sure that the required Collector port is open. See OneAgent configuration for the configuration instructions.

Manually change the AppMon Collector port to SSL

If the AppMon Collector is not connected to the AppMon Server, you can only change the port by editing the collector configuration file. The collector configuration file is installed to `<DT_HOME>/collector/conf/collector.config.xml` by default.

Set the port that the Collector should use to connect to the AppMon Server by changing the `serverport` attribute of the `<collectorconfig>` element. (XPath: `/dynatrace/collectorconfig/@serverport`). By default the Server listens for SSL connections on 6699.

If you want the tiers to communicate on a different port, you must also change it in Server settings. In the Client, select **Settings > Dynatrace Server > Services > General** and set the SSL Listen Port. When the Server is stopped, edit `collectorsslport` in `<DT_HOME>/server/conf/server.config.xml`.

```
<dynatrace ...>
   <collectorconfig serverport="6699" serveraddress="localhost" ... />
   ...
</dynatrace>
```

Configure authentication strings

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Set up SSL communication

Certificate management

A basic means for authentication in AppMon Server-Collector-communication is provided by authentication-strings.

If enabled, the Collector sends an MD5 hash of a configured authentication-string to the Server, which checks whether its own configured authentication-string matches.

> **Note**
>
> If communication is not encrypted such an MD5 hash can be captured quite easily via a man-in-the-middle attack, so it is advisable to use this feature in combination with encryption / SSL.

- For the Server-side, the authentication-string is configured using the `collectorauthstring` attribute in the corresponding `<collector>` element of `<DT_HOME>/server/conf/server.config.xml`, XPath: `/dynatrace/serverconfig/settings/collector/@collectorauthstring`. Add the plain-text password here and ensure this file cannot be read by unauthorized personnel.
- For the Collector-side (`<DT_HOME>/collector/conf/collector.config.xml`) the `authstring` attribute in the `<collectorconfig>` element must contain the same plain-text password (XPath:`/dynatrace/collectorconfig/@authstring`). Again, ensure that this file cannot be read by unauthorized personnel.

The following XML fragments illustrate how to configure an authentication-string between the Collector and Server.

- AppMon Server configuration file:

```
<dynatrace ...>
   ...
   <settings>
      <collector ... collectorauthstring="abc" ... />
      ...
   </settings>
   ...
</dynatrace>
```

- AppMon Collector configuration file:

```
<dynatrace ...>
   <collectorconfig .... authstring="abc"... >
   ...
</dynatrace>
```

Set up proxies

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

HTTP tunneling

Connect using forward proxies

Example reverse proxy configuration (using apache2 on ubuntu)

Related pages

Set up communication connections

Set up SSL communication

This page describes information needed to set up HTTP communication proxies and use HTTP tunneling for situations where you require communication through complex proxies or firewall chains.

Overview

AppMon endpoints such as the AppMon Client, AppMon Collector, and the AppMon Server are able to communicate through HTTP proxies to ensure compatibility with the vastness of the web and the complexity of intranet setups. AppMon supports *HTTP Tunneling*, which makes it possible to encompass all traffic between components in HTTP requests.

HTTP tunneling

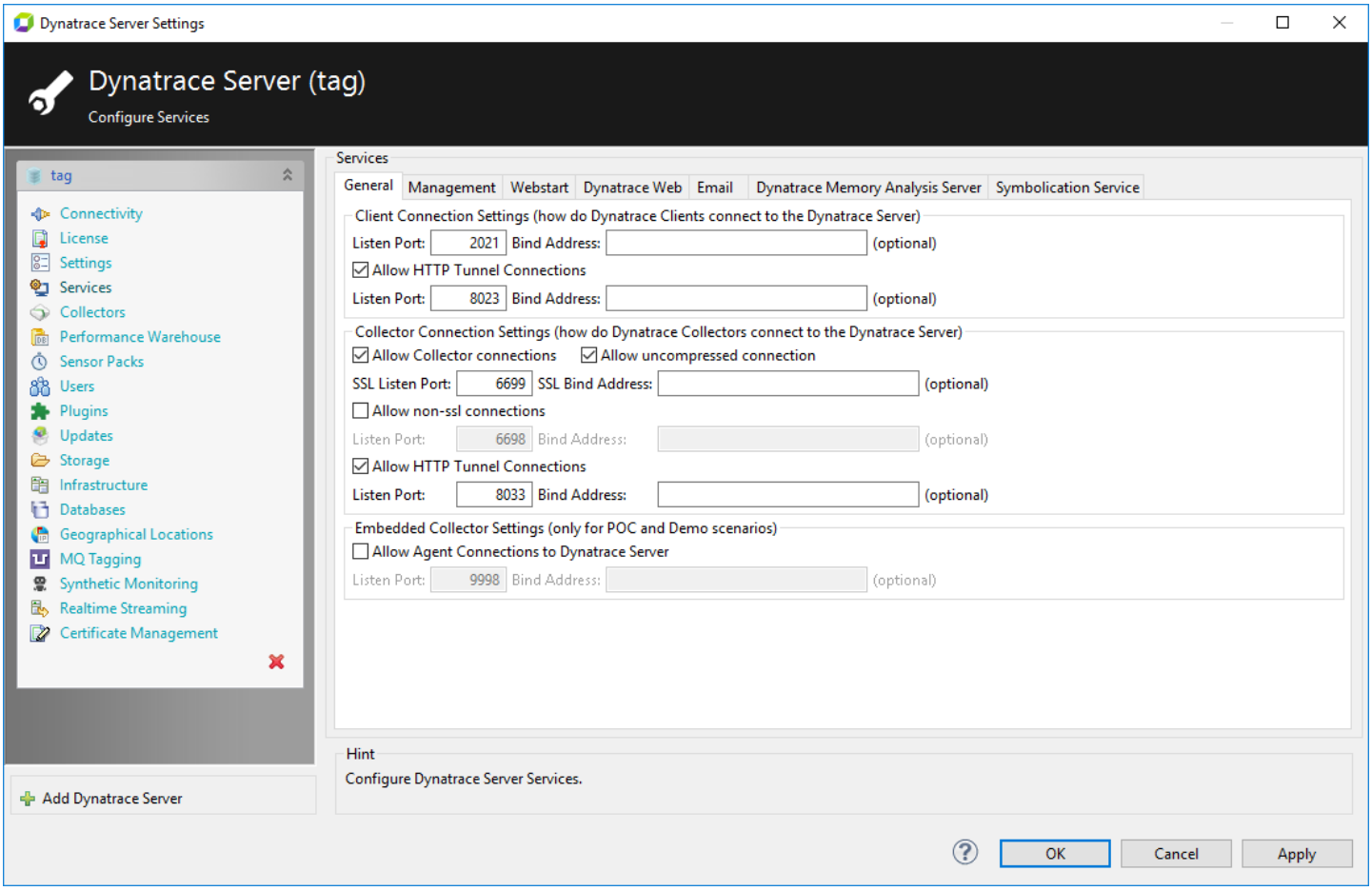
Use HTTP tunneling when you want to communicate through complex proxy or firewall chains. The AppMon HTTP tunneling approach uses standard HTTP POST messages to transmit all data necessary for communication between the AppMon endpoints. This makes AppMon fully HTTP reverse proxy compatible. **Use the HTTP Tunneling feature whenever reverse proxies are involved** in connecting two AppMon endpoints. However, due to the technical limitations of the HTTP protocol, this tunneling approach will increase traffic and affect throughput and performance.

Default ports

If enabled, the AppMon Server listens to the default HTTP tunnel port on 8023 for both the Collector and Client. Change this port in the **Services** pane of the **Dynatrace Server Settings** dialog box.

AppMon Server

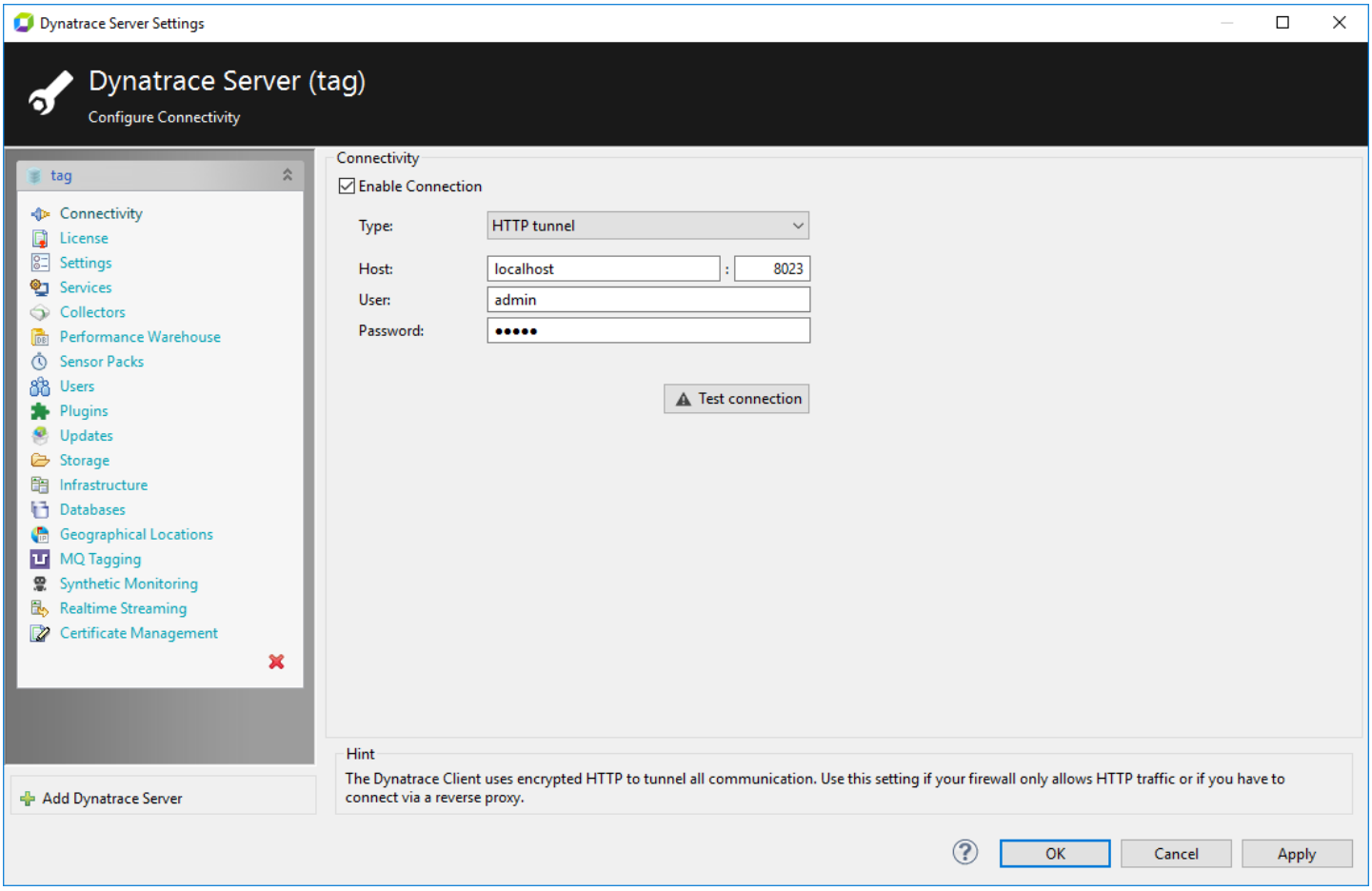
You can configure all other related settings in the AppMon Client in the **Services** pane of the **Dynatrace Server Settings** dialog box, as shown below.



Services pane of the server settings dialog box

AppMon Client

To connect the AppMon Client to the AppMon Server using a HTTP tunnel, the Server's HTTP tunnel port has to be configured in the AppMon Client. All options are available in the **Connectivity** pane of the **Dynatrace Server Settings** dialog box for the AppMon Server.



Connectivity pane of the server settings dialog box

AppMon Collector

While the AppMon Client and AppMon Server connectivity settings are configured in the UI, the AppMon Collector's setup is stored in `<DT_HOME>/collector/conf/collector.config.xml` on the machine where the AppMon Collector is installed.

This table lists the related XML attributes, values, and defaults:

XML attribute	Description	Value type	Default

usetunnel	Whether to use HTTP tunneling for this collector's connections.	true, false	false
tunnel	Server tunnel address. Change hostname or port on demand.	String	http://localhost:8023/tunnel

Connect using forward proxies

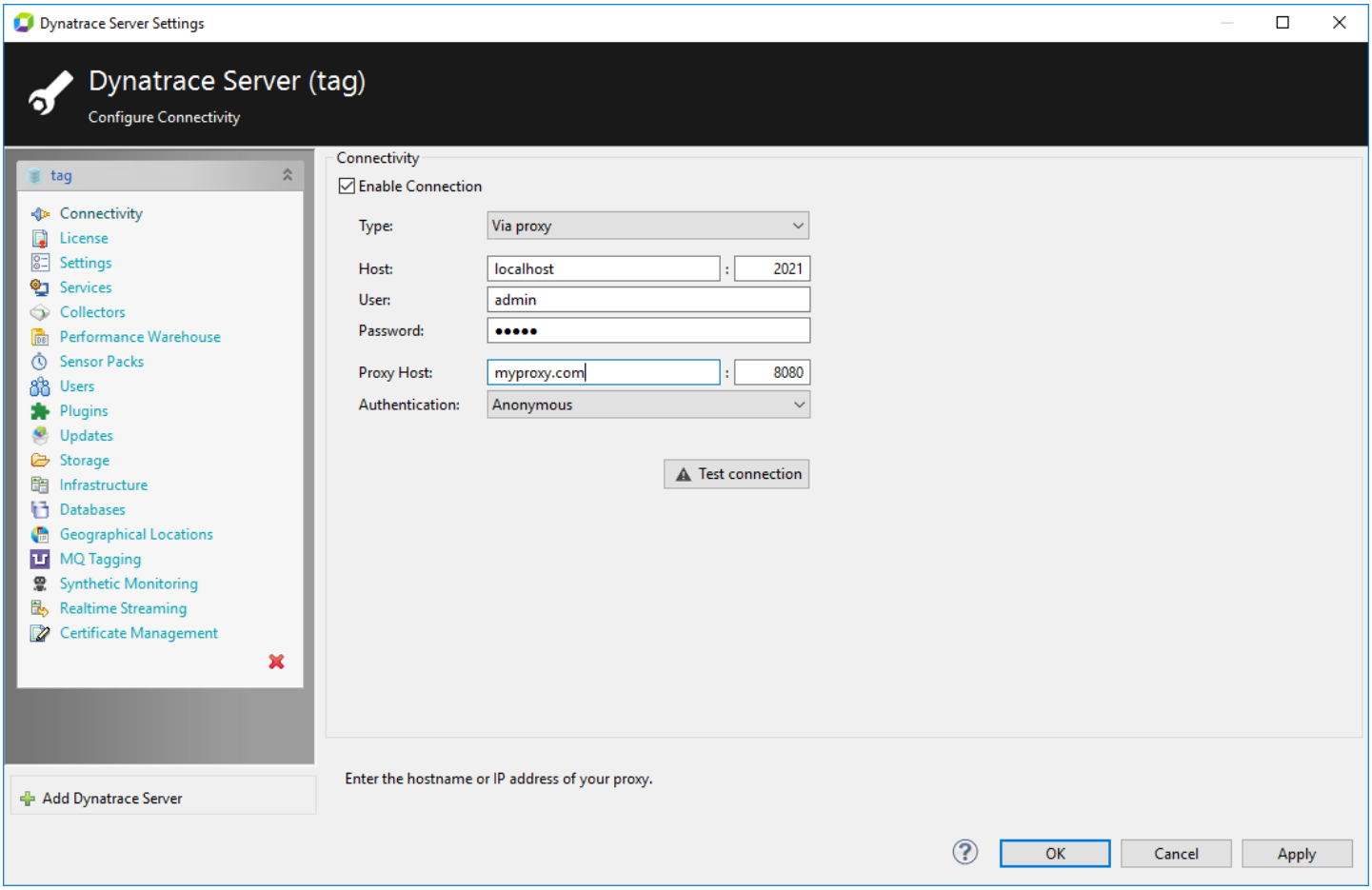
HTTP CONNECT is used for forward proxies. The active component initializing the connection must tell the proxy to which endpoint it wants to be connected using the CONNECT request. As soon as the connection is established, all communication is made without HTTP requests, so communication has less overhead and performance is better.

Connect the AppMon Client

Use the **Via Proxy** setting in the **Connectivity** pane of the **Server Settings** dialog box to connect an AppMon Client and the AppMon Server using a forward proxy.

Use the **Via Proxy** setting in the **Connectivity** pane of the **Server Settings** dialog box to connect an AppMon Client and the AppMon Server using a forward proxy.

Enter the proxy's address and port, and optionally enter user credentials if authentication is required. Preemptive authentication is used by default. Preemptive authentication can only be turned off by editing the Client's `<DT_HOME>/client/conf/client.config.xml` file and setting the `usepreemtiveproxyauth` attribute to false.



Connect the AppMon Collector

To configure an AppMon Collector, open the `<DT_HOME>/collector/conf/collector.config.xml` file and set the following XML attributes to your environment:

```
<dynatrace>
  <collectorconfig serveraddress="dynatrace" serverport="6699"
    useproxy="true" proxyhost="companyproxy" proxyport="8080"
    useproxyauthentication="true" proxyusername="myusername" proxypassword="mypass"
    compress="true">
  </collectorconfig>
</dynatrace>
```

Note

You must set the useproxy and optionally the useproxyauthentication attribute to true. Otherwise, the AppMon Collector will ignore your proxy settings.

Use a proxy password

To avoid saving any clear-text passwords in configuration files, the password to authenticate at the proxy server is transferred to a password-protected KeyStore.

AppMon Collector

There are two options to configure the proxy password at the Collector, but the effect of all options is the same. The password is transferred at the first startup to the KeyStore and deleted in the according config file, including the identifier of the option. To update the proxy password, you can add the identifier and the password again to one of the two configuration files, thus updating the existing password. Similarly, to delete the password, just provide an empty string as the password.

1. In `<DT_HOME>/collector/conf/collector.config.xml via "proxypassword=":`

```
<dynatrace>
  <collectorconfig serveraddress="dynatrace" serverport="6699"
    useproxy="true" proxyhost="companyproxy" proxyport="8080"
    useproxyauthentication="true" proxyusername="myusername" proxypassword="mypass"
    compress="true">
  </collectorconfig>
</dynatrace>
```

2. In `<DT_HOME>/dtcollector.ini`:

```
-Dtransfer.to.keystore.collector.proxypassword=mypass
```

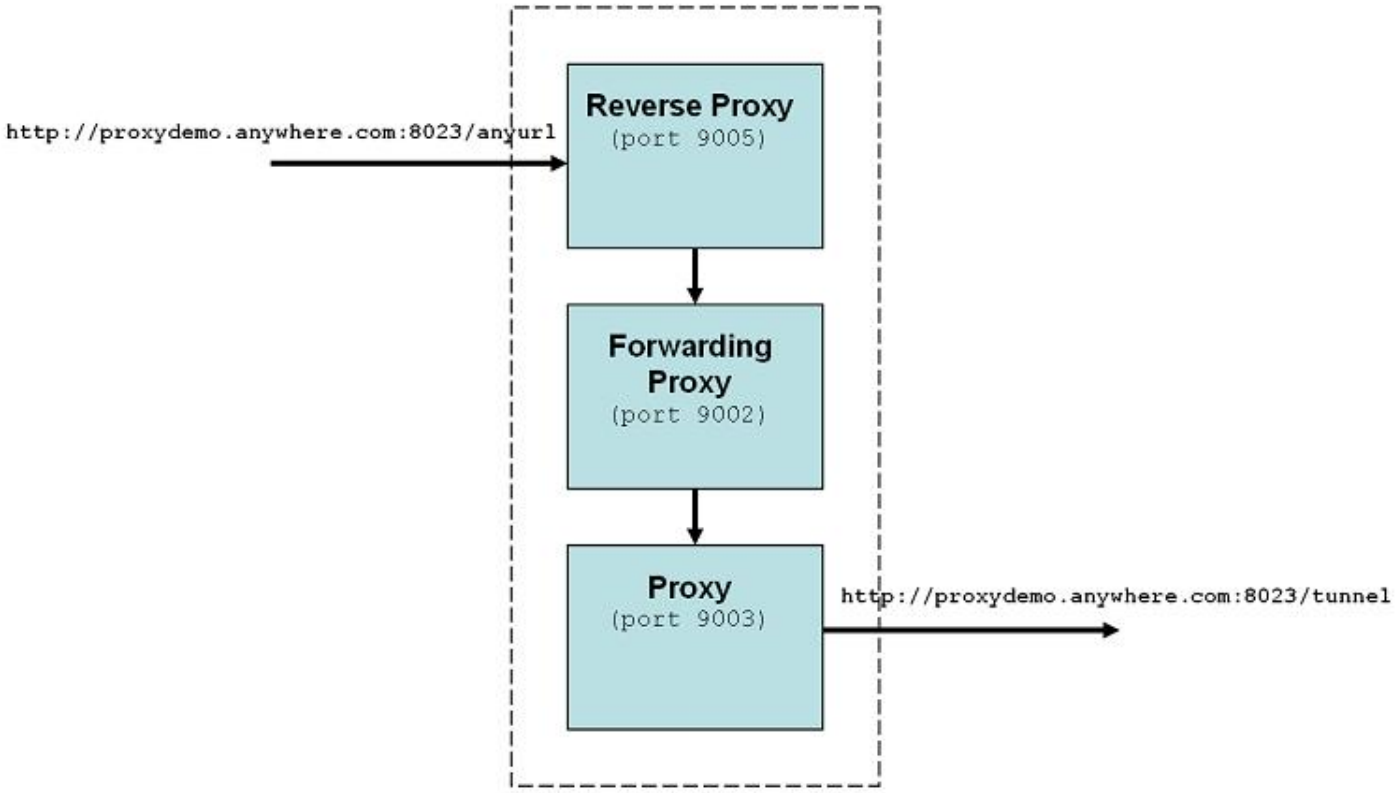
Example reverse proxy configuration (using apache2 on ubuntu)

Reverse proxy configuration is made through the `/etc/apache2/httpd.conf` file.

After changing the configuration, restart the apache2 daemon with the command: `/usr/sbin/apache2ctl restart`.

Log files can be found at `/var/log/apache2`.

The following example sets up a proxy chain consisting of three proxies running on one host using `httpd.conf`.



```
ServerName proxychain
LogLevel INFO

<Proxy \*>
  Order deny,allow
  Allow from all
</Proxy>

Listen 9005
NameVirtualHost \*:9005
<VirtualHost \*:9005>
  ProxyRequests Off
  Header set Host "proxydemo.anywhere.com:8023"
  ProxyPass /anyurl http://proxydemo.anywhere.com:8023/tunnel
  ProxyPassReverse /anyurl http://proxydemo.anywhere.com:8023/tunnel
  ProxyRemote * http://localhost:9002/
</VirtualHost>

Listen 9002
NameVirtualHost \*:9002
<VirtualHost \*:9002>
  ProxyRequests On
  ProxyRemote * http://localhost:9003/
</VirtualHost>

Listen 9003
NameVirtualHost \*:9003
<VirtualHost \*:9003>
  ProxyRequests On
</VirtualHost>
```

Setup reverse proxy to support SSL

Load SSL modules at Apache startup

Set links to SSL library and proxy SSL support in `/etc/apache2/mods-enabled`:

```
ln -s ../mods-available/ssl.load ssl.load
ln -s ../mods-available/ssl.conf ssl.conf
ln -s ../mods-available/ proxy_connect.load proxy_connect.load
```

Retrieve a certificate/key file

Either you already have a certificate or you can create a self-signed file (for testing purposes only - should not be used in production). See: http://www.debuntu.org/how-to-create-a-self-signed-certificate
See also: http://www.debuntu.org/ssh-key-based-authentication

Enable SSL for the reverse proxy

```
...

Listen 9005
NameVirtualHost \*:9005
<VirtualHost \*:9005>

  SSLEngine on
  SSLOptions +StrictRequire
  SSLCertificateFile /path/to/server.crt
  SSLCertificateKeyFile /path/to/server.key

  ProxyRequests Off
  Header set Host "proxydemo.anywhere.com:8023"
  ProxyPass /anyurl http://proxydemo.anywhere.com:8023/tunnel
  ProxyPassReverse /anyurl http://proxydemo.anywhere.com:8023/tunnel
  ProxyRemote * http://localhost:9002/
</VirtualHost>
...
```

Set up a forward or reverse proxy

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Authentication

General proxy setup

Related pages

Set up communication connections

Set up SSL communication

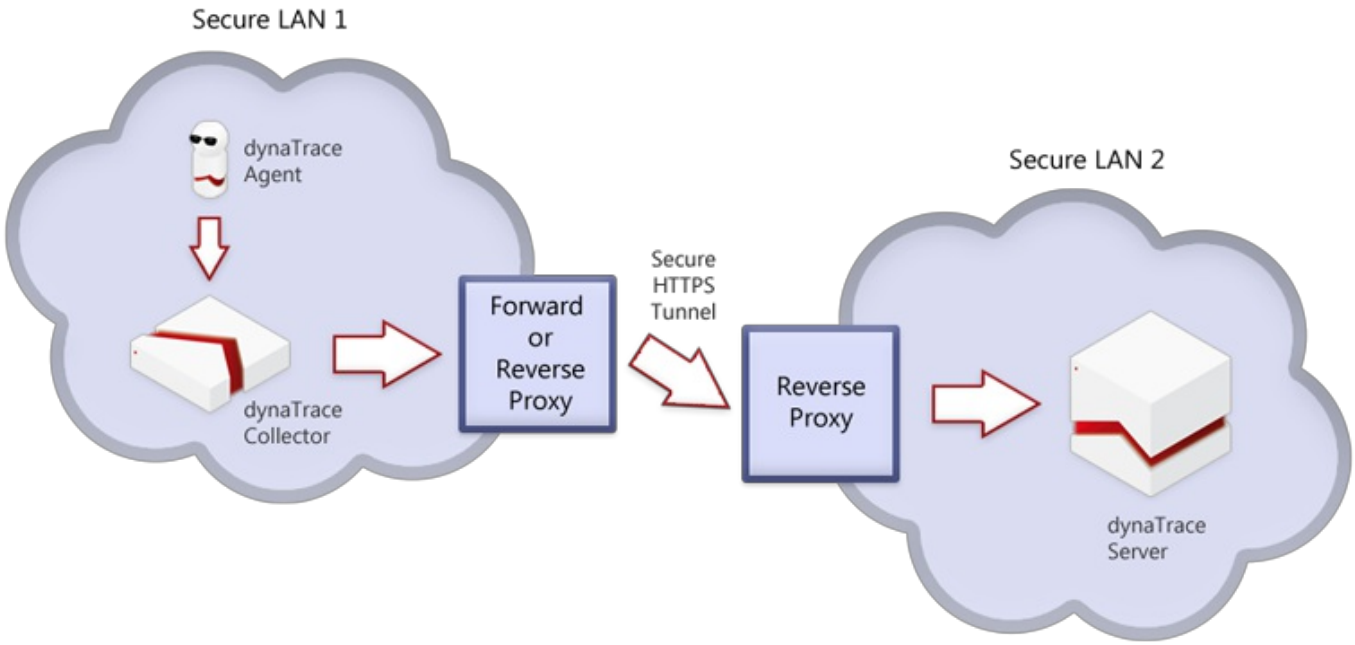
Set up proxies

This page explains how to set up a proxy server that is able to tunnel AppMon traffic using Apache 2.0.

Overview

AppMon can tunnel through any HTTP forward and reverse proxy. For more information on the differences between forward and reverse proxies, see http://en.wikipedia.org/wiki/Reverse_proxy.

Proxy tunneling is available for connections between the AppMon Server and AppMon Client and for connections between the Server and AppMon Collector. Tunneling traffic between an AppMon Agent and a Server or Collector is not possible. In case Agent traffic has to be tunneled, we recommended passing the Agent traffic through a local Collector, which is then able to tunnel the traffic through the proxy server.



AppMon uses the HTTP CONNECT method to create a TCP/IP tunnel through a HTTP proxy, which means it is necessary that the proxy server understands this HTTP 1.1 extension. It is also possible to relay an HTTP CONNECT tunnel through a chain of proxy servers. Any proxy server that understands the HTTP CONNECT method can relay its traffic to the next link in the chain until the final destination is reached.

Authentication

AppMon enables authentication with a HTTP forward proxy using *Basic HTTP Authentication*. You can enter your username and password in the AppMon Client GUI or the Collector config file and AppMon will initiate an authentication request against the configured proxy. Authentication with a *reverse* proxy is currently not available.

General proxy setup

Apache 2.0 must work as an HTTP CONNECT enabled proxy server to enable following Apache modules:

- mod_proxy (http://httpd.apache.org/docs/2.0/mod/mod_proxy.html)
- mod_proxy_http (http://httpd.apache.org/docs/2.2/mod/mod_proxy_http.html)
- mod_proxy_connect (http://httpd.apache.org/docs/2.0/mod/mod_proxy_connect.html)

A typical Apache installation contains those modules already - they just need to be enabled. Apache 2.0 modules may be enabled using the *LoadModule* directive within the `httpd.conf` configuration file.

Add the following lines to your `httpd.conf` file:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
```

Forward proxy setup

Usually a forward proxy works as a *gateway* for outbound traffic. The caller passes information about where it wants to connect to the proxy server and the forward proxy sets up a tunneled connection between the caller and the target machine. In AppMon, the AppMon Client or Collector passes an HTTP header to the proxy server that contains information about the target AppMon Server. Modify or add the following configuration lines to your `httpd.conf` to allow AppMon to tunnel through the Apache 2.0 proxy.

```
# forward proxy config for dynaTrace 3.x
Listen 9005
NameVirtualHost \*:9005
<VirtualHost \*:9005>
 ProxyRequests On
 AllowCONNECT 2020
</VirtualHost>
```

The above configuration setting tells Apache 2.0 to:

- Listen to port 9005
- Create a virtual host for port 9005
- Allow proxy requests to the virtual host
- Allow HTTP CONNECT requests to port 2020 on the target machine.

Reverse proxy setup

A reverse proxy redirects proxy requests to the configured machine and port and is mainly used to ensure network security and enable some load balancing. Modify or add the following configuration lines to your `httpd.conf` to make sure AppMon can tunnel through your reverse proxy installation.

```
# reverse proxy config for dynaTrace 3.x
Listen 9006
NameVirtualHost \*:9006
<VirtualHost \*:9006>
 ServerName bruckner
 ProxyPass / bruckner:2020
 ProxyPassReverse / bruckner:2020
 AllowCONNECT 2020
</VirtualHost>
```

The above configuration setting tells Apache 2.0 to:

- Listen to port 9006
- Create a virtual host for port 9006
- Forward all proxy requests to bruckner:2020
- Allow HTTP CONNECT requests to port 2020.

Certificate management

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Setting up SSL communication

Migration from customer-provided keystores

Currently used certificate

Deploy a new private key pair and certificate

Troubleshooting SSL communication

Related pages

Set up SSL communication

This page describes how to manage the certificate chain and key pair used by the AppMon SSL communication subsystem.

Setting up SSL communication

By default, the AppMon Client connects on port 2021 using SSL (no UI to change that), so they use the mechanism right away.

By default Collectors connect to an AppMon Server on port 6699 (SSL), so if you want to use non-SSL, select connected Collectors in the Client by selecting **Settings > Dynatrace Server > Collectors** and changing Connect to Server to 6698.

For disconnected Collectors change `serverport` in `<DT_HOME>/collector/conf/collector.config.xml` (or in `<DT_HOME>/collector/instances/<instanceName>...`)
So far Agents do not use SSL.

See Set up SSL Communication for a more detailed description.

Migration from customer-provided keystores

Customer-provided `.jks` keystores are not supported for AppMon 6.3 and later and are not automatically migrated. Instead of replacing keystores on every component manually, use settings in **Settings > Dynatrace Server > Certificate Management** to change the certificate and key pair used to secure the communication. The certificate and keys are automatically deployed to all components where they are needed.

In case you want to avoid the keys to be deployed to the Client and Collector, please consult the Advanced Features - Certificates, Private Keys and Keystore page for further information.

The customer-provided keystore used by the Web UI (given as system property) is not automatically migrated. Please consult the Upgrade and Migration Guide for further information.

Currently used certificate

To examine the certificate and key pair that are currently in use, switch to the Overview horizontal tab. For details see the Certificate Overview page.

Deploy a new private key pair and certificate

The Deploy Certificate page documents the necessary steps to replace the currently used certificate with a new one.

Troubleshooting SSL communication

If you have components, that can't connect after replacing the certificate or wish to setup a more complex environment, please consult the Advanced Features - Certificates, Private Keys and Keystore page.

Certificate overview

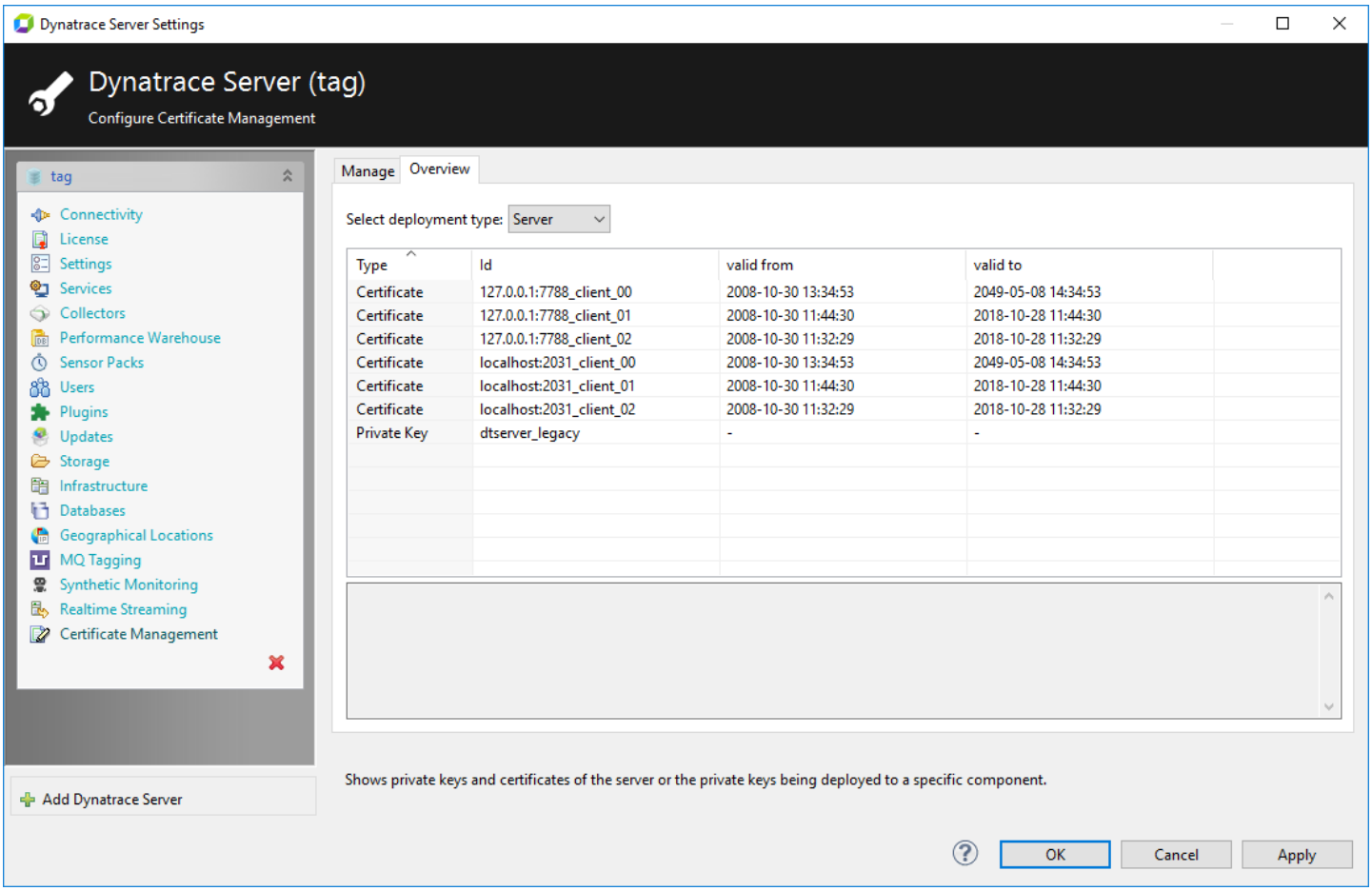
Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Certificate management

The **Settings > [Dynatrace Server or Dynatrace Client] > Certificate Management > Overview tab** displays the currently used key pair and an overview of all well known certificates of the respective AppMon tier when running.

Server certificate overview



Custom certificate requirements

```
$ openssl x509 -in certificate-chain.pem -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 6495580910137006250 (0x5a24f06c368658aa)
    Signature Algorithm: sha512WithRSAEncryption
        Issuer: CN=appmon CA
        Validity
            Not Before: Oct  3 22:00:00 2016 GMT
            Not After : Oct  3 22:00:00 2018 GMT
        Subject: CN=appmon
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:e9:60:00:31:c4:d0:04:12:cb:ea:e6:2e:92:d3:
                    8c:19:08:fb:5a:f5:5e:b6:41:3f:20:e3:dd:46:14:
                    4a:76:8b:fd:80:b2:54:a4:6b:b9:24:1a:f6:eb:bf:
                    fb:63:0b:ec:09:80:4a:87:aa:74:b6:08:c4:3a:8d:
                    ba:42:cd:e9:6d:ae:f6:f2:55:e8:0e:5f:90:7a:6e:
                    e2:5e:ad:d5:41:ed:0f:60:d1:b2:b3:39:49:c7:05:
                    29:a2:40:5f:90:4f:45:30:7e:89:6d:9b:65:cf:f5:
                    e5:c2:c7:35:ba:e3:30:f9:04:2d:cf:bb:b4:d9:55:
                    84:2f:50:46:71:cb:bd:e4:45:2f:e5:7f:1e:be:cd:
                    43:de:e1:d9:43:e0:b6:99:d4:01:24:a7:25:fd:06:
                    63:b5:dd:cf:16:ec:64:8a:3b:d0:2b:e6:64:f0:18:
                    b5:66:6c:81:34:04:56:16:c4:f2:1a:20:bb:cd:6b:
                    46:11:15:47:1b:24:1c:e5:d0:a9:e2:92:2b:33:8a:
                    07:ca:11:c2:5c:e7:d8:c1:86:83:14:d6:64:f1:f1:
                    47:0f:f8:df:ee:02:78:0c:a4:fe:de:a4:ef:ea:b3:
                    50:27:2b:27:af:3c:eb:c2:7d:31:f4:ee:54:85:1e:
                    00:d2:c4:58:e5:78:1a:8d:8a:2a:d4:9e:14:18:71:
                    ca:68:bf:a4:ba:13:ac:49:23:06:b8:6e:d4:5e:1f:
                    58:20:39:a7:ee:ef:b7:00:0b:50:cf:77:2b:eb:07:
                    64:f6:94:3b:dc:7c:ee:4f:40:7d:a4:8e:bc:50:09:
```

```
                16:07:c4:d6:25:f1:d2:80:dc:87:b6:f1:b3:b9:93:
                0c:6d:33:ff:1c:43:cf:f2:66:d1:24:eb:0e:e4:e7:
                30:51:40:ec:a1:83:eb:dc:c1:6b:56:bb:ba:40:ce:
                95:01:20:9e:fe:ad:85:5a:3b:cd:38:e7:d4:eb:79:
                60:5e:ec:63:e4:db:0c:d5:b2:67:96:c4:57:bf:d9:
                ea:c7:3d:35:d1:7e:06:1e:c0:79:b4:f3:37:c5:71:
                f2:92:b1:95:a6:fd:37:a6:4d:34:ad:f4:b0:51:a6:
                77:0c:fe:5c:c8:e9:33:46:ff:4f:86:f8:f7:92:2c:
                fc:a4:c2:7a:e1:68:e5:30:8f:4d:d4:c8:f2:a2:31:
                7e:c4:cf:89:59:f8:ee:30:40:43:f1:78:29:94:34:
                bb:39:53:fc:52:92:95:2e:6d:e0:8f:8c:3d:08:84:
                9f:73:2c:65:9d:6d:5f:51:c7:cf:51:77:d9:8b:21:
                9e:97:f3:80:27:07:ef:4a:1b:de:82:fa:7e:35:60:
                38:eb:40:29:a7:a1:1c:cc:5b:3f:88:6c:35:8c:4d:
                4d:ac:3b
            Exponent: 65537 (0x10001)
    Signature Algorithm: sha512WithRSAEncryption
         22:6d:bc:f1:7b:60:cf:fc:3c:a9:a0:b3:de:59:fb:12:4b:17:
         c8:0b:6e:86:09:82:89:65:d9:83:b3:9a:58:14:b8:c1:95:f0:
         74:8c:4e:dc:95:8e:46:64:39:4d:a2:f0:cf:28:4b:11:20:96:
         a3:43:fd:6e:10:85:e2:32:89:c5:9e:3d:f6:c2:8d:82:62:cd:
         db:aa:16:f3:ea:7c:51:ec:7c:96:61:ad:74:16:50:8f:95:65:
         b3:04:23:40:7f:86:25:01:58:90:d6:f4:67:65:7c:54:74:b1:
         49:0f:9f:1d:94:e6:cc:24:44:7d:81:dd:93:04:5e:8c:bf:1a:
         6c:0f:92:98:cf:4f:b8:f3:c2:54:66:44:8d:51:80:be:95:a0:
         07:e2:71:ea:ee:2c:77:80:cc:cc:65:76:ec:5b:95:1d:27:6d:
         22:f0:e3:61:a8:76:de:67:6e:2f:44:8d:52:b0:c0:6b:04:88:
         c9:a2:94:5a:4e:65:e3:79:97:10:90:aa:8b:68:f2:99:16:d1:
         ad:d2:07:eb:67:be:c1:1a:7e:04:b5:5d:8c:01:23:3f:4d:eb:
         83:8f:a5:38:28:2e:35:b4:d3:f7:2b:25:ac:8d:11:b6:21:2f:
         22:d8:52:c5:00:48:53:61:4f:e5:8a:47:b9:99:10:11:7a:3b:
         dc:e6:57:e2:ca:5f:94:de:4b:cd:e7:f6:9d:05:a8:e6:cc:a1:
         bd:bd:d4:9b:85:20:3f:bb:a9:21:2a:8c:58:bd:b3:95:bb:6c:
         ec:09:db:9b:c9:90:9a:ce:40:bd:dc:ac:0b:41:f4:b8:ff:74:
         13:04:79:ae:70:a9:5a:45:3c:4b:e2:ab:39:39:38:52:72:28:
         a7:40:b8:08:b2:50:3e:a3:bf:84:43:28:a3:11:e0:f6:8b:e7:
         fb:81:e8:a8:e3:4c:23:aa:32:4f:26:ce:f0:ca:3c:24:c0:2e:
         63:a1:bb:7d:33:cb:cb:b2:e5:43:af:74:db:bc:2e:78:0e:ae:
         79:71:2c:e9:c7:d1:20:76:a4:9f:fd:c8:1e:45:46:f3:7d:c9:
         d0:bb:3b:cf:57:22:64:92:12:f5:a9:84:bf:cf:be:85:23:8e:
```

```
          3a:c1:ae:76:b0:2e:41:d1:05:69:14:ef:95:9a:0e:f4:7a:af:
          3e:17:54:7e:65:04:5d:60:d9:88:97:b6:d4:5f:d6:6c:9d:c2:
          b1:6f:8a:94:2f:dc:53:cc:0c:e2:5a:78:ef:4f:3b:33:70:5d:
          95:3f:df:c5:6d:78:0e:bd:b5:28:0f:a5:f4:5b:76:72:2b:c0:
          ad:24:45:42:45:e0:e7:2d:00:51:20:35:e8:8e:73:73:68:b2:
          cf:a0:8a:cd:f3:d8:c1:54
```

```
$ openssl rsa -in key-pair.pem -check
RSA key ok
writing RSA key
-----BEGIN RSA PRIVATE KEY-----
MIIJKgIBAAKCAgEA6WAAMcTQBBLL6uYuktOMGQj7WvVetkE/IOPdRhRKdov9gLJU
pGu5JBr267/7YwvsCYBKh6p0tgjEOo26Qs3pba728lXoDl+Qem7iXq3VQe0PYNGy
szlJxwUpokBfkE9FMH6JbZtlz/Xlwsc1uuMw+QQtz7u02VWEL1BGccu95EUv5X8e
vs1D3uHZQ+C2mdQBJKcl/QZjtd3PFuxkijvQK+Zk8Bi1ZmyBNARWFsTyGiC7zWtG
ERVHGyQc5dCp4pIrM4oHyhHCXOfYwYaDFNZk8fFHD/jf7gJ4DKT+3qTv6rNQJysn
rzzrwn0x9O5UhR4A0sRY5XgajYoq1J4UGHHKaL+kuhOsSSMGuG7UXh9YIDmn7u+3
AAtQz3cr6wdk9pQ73HzuT0B9pI68UAkWB8TWJfHSgNyHtvGzuZMMbTP/HEPP8mbR
JOsO5OcwUUDsoYPr3MFrVru6QM6VASCe/q2FWjvNOOfU63lgXuxj5NsM1bJnlsRX
v9nqxz010X4GHsB5tPM3xXHykrGVpv03pk00rfSwUaZ3DP5cyOkzRv9Phvj3kiz8
pMJ64WjlMI9N1MjyojF+xM+JWfjuMEBD8XgplDS7OVP8UpKVLm3gj4w9CISfcyxl
nW1fUcfPUXfZiyGel/OAJwfvShvegvp+NWA460App6EczFs/iGw1jE1NrDsCAwEA
AQKCAgACLP3zeNVDpHU4PGskvUoqFZZaU71NT7Bp+xXeCGRlVIrJf2AgpEx2lLXG
nuSWGM9bi1GmkCVEaTKDJT0eUFMjzaTJv5lJQQTUz2HW5iH92lrxWSMkXp1ceKTZ
uWXqr4KCoDthxTgexzkcJsVOd73GJI7Ux/g4qn5v/nyZpP+hOC/pINVFjB5c3pgq
9NZe+sYD9Hb7M+sMEARVP2D0lPgs4JoCTwip6ssNlBjFX5KuQa3ejWh60kr2atDG
J7rdvOUgNuI5tQ6t1Q+FRl7Ee/c29jvmXzifPHCoFYW2x6zg8EkIBHP7Q0Mjbg1g
LjSlhREW5fO6N/NvbGFX2B1x2OW9LbUJIEhIXWdopWNLm01etIfhmtWxPo06dP21
iBTfDxc4w7wADi3SvQfM1UdmUCQ64633WWAho+pVxnAfkG4ZclkDwMrwbS8OigYn
TakIX/G63rPhUu5H8Baf8KpxLyYiPWW99Uebmwqn7C+zhbmHR3eo+VOJDKTsLS89
9XOqv4slxZkMORcopWqsQ6zDU7cWo6H63SqUxkkEs/CAsnJsOo0usw+uwlR2c/3A
+XGFlj+b5lvq7dexcFvVMNB0PSEQEJHKjfsbTy/egkSglXri4C92+HCfxW+0vM9Q
7B03Zs3LTEaQBNY6O1DuUS/DyNh0KYcjQyiDf/fGJ23LJtx2IQKCAQEA+y26nan4
gT19IJZv+Wn6qbwXRhV1/sullIQ2dH9dTjhVxgQXfrtPIFGLAYt3th5UEHSram3E
hGGZ3mYS5vNO7HxU+Y4vkq++y6Mozgqrnp+LR3U4m4S+43/VJWFOhLArNNq4ZIRL
yHCYoAIhEhXzEkOZrEYJcOS+vwpNg64JllRynsYD4VVRfuLf5CHhg9Qtqrhl/4FT
D2nAMIAdKI8uaL5oDMQg/GHKSr16FerI4t2FO/SrxGnAp9+er2JkThatHHD6Jw0U
G7U5PoIBE1xqtS1xT6oHZNUekTIBy7HfWkxMg6P0+9QgI4XxEs6ZOby8eX3SVESR
aze08pAZI+vo8QKCAQEA7drJR1SVehPKQ8q2LDmPm2g5Z7nOqQXJ2IekvXbtgWRf
```

```
WSknV2NCOCRv8TzNa+yZWu7KaK6uHOLNrf0U78Da+vThzVaPD69KhoV6pdupOyVu
NIO5OsVGRIKD2LlaBRX2qRooGRMUvCO+Z1rDm0YSZEBLczHRJmqqLSFmW7SYOAQG
8M0rgqC3h55hjs71m4iTQpYWvxMUfX25IcdtUQt67lKeFSrDdE2N618IgzwRcRp/
bx8ix9M5xCn9z+Cw8+X9dIbymMFz+K97ny5vbBxh3YxWiYZJVW7kpZ+EtA68HEsh
Fxf8qzegUK8I2Py3IPA9B3z+p9u5sDrkJkpJaGRH6wKCAQEAjmDMgKShXbkaHYfI
7ytY0C/zTWHK4eRT9oy/RWgQ3s7mJ/mXy9mnL52pWvXQpOYqJq3LoOeZ1qRwkPVx
T4KpwTl0/YTQiKigUVQqRCkeQSsUyyuyK82iWtjSXH79AA/QXgSBM1Eel/7jNQgf
8N2KTFcPW+zTWy/+w9G2QKc9ff5efoumFWvEZ1V5Lhrad1ylijk+cAfBqmH6dEOa
w6vt1afg2SOvwwA3qAghlRziRF7nMCYv24VKlJ4YmDgAvZbzB8Uut0YdmNswGLOe
onfncM3ShiLOUCTST+F1zR4FimznjlM5AiqrERfSoCOGDYljmMZunB/GAm2wXzK5
XXhTMQKCAQEAxIF8yPGYjMLM2DUhTGhBvKLeylM36Wxe1gU6lU0r6zqBi6f0WL4o
dzmjQip5jDBsgjPGBbZLIpI+WwA7y9gJEkw+QObCSrMHXxvnuB9CMRs7cSdiYKua
4f7x7pszRABhSFPTzON2vfjJ1ZmnASFjEHAN5Z3CrXpyHdb6Ocl5+M1a7paS29jr
x5+R+eoB//1VrC1TnYil5Qd5a+7W4Z4accrPz60EzZFFLq/U2JdH0amSnruagHyB
vQ70Ikn/54CbZSoYQxuQNGTGPTyf0TGhqOfIltXawDpdrHedqySBRN7M98dZeBr8
16vEI4ClyxCvWwBpY0SkpHQDCZ9WBmjTAQKCAQEAxeAO4Xo9HWsUkmuyH0VxL5PX
0mhVcYkHBvaBMU2+GBmjq/LJZlDE/bbwBitAf/t43y+IzuY8kB2eul3xiTwSilZs
mSgEjMVVfbB49Tb26WWS4ZBEo4HiC4aNAm0yX6Y1hJAfjC3QsIYOs2IlVTWWWZN6
vcEzBIPd6Mg2nzgtqVeWWKk/VtjNnTAjN68YsPgaN/ZuTw0DdPcQ+4dP8KZZjFaN
fEbSuPUzd5o8DBATwZAwbFZ+G4Ft9F3OoSw+bnjyahyg3JAeakalCE+1vhy1LEJ8
QIiaiT1tYHcc6UGQZMpBoywXVNPEZ1e8u4BjcfgeF/xx0BR8B7mN+JIiWo1hhA==
-----END RSA PRIVATE KEY-----
```

You can verify the pre-conditions with OpenSSL like following example:

- No password protected ZIP archive allowed.
- Only archive files ending with `.pem` are allowed.

Requirements to ZIP archive:

The wizard only accepts one file where all items are included. This file can be in PEM format, or a ZIP archive that contains only PEM entries.

- Only RSA key pairs are supported.
- Key length must be at least 2048 bit.
- The key pair may not be encrypted.
- The customer must provide the whole certificate chain, including CA certificates.
- Key pair and certificates must be provided in PEM format.

- No additional certificates or chains are allowed. This means the chain may not contain any certificates which are not used by the supplied trusted chain.
- More than one certificate chain is supported, but there may be only one certificate that signs the key pair itself. This means every trusted chain must start with the same certificate.

If you want to use your own certificate to secure the communication between AppMon components, the provided key pair and certificate chain must meet the following criteria:

Deploy certificate

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Deploy a new key pair and certificate chain

Certificate wizard

Deploy a new key pair and certificate chain

The currently used key pair can be replaced by:

- A wizard-generated self-signed certificate.
- A customer-provided self-signed certificate.

See Custom Certificate Requirements if you plan to import your own.

See Disallow Private Key deployment to Client and Collector for information on securely deploying a new Private Key and Certificate in a a potentially insecure environment.

Restart needed

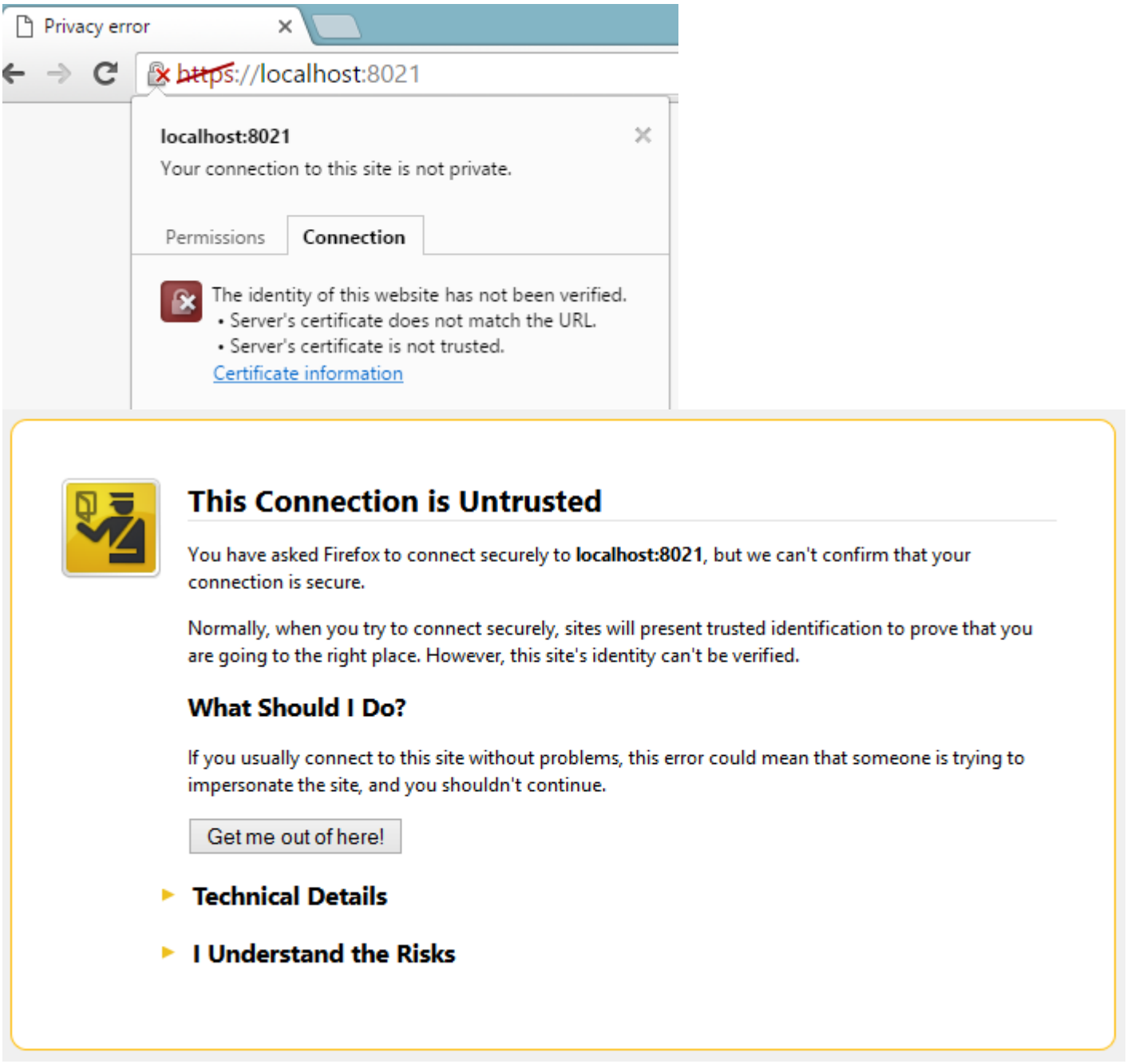
You must restart the AppMon Server and, if applicable, the Memory Analysis Server after deployment.
You must also restart all Collectors to apply changes after a certificate deployment.

Do not apply the certificate management wizard if you can not restart at least the AppMon Server processes.

Accepting self signed certificate

If you choose to create a self-signed certificate, remember that the created certificate is not an officially trusted one.

When connecting using a browser, you typically receive a warning about an unknown identity, as shown in the following:



You must accept the certificate to proceed. For details consult the documentation of your browser.

Certificate wizard

Use the Certificate wizard to:

- Check preconditions.
- Create a new self-signed key pair and certificate, or import a customer provided one.
- Deploy the items to the Dynatrace Server, Memory Analysis Server, Collector and Client
- Restart mandatory and optional processes.

Start the wizard using the Client by selecting **Settings > Dynatrace Server > Certificate Management > Manage** and clicking **Start wizard** under either **Generate and Deploy** or **Import and Deploy**.

Step 1 - Check preconditions

The wizard informs about the necessary restarts after a deployment. It also gives feedback if the Collectors and the Memory Analysis Server are connected and the Collectors are the same version. This helps to avoid manual steps after certificate deployment.

Offline Memory Analysis Server and / or Collector(s)

If you do not operate a Memory Analysis Server in the AppMon tool chain, ignore the offline warning. Otherwise, start it and restart the wizard.

Click **Create list of Collector details** button to export the list of affected Collectors.
Be aware that you can proceed, but offline collectors do not receive the new key pair and therefore they can not connect later. Ignore this warning only if you know what you are doing.

Collector version mismatch

Similar to the offline warning, an error warning appears if Collectors earlier than 6.3 are present. In that case you can not proceed. You must upgrade all collectors first. Export the collector details list to assist in this.

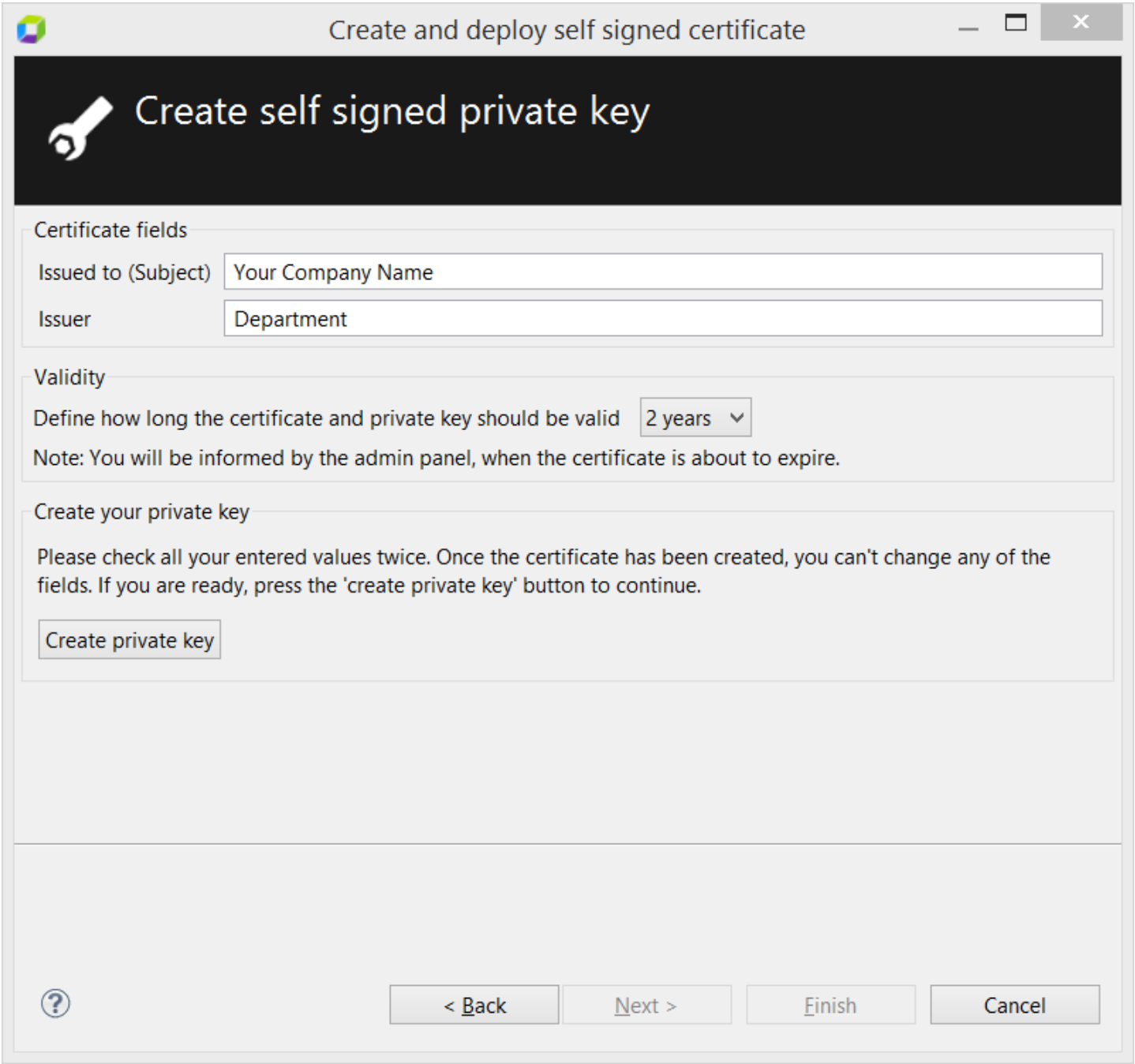
Resolve issues manually later

See Advanced Features - Certificates, Private Keys and Keystore to handle components that were offline or version-mismatched (Collectors) during deployment.

Step 2 - Certificate chain and keys - create or import

Alternative 1 - create a self-signed certificate

Enter values for CN and ISSUER, choose the validity and click **Create private key**.



Now optionally export generated certificate and key pair.

> **Backup to a safe location**
>
> The exported ZIP contains the generated private key used to encrypt communication. Handle the file with care and store it at a safe location.

Alternative 2 - import certificate and key pair

Step 3 - Deploy to components

This dialog box shows the result of deploying the key pair to all currently connected tiers. This is not enabled if you do not operate a Memory Analysis Server.

Step 4 - Select components to restart

AppMon Server and Memory Analysis Server (if applicable) always need to be restarted immediately. Depending on your environment this may take a few minutes to complete.
Check the Collectors checkbox to restart all currently connected ones now – or postpone it to later.

After the deployment has finished you can verify usage by navigating to the Certificate Overview horizontal tab. In the case the collectors were not restarted, the newly deployed private key is not applied until a manual restart triggers.

Advanced features - certificates, private keys and keystore

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Enable logging of keystore operations

Clear stored certificates to force usage of new ones

Clear stored private key to revert to default

Clear DC-RUM certificates

Reinitialize passwords

Disallow private key deployment to client and collector

Enable logging of keystore operations

To enable verbose output about keystore operations (e.g. addition of new keys and certificates) you need to put it the following system property into the `<DT_HOME>/<component>.ini` file: `-Dcom.dynatrace.diagnostics.debugKeyStore=true`

Clear stored certificates to force usage of new ones

If a collector does not connect to an AppMon Server because the Server's private key was changed while the Collector was offline, the following message appears in its logfile:

```
WARNING [DynaTraceSSLSocketFactory] Unable to connect to localhost:6699 probably untruste
```

Clear the certificate stored locally on the Collector to force it to accept any when presented by the AppMon Server. Start it with `<DT_HOME>/dtcollector -Dclearcerts`.

To remove certificates from the backend, frontend or client, the `-Dclearkeys` option is available on each component and also via Certificate Overview, when doing a right click on an entry of the certificate chain that should be removed and selecting `Delete certificate chain`.

Clear stored private key to revert to default

If you changed the AppMon Server process' private key and want to revert to the built in default private key (as it would be in a new AppMon installation), start backend and frontend process with `<DT_HOME>/dtserver -Dclearkeys -Dclearcerts` and `<DT_HOME>/dtfrontendserver -Dclearkeys -Dclearcerts` respectively.

> **Note**
>
> If a Collector wants to connect to the Server with the now removed key pair, you must Clear Stored Certificates as described above (for example, `<DT_HOME>/dtcollector -Dclearcerts)`. If you also want to remove the private key at the collector and revert back to the built in default private key, add `-Dclearkeys` too.

Clear DC-RUM certificates

DC-RUM integration server certificates are stored in the same manner as those for internal communication. In case the certificate of the DC-RUM server has changed, they must be re-accepted by clearing them first by entering `<DT_HOME>/dtserver -Dcleardcrum` or via Certificate Overview, when doing a right click on an entry of the certificate chain that should be removed and selecting `Delete certificate chain`.

Reinitialize passwords

Adding `-Dreinitpasswords` to any component when starting up (for example `<DT_HOME>/dtserver -Dreinitpasswords`) reinitializes several passwords such as AES tunnel encryption, keystore masterkey stored within the `dt_pwdstore`. This generates a `dt_pwdstore.zip` which has to be copied and extracted to every component's `/conf` folder (Server, Client, Collector). This overwrites all existing `dt_pwdstore` files. The migration occurs automatically when the component starts for the first time after the `.zip` extracts.

Important

Remove the `-Dreinitpasswords` option for the next startup, otherwise regenerates new passwords and overwrites the `.zip` file. In case something went wrong and you want to revert back to the default passwords, simply delete all `dt_pwdstore.*`in the `/conf` folder.

Disallow private key deployment to client and collector

To avoid deploying a newly imported/generated private key to Client and Collector, set the property `-Dcom.dynatrace.diagnostics.ssl.allowprivatekeydeployment=false` in `dtserver.ini` (to avoid deployment to collector) and `dtfrontendserver.ini` (to avoid deployment to client). Removing the flag afterwards causes the server to redeploy to components that are connecting to it.

Tip

This may be used as a workaround to deploy different private keys and Certificates to the various components.

For example: Deploy a new private key and Certificate to all components, then add the property to dtserver.ini and dtfrontendserver.ini and deploy another private key and Certificate, will give you a different private key for Client and Collector than the one at the Server.

Deploy new private key within a potentially insecure environment

This may be only done via a locally connected Client or by placing a customized `keystore.jks` in `<DT_HOME>/server/conf` which contains the new private key accompanied by the certificate chain (with alias `dtserver`) and has it's password defined in the `dtserver.ini` using `-Dcom.dynatrace.diagnostics.communication.sslkeystorepassword=mypassword`. Please follow the subsequent steps very carefully.

1. Define `-Dcom.dynatrace.diagnostics.ssl.allowprivatekeydeployment=false` in `dtserver.ini` and `dtfrontendserver.ini`.
2. Start Frontend and Backend Server and connect with a local Client.
3. Go to **Settings > Dynatrace Server > Services > General** and clear all check boxes except the **Allow Agent Connections to Dynatrace Server** check box (this one must be enabled) and restart the Server.
4. Generate/import and deploy new private key and certificates and restart all components.
5. Shutdown the Backend and Frontend Server.
6. Define `-Dcom.dynatrace.diagnostics.ssl.allowprivatekeydeployment=true` in `dtserver.ini`, but do not modify `dtfrontendserver.ini`.
7. Start the Backend Server briefly and immediately shutdown after successful startup.
8. Define `-Dcom.dynatrace.diagnostics.ssl.allowprivatekeydeployment=false`again in `dtserver.ini`
9. Startup the Server and go to **Settings > Dynatrace Server > Services > General** again and restore all previously selected and cleared check boxes. Then restart the Server.
0. Copy `<DT_HOME>/server/conf/dt_co_keystore.*` to an USB stick and import in every `<DT_HOME>/collector/conf` where you wish to exchange the private key.

Set up system components

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Supported plattforms

Network communication

Related pages

SELinux and Dynatrace product issues on Redhat

An AppMon system environment consists of the components listed below. Follow the links for detailed installation and configuration steps.

Set up the AppMon Server
Set up Collectors
Set up a Memory Analysis Server
Set up Agents
Set up the AppMon Client
Set up the WebStart Client

Supported plattforms

Multiple on operating system platforms are supported to run these components. See the Release notes for details on supported platforms for AppMon components.

Network communication

AppMon components communicate via a standard TCP/IP unicast protocol. The required port, firewall, and proxy configurations are described in Set up Communication Connections.

Set up the AppMon Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Memory settings

Additional considerations and troubleshooting

Related pages

Server administration

Set up collectors

Memory settings

You configure memory settings in the AppMon Client at initial startup. After importing a license you can select them from a list of predefined deployment sizes.

Change the selected deployment size at any time after initial setup by choosing **Settings > Dynatrace Server > Settings > Sizing** or directly under `-memory` in `DT_HOME/dtserver.ini`. Find more information in Sizing Settings.

Additional considerations and troubleshooting

Versioned profiles

All System Profiles are stored in the `profiles` folder (see above) as XML files (ending with `.profile.xml`). Those files should not only be backed up on a regular basis but can also be archived with common version control systems. Doing so provides easy history control and rollback over the course of time.

Parallel startup of server and analysis server

If you have the AppMon Server and AppMon Analysis Server installed on the same machine (sharing the same core executables), it is not possible to start both servers at the same time, for example by using a UNIX shell script.

In this case, there are two possible solutions:

- Install the Analysis Server standalone on the machine.
- In your shell script, first start the Analysis Server, wait a short time (typically 10 seconds is sufficient), then start the AppMon Server.

Number of open files limit on Linux

Some UNIX-based systems impose a low limit on the number of files or file descriptors a user or process can have open. This limit includes open socket connections.

Because the AppMon Server needs to read/write many files from/to the session store and have many socket connections open, the system default limits for the number of open files may be too low. Some Linux distributions use a default of 1024 file handles, which is most likely too low for medium to large AppMon installations. Therefore, the AppMon server enforces a minimum of 2048 file handles. Use `ulimit -n` to query the current number of file handles.

> **Note**
>
> Don´t confuse the `ulimit` (thousands) with the system-wide limit (hundreds of thousands) that you can check with `more /proc/sys/fs/file-max`.

You may learn of an insufficient limit through an error message like “IOException: Too many open files” or “java.net.SocketException: Too many open files”. Before changing anything, check the current (soft) limit with `ulimit -n` or `ulimit -Sn`. Use `ulimit -Hn` to determine how much you can raise the soft limit as non-root.

Use `ulimit -n <numberOfFileDescriptors>` to raise the (soft) limit temporarily for the current user. Add entries for other users or groups to `/etc/security/limits.conf`. Use `ulimit -n` to check that the change was successful.

Set up collectors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Configure memory

JRE on Solaris, HP-UX, AIX, z/Linux, z/OS

Configure a Collector other than localhost

Related pages

Set up the AppMon Server

Set up a Memory Analysis Server

Set up agents

Install collectors

Collectors

Overview

Installation takes care of AppMon components and their configuration for basic local usage / small implementations. You can have just everything (all AppMon and your components) on one machine and use the stand-alone Collector service / daemon or even the AppMon Server-embedded Collector for a still smaller memory footprint. (see Embedded Collector Usage Restrictions)
With more complex, larger deployments you may need quite a number of Collectors. Depending on the architecture of your application the Collectors may be on one or more (virtual) machines.

If you run the Collector on a different machine than the AppMon Server you must reference the Server in the registration (Windows) or in the *NIX shell script. See Configure a Collector Other Than Localhost) for more information.

To run more than one Collector on a machine you must install extra instances of the installed default instance. If doing this, make sure Collector names and ports used listen for Agents are unique.

Once a Collector is up and connected to the AppMon Server, you can change settings or set up additional ones in the Collector Group by selecting **Settings > Dynatrace Server > Collectors** in the AppMon Client, then selecting a Collector in the upper pane and configure settings in the lower pane.

Configure memory

The AppMon Collector uses by default 2 GB of memory. Depending on the number of Agents, you may need to adapt this default setting. See the Collector Best Practices for more information.
To change the amount of used memory, please open the main configuration file (`dtcollector.ini*`) in a text editor and adjust the values for the `-Xmx` and `-Xms` parameters. Always use the same value for both parameters, and restart the AppMon Collector to apply your changes.

The following shows an example of an adapted `*dtcollector.ini`, where memory usage was increased from the default 2 GB to 4 GB (see lines 6,7):

```
1   ...
2   -basedir
3   collector
4   -restartonfailure
5   -vmargs
6   -Xmx4G
7   -Xms4G
8   ...
```

See the Deployment Guide for detailed sizing recommendations.

JRE on Solaris, HP-UX, AIX, z/Linux, z/OS

If you install the Collector on HP-UX, AIX, z/Linux, or z/OS, set the Java used to run the Collector. In the `dtcollector.ini` file, edit the line after `-vm` to point to the Java executable of the desired JRE. For example:

```
-vm /usr/java/jre1.7.0_51/bin/java
```

Configure a Collector other than localhost

The configuration settings below can later on be reconfigured in a Collector Group in the AppMon Client by selecting **Settings > Dynatrace Server > Collector**, then selecting a Collector in the upper pane and configuring settings in the lower pane.

Windows

1. Unregister the service that was registered by default during installation:
   `%DT_HOME%/dtcollector.exe -service uninstall`
   or
   `sc delete "dynaTrace Collector 7.0"`.
2. Register the Collector to connect to a AppMon Server on a different machine. The general syntax is:
   `%DT_HOME%/dtcollector -service install -server <Server(Host)Name> [:PortToConnectToServer] [-listen:PortToListenForAgents]`
   For example, to register a Collector that connects to an AppMon Server / host named `myAppMonServer` on the default port and listens for Agents on the default port:
   `%DT_HOME%/dtcollector -service install -server myAppMonServer`

   The registration parameters are reflected in `%DT_HOME%/collector/conf/collector.config.xml`.

Linux and Unix

On *NIX there is no automatic registration at install time, so you don't need to unregister a daemon initially. However, to connect to a AppMon Server on a different machine you must at least adapt `OPT_ARGS= -server myDynatraceServer"` in the `$DT_HOME/init.d/dynaTraceCollector` shell script, Save As in `/etc/init.d`, and register the daemon for boot with `chkconfig` or the equivalent on your distribution.

The general syntax is as follows:

```
OPT_ARGS="-server <Server(Host)Name>[:PortToConnectToServer] [-listen:PortToListenForAger
```

Set up a Memory Analysis Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Configure memory

Configure the port

Start and stop the Memory Analysis Server

This page describes how to set up a Memory Analysis server.

Overview

The AppMon Memory Analysis Server is used solely for post-processing memory snapshots. It should not be confused with the AppMon Frontend Server, which serves other analysis data to the AppMon Client.

Configure memory

Before starting the AppMon Memory Analysis Server, adjust the memory settings in the main configuration file `dtanalysisserver.ini`. Set the option `-Xmx` to the maximum heap size of applications + 25% to analyze.

The following is an example of the code to add to `dtanalysisserver.ini`.

```
...
-Xmx2048M
-Xms128M
-XX:MaxPermSize=128M
-XX:PermSize=128M
...
```

The `-XX:+UseCompressedOops` option is automatically appended when you use the 64-bit installer.

Configure the port

The Memory Analysis Server listens on a specified port for incoming connections from the AppMon Server. To change the default port number, add `-Dcom.dynatrace.diagnostics.listen=[<ip>]:<port>` to the `dtanalysisserver.ini` file.

Start and stop the Memory Analysis Server

Windows service

On Windows, you can start the Memory Analysis Server service from the **Start** menu, the command line, or the **Services** control panel. The registered service name is `*Dynatrace Memory Analysis Server* <version>`. The service runs with restricted permissions as Local Service.

> **Note**
>
> If you change any output directories (such as the session directory), make sure the AppMon Server service has sufficient privileges in the new directory.

To start the Memory Analysis Server from the command line, change to the `DT_HOME` and execute:

```
dtanalysisserver.exe -service start
```

To list all the available options, execute `dtanalysisserver.exe -?`, which displays the following:

```
Dynatrace Memory Analysis Server
Version: 7.0.0.2469 - built 2017-05-30 13:45:26 +0200
Copyright (C) 2004-2017, Dynatrace

Usage: dtanalysisserver [options]

Available options:
   -?, -h, --help Print this help message
   -v             Display version information
   -loglevel [0..8]
                  Set loglevel, valid values are: [0..8] or
                  [finest|finer|fine|config|info|warning|severe|debug|none]
                  Default setting: 4 (info)
   -listen [<ip>:]<port>
                  Listen for dynaTrace Server instances on [<ip>:]<port>
   -service [install | uninstall | start | stop | list(all)]
                  Install/uninstall Windows Service
                  (uninstall might require you to log off)
   -basedir <dir> Set working directory for application
   -logdir <dir>  Override default log directory (command line argument only)
   -ini <config>  ini file to use, default is dtanalysisserver.ini
                  A value of 'ignore' will skip reading the ini file.
   -timeout <sec> Timeout value for watchdog feature, 0 turns the watchdog off.
                  Default value: 180, valid value range [30..86400] sec
   -watchdogaddress 127.0.0.1
                  IP address to be used for watchdog feature
   -portrange <n>[:<m>]
                  Port range to be used by watchdog feature
                  By default a port within range 50000:65535 is used
   -vm <javaVM>   The Java VM to be used for launching the application
   -startup <startup.jar>
                  The startup jar to execute. The default is to find the
                  plugins/org.eclipse.equinox.launcher jar with the highest
                  version number.
   -ea[:<package>"..."|:<class>]
                  Enable assertions (assertions are disabled by default).
                  A <package> or <class> name can be specified (cf. Java docs)
   -Dosgi.*=<value>
                  Set arbitrary OSGI configuration parameters
   -D*=<value>    Set properties for Java application
```

```
    -X*=<value>    Set (non standard) options for Java VM
```

Linux Daemon

The syntax for the Memory Analysis Server is:

```
<DT_HOME>/init.d/dynaTraceAnalysis start | stop | version | status
```

> **Note**
>
> Do not log-rotate the Memory Analysis Server's log files. This is automatically done by the Server.

Set up agents

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Agent settings

Related pages

Apache Agent configuration

Database Agent configuration - DB2

Database Agent configuration

IIS Agent configuration

Java Agent configuration

.NET Agent configuration

NGINX Agent configuration

Node.js Agent configuration

OneAgent configuration

PHP Agent configuration

Web Server Master Agent configuration

Install agents

Unique agent names with index

Agent timers

Agent Group - Agent Mapping

SELinux and Dynatrace product issues on Redhat

Watch and learn

Managing agents

Depending on the technology to monitor and the operating system, initial setup varies. Some examples:

- Host Agent and Web Server master Agent bootstraps have `.ini` files in `DT_HOME/agent/conf`. Both are installed as services or daemons on Windows or *NIX respectively. The service needs to be uninstalled and reinstalled and the daemon's `init.d/<shellScript>` needs to be adapted if parameters like name or port change.
- Apache and IIS Web Server Agents, NGINX, and PHP Agents are slaves to the Web Server master Agent and as such options like `logfilesize`, are configured via entries in its `.ini`.
- The Agent for Java technologies can be configured with `<option>=value` appended to the configuration string, which is appended to `java -options` or a `JAVA_OPTS` variable in a Java configuration or start file.
- For all Agents, environment variables named `DT_<OPTION>` (always capitalized) have the same effect as configuration string `<options>`, but are less flexible.
- Configure .NET Agent options by choosing **Start > Dynatrace > <version> > Agent Configuration** Tool, set directly in the registry or with environment variables, which take precedence if set.

Agent settings

Most arguments and options (`<ArgParOrOptName>`) have an environment variable equivalent (`DT_<ArgParOrOptName>`).

- `logfilesize` is equivalent to `DT_LOGFILESIZE`
- `name` and `agentname` are equivalent to `DT_AGENTNAME`

Note

Environment variables are global, so setting one may affect other Agents. Also, there are different orders of precedence if you happen to have both the option and the environment variable set. For example, the environment variable for .NET precedes over the argument / parameter / option.

Mandatory arguments / parameters are applied in the following:

- Connection string in the Java options
- .NET Agent Configuration Tool
- Web Server master Agent `agent/conf/dtwsagent.ini`
- Main `Node.js` JavaScript file. This must be the first lines at the beginning of the file to capture PurePaths.
- Host Monitoring Agent `agent/conf/dthostagent.ini`

Argument/Parameter=value ENVIRONMENT_VAR	Default value	Comment
`name=<AgentName>` or `agentname=<AgentName>` `DT_AGENTNAME`		`name` is actually legacy, but used everywhere. `name <agentname>` with a separating space in `.ini`. Unquoted string according to variable-naming-rules. If you opt for the environment variable, it must be `DT_AGENTNAME`.
`collector=<Colle/ctorName>` `[:port]` or `server=<CollectorName>` `[:port]` `DT_COLLECTOR` or `DT_SERVER`	[:9998] or [:8042] if via HTTP	The Collector (name or IP and optional port if non-default) to connect to. The legacy `server` parameter stems from the Collector that was originally embedded in the AppMon Server.

Based on scope and precedence, options are applied as environment variables if an environment variable is available for the respective option. If not available, the option is applied to the following:

- Java options / connection string as `<option>=<value>`, no spaces and in any order as they are named.
- .NET as registry keys.

For the Web Server Agent and Host/Infrastructure Monitoring Agent services/daemons only `loglevel` and `logfilesize` are exactly the same (names), so `tech type = all` in the table means Agent is started by the `dtagent` bootstrap.
For all other Agents, see the specific page in the help for specific settings by Agent type.

Options with corresponding ENVIRONMENT_VARs for all tech types (Java, .NET, native)

Option=value	ENVIRONMENT_VAR	Default value	Comment
logfile=<path>	DT_LOGFILE	<DT_HOME>/log/dt_<agentname>.<pid>.#.log [3]	

loglevel=<level> or loglevelfile=<level>	DT_LOGLEVELFILE	info levels: none | off, finest, finer, fine, info, warning, severe	
loglevelcon= <level> logstdout= <level>	DT_LOGLEVELCON	info levels: none | off, finest, finer, fine, info, warning, severe	Loglevel on console
logfilesize=<bytes>	DT_LOGFILESIZE	10000000 Bytes (=10MB)	Value in bytes
collectorlist= <path>	DT_COLLECTORLIST	<DT_HOME>/agent/conf/collectorlist. <tier><SystemProfile>	Change default path if
debugflags=<list>	DT_DEBUGFLAGS		Comma-separated list AgentDebugFlag=valu
sotimeout= <seconds>	DT_SOCKTIMEOUT	30	Specifies the native Ag for sending/receiving
hirestimer=<type>	DT_HIRESTIMER	auto	Sets the high-resolutic used for this Agent. Ty runtime, auto
storage	DT_STORAGE	DT_HOME/agent/downloads	Points to AppMon hom an `agent/downlo` where the all configura and log directories for core Agent are stored part.[2]
wait=<seconds>		20	Time the Agent waits f Collector. Else uninstr
environmentname= <string>			Results in `agentname[<env`
usessl=true|false	DT_USESSL	false	Enables SSL encryptic and the Collector. Enc uses the same port as enable SSL you need as true before launchi

Options for selected tech types

Option=value	Default value	Tech Type	Comment
maxsamplingstackdepth=<#>	2048	Java, .NET	Max stack depth that is captu

maxsamplingmethodtablesize	65536	Java, .NET	Max number of methods that a
environmentdetection=<yes no>	yes	Java	**Yes**: The Agent tries to detec **No**: Prevents the Agent from t
transformationsamples=<# of samples> DT_TRANSFORMATIONSAMPLES	1333	Java	Number of samples to take to
transformationmaxavgwait=<milliseconds> DT_TRANSFORMATIONMAXAVGWAIT	60	Java	Maximum average wait time fo
ctimeout=<seconds>	10	Java	Connection timeout — time to
excludeclasses=<yes no>	yes	Java	Prerequisite for `exclude.`
exclude=<expression>	not set	Java	[1]**semicolon-delimited** list of
optionTestRunIdJava=<testRunId>	GUID string	Java	Identify the test to be run with
disableperfcounters=true|false=not set DT_DISABLEPERFCOUNTERS	not set	.NET	Add a key `disableperfc` `HKEY_LOCAL_MACHINE\`
DT_TESTRUN_ID=<testRunId>	GUID string	.NET	Identify the test to be run with
perfcounterinterval=<multiples_of_10_secs> DT_PERFCOUNTERINTERVAL	10	Native	Perfcounter poll interval.
enablenetworkdrivemonitoring=<true false>	false	Native	Network drive monitoring is by
disablewifiadaptermonitoring=<true false>	false	Native	
overridehostname=<host name>	not set empty string	Native	Overrides the auto detected h
buffers=<# of event buffers>	100000	Native	Number of event buffers to all

[1] Excluded classes are not sent to the Collector for instrumentation. `excludeclasses` *needs to be enabled (default). The classname must be delimited by '/' instead of '.'. For example:* `-agentpath:...,exclude="starts:com/dynatrace/diagnostics/foobar;contains:` *would exclude classes that start with com.dynatrace.diagnostics.foobar and classes that contain the string "Proxy" in their name. More examples:* `exclude="starts:java/util/LinkedHashMap;contains:LinkedHashMap;ends:Hash` *Quotes are optional, for example:* `exclude=ends:MyImpl`

[2] The bootstrap part of an Agent tries to save the core into a subdirectory `agent/downloads` *to the following directories in the following order:*

- `DT_STORAGE`
- `DT_HOME`
- `DTD_HOME` *(precursor of* `DT_HOME`*)*
- *The path of the Agent,* `tmp`*, current directory So having no environment variable set is fine because write permissions to* `agent/downloads` *are set at install time. If you did not install correctly and everything else fails, updates are still available.*

  [3] Log rotation and log aging prevent the Agent from filling up the host's file system.

Log rotation

Each Agent limits its number of logs to 5, rotating the log file index. Because Index 0 contains environment and start up information, it is never overwritten. Each of these 5 files is limited to whatever `logfilesize` is configured as <option> passed to the JVM as described above.

Log aging

Because log file sizes and their number grow over time, the Agent purges logs files when:

- A log file is older than 14 days
- The combined size of all log files exceeds 1 GB of storage space
- There are more than 1,000 log files

To prevent unnecessary delays in your application's start up, the Agent's bulk purge limit is set to delete a maximum of 50 of the oldest files. This logic is shared between bootstrap and core Agents, so in total 100 of the oldest files may be deleted at one time.

Java Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Java Agent configuration string / JVM arguments / parameters

Agent configuration for Java-based application servers

Agent Configuration for Java-based Windows Services

Parameters and options

Environment variables

Debugging and agent conflicts

OneAgent usage

Related pages

Apache Tomcat

Fujitsu Interstage Application Server

GlassFish Server Open Source Edition

JBoss, Wildfly Application Server

Oracle Application Server

SAP Netweaver

WebLogic Application Server

WebSphere Application Server

WebSphere Liberty Profile

Java Agent

Note

For AppMon 6.3.4 and later, there is an important new security setting `enforcepci` for PCI-compliance. See the options table for details.

Java Agent configuration string / JVM arguments / parameters

The Java Agent uses the JVM Tool Interface. Configure a Java Agent by passing parameters to the Java Virtual Machine. You can pass parameters using the command line (for example `java -<option> -jar <jarfile>`) or add it to an `.ini` file or the JAVA_OPTS variable in a configuration or start file, such as `setenv.sh` or a batch file.

In addition to the mandatory `agentpath`, `name` and `server` parameters you can add options as listed below. Options are separated with a comma, contain no spaces in between and can be in any order with the syntax `optionName=value`.

The generic syntax is as follows:

- J2SE 5.0 and later: `-agentpath:<agentlib>=<parameter>[,<option>]`

Use `collector=` instead of `server=`. See the explanation in the table below for details.

For Linux, to instrument a Java application with a default installation of AppMon, all on one machine and all 64-bit, and the Agent mapping into the `mySystemProfile`, add the following:

```
`-agentpath:/opt/dynatrace-7.0/agent/lib64/libdtagent.so=name=Java_mySystemProfile,collec
```

For Windows, add the following:

```
`-agentpath:C:\Program Files\dynaTrace\dynaTrace 7.0\agent\lib64\dtagent.dll=name=Java_my
```

Agent configuration for Java-based application servers

The configuration string for the Java Agent is very similar for all platforms using Java, but there are some differences where and in what dialog or file to place it. See the related topics on this page for links to configuration string information for supported Java-based application servers.

Agent Configuration for Java-based Windows Services

The configuration string is similar for Windows services implemented in Java, but the configuration mechanism is different. These services typically start using `wrapper.exe`, which uses a `wrapper.conf` file to pass in Java options. The following is an example of adding a parameter to start the Agent:

```
# Java Additional Parameters
#wrapper.java.additional.1=
wrapper.java.additional.1=-agentpath:C:/Program Files/dynaTrace/dynaTrace/agent/lib64/dta
```

Note

When adding the `agentpath` to a `wrapper.conf` file, you must use frontslashes (`/`) in the path to the `dtagent.dll`.

Parameters and options

The following table lists and describes parameters and options. All items can not only be passed as connection string arguments, but also expressed as environment variables. For example, `DT_LOGLEVEL` is functionally similar to `loglevel`, but less flexible if you have several components on one machine.

See Application Integration for examples and details of parameter usage on various application servers.

Parameter	Description

`name=<agentname>`	This mandatory parameter assigns a descriptive name to the AppMon Agent. The name should reflect the tier (here: Java) or application that the agent observes, possibly concatenated with _SystemProfileToMapInto (`mySystemProfile` in the above example). The agentname must be unique among all deployed Agents and must not contain spaces, tabs, commas or other special characters.

`collector=<host>[:port]`
or
`server=<host>[:port]`
	The `server` parameter refers to the Server-embedded or stand-alone Collector. It sets the host, and optionally, the port to which the Agent connects. The host may be a (fully qualified) host name or an IP Address. The port defaults to 9998.

`loglevel=<level>`	Optional: Specifies the overall log level. The level can be one of the following: `finest | finer | fine | config | info | warning | severe | none`. When the `loglevel` option is omitted, the level defaults to info.
`logstdout=<level>`	Optional: Specifies the log level to write to the console. When you set this option to NONE, it suppresses all log output to console, but still writes to the log file. The level is one of the following: `finest | finer | fine | config | info | warning | severe | none`. When the `logstdout` option is omitted, the level defaults to info.
`logfile=<filename>`	Optional: Specifies the filename of the logfile. If omitted, the filename defaults to `<dt_agentname_pid.log>`.
`logpath=<path>`	Optional: Specifies the target path to write the log files. If omitted, the logpath defaults to `../../log` relative to the location of the native agent library. **Note** Make sure that the directory is writable for the application process.
`hirestimer=<timertype>`	Optional: This option sets the high-resolution timer you must use for this AppMon Agent. The `timertype` is one of the following values: `cpu | os | java | runtime | auto`. When the hirestimer option is omitted or set to `auto`, the `timertype` defaults to the best option for your platform.

`wait=<seconds>`	Optional: Specifies the initial wait timeout — the maximum time to wait for a connection to an AppMon Collector in seconds. If the connection cannot be established within this timeframe, the application continues uninstrumented. Defaults to 20 seconds now; was 60s until 5.5. **Note** AppMon Collector often referred to as Server (also in Client dialogs), because the Collector was integrated in the AppMon Server. You can still use the Server-embedded Collector to save memory when you test AppMon, but the standalone Collector is enforced (with the license) in real-world scenarios.
`sotimeout=<seconds>`	Optional: Specifies the native AppMon Agent´s socket timeout to send and/or receive data in seconds. Defaults to 30 seconds.
`ctimeout=<seconds>`	Optional: Specifies the maximum time that the native AppMon Agent waits until a connection to the AppMon Server is established. Default is 10 seconds.
`tmtimeout=<milliseconds>`	Optional: If specified, AppMon issues a warning log message if the transformation of a single class exceeds the specified time.
`transformationmaxavgwait=<milliseconds>`	Optional: If the average class transformation time (based on the first 1333 transformed classes) exceeds this timeout, the instrumentation stops, but the AppMon Agent stays connected, and resource dumps are possible. Defaults to 60 milliseconds.
`overridehostname=<custom hostname>`	Optional: This flag disables the auto detection of the hostname. It overrides the equivalent option in an agent configuration `.ini` file.
`optionTestRunIdJava=<testRunId>`	Identify the test to be run with an ID obtained from a REST call for test automation.
`enforcepci=true|false`	Prevents capturing of security/privacy relevant info (method arguments and memory dumps) directly on the Agent, meaning it cannot be changed on the Client. For AppMon 6.3.4 and later, the default is false. For PCI-compliance set this to true.
`usessl=true|false`	Enables SSL encryption between the Agent and the Collector. Encrypted communication uses the same port as non-encrypted. To enable SSL you need to set this parameter as true before launching the Agent.

Environment variables

The use of DT_HOME is optional.

If set, `<DT_HOME>` should refer to the AppMon installation directory. With a default full installation:

- On Windows, `%DT_HOME%` should point to `%ProgramFiles%\dynaTrace\dynaTrace 7.0`
- On *NIX, if you change to the `opt` directory (`cd /opt`) before installation as recommended, `$DT_HOME` should point to `/opt/dynatrace-7.0`.

If specifying the Agent path, you only need to put `dtagent` as <agentlib>. Path statements are:

- Windows: `set PATH=%PATH%;%DT_HOME%\agent\lib[64]`.
- *NIX: `export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$DT_HOME/agent/lib[64]`
  Use `export` for bash or `setenv` for csh.

Debugging and agent conflicts

When enabling debugging, the Agent conflicts with `jdwp` tools. To prevent this, use the Java option:

```
`-agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=10284`
```

instead of

```
`-Xdebug`

`-Xrunjdwp:transport=dt_socket,address=10284,server=y,suspend=n`
```

OneAgent usage

If you're using OneAgent, don't forget to configure ports for it.

OneAgent released as BETA functionality for AppMon 2017 May. Please be aware that:

- Not all features/sensors available in Classic Agent are already available in the OneAgent for AppMon
- Features/Sensors known from Classic Agent can behave differently

To use OneAgent in AppMon 2017 May you need to install Classic Agents first, and then roll them out to the OneAgent. No additional configuration needed. Before you switch to OneAgent, double check your major requirements with the available feature set.

Warning

You must **not** use the OneAgent in your production environment. The aim of BETA is to test it in pre-production environment, and see how the new Agent affects your custom dashboards/configurations you use to monitor your production environment.

.NET Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

.NET Agent configuration tool

Creating a .NET Agent configuration

Editing existing Agent configurations

Enable/disable Agent Configurations

Other settings

Upgrade

OneAgent usage

Related pages

.NET Agent

Watch and learn

Evaluating Dynatrace for .NET Applications

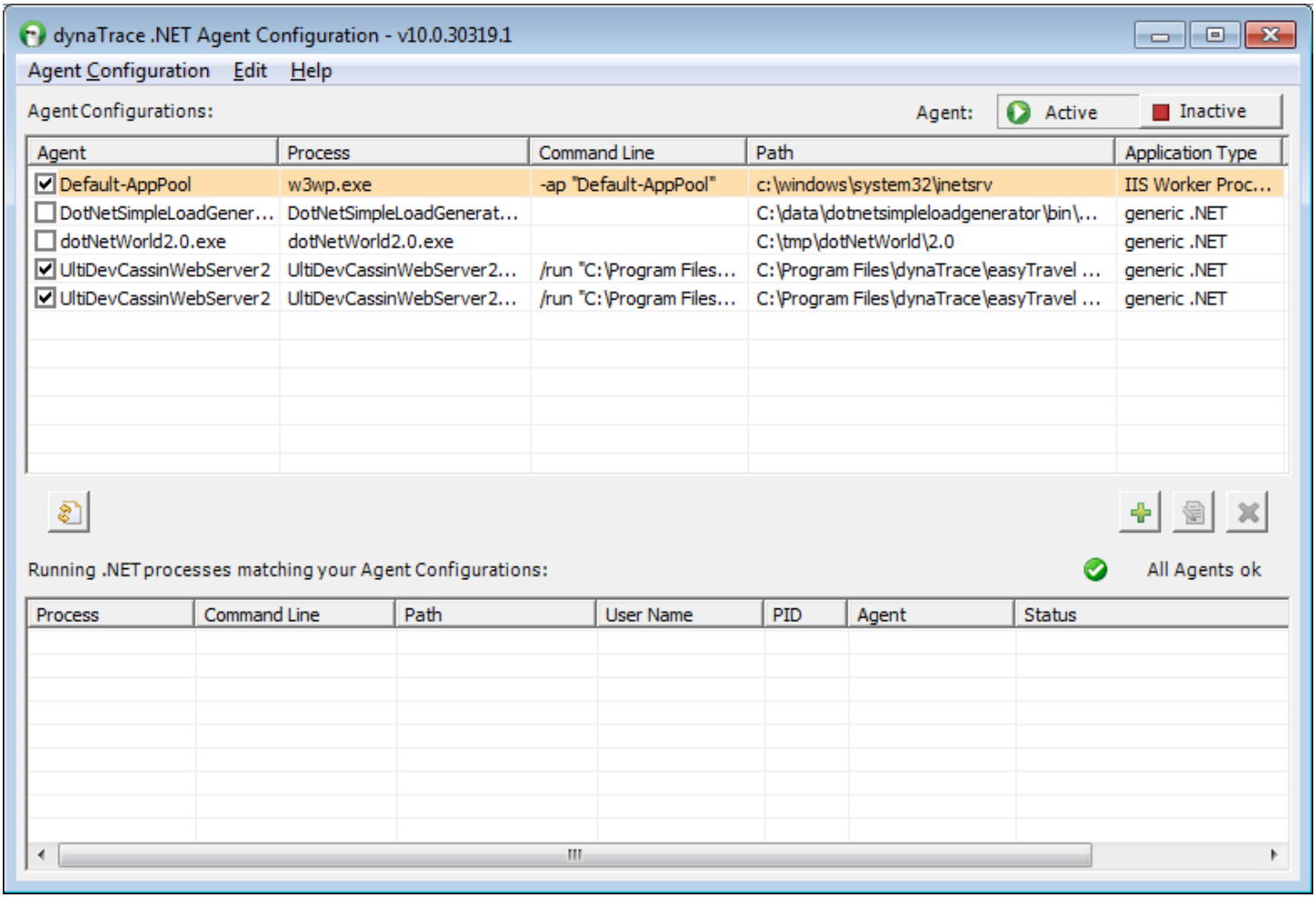
Use the .NET Agent Configuration tool to create or edit .NET Agent configurations and to enable/disable the Agent.

.NET Agent configuration tool

Note

The .NET Agent Configuration tool can cause small CPU spikes every ten seconds when open.

You can start the .NET Agent Configuration tool by selecting **All Programs > Dynatrace > Agent Configuration Tool for .NET** in the **Start** menu, or by double-clicking the executable `dtagentconf.exe` in the `<DTHOME>` directory.



Main window

The **.NET Agent Configuration** tool main window has two parts. The upper part shows a list of Agent configurations. They define in which .NET process an AppMon Agent should be placed.
The **Application Type** column displays the type of .NET process. ASP.NET Worker Processes are highlighted to distinguish them from other .NET processes.

The first time the **.NET Agent Configuration** tool starts, this list is empty because no agent configurations have been defined yet. See Creating a .NET Agent Configuration (below) for details on how to create an Agent configuration.

The bottom part of the window lists all currently running .NET processes that the active Agent configurations in the upper list. This list is updated automatically every 20 seconds and when an Agent configuration is activated/deactivated by clicking the checkbox next to it. You can manually refresh the processes list by clicking the refresh button.

Status messages

Status message	Agent Loaded/Unloaded Successfully? The following should be loaded: `dtagent.dll` (Bootstrap agent) `dtagentcore.dll` (.NET agent)	Environment Set Correctly? The following variables should be set: COR_ENABLE_PROFILING COR_PROFILER	Comments
Enable profiling:			
.NET agent successfully loaded.	(Bootstrap agent loaded) (.NET agent loaded)	(active)	Everything
Agent is not activated (environment variables COR_PROFILER and COR_ENABLE_PROFILING are not set). Press **Active** to set environment variables. Process restart or system reboot is required.	(Bootstrap agent not loaded) (.NET agent not loaded)	(inactive)	• Enviro (`COR_` are no • Click **A** neces
Bootstrap agent (dtagent.dll) is not loaded, but has been activated. Process restart or system reboot is required.	(Bootstrap agent not loaded) (.NET agent not loaded)	(active)	• The `d` proces • In som detect althou Explor `dtag`

Bootstrap agent (`dtagent.dll`) is loaded, but .NET agent (`dtagentcore.dll`) is not. Process restart or system reboot is required. Please make sure that bootstrap agent can connect to collector and the environment variables COR_PROFILER_PATH and DT_AGENTACTIVE are not set.	(Bootstrap agent loaded) (.NET agent unloaded)	(active)	• If an al proces • Make s Reviev • Check `COR_` are se reboot • Make s not, Cl proces
Agent did not connect or did not find matching system profile. Please make sure that agent can connect to collector, and that agent settings system profile name is valid. (.NET agent version: 'v…', Bootstrap agent version: 'v…').	(Bootstrap agent loaded) (.NET agent loaded)	(active)	• No inst • Make s Reviev • If the A in the • Make s does r proces
Disable profiling:			
Profiling disabled.	(Bootstrap agent unloaded) (.NET agent unloaded)	(inactive)	Everything
Bootstrap agent (`dtagent.dll`) is still loaded, but has been deactivated. Process restart or system reboot is required.	(Bootstrap agent loaded)	(inactive)	• Althou still loa profilin • Restar resolve
.NET agent (`dtagentcore.dll`) is still loaded, but has been deactivated. Process restart or system reboot is required.	(Bootstrap agent loaded) (.NET agent loaded)	(inactive)	• Althou still loa profilin • Restar help.

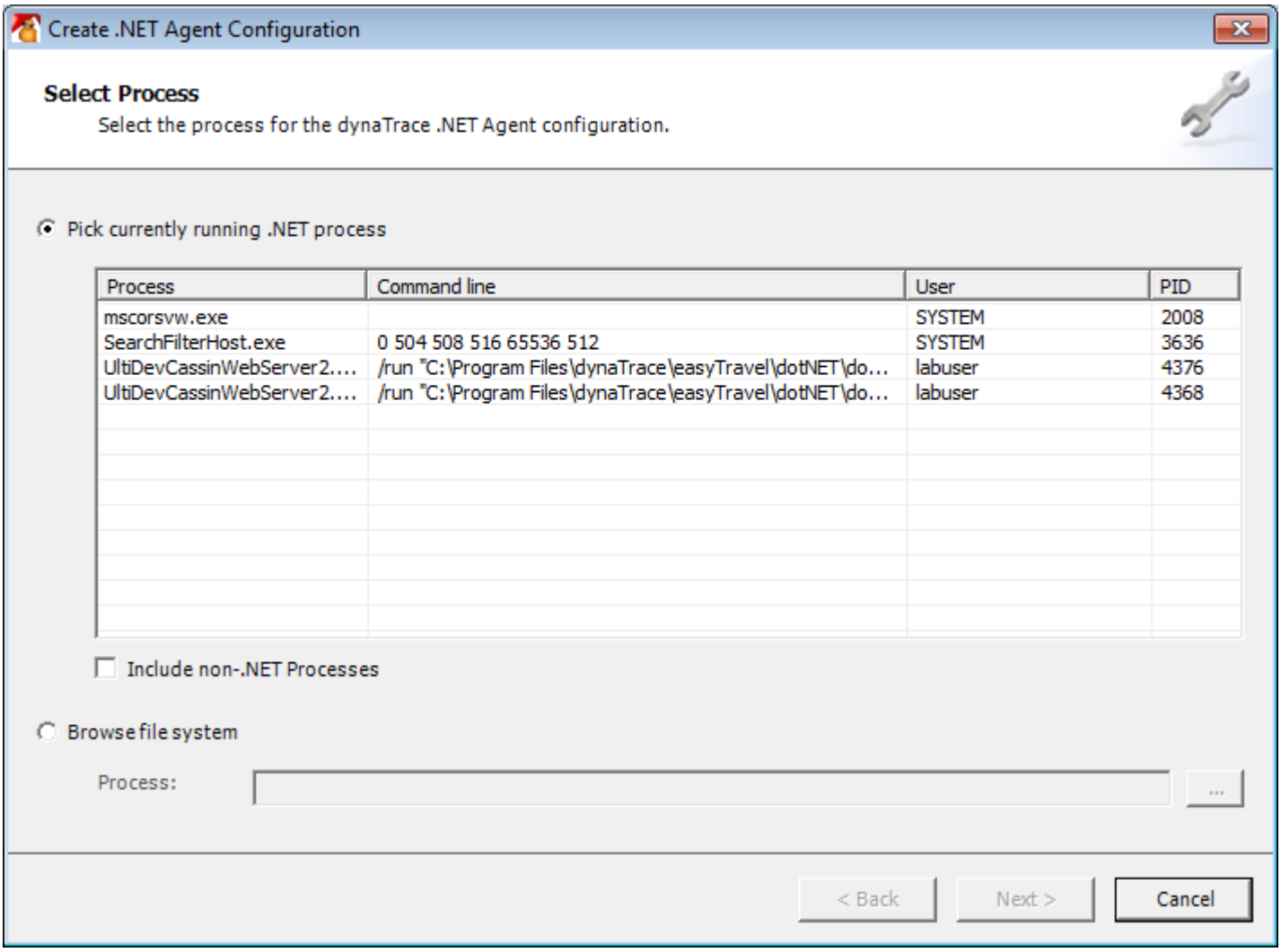
The following table lists possible resolutions for special cases.

Special Case	Comments / possible solutions
Status of 64-bit processes can only be determined for Vista / Server 2008 and later.	On x64 systems earlier than Windows Vista or Server 2008, modules of 64-bit processes cannot be enumerated. Therefore it is impossible to check whether the Agent (`dtagent.dll`) was loaded into a 64-bit process.

No status information available.	No status could be determined for the process. Check the agent status manually: • Open the **Agent Overview** in the AppMon Client and check whether the Agent is connected. • Use Process Explorer to check whether the environment is set correctly and `dtagent.dll` is loaded for the process.
Restart of IIS Services pending. For Restart, click Button 'Restart IIS-Services'.	A restart of the IIS services is pending if you activated or deactivated the .NET Agent but clicked No in the confirmation prompt asking if the IIS services should be restarted by the Agent Configuration Tool). In this case, a button is displayed in the main window to allow restart of these services at a more suitable time (for example, at night).

Creating a .NET Agent configuration

Select **Agent Configuration > New** from the menu, or click the **Add** button to display the **Create .NET Agent Configuration** wizard and create a new .NET Agent configuration.

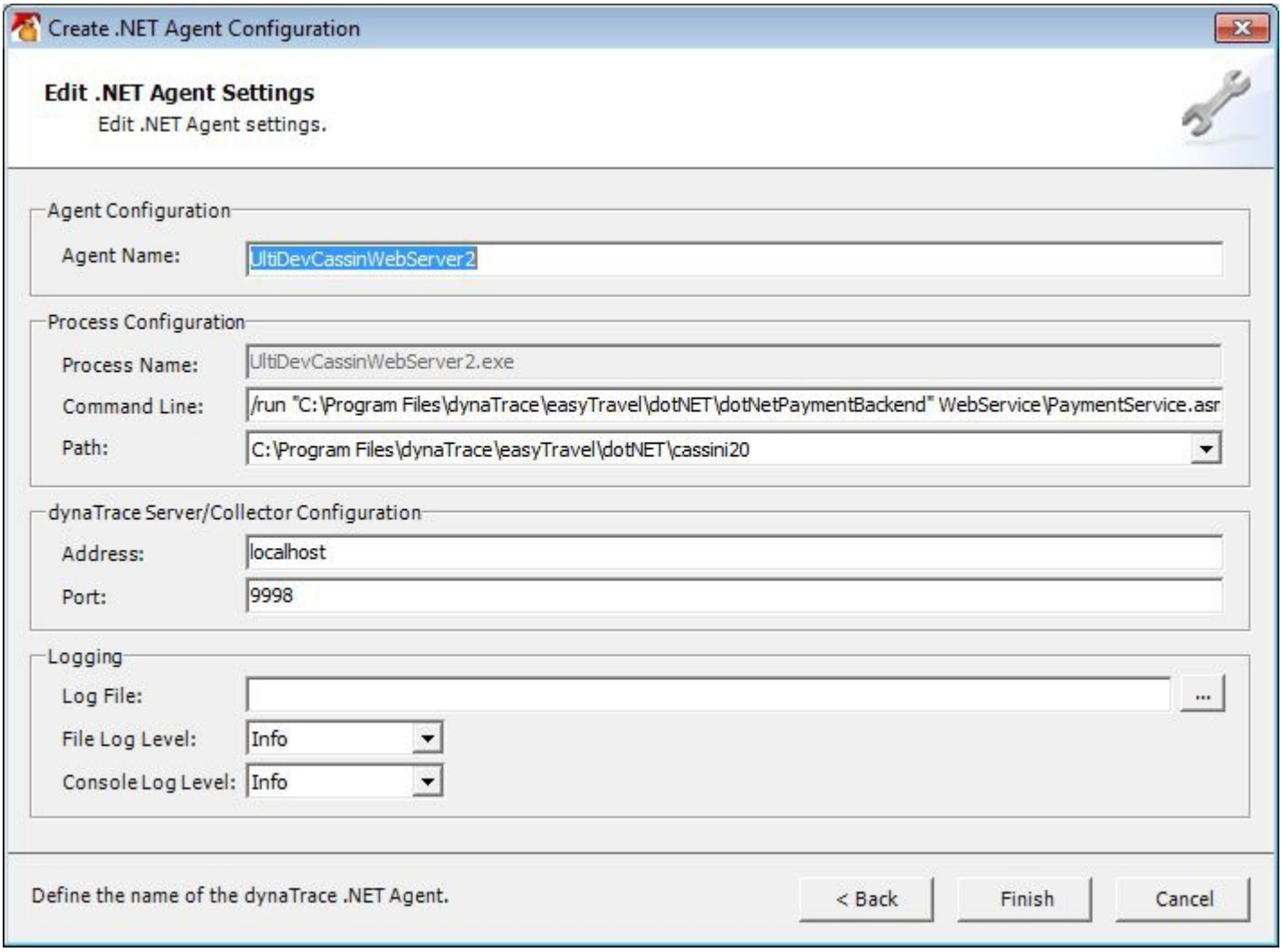


Select a process for .NET Agent configuration

On the **Select Process** screen, all running processes that host the .NET runtime are shown by default. Select **Include non-.NET Processes** to show all currently running processes. If the desired process is not currently running, you can locate the executable by selecting **Browse file system**, clicking the browse button to the right of the text box, and navigating to the executable.

> **Note**
>
> In some environments, especially 64-bit environments, the list of .NET processes may not show all running .NET processes. In this case, use the **Include non-.NET Processes** option.
>
> If an IIS worker process is not listed, set up a web request to the site. This starts the worker process for the application pool that hosts the site.

Select a process from the list and click **Next** to display the **Edit .NET Agent Settings** dialog box.



Adjust the settings of the Agent Configuration

Use this page to set details of the Agent configuration. The Agent name and the host name or address for the AppMon Server or Collector are required. All other settings are optional.

The following table describes the settings.

Setting	Description
Agent name	This mandatory setting assigns a descriptive name to the AppMon Agent. The name should reflect the tier or application that is observed by this AppMon Agent. The Agent name must be unique among deployed AppMon Agents. It must not contain spaces, tabs, commas, or other special characters.
Process name	This setting shows the name of the executable for the .NET process. It cannot be edited.
Command line	If a command line is provided, the AppMon Agent only handles processes that match the executable and path and also contain the value specified in the command line. The check is case-insensitive (for example, `WebService` is the same as `webservice`). If this field is blank, the process will be handled regardless of the actual command-line arguments.

Path	This setting specifies the location of the executable. You can use the list to select the path to either a specific executable or all executable files with the executable name.
Address Port	Use these settings to configure the host and port for connection to the standalone or embedded Collector. In earlier AppMon releases, this setting applies to the AppMon Server. In the current release, an Agent must always connect to a Collector. The **Address** value can be a fully qualified host name or an IP Address. The default **Address** and **Port** values are set to `localhost:9998`.
Log file	Specify the path and file name for writing the log files. If omitted, the file name defaults to `<agentname>.log` in the agent log directory. **Note** The executable process must have the privilege to write into the specified log folder. This might be problematic for some processes, such as the ASP.NET worker process, which is started by default with restricted privileges. If the process is not allowed to access the specified log folder, no logs will be generated.
File log level	Specify the overall log level. The level can be one of the following: `finest | finer | fine | config | info | warning | severe | off`.
Console log level	Specifies the log level for writing to the console. Setting this option to **off** suppresses all log output to the console while still writing to the log file. The level can be one of the following: `finest | finer | fine | config | info | warning | severe | off`.

Click **Finish** to create the Agent configuration with the current settings and return to the **.NET Agent Configuration** window. This new Agent configuration is now shown in the upper list and is activated by default. All currently running processes matching this AppMon Agent configuration automatically appear in the bottom list. After these applications are restarted, they are ready to connect to the AppMon Server.

Note

To place the AppMon Agent in an IIS worker process (`w3wp.exe`) after activating the AppMon Agent, restarting IIS is not enough because this does not restart `svchost.exe`, the parent process of `w3wp.exe`. You must restart all IIS services on activation or deactivation of the .NET Agent.

Choose **Yes** when prompted to restart the listed IIS services. Be aware that restarting affects **all** connected processes.

You can restart at a convenient time by clicking the **Restart IIS-Services** button that appears in the center of the main window if an IIS service restart is pending.

If restarting the IIS services does not resolve all issues, a system reboot is recommended.

If the Agent could not be placed in a process, a hint for solving the problem is shown in the **Status** column for the process.

Note

Certain Windows API calls are used to determine if the Agent was successfully injected. For 64-bit processes, these calls are available only in Windows Vista, Windows Server 2008, and later Windows versions. In earlier 64-bit environments, incorrect hint texts may appear in the processes list. In this case, these hints can be ignored.

Editing existing Agent configurations

To edit an existing Agent configuration, select the Agent in the upper list and click the **Edit** button, or simply double-click the Agent's row. The same wizard as described in Creating a .NET Agent Configuration is shown, but it displays the second page with the current settings. Click the **Finish** button saves your changes.

Delete one or more selected Agent configurations by clicking the **Delete** button or by pressing the **Delete** key. You can copy and paste an existing configuration, as in all Windows applications, then edit the new configuration to provide unique details. This is especially useful for creating several similar Agent configurations.

Enable/disable Agent Configurations

In the **Agent Configuration** list, the check box adjacent to the Agent Configuration name controls whether that Agent Configuration is enabled (checked) or disabled (unchecked) when the .NET Agent is active. Click the check box for an Agent Configuration to toggle it between enabled and disabled. Disabling an Agent Configuration does not delete it from the **Agent Configuration** list.

Other settings

Application pool handling for IIS 7.x and IIS 8.*x*

ASP.NET worker processes on IIS 7.x and IIS 8.*x* are handled differently from other .NET processes. IIS 7.x and IIS 8.*x* use *application pools*, which can host one or more ASP.NET web applications. An application pool is uniquely identified by its name. The name is passed to the ASP.NET worker process via the `-ap` command line option. To distinguish the different application pools, AppMon captures this part of the command line and creates a separate process entry. This facility allows easy configuration of the ASP.NET worker processes for each application pool.

Activate or Deactivate the .NET Agent

Click the **Active** button at the top right corner of the window to activate the .NET Agent and enable all checked Agent Configurations.

If the AppMon Agent is enabled successfully, the button changes to appear “pressed”.

Click the **Inactive** button at the top right corner of the window to deactivate the .NET Agent and disable all Agent Configurations.

Enable local profiling

When you don't have the required permissions to set system-wide environment variables, enable profiling in local files (for example, batch scripts) by setting the following environment variables.

```
@echo off
REM set environment variables necessary for loading the dynaTrace Agent
set COR_ENABLE_PROFILING=0x1
set COR_PROFILER={DA7CFC47-3E35-4C4E-B495-534F93B28683}
set DT_AGENTNAME=MyDotNetAgent
```

Environment variables

Most of the settings for the .NET Agent can be configured via environment variables. The values of the environment variables overwrite the values that you configured with the **.NET Agent Configuration** tool.

The following environment variables are recognized.

Environment variable	Description
COR_ENABLE_PROFILING	0x1: Enable profiling for your .NET process.
COR_PROFILER	Specify the AppMon .NET Agent DLL that will be loaded by the .NET runtime. Use the value `{DA7CFC47-3E35-4C4E-B495-534F93B28683}` to inject the .NET Agent.
DT_AGENTNAME	Assign a descriptive name to your AppMon Agent. For details, see the **Agent Name** setting above.
DT_SERVER	`[hostport]` Specify the AppMon Server host and port to connect to.
DT_SOCKTIMEOUT	Specify the instrumentor channel send/receive (socket) timeout. The default is 120 seconds.
DT_AGENTACTIVE	`["true"|"false"]` Activate (`true`) or deactivate (`false`) the .NET Agent.
DT_LOGFILE	`[String]` Override the log file path. Specify the full path where the Agent log file should be written to. The Agent inserts the PID before the last period (**.**) in the file name (registry key `'logfile'`).

DT_LOGLEVELCON	`[level]` Override the log level for console logging (registry key `'loglevelcon'`). Possible values for log levels are: • NONE / OFF • FINEST • FINER • FINE • INFO • WARNING • SEVERE
DT_LOGLEVELFILE	`[level]` Override the log level for the log file (registry key `'loglevelfile'`).
DT_LOGFILESIZE	[num] Sets the maximum size in bytes for a log file. Default is 10 MB (10*1024*1024). Reaching the maximum triggers log file rotation (registry key `'logfilesize'`).
DT_MEASUREOVERHEADNATIVE	`["true"|"false"]` Enable overhead log functionality for the native Agent part (registry key `'measureoverheadnative'`).
DT_MEASUREOVERHEADNONNATIVE	`["true"|"false"]` Enable overhead log functionality for the managed Agent part (registry key `'measureoverheadnonnative'`).
DT_MEASUREOVERHEAD	`["true"|"false"]` Enable overhead log functionality (registry key `'measureoverhead'`).
DT_TRANSFORMATIONSAMPLES	`[num]` Number of samples to take for determining slow transformation (modules) (registry key `'transformationsamples'`)
DT_TRANSFORMATIONMAXAVGWAIT	`[ms]` The maximum average wait time in milliseconds for determining slow transformation (registry key `'transformationmaxavgwait'`).
DT_WAIT	[num] This is the registry key 'wait'. It is the maximum time in seconds (default: 20; was 60 until 5.5) an Agent waits for a connection to an AppMon Collector[1]. If the connection cannot be established within this time-frame, the application continues uninstrumented.

Note

[1] The Collector is often referred to as "Server" (not only in configuration strings and files, but also in Client dialogs), because the Collector used to be integrated in the AppMon Server. You can still use the Server-embedded Collector when you test AppMon, but the standalone Collector is enforced (by the license) in real-world scenarios.

DT_HIRESTIMER	`[cpu|os|runtime|auto]` Specify the timer the Agent should use (registry key `'hirestimer'`).
DT_DISABLEINITIALLOGGING	`["true"|"false" (default)]` Disable logging until the Agent gets valid settings (omit the startup banner for disabled Agents).
DT_TESTRUN_ID	Identify the test to be run with a `<testRunId>` obtained from a REST call for test automation.
DT_USESSL	Enables SSL encryption between the Agent and the Collector. Encrypted communication uses the same port as non-encrypted. To enable SSL you need to set this parameter as true before launching the Agent.

Upgrade

You do not have to upgrade certain Agents (7.0: Java, .NET, Web Server, PHP, Native ADK, z/OS, and Host Agent). A bootstrap part checks for the correct AppMon Agent library when your application starts.

See Upgrade and Migration Guide for information on upgrades and migration.

Command-line options

The **.NET Agent Configuration tool** provides the following command-line option that executes registration actions without launching the user interface. This option is helpful for automated scenarios.

Option	Comment
/repairnative	Re-registers the AppMon native Agent.

OneAgent usage

If you're using OneAgent, don't forget to configure ports for it.

OneAgent released as BETA functionality for AppMon 2017 May. Please be aware that:

- Not all features/sensors available in Classic Agent are already available in the OneAgent for AppMon
- Features/Sensors known from Classic Agent can behave differently

To use OneAgent in AppMon 2017 May you need to install Classic Agents first, and then roll them out to the OneAgent. No additional configuration needed. Before you switch to OneAgent, double check your major requirements with the available feature set.

Warning

You must **not** use the OneAgent in your production environment. The aim of BETA is to test it in pre-production environment, and see how the new Agent affects your custom dashboards/configurations you use to monitor your production environment.

.NET Agent troubleshooting

> **Note**
>
> To use AppMon User Experience Management (UEM) for .NET applications, you must use the Web Server Agent for IIS. There is no option that would allow you to use UEM without a Web Server Agent.

```
net localgroup "Performance Monitor Users" /add "IIS APPPOOL\<AppPool Name>"
```

The resolution is to add the user `ApplicationPoolIdentity` to the local application server user group called `Performance Monitor Users`. Because the GUI to add users to this group may also be restricted, a reliable way is to do this via command prompt:

```
warning [native] Unable to get Performance Counter Value
```

For IIS 7 and later, by default the user `ApplicationPoolIdentity` is set to be used for the application pool worker processes. This causes the worker process (including the Agent) to run under restricted permissions that might not contain the rights to fetch the Windows performance monitor measures for the process health of its own CLR. This results in not seeing all process health metrics (for example, garbage collection values), and seeing messages in the Agent log like the following:

IIS may sometimes start a worker process, but not load or only partly load the CLR with the web application if there are no requests on the web server or only requests to static resources. If you see that a monitored `w3wp.exe` instance is loaded but no Agent is connected, no bootstrap log is written (even if `dtagent.dll` is loaded inside), and the memory usage of the `w3wp.exe` process is quite low compared to other started worker processes), then it is most likely the case that you need to give some load to the web server that needs the web application being loaded.

Special behavior of IIS worker processes

8. The main Agent is loaded (a new log `dt_<agentname>_<pid>.log` is created) and tries to

instrument the application according to the settings. The assemblies are sent to the Collector, instrumented, and sent back to the Agent, where the code is executed.

Possible problems

The application takes a long time to come up.

Solution

Place a Collector near the Agent or directly on the machine where the Agent and application are running, to shorten the path and avoid network latency.

7. If the agent is successfully connected to the Collector, it checks whether a newer Agent version is available and tries to download it.

   Possible problems

   No write access is allowed for the user the process is running with under `<DT_HOME>\agent\downloads`.

   Solution

   Give the user modify, read, and execute rights in the file system to that folder (check the bootstrap log).

6. If a configuration is found, the Agent tries to connect to the given Collector (**Server** setting) and creates a new `dt_<agentname>` *bootstrap* `<pid>.log` file in the `<DT_HOME>\log` folder. If connection problems occur, the Agent tries to connect for 60 seconds (default timeout setting) and blocks the process from executing. When the Agent times out, no instrumentation is done (the same as when no configuration for this process exists).

   Possible problems

   - Windows services that use a .NET CLR cannot be started if AppMon tries to monitor them with a wrong Collector setting.
   - The IIS worker process is not starting.

   Solution

   - Check the Collector settings.

- Increase the service start timeout for the service control manager.
- Check the startup time limit in the advanced settings of the IIS application pool.

5. If no configuration or an inactive configuration is found for that process (no whitelist entry or no `DT_...` environment variables), no instrumentation is done.

   Possible problems

   On unmonitored .NET command-line applications (for example, MS PowerShell), you can see the two messages:

   ```
   info [native] No Registry-Settings exist for this process
   ```

   ```
   info [native] Agent has no active settings - running normally
   ```

   Solution

   For debugging reasons, the Agent also writes the logs to the console until the log settings to that process are read from the whitelist. You can disable this by using the environment variable `DT_DISABLEINITIALLOGGING`, set to `TRUE`.

4. The bootstrap agent goes through the whitelist entries, compares the executable name and path for equality, and checks the command-line parameters if they contain the related whitelist setting.

   Possible problems

   No active configuration can be found for a configured process, so no Bootstrap Agent log is created.

   Solution

   - Remove any inactive configuration entries that match this process (no parameter, same parameter), because the Agent quits checking after the first matching entry and the order in which the entries are checked (how Windows enumerates them) is random.
   - Check whether the executable path setting equals the whitelist entry. For example, when you change a `w3wp` worker process to use 32-bit instead of 64-bit, the path changes from `C:\Windows\System32\inetsrv` to `C:\Windows\SysWOW64\inetsrv` and the whitelist entry does not match anymore.

3. The `dtagent.dll` tries to determine whether the process that it is loaded into should be instrumented. It first checks the optional environment variables. If they are not set, then it checks the

whitelist (created and managed by the .NET Agent Configuration tool) in the registry:

- x86 (32-bit)

```
HKEY_LOCAL_MACHINE\SOFTWARE\dynaTrace\Agent\Whitelist
```

- x64 (64-bit)

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\dynaTrace\Agent\Whitelist
```

Possible problems

- The user under which the process is running does not have registry read access rights.
- The whitelist is ignored because of the presence of the `DT_AGENTACTIVE` environment variable.

Solution

- Give the user read permissions to that registry key, subkeys, and values.
- If this cannot be accomplished (because of security guidelines, for example), a workaround is to use of the optional AppMon environment variables (including Microsoft `COR_ENABLE_PROFILING` and `COR_PROFILER` variables) with the possibility to set them only for the user, the process, or the IIS worker processes.
- If `DT_AGENTACTIVE` is set to `TRUE`, all whitelist settings are ignored and all .NET processes are instrumented just by using the `DT_SERVER` and `DT_AGENTNAME` environment variables. If these variables are not set, the default values are used (`localhost:9998` and the process name).

2. The CLR tries to load the DLL under the sub-key `\InprocServer32`.

Possible problems

- The user under which the process is running has no file system permission to read and execute the referenced `dtagent .dll`.
- The bitness (32-bit or 64-bit) of the referenced `dtagent.dll` does not match with the bitness of the process or CLR.

Solution

- Give the user read and execute rights in the file system to the referenced `dtagent.dll`.
- Check that the correct bitness versions are referenced in the registry keys mentioned above (for both cases, see the Knowledge Base article Error - Failed to CoCreate Profiler).

1. The CLR checks whether `COR_ENABLE_PROFILING` is set to `0x1`. If yes, it does a lookup into the registry:

   - x64 CLR on x64 OS or x86 CLR on x86 OS

   ```
   HKEY_CLASSES_ROOT\CLSID\<COR_PROFILER>
   ```

   - x86 CLR on x64 OS

   ```
   HKEY_CLASSES_ROOT\Wow6432Node\CLSID\<COR_PROFILER>
   ```

   Possible problems

   - No registry key exists.
   - The user under which the process is running does not have registry read access rights.
   - The profiler GUID is wrong.

   Solution

   - Reinstall the AppMon Agent with the installer, or use the following to register the DLL in a command prompt with administrator rights:

   ```
   regsvr32 <DT_HOME>\agent\lib(64)\dtagent.dll
   ```

   - Give the user read permissions to that registry key, subkeys, and values (possible with `regedit` executed with administrator rights).
   - Check that a correct AppMon .NET Agent GUID is used for `COR_PROFILER` (see the Knowledge Base article .NET agent cannot startup).

- For instrumenting IIS application pools, you normally need to run `iisreset` from a command prompt as administrator to get the `COR_*` environment variables passed to the worker processes, because restarting or recycling the application pool is not sufficient. But this is sometimes not enough to get all current environment variable changes into the worker process (for example, removal of `DT_AGENTACTIVE`), because these are inherited from the parent process (for IIS application pool worker processes, the parent process is `svchost.exe`). Killing the parent process might work, but the only safe way is a server restart. You can verify which environment variables are currently seen by a process/CLR by using Sysinternals Process Explorer (right-click the process and select **Properties** and click the **Environment** tab).

.NET CLR load

This page explains how the Microsoft .NET CLR loads the AppMon .NET Agent, lists problems which may occur, and suggests solutions them. It also contains info about special IIS behavior. For information about the Agent configuration causing high runtime overhead, see Agent Group - Agent Mapping > Advanced Settings.

PHP Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Architecture

Install, configure and verify

Limitations

OneAgent usage

Related pages

PHP API Breakdown

Standalone PHP Agent configuration

Usage example - PHP and Auto sensors

Usage example - user-defined PHP sensor

PHP Agent

Overview

PurePath technology provides full visibility into PHP application performance and behavior: From a browser click to the database and back, for all transactions, in real time 24x7. Smart auto-detection of PHP pages lets you see which page has performance issues. The core of the solution is the instrumentation of an Agent for the Apache HTTP server PHP extension.

AppMon has extensions to instrument PHP on Windows, either in combination with the Apache HTTP Server or IIS.

The following table lists the supported PHP versions by operating system.

Operating system	PHP 5.3	PHP 5.4	PHP 5.5	PHP 5.6	PHP 7.0	PHP 7.1[5]
Windows (x86)		✓ 1,3	✓ 1,3	✓ 1,3	✓ 1,3	
Linux (x86/x64)	✗	✗	✓ 1,2,3	✓ 1,2,3	✓ 1,2,3	
	✓ 1,2,3	✓ 1,2,3,4	✓ 1,2,3,4			

[1] mod_php (for Apache Web Server) and PHP CGI (for IIS).

[2] PHP FPM (fast process manager; for Nginx)

[3] PHP threaded MPM is not supported as it is not production setup safe. (reason: PHP_TS)

[4] Supported only via PHP OneAgent [5] As of 7.0.3

Supported PHP flavors include:

- PHP Zend Engine
- PHP FastCGI
- PHP-FPM
- suPHP

Architecture

Web servers like the Apache HTTP Server typically consist of several worker processes that handle different requests. Sometimes the processes contain several threads or one process is created for each request. Since the lifetime of worker processes is not known in advance, the AppMon Agent for web servers consists of the following parts:

- The PHP interpreter loads the extension. It gathers the information the AppMon Server uses for analysis.
- The Web Server Master Agent is a separate process that collects all worker process data. It handles communication with the Collector, retrieves configuration data, and performs other functions.
- Start the Web Server Master Agent on Linux separately as a daemon (like the service on Windows) and not with the sub-Agent. See Web Server Agent Configuration.

The PHP Agent communicates with the Web Server Master Agent through these channels:

- The shared memory segment propagates configuration data to the worker processes that the Web Server Master Agent receives from the collector, and holds other information shared across all worker processes. The shared memory segment is backed by a file on all platforms. The worker processes and the Master Agent process must have read/write access. This file is located in `<dynatrace install dir>/agent/conf`.
- The PHP Agent send collected data to the Web Server Master Agent via a stateless UDP connection. The Master Agent collects this data, transforms it, and forwards it to the AppMon Collector.

Install, configure and verify

1. Install, configure and start the Web Server Master Agent service / daemon. You need to at least configure the Agent name in `<dynatrace install dir>/agent/conf/dtwsagent.ini` that matches the mapping in your active System Profile. See Web Server Agent Configuration and Apache Web Server Agent Configuration for more information.

2. At the bottom of the `php.ini` file (or at least as last extension) add a reference to PHP Agent extension. The revision is not a part of the default installation path, so the version portion is 7.0 and not 7.0.0. A 64-bit Agent example for an AppMon default installation is as follows:

   ```
   extension=/opt/dynatrace-7.0/agent/lib64/libdtagent.so
   ```

   A 32 bit Agent example for an AppMon default installation on Windows in combination with Apache or IIS is as follows: `extension="C:\Program Files\dynaTrace\dynaTrace7.0\agent\lib\dtagent.dll" php.ini`

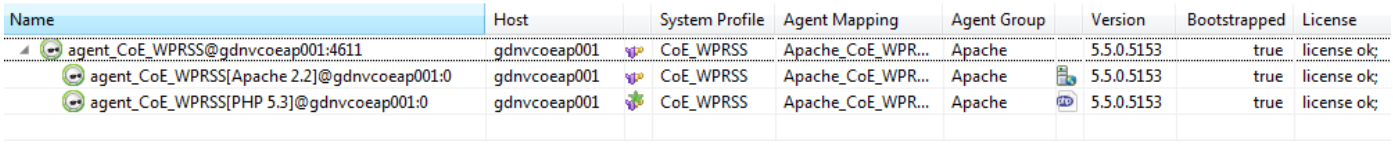
   This is typically is in `xampp\php` for XAMPP or `C:\Program Files (x86)\PHP\v<major.minorVersion>` when installed with the Web Platform Installer.

   The bootstrapper (`libdtagent.so` for *NIX or `dtagent.dll` for Windows) chooses the right PHP extension for the PHP version in use and threadsafe (`_ts`) or not. (`dtphpagent<majorMinorPHPversion>[_ts].so` or `.dll`)

   Regardless of the PHP distribution you deploy, the AppMon Agent start always triggers in the `php.ini` file. The `php-fpm.conf` file is not supported.

3. Restart the Apache HTTP Server, PHP-FPM or IIS.

4. You must browse the site to start the worker process and / or PHP. Verify that the Agents operate correctly. Choose **Configuration Panel > Agent Connection Status** or **Agent Overview** in the AppMon Client if Agents are connected and show up in a System Profile.



- Resort to the **System Information** dashboard or directly to the logs in `<dynatrace install dir>/log` if needed.
- The `dtphpagent` section in `phpinfo()` contains information as well, including Agent version, status (enabled or not) and the path to the log file. See the Apache sample page in a web browser so see this information.

Limitations

Reserved PHP keywords

AppMon does not support PHP reserved keywords instrumentation. See the PHP website for a complete list of keywords.

Include or require nodes in a PurePath do not refer to any PHP keywords, and have different meaning than the PHP keywords. They refer to the code in the PHP files, and not to the files. For example, if the PurePath shows the execution time for an include node, it shows how much time it took to process the code included in the files.

The only exception to this rule are the die() and exit() keywords, for which AppMon can report an exception, but only if the keyword occurs inside a function body.

Unsupported functions

- **curl_setopt**: The **curl_setopt** function calls are suppressed to keep the PurePath size small with the PHP cURL sensor. The limitation applies also to user-defined sensor rules. See PHP documentation for information about the function.

Capture default PHP argument

AppMon does not support default PHP arguments capture.

Accelerators

If you use PHP accelerator extensions in your PHP deployment, like eAccelerator, it does not report compilation time.

OneAgent usage

If you're using OneAgent, don't forget to configure ports for it.

OneAgent released as BETA functionality for AppMon 2017 May. Please be aware that:

- Not all features/sensors available in Classic Agent are already available in the OneAgent for AppMon
- Features/Sensors known from Classic Agent can behave differently

To use OneAgent in AppMon 2017 May you need to install Classic Agents first, and then roll them out to the OneAgent. No additional configuration needed. Before you switch to OneAgent, double check your major requirements with the available feature set.

> **Warning**
>
> You must **not** use the OneAgent in your production environment. The aim of BETA is to test it in pre-production environment, and see how the new Agent affects your custom dashboards/configurations you use to monitor your production environment.

Standalone PHP Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Prior to installation

Installation

Related pages

Related topics

When PHP runs in FPM mode, served for example by nginx or without activating a Web Server Agent in general, you can use the PHP monitoring features, but you need to perform a special installation procedure.

Prior to installation

Obtain the Web Server Agent installation package by adding a PHP application tier in your monitoring System Profile. Transfer the package to your Server and unpack it in a Linux FHS compliant directory (for example, `/opt/dynatrace-7.0`). Ensure that the `dtwsagent.ini` file contains correct values for the Agent name and the AppMon Server address. See Web Server Agent Configuration for more information on allowed values for configuration directives.

Make sure you know where the `php.ini` file is located. This depends on your Linux distribution and installation method (packages or from source).

Installation

To deploy PHP monitoring, you need to edit the `php.ini` file and enable the automatic start of the AppMon Agent manually.

1. Activate the PHP extension. to do this, edit the `php.ini` file to activate PHP monitoring. Use the **extension** directive supplemented with the path to the Agent library (`libdtagent.so`). Regardless of the PHP distribution deployed, the Agent start always triggers in `php.ini`. The `php-fpm.conf` file is not supported.

   ```
   extension=<agentPath>
   ```

   On a 64-bit Linux platform and PHP 5.4, add the following line:

   ```
   extension=/opt/dynatrace-<major.minorVersion>/agent/lib64/libdtagent.so
   ```

   > **Note**
   >
   > Add the directive at the bottom of `php.ini` file. The agent should be the last loaded extension.

2. Add the Agent start to your Linux boot sequence to ensure Agent start.
   You can use the `dynaTraceWebServerAgent` script to have Agent started each time your system boots. This script is available in your AppMon Agent installation directory, `<agentPath>/init.d/dynaTraceWebServerAgent` (for example, `/opt/dynatrace-7.0/init.d/dynaTraceWebServerAgent`). This way no manual Agent start is necessary when your system is restarted.

   a. Edit `dynaTraceWebServerAgent` and make sure the lines starting with # Default-Start and # Default-Stop contain correct runlevel numbers.
   By default, the Agent is set to start in runlevel 3 and 5. Default settings make Agent stop in runlevels 0, 1, 2, and 6. For example, to start the Agent also in runlevel 2, edit the script as follows:

```
# Default-Start: 2 3 5
# Default-Stop: 0 1 6
```

b. Copy the file to the `/etc/init.d/` folder.

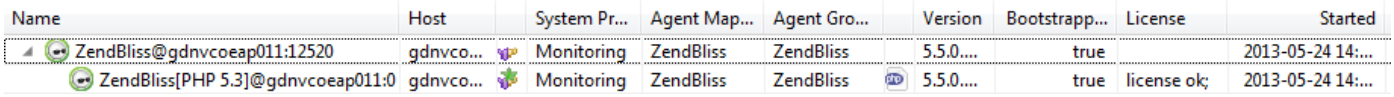
c. Use a distribution tool such as `chkconfig` or `update-rc.d` to enable proper startup and shutdown of the Agent.
You can also edit the /etc/rc.local script and add `<agentPath>/init.d/dynaTraceWebServerAgent start` there instead of using distribution tools.

3. Execute `/etc/init.d/dynaTraceWebServerAgent start` to start the AppMon Agent manually.

4. Execute `ps -ef | grep dtwsagent` to find the agent process and verify the agent properly loads.

```
ps -ef |grep dtwsagent
root     18290     1  0 15:35 pts/0    00:00:02 /opt/dynatrace-<major.minorVersion>/ag
```

You can also verify Agent communication with the AppMon Server using **Agents Overview** in the AppMon Client.



`phpinfo()` also contains information about AppMon Agent. The section **dtphpagent** contains information about agent version, status (enabled, not enabled), and the path to the log file.

5. Restart your web server. You may also need to restart your `php-fpm` daemon.

Usage example - PHP and Auto sensors

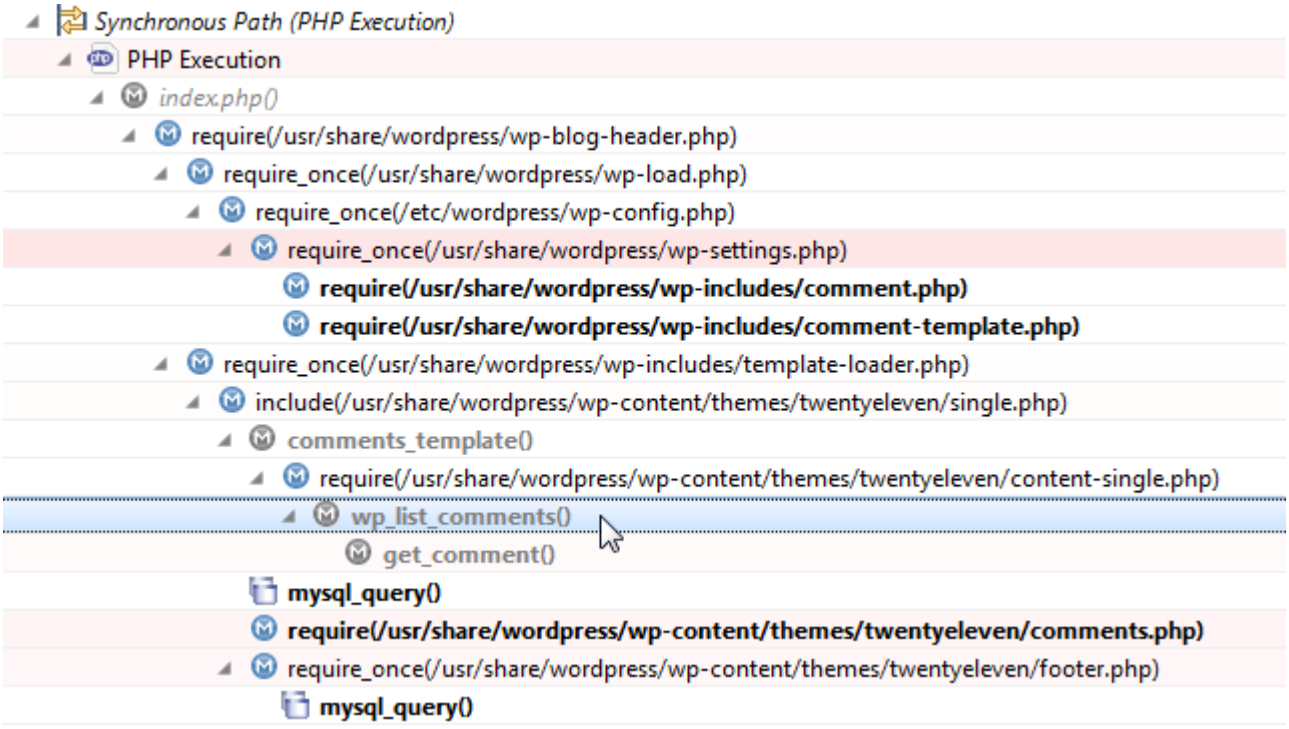
Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Usage example - user-defined PHP sensor

Sensor Configuration

Auto Sensors provide additional information on methods that affect the performance of your transaction. You can use this mechanism to identify problematic methods for PHP applications while a transaction is running and enrich your PurePaths with that data. Whenever there is a potential performance problem, the Auto Sensor brings up the function or method that is causing problems. The Auto Sensor triggered is marked by a gray method icon in the PurePath dashlet.

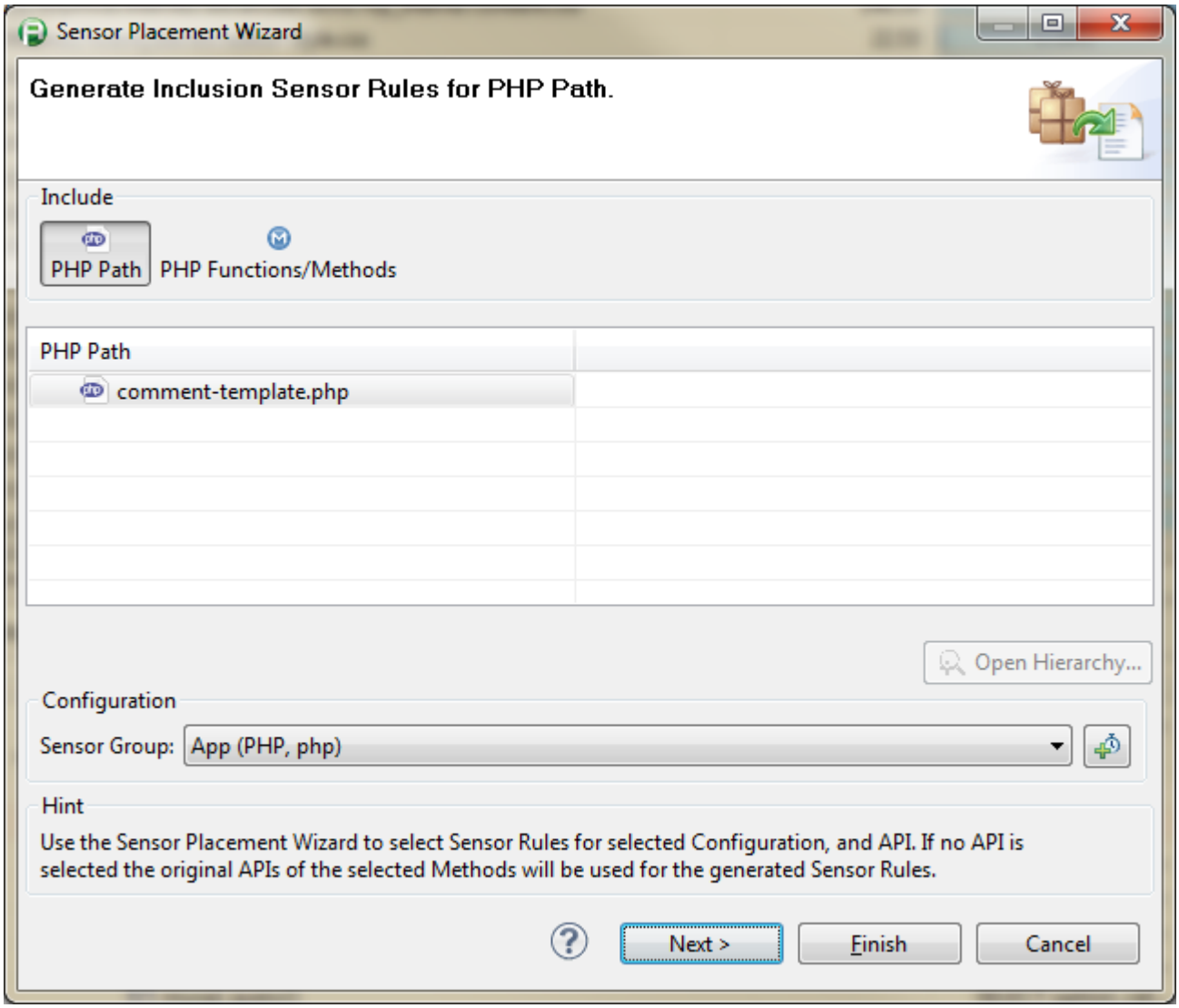


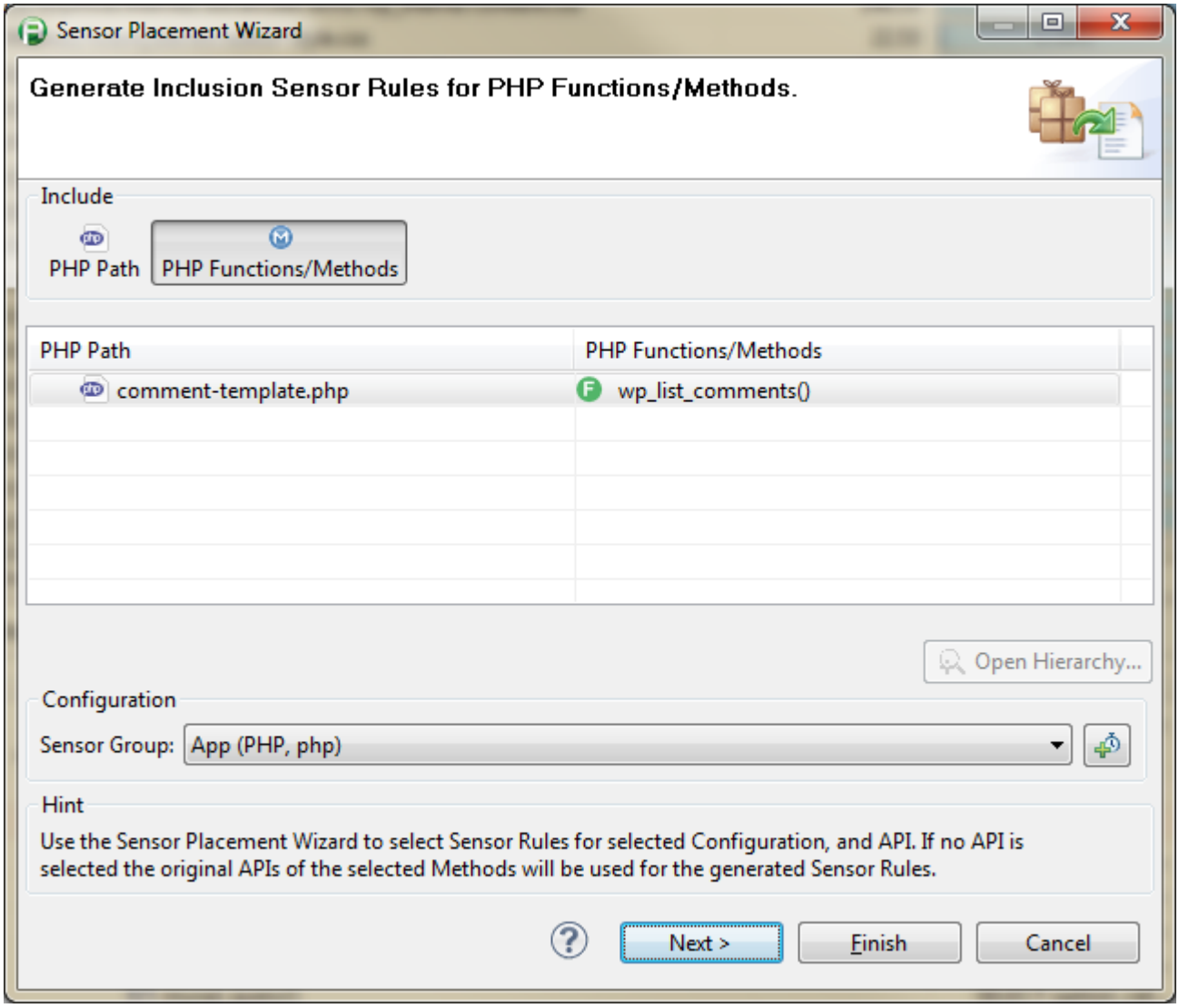
Adding sensor rules for PHP methods picked up by Auto sensors

1. Right-click the function or method picked up by Auto Sensors.

2. Select **Add Sensor Rules gt; Include Selected Methods** for the context menu.

3. Indicate whether you need to include only path or path and method or function.





4. Select the sensor group where you want the rule added and click **Next**. You can use the default **App** group or add a new one.

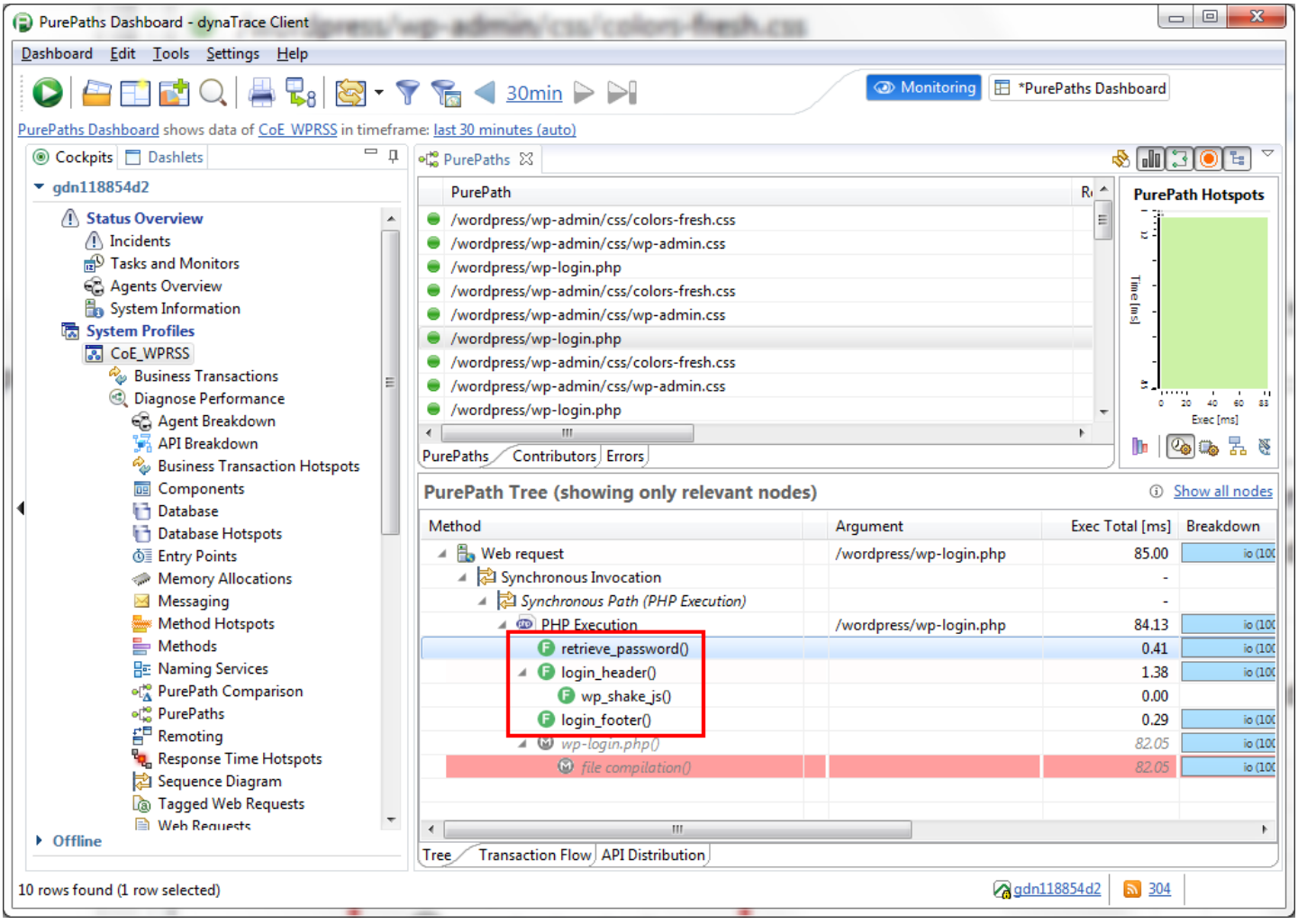
5. Click **Finish** to save your changes and proceed with monitoring.
   To enhance the rules for your function or method (for example, to add a new path), click **Next** and continue rule tuning. See Usage Example - User-Defined PHP Sensor and Sensor Configuration for more information.

Usage example - user-defined PHP sensor

- The maximum number of captured arguments is limited to four (4). For example, to capture four arguments type `1,2,3,4` in the **Capture Arguments** field.
- Use an * (asterisk) symbol to capture all arguments. It is recommended that you use this as a temporary diagnostic measure. Capturing all arguments negatively affects performance.
- Clear the **Capture Events** check box to disable a function or method rule.

PHP function or Method Rule notes

The new settings apply automatically to the next PHP request, and there is no need to restart Apache.

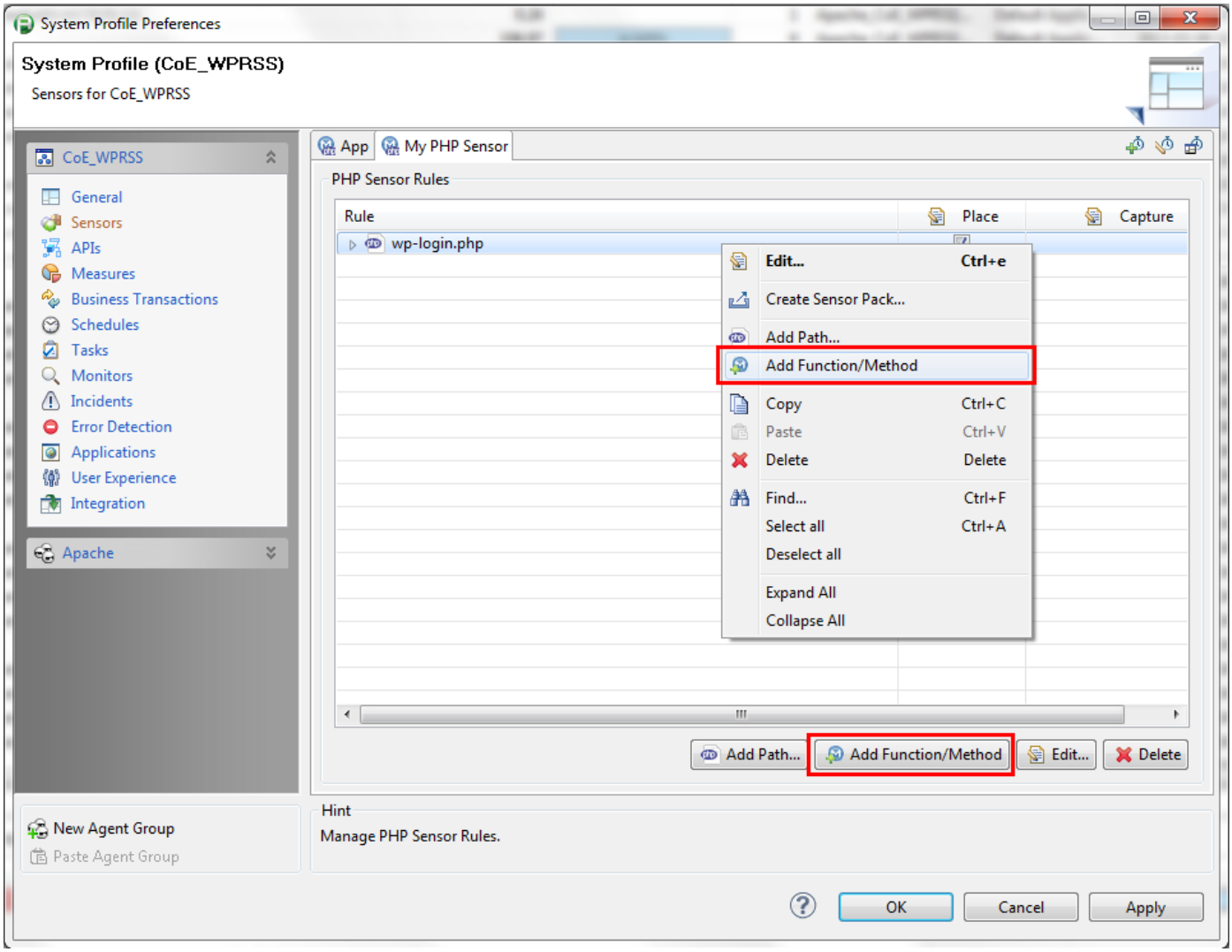


Result

See Sensor Configuration for more information about user-defined sensors.

7. Click **OK** to save the configuration.

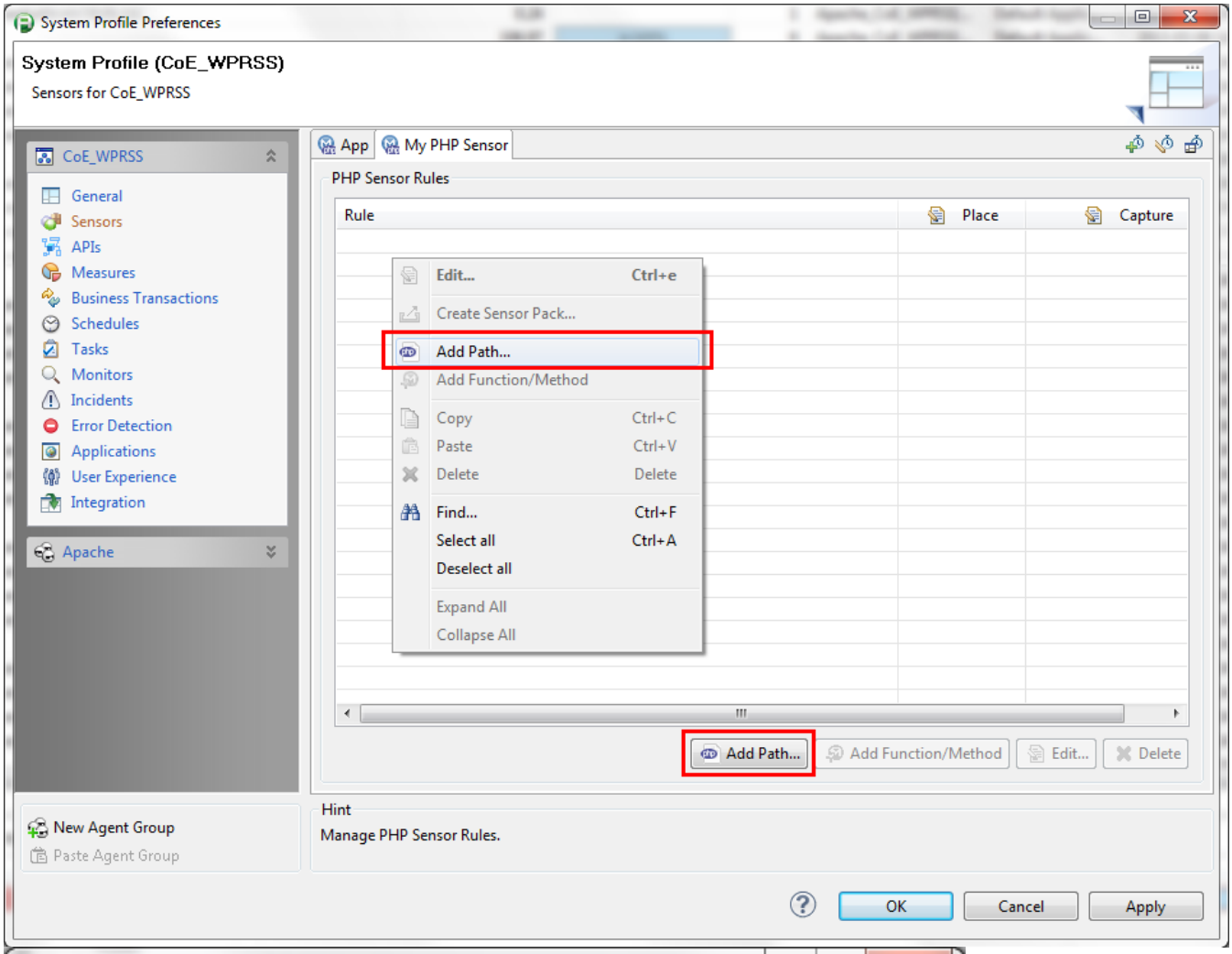
6. Optionally specify which functions or methods should be included in the Path Rule. For a selected Path Rule, click **Add Function/Method** to access the Rule properties.

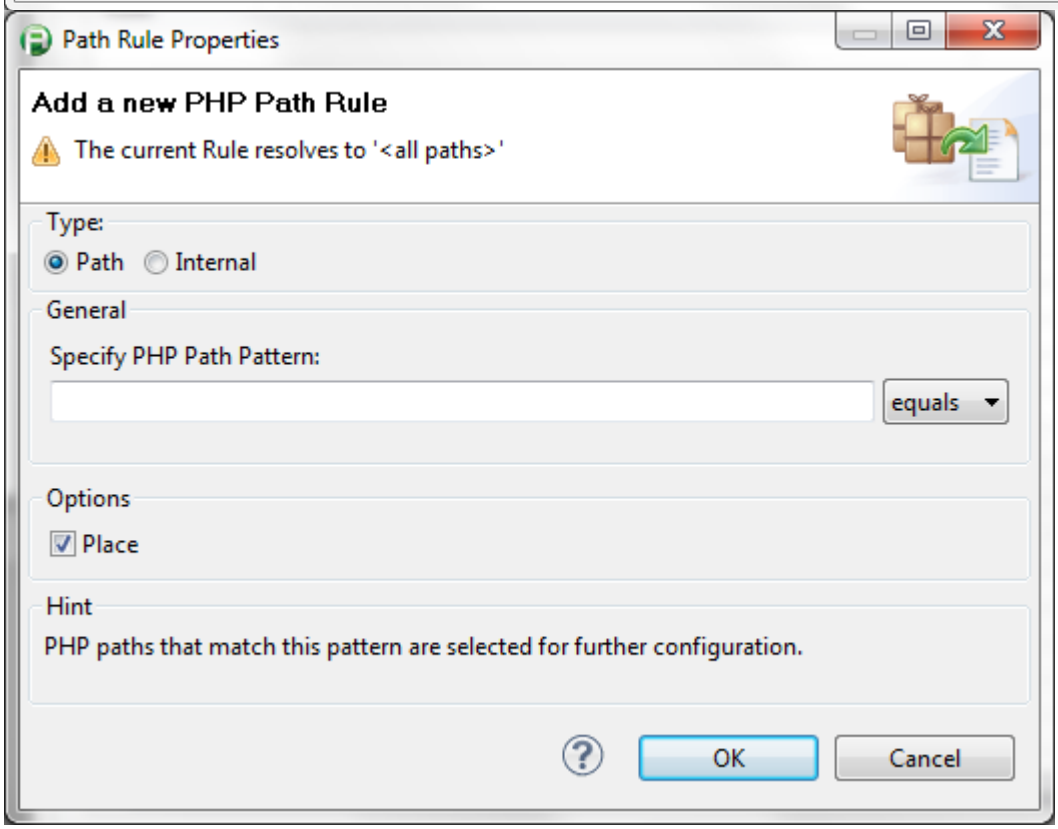


The example skips this step.

5. Define one or more sensor rules.

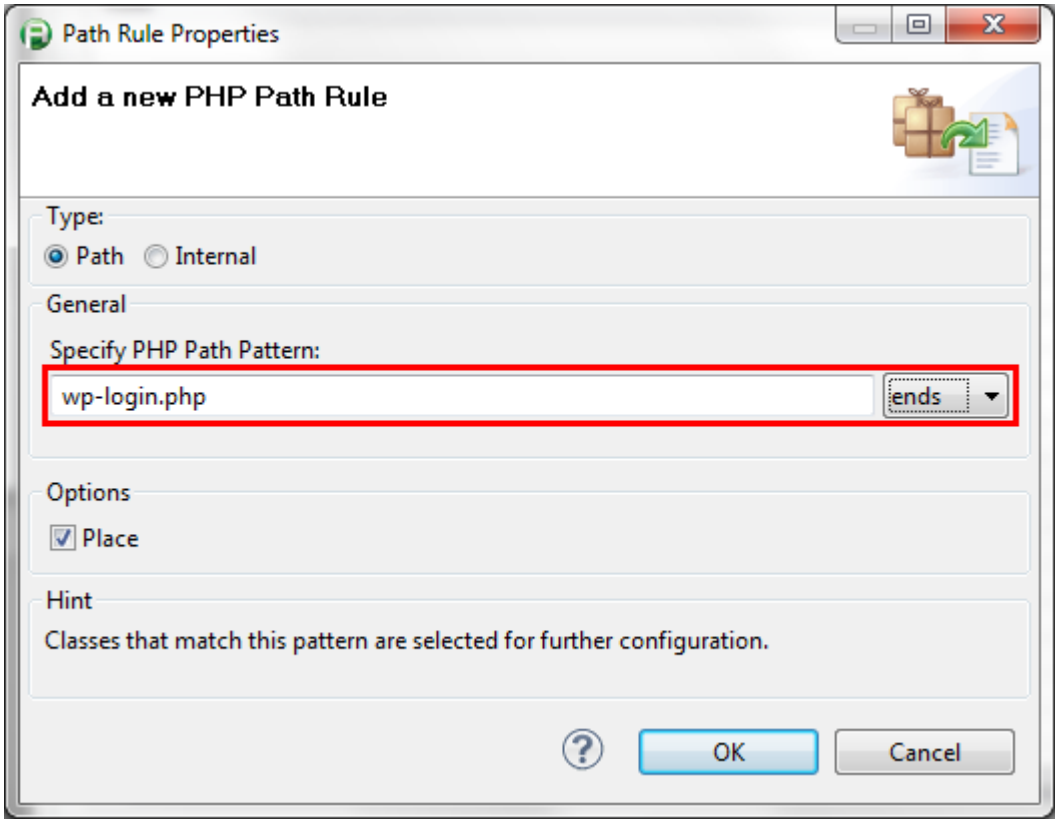
   a. Click **Add Path**. Optionally, you can also right-click in the **PHP Sensor Rules** table and choose **Add Path**. This opens the **Path Rule Properties** window.


System Profile Preferences
System Profile (CoE_WPRSS)
Sensors for CoE_WPRSS
CoE_WPRSS
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Integration
Apache
App
My PHP Sensor
PHP Sensor Rules
Rule
Place
Capture
Edit... Ctrl+e
Create Sensor Pack...
Add Path...
Add Function/Method
Copy Ctrl+C
Paste Ctrl+V
Delete Delete
Find... Ctrl+F
Select all Ctrl+A
Deselect all
Expand All
Collapse All
Add Path...
Add Function/Method
Edit...
Delete
New Agent Group
Paste Agent Group
Hint
Manage PHP Sensor Rules.
OK
Cancel
Apply


Path Rule Properties
Add a new PHP Path Rule
The current Rule resolves to '<all paths>'
Type:
Path
Internal
General
Specify PHP Path Pattern:
equals
Options
Place
Hint
PHP paths that match this pattern are selected for further configuration.
OK
Cancel

b. Leave the **Path** rule type selected. This rule type applies to the user functions that match the PHP Path Pattern.

c. Specify the **PHP Path Pattern**. For example, to see all functions called by the wp-login.php file in the PurePath:



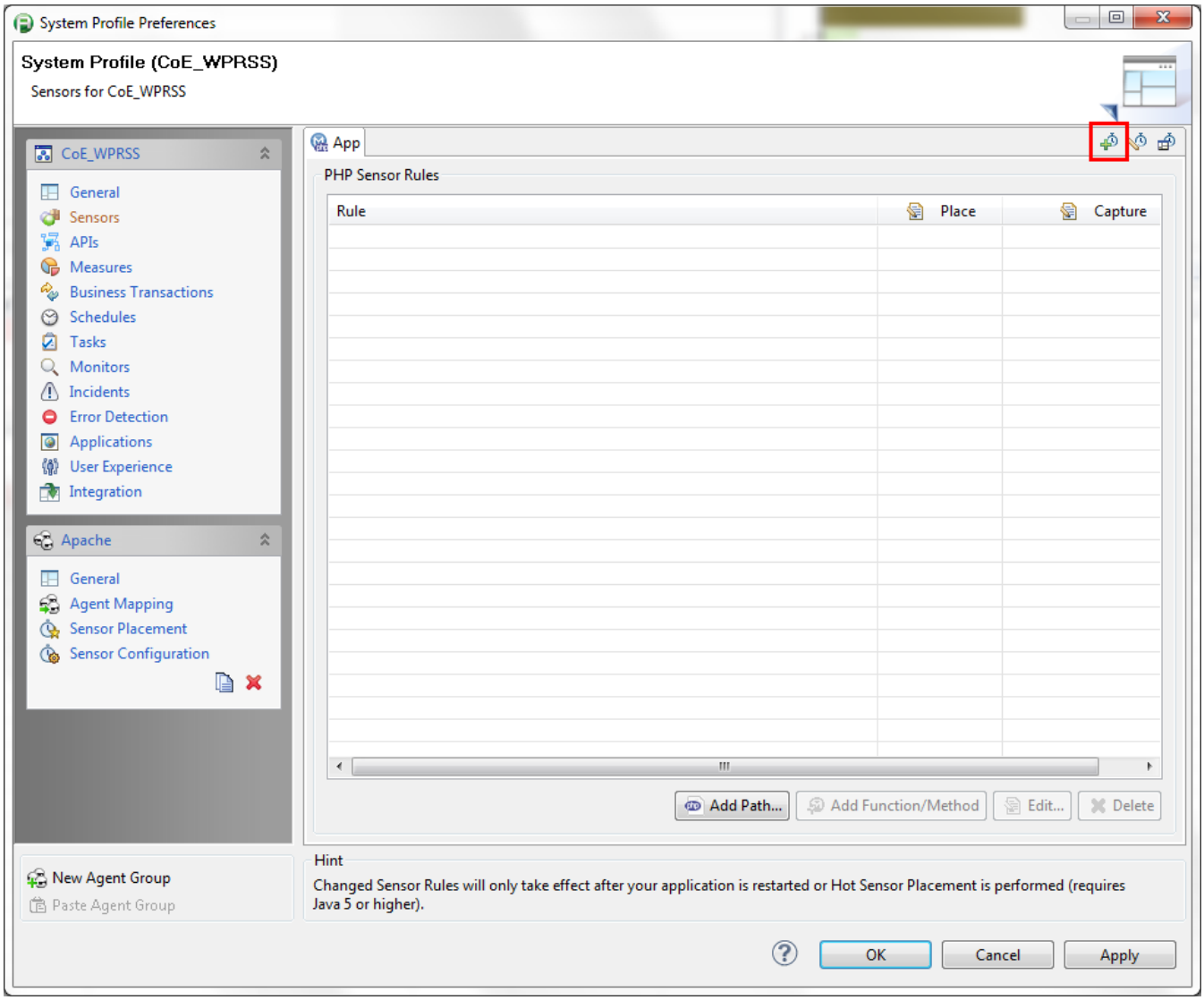
> **Note**
>
> If you leave the pattern field empty, it will resolve all paths, which should be avoided, because it may result in substantial overhead.

d. Click **OK** to save the settings. By default, the sensor is automatically placed. You can clear the **Place** option to skip sensor placement (not activate it).

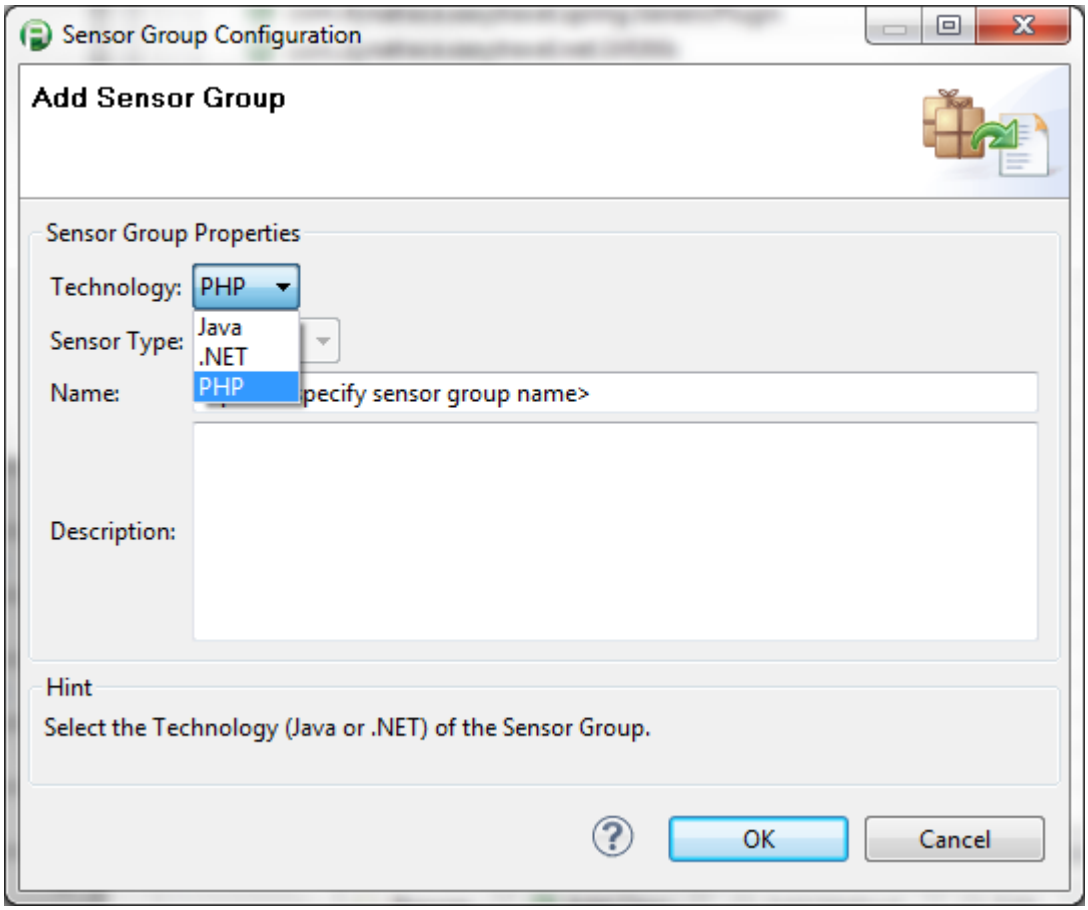
4. Add a new sensor group or use pre-defined **App** sensor.

a. Go to the **Sensors** section for your application.

b. To add new sensor group, click the **Add Sensor Group** icon. This opens the **Sensor Group Configuration** window.



c. Under **Sensor Group Properties**, choose **PHP** from the **Technology** list. The sensor type automatically sets to PHP.



d. Provide the group name and, optionally, a group description.

e. Click **OK** to save the new sensor group.

3. Make sure that PHP monitoring is enabled. To do this, navigate to **General** settings for your application, and verify if the PHP check box is selected in the **Technology** section.

2. Open the **System Profile Preferences** window for your application. Under **System Profiles**, right-click your application and choose **Edit System Profile**.

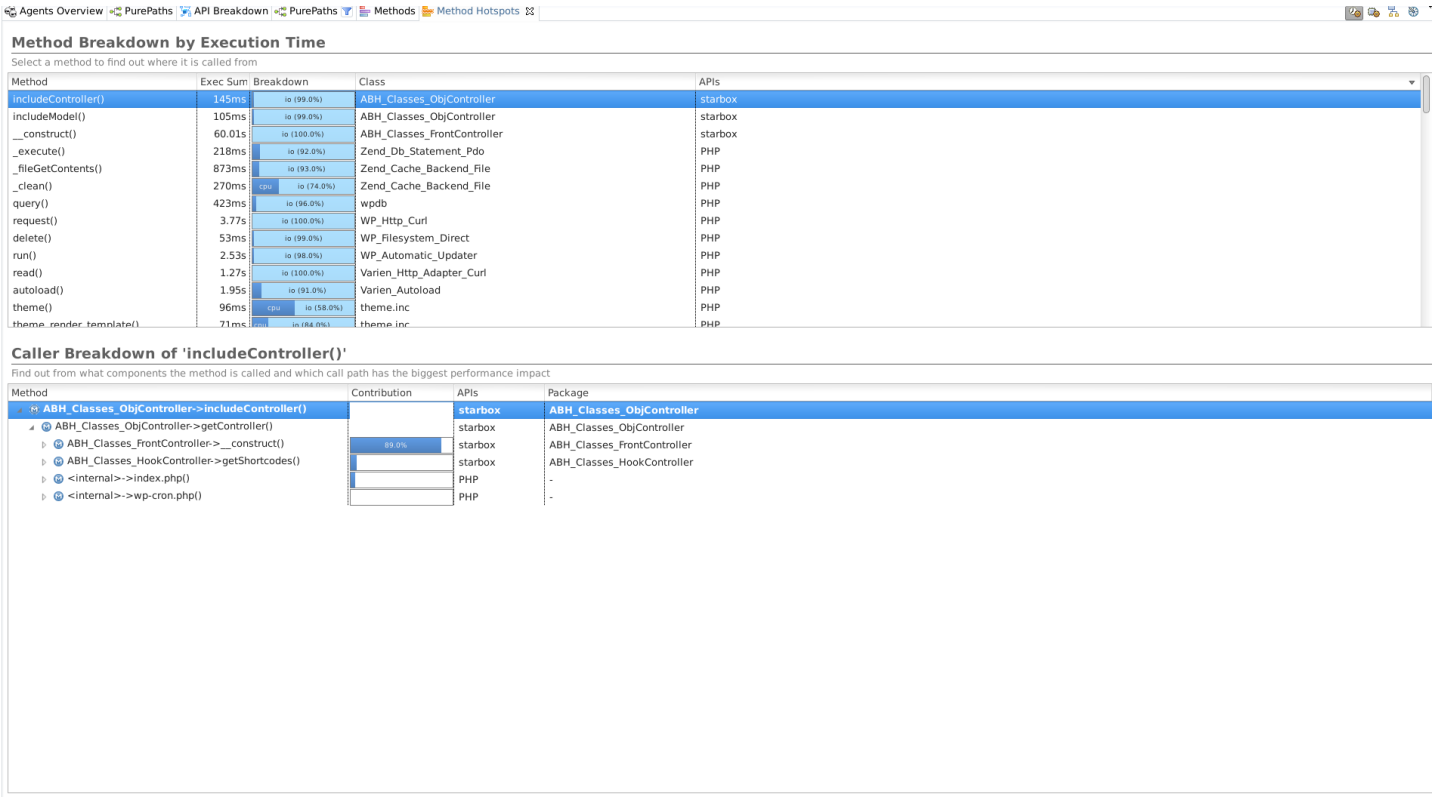
1. Start the AppMon Client

1. Start the AppMon Client.

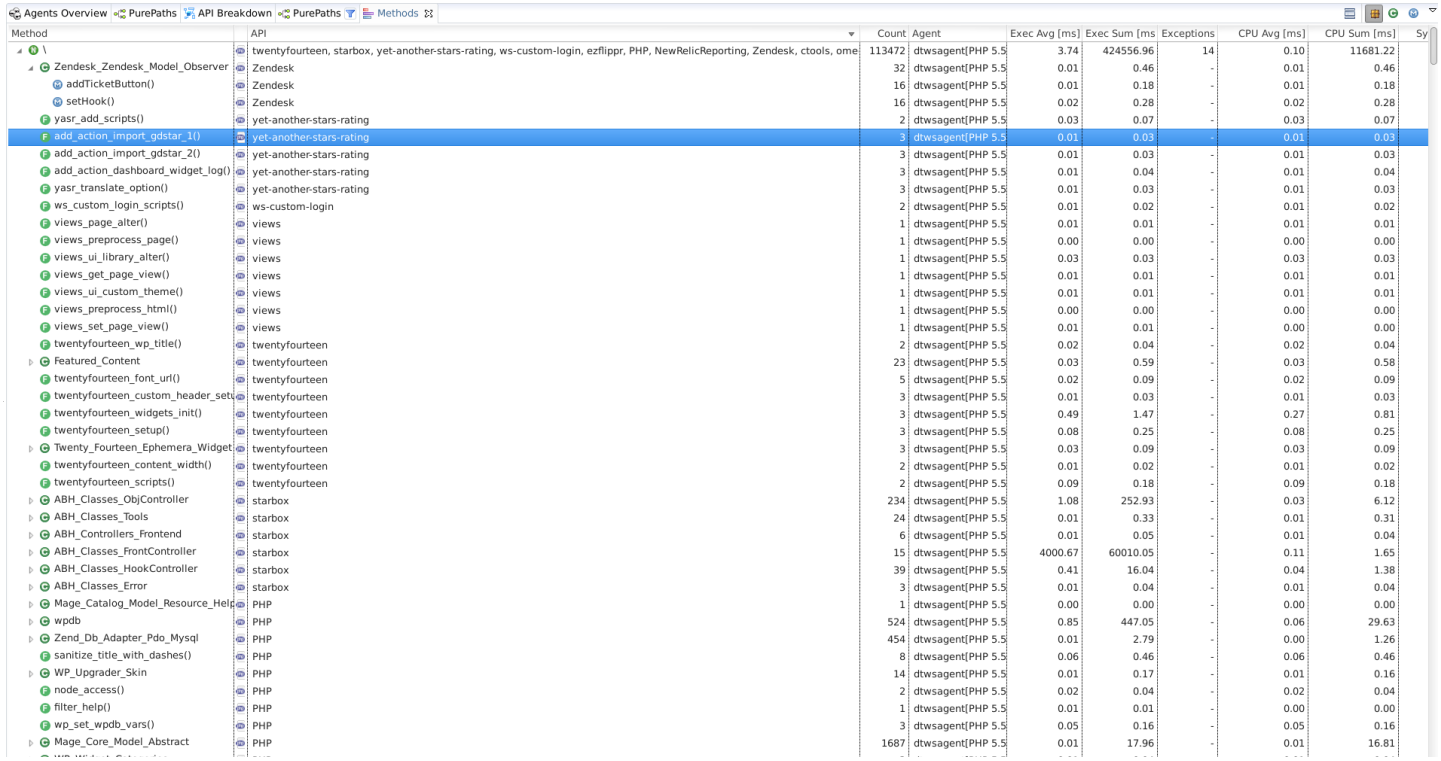
Define a custom sensor

This example shows how to create your own sensor to access specific information in a selected PHP file.

PHP API Breakdown

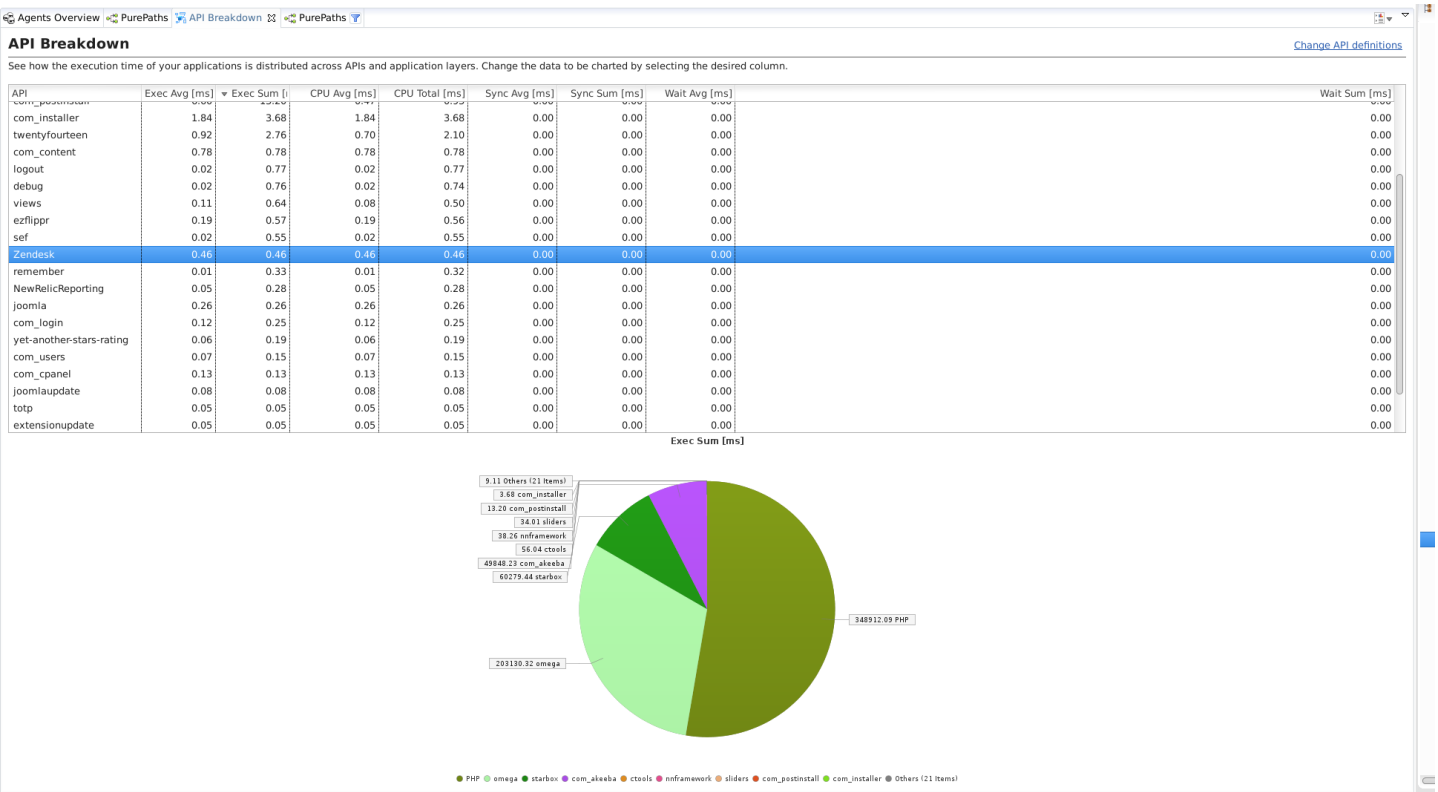


Methods Hotspot dashlet view for PHP api





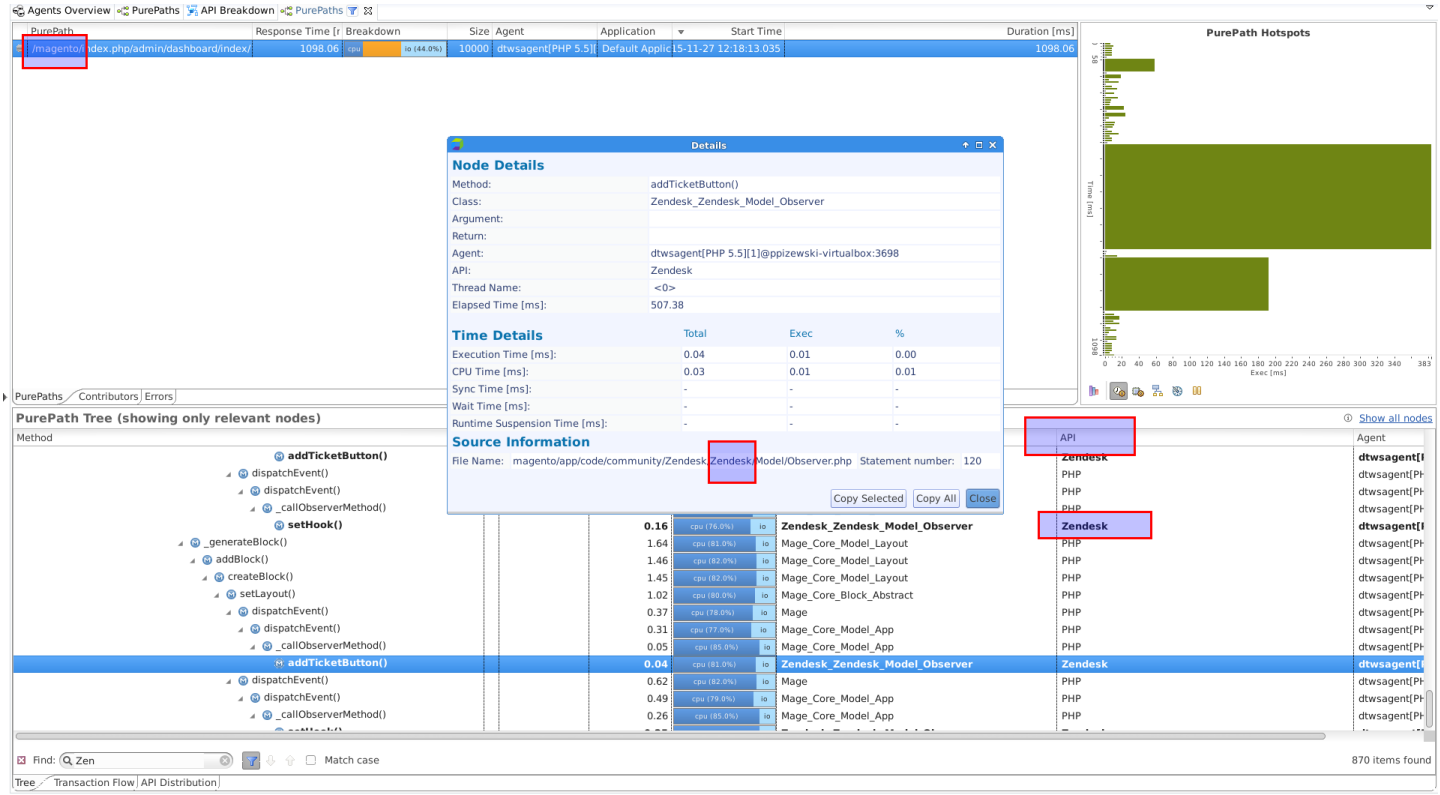
Methods dashlet view mixed supported PHP frameworks



API Breakdown dashlet view for mixed Drupal/Magento/Joomla/wordpress plugins

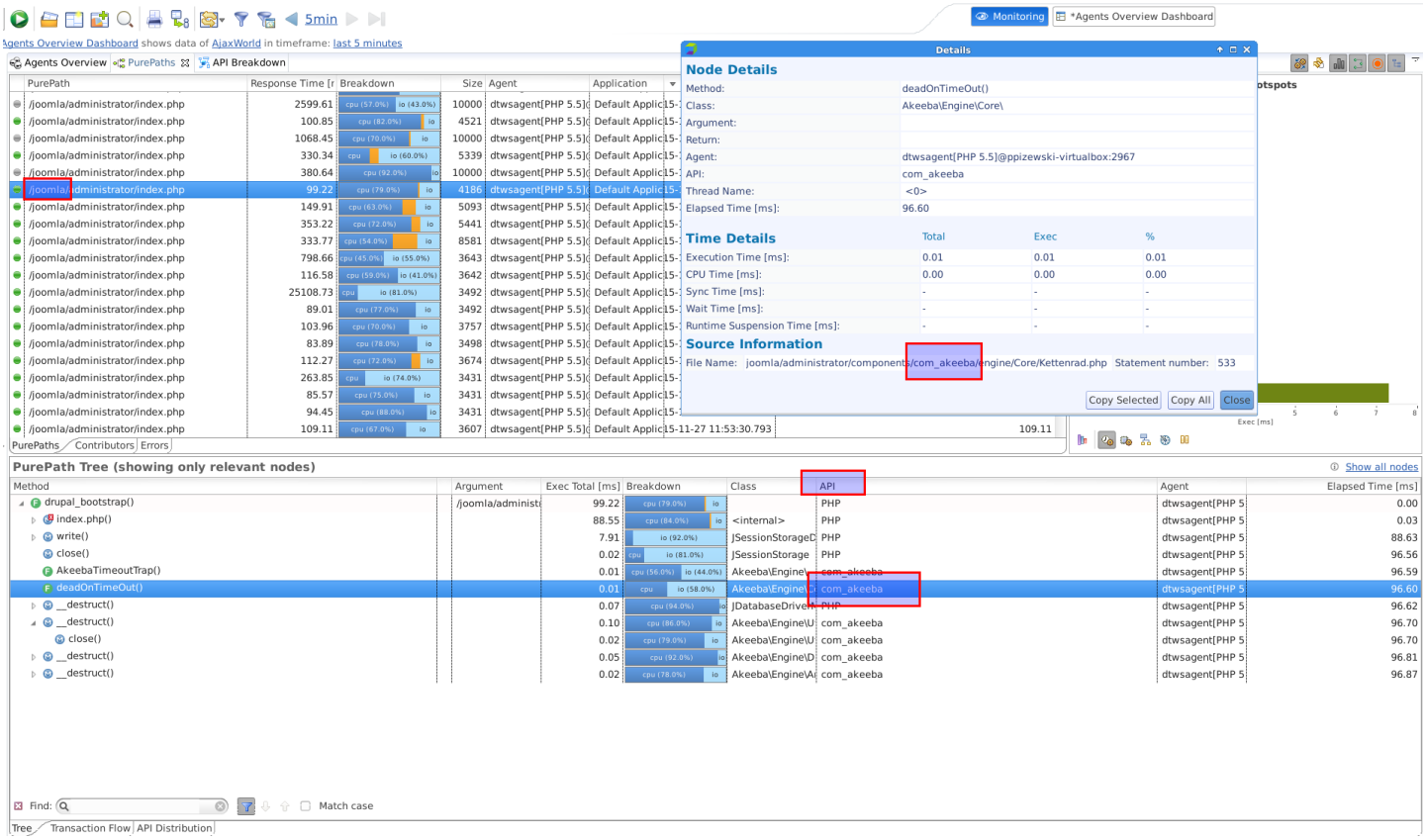
Other examples:

Magento Extension example:



Magento Extension: Patterns: magento/app/code/community/Company/**Extension** /script.php

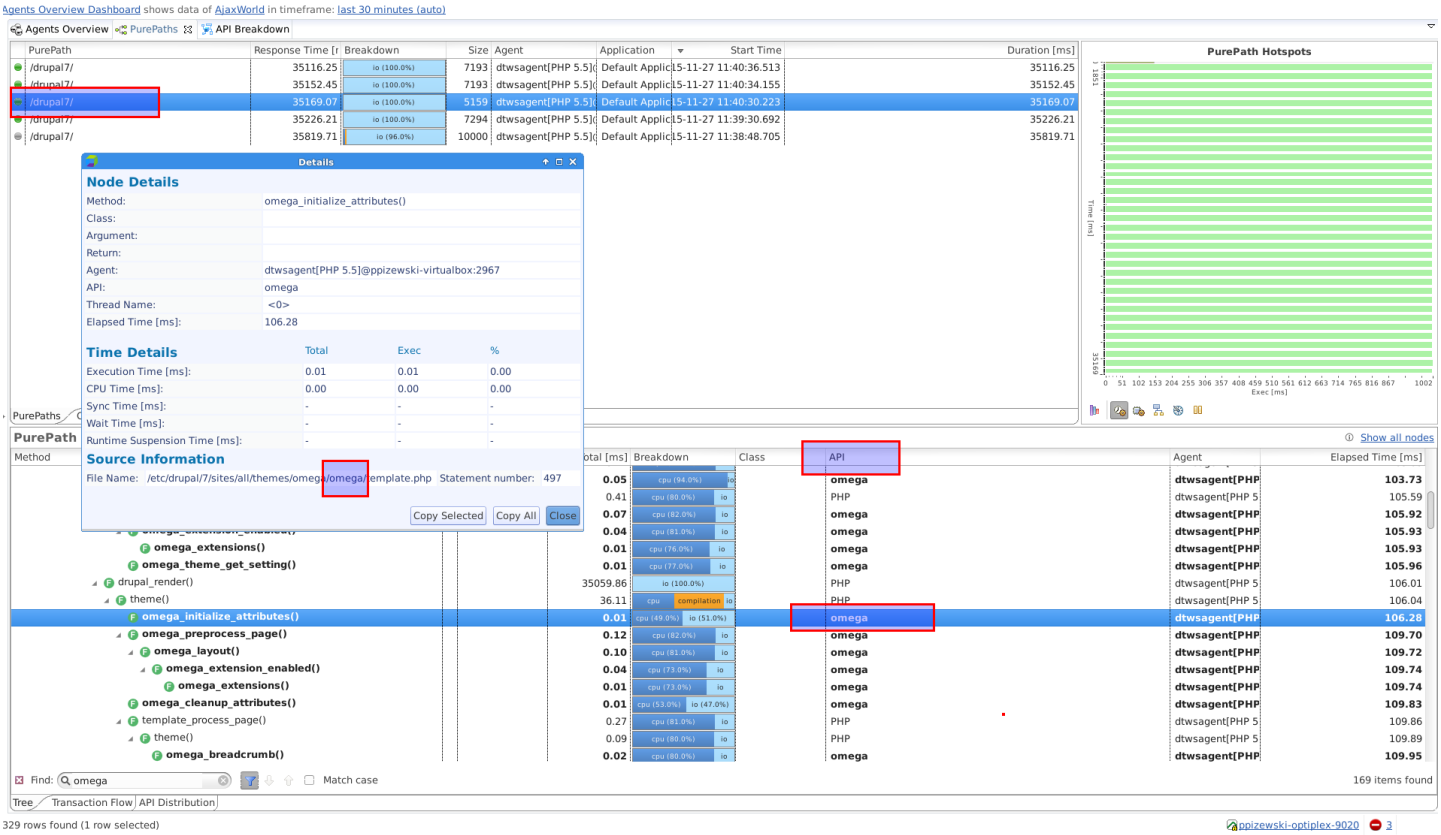
Joomla Component example:



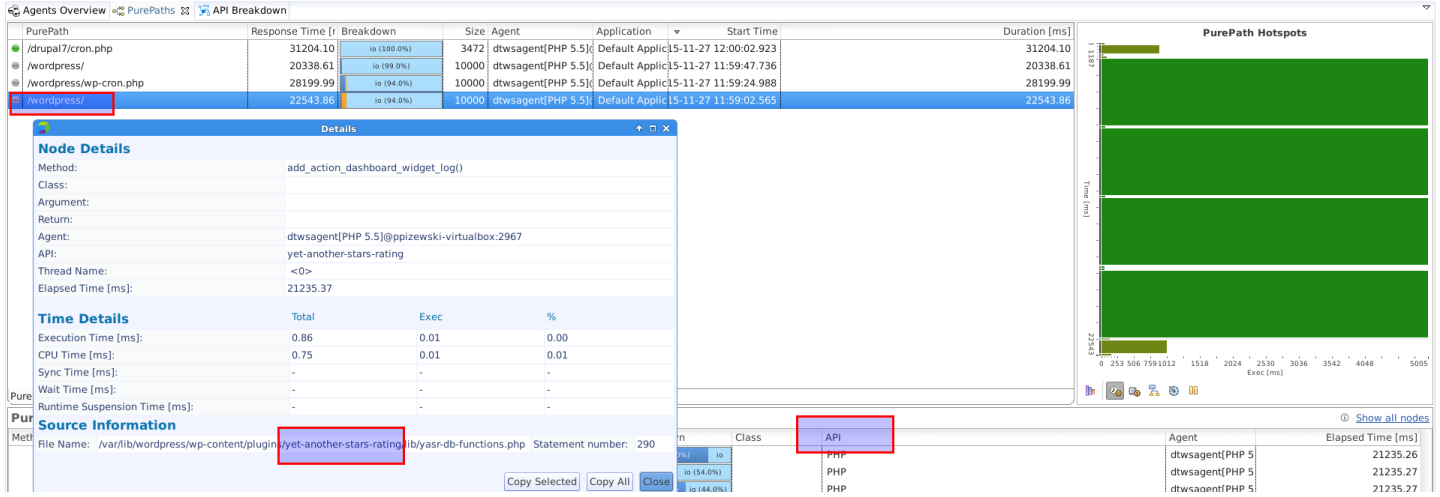
Joomla Component\Plugin\Module: Patterns: joomla/components/**Component**

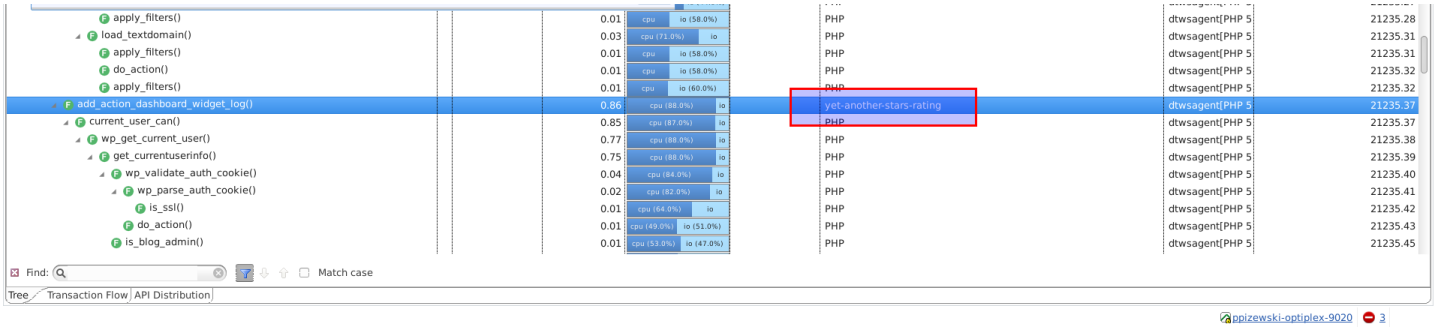
/script.php joomla/administrator/components/**Component**
/script.phpjoomla/plugins/**Plugin**/Company/script.php joomla/modules/**Module** /script.php

Drupal Theme example:



Wordpress Plugin example:





Drupal Module\Theme: Patterns: drupal/**version**/sites/all/modules/**Module** /script.php drupal/**version** /sites/all/themes/**Theme** /script.php

Wordpress Plugin\Theme: Patterns: wordpress/wp-content/plugins/**Plugin** /script.phpwordpress/wp-content/themes/**Theme** /script.php

The extension name is extracted from path of executed **PHP** script using dedicated rules form below for each one framework and type of component:

- **Magento**

- **Wordpress**
- **joomla**
- **Drupal (version 6 to 8)**

The **API Breakdown** functionality provides information about modules, extension and themes for the most popular **PHP** frameworks:

Apache Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Architecture

Installing the agents

Configuring Apache

Deployment and file access permissions

Related pages

Apache Agent

Supported web servers and platforms

Architecture

The Web Server Agent for Apache is provided as a module for the Apache HTTP server (`dtagent.dll` on Windows, `libdtagent.so` on Linux or UNIX), in `<DT_HOME>/agent/lib[64]` with configuration in `<DT_HOME>/agent/conf/dtwsagent.ini` .

Classic Agent

The *NIX Web Server (master) Agent should be separately started as a daemon (at system startup). See Web Server Agent Configuration for details.

The Apache Agent doesn't communicate with the Collector directly, but via the Web Server master Agent.

On Windows, the Web Server Agent must be running as a service when the Apache module is loaded.

For an overview and technical explanations of the AppMon Web Server Agent technology, see Web Server Agent Configuration.

This page also applies to the IBM HTTP Server (IHS), which is based on the Apache HTTP server.

OneAgent

The Apache Agent doesn't communicate with the Collector directly, but via the OneAgent. Web Server Master Agent is not needed. See OneAgent configuration.

Installing the agents

On Windows you can either use the full or the Agent-only installer, do a basic or custom install, interactively or unattended.

On *NIX: On Linux you can use the full installation with `java -jar <packagename>.jar`.
For *NIX flavors there are Agent-only `dynatrace-wsagent-<version>-<platform>.tar` archives.
Un-tar with the Archive Manager or use the command `tar xvf Downloads/dynatrace-wsagent-7.0.0.1289-linux-x86-64.tar -C Downloads cd /opt sudo ~/Downloads/dynatrace-wsagent-7.0.0.1289-linux-x86-64.sh`.

This results in the same directory structure as the `.jar` installation and consequently in the same Apache httpd `LoadModule` line (see below), but only installs the Web Server master Agent and Apache and PHP slave Agents (actual modules / extensions) of the respective platform.
While the `.jar` installer generates `DT_HOME` in the `init.d/dynaTraceWebServerAgent` script according to the installation location, `DT_HOME` must be adapted in the script resulting from the `.tar` if it is not the default `/opt/dynatrace-<majorMinorVersion>` directory.

Configuring Apache

> **Note**
>
> It is not necessary to upgrade the AppMon Agent libraries for the Web Server Agent or choose the correct Apache Agent variants. The Bootstrap Agent takes care of checking for the correct AppMon Agent library when your application starts.

Add the following line to your Apache configuration file `httpd.conf` for XAMPP, located in `<XAMPP_home>/...` or in `/opt/lampp/<etc>`.

```
LoadModule dtagent_module "
<DT_HOME>/agent/lib[64]/<dynaTraceAgentModule>"
```

It is recommended that you put the line at the end of `httpd.conf`, or at least add it as the last `LoadModule`.

For Apache2 installations that do not use a `httpd.conf` file, you will need to create a new module for the agent and then enable it:

1. In the `apache2/mods-available` directory, create a new `dtagent_module.load` file that contains the following content:
   `LoadModule dtagent_module "`
   `<DT_HOME>/agent/lib[64]/<dynaTraceAgentModule>"`
   Substitute your values for the home directory and module file name.
2. Enable the new agent module using the `a2enmod` command. For example, on *NIX systems:
   `sudo a2enmod dtagent_module`
   Once enabled, the agent module will load the next time you restart Apache.

Configuring an Apache node name

You can configure an Apache node name. This name shows in the Agent overview as part of the Agent name (for example: `dtwsagent[Apache 2.2-My Apache Node Name]@myhost:8276`). When starting multiple Apache instances on one host, you can use that node name to distinguish between those instances in the Agent overview. To configure the Apache Node Name, add the following line to the `httpd.conf:`

```
ApacheNodeName "My Apache Node Name"
```

Add this line somewhere after the "`LoadModule dtagent_module ...`" line.

> **Note**
>
> The Apache Node Name can not be used for mapping the agent into a System Profile

Agent injection copy/paste examples Windows

64-bit full AppMon default installation with 64-bit Apache HTTP server on 64-bit Windows:
`LoadModule dtagent_module "C:\Program Files\dynaTrace\dynaTrace 7.0\agent\lib64\dtagent.dll"`

64-bit Agent-only default installation with 64-bit Apache on 64-bit Windows:
`LoadModule dtagent_module "C:\Program Files\dynaTrace\dynaTrace Agent 7.0\agent\lib64\dtagent.dll"`

Agent injection copy/paste examples Linux / Unix

64-bit with 64-bit Apache HTTPd: 64-bit full dynaTrace `.jar` default installation to `/opt` or 64-bit Agent-only default `tar` shell script installation:

```
LoadModule dtagent_module /opt/dynatrace-7.0/agent/lib64/libdtagent.so
```

Deployment and file access permissions

Usually the file access permissions of the files distributed with the AppMon Web Server Agent are automatically set by the installer shell script or MSI. If it is necessary to manually copy installation files, for example to automatically deploy a new version, make sure to retain file access permissions or that the Apache worker processes have the necessary permissions to access the files. The following is a list of files and the necessary permissions.

For Linux and UNIX systems:

Path	Description	Access permissions req

`/agent/conf/dynaTraceWebServerSharedMemory`	The backing shared memory file used for communication across the worker processes.	read/write
`/agent/conf/dtwsagent.ini`	Agent configuration file.	read
`/agent/lib[64]/libdtagent.so`	Agent modules.	read/execute
`/agent/lib[64]/dtwsagent`	Agent service binary.	read/execute
`/agent/downloads`	Downloaded Agent modules.	read/write

For Windows systems:

Path	Description	Access permissions requi
`agent\conf\dynaTraceWebServerSharedMemory`	The backing shared memory file used for communication across the worker processes.	read/write
`agent\conf\dtwsagent.ini`	Agent configuration file.	read
`agent\lib[64]\dtagent.dll`	Agent modules.	read
`agent\downloads`	Downloaded Agent modules.	read/write

Apache Agent troubleshooting

On this page

Apache starts but no Agent connects to the AppMon Collector

ModSecurity blocks the signal to the AppMon monitor (403 error)

Loading Apache Agent on SELinux

Apache starts but no Agent connects to the AppMon Collector

- Windows – On Windows platforms, you must start the master Web Server Agent service automatically registered by the installer.
- Linux and UNIX – On Linux platforms, you must start the master Web Server Agent process.

If the problem persists, check the Agent log files (usually contained in the `<DT_HOME>/log` directory) for output like this:

```
2011-09-05 16:36:46 severe  [native] Failed to launch Webserver Agent
binary /home/labuser/GG/wsagent/agent/lib/dtwsagent: Permission denied
```

This indicates a permission problem with the Web Server Agent (`dtwsagent`). See Deployment and File Access Permissions for further information.

ModSecurity blocks the signal to the AppMon monitor (403 error)

A restrictive configuration of the ModSecurity plugin of the web server can block signals to the AppMon Monitor. This can result in `Access forbidden (403)` error codes for some web content. To prevent this issue, paste the following snippet into the Apache configuration file after the inclusion of the `ModSecurity-config` files.

```
<IfModule security2_module>

    # !!! This rule is to ensure that dynaTrace gets all signals !!!
    <LocationMatch 'dynaTraceMonitor$'>
        SecRuleEngine Off
    </LocationMatch>


    # !!! All the following rules exist to not break the website !!!

    # default is 1500
    SecPcreMatchLimit 2000

    # SQL injection, Detect SQL Comment Sequences
    SecRuleRemoveById 981231

    # SQL injection, Restricted SQL Character Anomaly Detection Alert - Total # of specia
    SecRuleRemoveById 981172

    # SQL injection, String Termination/Statement Ending Injection Testing
    SecRuleRemoveById 981318

    # SQL injection, Detects MySQL comments, conditions and ch(a)r injections
    SecRuleRemoveById 981240

    # HTTP policy,   Restrict which content-types we accept (POST with custom Content-Typ
    SecRuleRemoveById 960010

    # if there are a lot of page problems due to blocked requests, use this
    #SecRuleRemoveByTag "WEB_ATTACK/SQL_INJECTION"
</IfModule>
```

Loading Apache Agent on SELinux

If you are using Linux with SELinux enabled, agents might not start because SELinux is blocking essential operations. In this section we show possible ways to solve it for the Apache HTTPd Agent.

Check if SELinux is involved

If your Apache Agent is not starting, check whether SELinux is enabled, by executing the `sestatus` command:

```
> sestatus

SELinux status:                 enabled
SELinuxfs mount:                /selinux
Current mode:                   enforcing
Mode from config file:          enforcing
Policy version:                 24
Policy from config file:        targeted
```

If **SELinux status** is *enabled* and **Current mode** is *enforcing*, SELinux might block essential operations which are required for the Apache Agent to start.

To check whether SELinux actually denies something important, take a look at the audit log. It is usually located in the `/var/log/audit.log` file. Denied operations are logged as `type=AVC`.

Right after an attempt to start the HTTPd + Agent, execute the `ausearch -m avc --start recent` command to look for recent audit log entries.

If there are denied operations, `ausearch` command returns something like this:

```
time->Fri Feb 24 10:09:13 2017
type=SYSCALL msg=audit(1487927353.999:24): arch=c000003e syscall=2 success=no exit=-13 a0
type=AVC msg=audit(1487927353.999:24): avc:  denied  { write } for  pid=2590 comm="httpd"
....
time->Fri Feb 24 10:09:18 2017
type=SYSCALL msg=audit(1487927358.556:34): arch=c000003e syscall=42 success=no exit=-13 a
type=AVC msg=audit(1487927358.556:34): avc:  denied  { name_connect } for  pid=2590 comm=
```

The first entry at `10:09:13` shows that the process with the id `pid=2590`, and the executable HTTPd `exe="/usr/sbin/httpd"`, which is running in the `unconfined_u:system_r:httpd_t:s0` domain, is denied to write to the `dynaTraceWebServerSharedMemory` file.

The second entry at `10:09:18` shows that the same HTTPd `exe="/usr/sbin/httpd` is denied to connect to the destination TCP port 9998 `tclass=tcp_socket, dest=9998`.

There are several ways to resolve it. Below we cover the following:

- Completely disable SELinux
- Create a policy module with audit2allow
- Create a policy module manually

Disabling SELinux

To disable SELinux, edit the `/etc/selinux/config` file and set the **SELINUX** property as disabled.

```
SELINUX=disabled
```

Reboot your system for the changes to take effect.

Creating a policy module with audit2allow

If you don't want to completely disable SELinux, you can use audit2allow to create a policy module, which grants the required permissions. The tool can build a policy module based on specific contents of the `audit.log` file. The procedure is the following:

1. Switch to **permissive mode**, by executing the following command:

   ```
   setenforce 0
   ```

   Executing of the `getenforce` or `sestatus` command should now indicate that SELinux is running in the permissive mode:

   ```
   > getenforce
   Permissive
   ```

In permissive mode, SELinux does't deny any access, but logs actions, that would have been denied in enforcing mode, as denials.

2. Disable dontaudit rules.
   In certain situations, AVC denials may not be logged. A policy can silence AVC denials by using dontaudit rules. To ensure we see all denials in the `audit.log` file, disable the dontaudit rules with the following command:

   ```
   > semodule -DB
   ```

   We will re-enable them again later.

3. Start httpd + Agent. Start dtwsagent and httpd.

   ```
   > ./dtwsagent
   > service httpd start
   ```

   Since SELinux is in permissive mode, everything should start now, but new SELinux denials will be written to the `audit.log` file.

4. Create a policy module with audit2allow.
   The new entries to the `audit.log` file can be used as input to audit2allow to create the module with the desired permissions:

   ```
   > grep /usr/sbin/httpd | audit2allow -M apacheagent
   ```

5. Install the module:

   ```
   > semodule -i apacheagent
   ```

6. Switch back to enforcing mode and restart HTTPd and Dynatrace Web Server Agent.

   ```
   > setenforce 1
   ```

   Restart HTTPd and Dynatrace Web Server Agent and everything should come up. Recheck audit log and verify that there are not any related denials.

   The dontaudit rules can then be enabled again:

   ```
   > semodule -B
   ```

Creating a policy module manually

You can also create a policy module manually. It is not recommended, unless you know exactly what you are doing. The recommended approach is one of the two described above: either disable SELinux completely or create a policy module with audit2allow.

Anyway, if you need to create a policy manually, the following example might help you. But it's just a demo with the purpose to show the general procedure of manual module creation.

> **Warning**
>
> The instructions below are tested to work on CentOS/RHEL 6 and with Apache HTTPd Agent only and might not be applicable to your system!

The strategy of this demo is the following:

1. Recursively tag all files in the Dynatrace installation folder with a custom type `dynatrace_agent_t`.
2. Permit full access to all files tagged `dynatrace_agent_t` for processes running in the `httpd_t` domain. This ensures that the agent can access all files in the installation folder and below, including the `dynaTraceWebServerSharedMemory` file.
3. Permit processes, running in the `httpd_t` domain, to connect to TCP ports.

To create a policy module, create the two following files in an empty directory:

- `dynatrace_agent.fc` — for the security contexts that should be applied to certain files/directories.
- `dynatrace_agent.te` — for the type enforcement rules.

Content of the `dynatrace_agent.te` file:

```
policy_module(dynatrace_agent_Policy, 0.1.0)

require {
        type httpd_t;
        type port_t;
        class file { open read write create append unlink getattr execute setattr rename
        class dir { open read write create search remove_name add_name getattr setattr rm
        class tcp_socket { name_connect };
}

# declare the new file type
type dynatrace_agent_t;
files_type(dynatrace_agent_t)

# file/directory permissions
attribute dynatrace_agent_t_allowed_app;
allow dynatrace_agent_t_allowed_app dynatrace_agent_t:dir { open read search remove_name
allow dynatrace_agent_t_allowed_app dynatrace_agent_t:file { open read write create apper

# permissions to connect to tcp socket
allow dynatrace_agent_t_allowed_app port_t:tcp_socket { name_connect };

# grant processes of domain httpd_t permissions specified for dynatrace_agent_t_allowed_a
typeattribute httpd_t dynatrace_agent_t_allowed_app;
```

Content of the `dynatrace_agent.fc` file:

```
# recursively label files in /opt/dynatrace-6.5 with our custom type dynatrace_agent_t
/opt/dynatrace-6.5(/.*)?        gen_context(unconfined_u:object_r:dynatrace_agent_t,s0)
```

Note that since we change file labels here, existing rules might stop to fulfill their purpose because they were written for the original labels!

To build the SELinux module, execute:

```
> make -f /usr/share/selinux/devel/Makefile
```

This “compiles” the module to a policy package file `dynatrace_agent.pp`. A policy package can be loaded/unloaded similar to kernel modules:

```
> semodule -i dynatrace_agent.pp
```

To verify that the module was loaded correctly, list the installed modules with the `selinux -l` command:

```
> semodule -l | grep dynatrace
dynatrace_agent_Policy  0.1.0
```

To unistall the dynatrace_agent module, execute the following command:

```
> selinux -r  dynatrace_agent     ... uninstall module
```

Installing a module with the `selinux -i` command is persistent, i.e. the module will be loaded again at a system reboot.

Once again — the above procedure of manual module creation might not be applicable for your system!

IIS Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Classic Agent

OneAgent

Confgure the IIS module

Deployment and file access permissions

Related pages

IIS Agent

IIS Agent troubleshooting

.NET Agent configuration

Web Server Master Agent configuration

Supported web servers and platforms

The following sections explain how to configure the Web Server Agent for IIS. It requires an HTTP Module for IIS 7. You would typically enable this module for one or multiple web sites hosted on IIS. It is, however, possible to configure this module globally for the full IIS instance. This enables the Web Server Agent for all hosted web sites on that IIS instance.

Click here for troubleshooting tips.

Classic Agent

Edit dtwsagent.ini and start Web Server Master Agent

In the `<DT_HOME>/agent/conf/dtwsagent.ini` file, you may want to at least change the Agent name to be more descriptive than the default (`dtwsagent`); and the AppMon Collector the Agent connects to (the default is `localhost`).

> **Note**
>
> In `dtwsagent.ini`, the setting for the AppMon Collector is named `Server`, because in earlier releases the Collector was embedded in the AppMon Server.

The IIS Agent doesn't communicate with the Collector directly, the Web Server master Agent service talks to the configured Collector (`Server`) on one side and the `dtagent.dll` loaded into IIS on the other. See Web Server Agent Configuration for more information.

You can start the Web Server Agent now with `sc start "dynaTrace Web Server Agent 7.0"` on the command line, or in the Windows **Services** window.

OneAgent

The IIS Agent doesn't communicate with the Collector directly, but via the OneAgent. Web Server Master Agent is not needed. See OneAgent configuration.

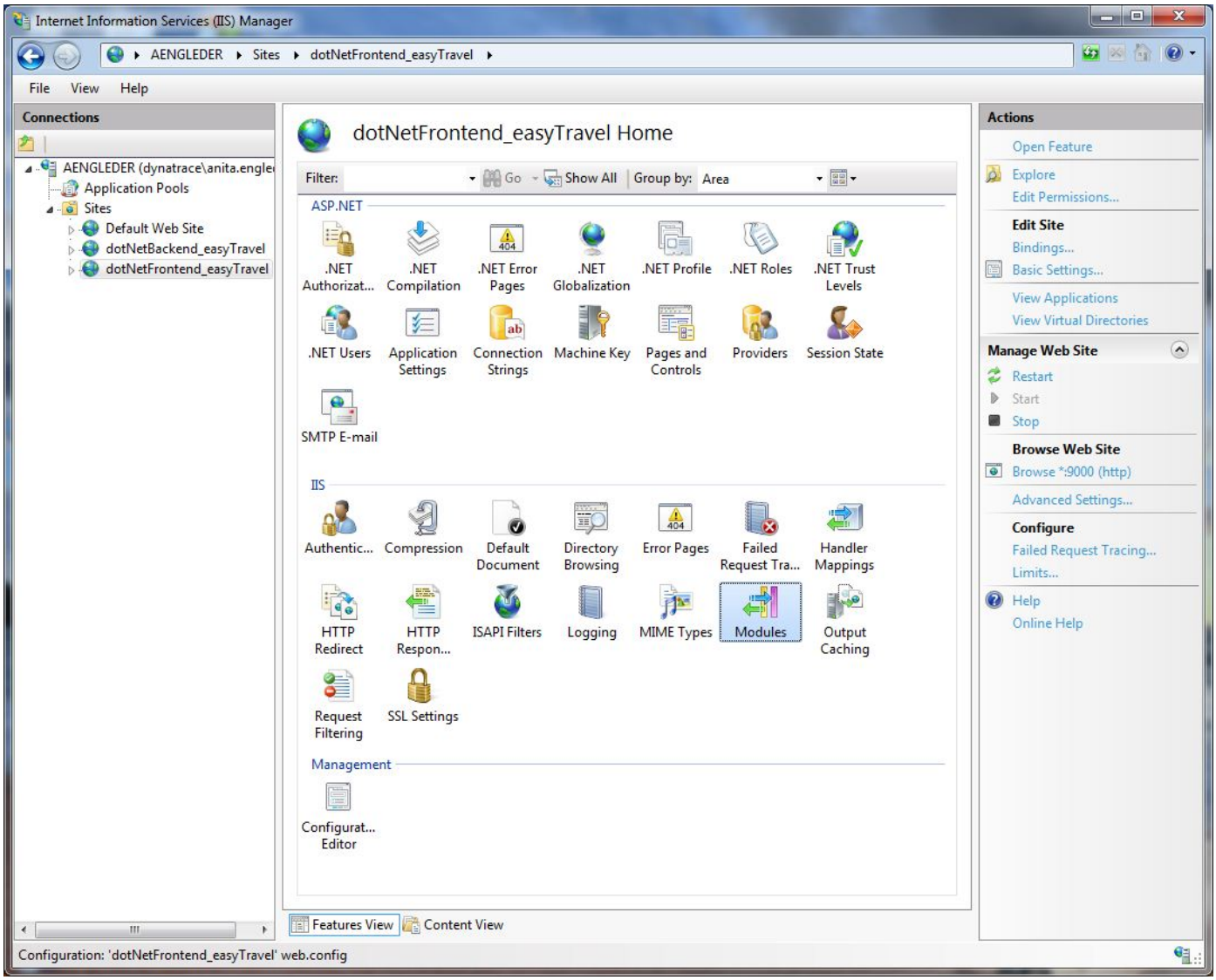
Confgure the IIS module

Configure IIS 7.x or later

The Web Server Agent for IIS is a module. It is registered with IIS automatically during installation. To enable it for your web sites, you have to tell IIS to load the module:

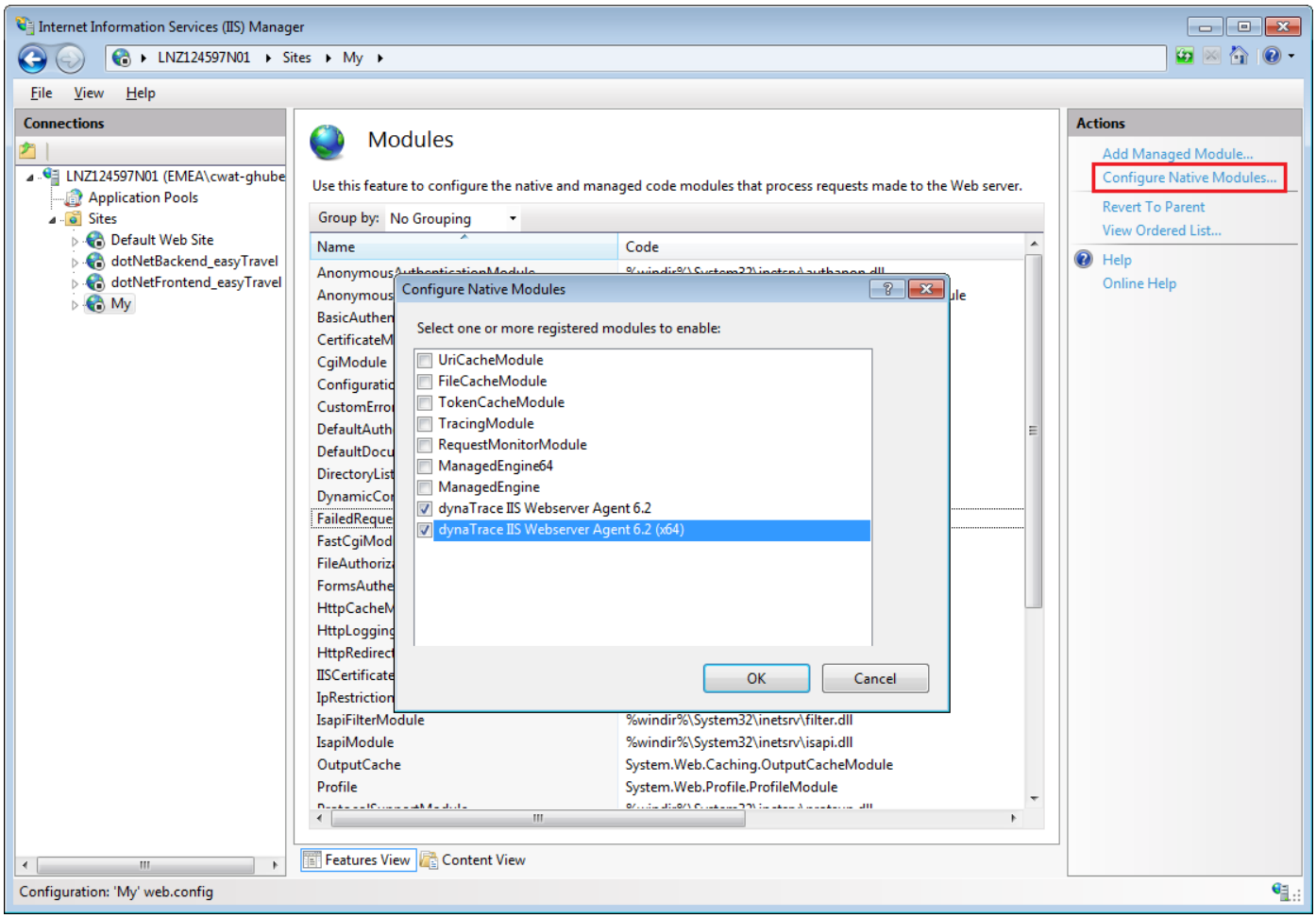
1. Open Internet Information Services (IIS) Manager Control Panel from the GUI or by running the command `inetmgr.exe`.

2. Select the site for which you want to enable the Agent in the left tree and double-click **Modules**.



3. From the **Actions** menu, select **Configure Native Modules**.

4. Select the Web Server Agents called **dynaTrace IIS7 Webserver Agent <version>**. You can safely select both variants (x86 and x64). IIS will only load the appropriate module once. You cannot, however, register two or more modules of the same ABI (32-bit or 64-bit) at the same time, because then IIS would load them all and then they would interfere with each other.



5. Click **OK**. After installation, you may have to restart IIS by doing one of the following:

- Run the command `iisreset` as administrator.
- Open Internet Information Services (IIS) Manager, right-click the server, select **Stop**, then select **Start**.

Configure IIS express

See How to use AppMon with IIS Express.

Deployment and file access permissions

The installer typically sets access permissions for files distributed with the Web Server. If you must manually copy installation files, for example to automatically deploy a new version, make sure you retain file access permissions or that the IIS worker processes have the necessary permissions to access the files. The following is a list of files and the necessary permissions.

Path	Description	Access permissions requir
`agent\conf\dynaTraceWebServerSharedMemory`	The backing shared memory file used for communication across the worker processes.	read/write
`agent\conf\dtwsagent.ini`	Agent configuration file.	read
`agent\lib[64]\dtagent.dll`	Agent modules.	read
`agent\downloads`	Downloaded agent modules.	read/write

IIS Agent troubleshooting

On this page

The native modules are not listed in the Modules Dialog box

Simultaneous IIS restart scenario

The native modules are not listed in the Modules Dialog box

Normally, after Agent installation, the 32 and 64 bit Web Server Agent modules of the respective AppMon major / minor version (for example, 7.0) should be registered and be selectable in the list of native modules. If those modules are deleted or unregistered, you can register them manually:

1. Open Internet Information Services (IIS) Manager by selecting **Control Panels > Administrative Tools** or by selecting **Run** from the Windows **Start** menu and entering `inetmgr.exe`.
2. Select the IIS server (not a specific site) in the left pane and double-click **Modules**.
3. From the **Actions** menu top right, select **Configure Native Modules**.
4. Click **Register**.
5. Register the modules `<DT_HOME>/agent/lib/dtagent.dll` and `<DT_HOME>/agent/lib64/dtagent.dll`. Give them a descriptive name, like the default one: AppMon IIS Web Server Agent 7.0 [(x64)]. You can register both bitnesses of the same AppMon version with your IIS. You can also configure both bitnesses for the site you want to instrument and the bootstrap Agent chooses the right one. However, you should not have registered modules of the same bitness and different AppMon versions at the same time. The configured module is used, but all registered modules be load and initialize.
6. Clear any check boxes left of the newly registered modules, because this would enable these modules for all sites of the selected IIS. Normally you just want to configure the modules for selected sites of this IIS.
7. Click **OK**. You can now configure the modules for the sites.

Simultaneous IIS restart scenario

If you collectively restart your IIS instances, you should not have more than 50 Web Server Agents connected to one Collector, so the load does not overwhelm the Collector.

NGINX Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Prerequisites

Activate the AppMon Agent

Related pages

NGINX Agent

NGINX Agent troubleshooting

Nginx distributions

Supported web servers and platforms

Click here for troubleshooting tips.

Prerequisites

You must have the following to perform this procedure:

- NGINX version 1.4.7 or later.
- AppMon Web Server Agent installer

The NGINX Agent doesn't communicate with the Collector directly, but via the Web Server master Agent, or OneAgent. See Web Server Agent Configuration or OneAgent Configuration for configuration instructions.

The Classic NGINX Agent requires the master Web Server Agent to be running. Make sure to start it before starting NGINX. See the Architecture section on the Web Server Agent Configuration page for details.

Activate the AppMon Agent

As NGINX offers no user-facing API for interfacing with its functionality programmatically, the NGINX Agent needs to be started in a somewhat non-typical way. The following factors are involved:

- A file containing addresses (offsets) of specific functions and data fields within the NGINX binary.
- Manual load of the Agent library into NGINX process memory using system LD_PRELOAD functionality.

Offset file

This data, stored in JSON format, modifies the NGINX process in RAM, and invokes its functionality by direct low-level jump. In most situations, the Agent generates the file at startup. However, certain circumstances require manual file generation and the Agent writes an appropriate script to disk to request the user to execute it from command line.

Loading the agent at NGINX startup

Invoke the Agent in an NGINX startup script. LD_PRELOAD needs the absolute path of the library:

```
LD_PRELOAD=/opt/dynatrace-7.0/agent/lib64/libdtagent.so nginx
```

Make sure to use the Agent library matching the bitness of your NGINX: one from the `lib` directory for 32-bit NGINX or one from `lib64` for 64-bit NGINX.

The agent looks for configuration, including the offset file, in `../conf/` starting from its location on disk. Logs are typically written to `../logs/`, although this can be changed in `dtwsagent.ini` located in the configuration directory.

NGINX Agent troubleshooting

On this page

Troubleshooting startup issues

Third party plugin issues and workarounds

Troubleshooting startup issues

Offset file availability

If the NGINX Agent won't start, it is almost always a question of the offset file. The Agent should be able to handle that automatically in the following cases:

- NGINX installed from distribution package, in APT or RPM (yum)-based distributions.
- Custom-compiled unstripped NGINX binaries.

With an additional manual step, the Agent can support an arbitrary NGINX binary when debug symbols are available.
To check whether your NGINX binary contains debug symbols, simply run the command `file` on it. If the output ends with "Not stripped", the debug symbols are included.

When you attempt to load the Agent with a binary that is not supported out of the box, it prints the required actions to `stdout`.

Agent cannot read offset file

The most common reason for errors of this type is file system permissions. Though NGINX starts using root privileges, workers almost always run as a special non-privileged user, `www-data` or similar. The NGINX Agent configuration directory (the one where it looks for the offset file) must be readable for this user. This also involves all its parent directories that must be executable for this user, otherwise it won't be possible to `chdir` to this directory).

Classic Agent

To use the offset self-generation functionality, the configuration directory must be writable for the non-privileged NGINX user when the file is generated. Later it can be read-only.

The easiest way to verify that the offset file is actually accessible for this user is:

```
$ sudo su www-data $ cat /opt/dynatrace-7.0/agent/conf/dtnginx_offsets.json | wc -l
```

Generating an offset file manually

You can extract the script for offset generation from the agent library. This way you always have the correct version for your Agent.

To dump the script to a file, set an environment variable when using `LD_PRELOAD` to load the Agent library. The actual command (**ls** in the example) run with the Agent preloaded is not important, as it won't even get executed – the Agent terminates after dumping the script:

```
$ DT_DUMP_SCRIPT=1 LD_PRELOAD=/opt/dynatrace-7.0/agent/lib64/libdtnginxagent.so ls
```

The script is written to file `generate_offsets.sh` in the working directory. It can be used in these modes:

- To generate offsets for a specific unstripped NGINX binary or a stripped binary + debug symbol file pair.
- To generate a single offset file covering multiple NGINX binaries or stripped binary + debug symbol file pairs included in specified directory tree.
- To verify that all system requirements for NGINX Agent are met and the specified NGINX binary can be supported.

You can run `generate_offsets.sh --help` for syntax details. Once the JSON file is produced, place it in the AppMon `conf` directory. For example:

```
/opt/dynatrace-7.0/agent/conf/
```

If offset generation fails, the script collects required system information and prints it to `stdout`. Forward it to AppMon Support, if possible accompanied by the NGINX binary itself, either unstripped or together with separate debug symbol file.

Validating offset files

If uncertain whether the given JSON offset file is compatible with their NGINX Agent version, the Agent can be ordered to validate a file. To do this, set an environment variable when using `LD_PRELOAD` to preload the Agent library. The actual command (**ls** in the example) run with the Agent preloaded is not important, as it won't even get executed – the Agent terminates after validating the offset file:

```
$ DT_TEST_OFFSETS=/opt/dynatrace-7.0/agent/conf/dtnginx_offsets.json
LD_PRELOAD=/opt/dynatrace-7.0/agent/lib64//libdtnginxagent.so ls
```

Bypassing NGINX binary hash check

In rare cases, you can alter binaries on disk during system lifetime (RedHat prelink functionality comes to mind). This changes the md5 hash value used for identifying binaries within an offset file and would require regenerating the offset file, which is not always feasible.

To support these cases, NGINX Agent bypasses the hash check and manually identifies the offset file binary. Set an environment variable DT_NGINX_MD5 to the value of md5 hash describing the desired section in the offset file:

```
$ DT_NGINX_MD5=785ff0b8da9b061eb0dc4d0692d66a9d
LD_PRELOAD=/opt/dynatrace-7.0/agent/lib64/libdtagent.so nginx
```

You must also change NGINX's configuration file (in most cases `nginx.conf`) by adding the following line in the main section: `env DT_NGINX_MD5;`

This makes the Agent use offsets specified in the section **785ff0b8da9b061eb0dc4d0692d66a9d** in the JSON file.

Ensure that this functionality is required before using it, as it bypasses several checks built into the Agent.

Upstart configuration

Edit `/etc/init/nginx.conf` to attach NGINX's agent to NGINX with an Upstart startup script. Adapt the example below to your script:

```
description "nginx - small, powerful, scalable web/proxy server"

start on filesystem and static-network-up
stop on runlevel [016]

expect fork
respawn

pre-start script
        [ -x /usr/sbin/nginx ] || { stop; exit 0; }
        /usr/sbin/nginx -q -t -g 'daemon on; master_process on;' || { stop; exit 0; }
end script

script
        set -a
        LD_PRELOAD="/opt/dynatrace-7.0/agent/lib64/libdtagent.so"
        exec /usr/sbin/nginx -g 'daemon on; master_process on;'
end script

pre-stop exec /usr/sbin/nginx -s quit
```

Systemd configuration

For `systemd` configuration `/lib/systemd/system/nginx.service` should look similar to below:

```
[Unit]
Description=A high performance web server and a reverse proxy server
After=network.target
[Service]
Type=forking
PIDFile=/run/nginx.pid
ExecStartPre=/usr/sbin/nginx -t -q -g 'daemon on; master_process on;'
Environment=LD_PRELOAD=/opt/dynatrace-7.0/agent/lib64/libdtagent.so
ExecStart=/usr/sbin/nginx -g 'daemon on; master_process on;'
ExecReload=/usr/sbin/nginx -g 'daemon on; master_process on;' -s reload
ExecStop=-/sbin/start-stop-daemon --quiet --stop --retry QUIT/5 --pidfile /run/nginx.pid
TimeoutStopSec=5
KillMode=mixed
[Install]
WantedBy=multi-user.target
```

Third party plugin issues and workarounds

Google PageSpeed

Google nginx_pagespeed is a module that optimizes page loading times by applying several techniques like image optimization and JavaScript and CSS inlining. The module is highly configurable by several filters and can interfere with AppMon UEM injection in following ways:

- **Blocked page loading**: As PageSpeed intercepts the loading of JavaScript, UEM injection can cause a page not to load because of blocked HTTP requests to `dtagent*.js` and `dynaTraceMonitor`.
- **Solution**: Use the 'pagespeed Disallow' filter as a solution to make it skip all requests related to AppMon UEM:

```
pagespeed Disallow "*dtagent*.js";
pagespeed Disallow "*dynaTraceMonitor*";
```

The default script names can be overridden in the AppMon Client, so make sure that the pattern matches your configuration.

- **Segmentation faults**: Using automatic UEM injection with PageSpeed can lead to segmentation fault errors visible in your servers error log. As a solution disable automatic UEM injection by the instrumented server and use manual injection instead.
  Place `<script src='/<http://<instrumentedWebOrAppServer>[/path]/dtagent_bootstrap.js'></script>` manually.
  See Manual Injection Details on the JavaScript Agent page to learn how.

> **Tip**
>
> The JavaScript Agent bootstrapping is available for manual injection.

Nginx distributions

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related Pages

NGINX Agent configuration

System Requirements

This page lists supported NGINX distributions.

URLs

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx//nginx-core_1.4.6-1ubuntu3.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx//nginx-core_1.4.6-1ubuntu3.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu0.16.04.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu0.16.04.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu0.16.04.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu0.16.04.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu0.16.04.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu0.16.04.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu0.16.04.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu0.16.04.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.0-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu1.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu1.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.1-0ubuntu5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.2-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.2-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.3-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.10.3-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.6_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.6_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.4.6-1ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.6.2-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.6.2-1ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.6.2-5ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.6.2-5ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.6.2-5ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.6.2-5ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.10-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.10-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.11-0ubuntu2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.11-0ubuntu2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.12-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.12-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.13-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.13-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.14-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.14-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.15-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.15-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.3-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.3-1ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.3-1ubuntu1.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.3-1ubuntu1.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.3-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.3-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.9-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.9-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.9-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/main/n/nginx/nginx-core_1.9.9-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx//nginx-extras_1.4.6-1ubuntu3.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx//nginx-extras_1.4.6-1ubuntu3.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx//nginx-full_1.4.6-1ubuntu3.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx//nginx-full_1.4.6-1ubuntu3.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx//nginx-light_1.4.6-1ubuntu3.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx//nginx-light_1.4.6-1ubuntu3.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx//nginx-naxsi_1.4.6-1ubuntu3.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx//nginx-naxsi_1.4.6-1ubuntu3.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.1.19-1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.1.19-1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.1.19-1ubuntu0.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.1.19-1ubuntu0.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.1.19-1ubuntu0.8_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.1.19-1ubuntu0.8_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu0.16.04.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu0.16.04.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu0.16.04.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu0.16.04.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu0.16.04.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu0.16.04.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu0.16.04.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu0.16.04.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.0-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu1.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu1.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.1-0ubuntu5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.2-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.2-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.3-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.10.3-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.6_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.6_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.4.6-1ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.6.2-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.6.2-1ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.6.2-5ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.6.2-5ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.6.2-5ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.6.2-5ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.10-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.10-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.11-0ubuntu2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.11-0ubuntu2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.12-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.12-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.13-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.13-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.14-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.14-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.15-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.15-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.3-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.3-1ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.3-1ubuntu1.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.3-1ubuntu1.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.3-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.3-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.9-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.9-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.9-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-extras_1.9.9-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.1.19-1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.1.19-1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.1.19-1ubuntu0.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.1.19-1ubuntu0.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.1.19-1ubuntu0.8_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.1.19-1ubuntu0.8_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu0.16.04.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu0.16.04.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu0.16.04.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu0.16.04.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu0.16.04.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu0.16.04.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu0.16.04.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu0.16.04.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.0-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu1.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu1.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.1-0ubuntu5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.2-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.2-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.3-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.10.3-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.6_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.6_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.4.6-1ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.6.2-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.6.2-1ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.6.2-5ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.6.2-5ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.6.2-5ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.6.2-5ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.10-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.10-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.11-0ubuntu2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.11-0ubuntu2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.12-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.12-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.13-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.13-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.14-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.14-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.15-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.15-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.3-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.3-1ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.3-1ubuntu1.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.3-1ubuntu1.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.3-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.3-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.9-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.9-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.9-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-full_1.9.9-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.1.19-1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.1.19-1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.1.19-1ubuntu0.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.1.19-1ubuntu0.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.1.19-1ubuntu0.8_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.1.19-1ubuntu0.8_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu0.16.04.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu0.16.04.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu0.16.04.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu0.16.04.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu0.16.04.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu0.16.04.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu0.16.04.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu0.16.04.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.0-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu1.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu1.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.1-0ubuntu5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.2-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.2-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.3-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.10.3-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.6_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.6_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.4.6-1ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.6.2-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.6.2-1ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.6.2-5ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.6.2-5ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.6.2-5ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.6.2-5ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.10-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.10-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.11-0ubuntu2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.11-0ubuntu2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.12-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.12-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.13-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.13-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.14-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.14-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.15-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.15-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.3-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.3-1ubuntu1.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.3-1ubuntu1.2_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.3-1ubuntu1.2_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.3-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.3-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.9-0ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.9-0ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.9-1ubuntu1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-light_1.9.9-1ubuntu1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.1.19-1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.1.19-1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.1.19-1ubuntu0.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.1.19-1ubuntu0.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.1.19-1ubuntu0.8_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.1.19-1ubuntu0.8_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.1_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.4_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.4_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.5_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.5_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.6_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.6_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.7_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3.7_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.4.6-1ubuntu3_i386.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.6.2-1ubuntu1.1_amd64.deb

http://archive.ubuntu.com/ubuntu/pool/universe/n/nginx/nginx-naxsi_1.6.2-1ubuntu1.1_i386.deb

http://archive.webtatic.com/yum/el6-archive/x86_64/nginx16-1.6.0-1.w6.x86_64.rpm

http://archive.webtatic.com/yum/el6-archive/x86_64/nginx16-1.6.0-2.w6.x86_64.rpm

http://archive.webtatic.com/yum/el6-archive/x86_64/nginx16-1.6.1-1.w6.x86_64.rpm

http://archive.webtatic.com/yum/el6-archive/x86_64/nginx16-1.6.2-1.w6.x86_64.rpm

http://archive.webtatic.com/yum/el6-archive/x86_64/nginx16-1.6.3-1.w6.x86_64.rpm

http://archive.webtatic.com/yum/el6-archive/x86_64/nginx18-1.8.0-1.w6.x86_64.rpm

http://archive.webtatic.com/yum/el6-archive/x86_64/nginx18-1.8.1-1.w6.x86_64.rpm

http://archive.webtatic.com/yum/el7-archive/x86_64/RPMS/nginx16-1.6.0-2.w7.x86_64.rpm

http://archive.webtatic.com/yum/el7-archive/x86_64/RPMS/nginx16-1.6.1-1.w7.x86_64.rpm

http://archive.webtatic.com/yum/el7-archive/x86_64/RPMS/nginx16-1.6.2-1.w7.x86_64.rpm

http://archive.webtatic.com/yum/el7-archive/x86_64/RPMS/nginx18-1.8.0-1.w7.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/6/i386/nginx-1.0.15-12.el6.i686.rpm

http://dl.fedoraproject.org/pub/epel/6/i386/nginx-1.10.1-1.el6.i686.rpm

http://dl.fedoraproject.org/pub/epel/6/i386/nginx-1.10.2-1.el6.i686.rpm

http://dl.fedoraproject.org/pub/epel/6/x86_64/nginx-1.0.15-12.el6.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/6/x86_64/nginx-1.10.1-1.el6.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/6/x86_64/nginx-1.10.2-1.el6.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/7/x86_64/n/nginx-1.10.1-1.el7.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/7/x86_64/n/nginx-1.10.2-1.el7.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/7/x86_64/n/nginx-1.6.3-6.el7.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/7/x86_64/n/nginx-1.6.3-7.el7.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/7/x86_64/n/nginx-1.6.3-8.el7.x86_64.rpm

http://dl.fedoraproject.org/pub/epel/7/x86_64/n/nginx-1.6.3-9.el7.x86_64.rpm

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/Debian_8.0/amd64/nginx-custom_1.10.0-1ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/Debian_8.0/amd64/nginx-custom_1.10.0-2_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/Debian_8.0/amd64/nginx-custom_1.10.0-3_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/Debian_8.0/amd64/nginx-custom_1.8.0-8ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/Debian_8.0/amd64/nginx-custom_1.8.0-9ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/Debian_8.0/amd64/nginx-custom_1.8.1-1ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.04/amd64/nginx-custom_1.10.0-1ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.04/amd64/nginx-custom_1.10.0-2_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.04/amd64/nginx-custom_1.10.0-3_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.04/amd64/nginx-custom_1.8.0-8ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.04/amd64/nginx-custom_1.8.0-9ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.04/amd64/nginx-custom_1.8.1-1ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.10/amd64/nginx-custom_1.10.0-1ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.10/amd64/nginx-custom_1.10.0-2_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.10/amd64/nginx-custom_1.10.0-3_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.10/amd64/nginx-custom_1.8.0-8ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.10/amd64/nginx-custom_1.8.0-9ppa_amd64.deb

http://download.opensuse.org/repositories/home:/rtCamp:/EasyEngine/xUbuntu_14.10/amd64/nginx-custom_1.8.1-1ppa_amd64.deb

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.10.0-55.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.10.1-58.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.11.2-62.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.11.4-63.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.11.4-64.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.11.8-68.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.8.0-46.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.8.0-49.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/i586/nginx-1.8.1-52.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.10.0-55.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.10.1-58.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.11.2-62.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.11.4-63.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.11.4-64.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.11.8-68.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.8.0-46.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.8.0-49.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.1/x86_64/nginx-1.8.1-52.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.10.0-55.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.10.1-58.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.11.2-62.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.11.4-63.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.11.4-64.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.11.8-68.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.8.0-1.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.8.0-49.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/i586/nginx-1.8.1-52.1.i586.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.10.0-55.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.10.1-58.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.11.2-62.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.11.4-63.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.11.4-64.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.11.8-68.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.8.0-1.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.8.0-49.1.x86_64.rpm

http://download.opensuse.org/repositories/server:/http/openSUSE_13.2/x86_64/nginx-1.8.1-52.1.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.10.0-1.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.10.1-1.1.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.10.1-1.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.2-1.1.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.2-1.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.4-1.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.4-2.1.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.4-2.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.4-2.4.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.4-2.5.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.8-1.1.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.8-1.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.9-1.1.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.9-1.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.11.9-1.3.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.8.0-1.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.8.0-1.3.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.8.1-1.1.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.8.1-1.2.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/i586/nginx-1.8.1-1.3.i586.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.10.0-1.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.10.1-1.1.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.10.1-1.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.2-1.1.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.2-1.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.4-1.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.4-2.1.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.4-2.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.4-2.4.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.4-2.5.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.8-1.1.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.8-1.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.9-1.1.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.9-1.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.11.9-1.3.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.8.0-1.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.8.0-1.3.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.8.1-1.1.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.8.1-1.2.x86_64.rpm

http://download.opensuse.org/tumbleweed/repo/oss/suse/x86_64/nginx-1.8.1-1.3.x86_64.rpm

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.10.0-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.10.0-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.10.1-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.10.1-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.10.1-2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.10.1-2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.11.2-1~exp1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.11.2-1~exp1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.11.3-1~exp1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.11.3-1~exp1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.11.3-1~exp2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.11.3-1~exp2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.11.3-1~exp2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.11.3-1~exp2+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.2.1-2.2+wheezy3_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.2.1-2.2+wheezy3_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.2.1-2.2+wheezy4_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u1~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u1~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u2~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u2~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.6.2-5+deb8u4_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1~bpo8+2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1~bpo8+2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1~bpo8+3_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1~bpo8+3_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1~bpo8+4_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.10-1~bpo8+4_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.14-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.14-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.3-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.3-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.4-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.4-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.4-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.4-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.4-1+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.4-1+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.6-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.6-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.6-2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.6-2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.6-2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.6-2+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.9-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-extras_1.9.9-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.10.0-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.10.0-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.10.1-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.10.1-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.10.1-2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.10.1-2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.11.2-1~exp1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.11.2-1~exp1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.11.3-1~exp1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.11.3-1~exp1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.11.3-1~exp2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.11.3-1~exp2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.11.3-1~exp2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.11.3-1~exp2+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.2.1-2.2+wheezy3_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.2.1-2.2+wheezy3_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.2.1-2.2+wheezy4_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u1~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u1~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u2~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u2~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.6.2-5+deb8u4_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1~bpo8+2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1~bpo8+2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1~bpo8+3_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1~bpo8+3_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1~bpo8+4_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.10-1~bpo8+4_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.14-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.14-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.3-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.3-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.4-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.4-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.4-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.4-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.4-1+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.4-1+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.6-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.6-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.6-2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.6-2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.6-2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.6-2+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.9-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-full_1.9.9-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.10.0-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.10.0-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.10.1-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.10.1-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.10.1-2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.10.1-2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.11.2-1~exp1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.11.2-1~exp1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.11.3-1~exp1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.11.3-1~exp1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.11.3-1~exp2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.11.3-1~exp2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.11.3-1~exp2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.11.3-1~exp2+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.2.1-2.2+wheezy3_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.2.1-2.2+wheezy3_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5+deb8u1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5+deb8u1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5+deb8u1~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5+deb8u1~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5+deb8u2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5+deb8u2~bpo70+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5+deb8u2~bpo70+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.6.2-5+deb8u2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1~bpo8+2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1~bpo8+2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1~bpo8+3_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1~bpo8+3_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1~bpo8+4_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.10-1~bpo8+4_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.14-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.14-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.3-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.3-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.4-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.4-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.4-1~bpo8+1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.4-1~bpo8+1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.4-1+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.4-1+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.6-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.6-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.6-2_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.6-2_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.6-2+b1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.6-2+b1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.9-1_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-light_1.9.9-1_i386.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-naxsi_1.2.1-2.2+wheezy3_amd64.deb

http://ftp.debian.org/debian/pool/main/n/nginx/nginx-naxsi_1.2.1-2.2+wheezy3_i386.deb

http://nginx.org/packages/centos/6/i386/RPMS/nginx-debug-1.8.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.0-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.0-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.0-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.0-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.0-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.0-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.0-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.0-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.1-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.1-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.1-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.1-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.1-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.1-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.1-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.1-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.2-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.2-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.2-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.2-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.2-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.2-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.2-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.2-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.3-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.3-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.3-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.3-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.3-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.3-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.3-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.10.3-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.0-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.0-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.0-1~squeeze_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.0-1~squeeze_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.0-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.0-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.1-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.1-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.1-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx_1.8.1-1~wheezy_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~jessie_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~jessie_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~squeeze_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~squeeze_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~squeeze_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~squeeze_i386.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~wheezy_amd64.deb

http://nginx.org/packages/debian/pool/nginx/n/nginx/nginx-debug_1.8.0-1~wheezy_i386.deb

http://nginx.org/packages/mainline/centos/5/i386/RPMS/nginx-debug-1.9.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/centos/5/i386/RPMS/nginx-debug-1.9.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/centos/5/x86_64/RPMS/nginx-debug-1.9.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/6/i386/RPMS/nginx-debug-1.9.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/centos/6/i386/RPMS/nginx-debug-1.9.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/centos/6/i386/RPMS/nginx-debug-1.9.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/centos/6/i386/RPMS/nginx-debug-1.9.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/centos/6/x86_64/RPMS/nginx-debug-1.9.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/6/x86_64/RPMS/nginx-debug-1.9.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/6/x86_64/RPMS/nginx-debug-1.9.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.10-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.10-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.4-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.4-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.5-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.5-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.6-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.6-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.7-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.7-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.8-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.8-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.9-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.11.9-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.10-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.10-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.11-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.11-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.12-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.12-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.13-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.13-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.14-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.14-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.15-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.15-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.4-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.5-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.6-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.7-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.8-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.8-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.9-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-1.9.9-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.4-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.4-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.5-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.5-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.6-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.6-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/centos/7/x86_64/RPMS/nginx-debug-1.9.7-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.0-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.0-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.0-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.0-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.0-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.0-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.0-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.0-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.10-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.10-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.10-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.10-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.10-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.10-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.10-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.10-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.1-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.1-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.1-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.1-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.1-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.1-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.1-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.1-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.2-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.2-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.2-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.2-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.2-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.2-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.2-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.2-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.3-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.3-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.3-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.3-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.3-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.3-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.3-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.3-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.4-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.4-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.4-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.4-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.4-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.4-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.4-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.4-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.5-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.5-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.5-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.5-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.5-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.5-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.5-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.5-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.6-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.6-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.6-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.6-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.6-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.6-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.6-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.6-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.7-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.7-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.7-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.7-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.7-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.7-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.7-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.7-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.8-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.8-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.8-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.8-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.8-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.8-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.8-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.8-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.9-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.9-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.9-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.9-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.9-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.9-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.9-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.11.9-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.0-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.0-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.0-1~squeeze_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.0-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.0-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.0-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.10-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.10-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.10-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.10-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.10-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.10-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.10-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.10-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.1-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.1-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.1-1~squeeze_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.1-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.1-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.1-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.11-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.11-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.11-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.11-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.11-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.11-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.11-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.11-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.12-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.12-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.12-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.12-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.12-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.12-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.12-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.12-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.13-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.13-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.13-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.13-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.13-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.13-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.13-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.13-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.14-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.14-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.14-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.14-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.14-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.14-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.14-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.14-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.15-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.15-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.15-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.15-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.15-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.15-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.15-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.15-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.2-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.2-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.2-1~squeeze_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.2-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.2-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.2-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.3-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.3-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.3-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.3-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.4-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.4-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.4-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.4-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.5-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.5-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.5-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.5-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.6-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.6-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.6-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.6-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.7-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.7-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.7-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.7-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.8-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.8-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.8-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.8-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.8-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.8-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.8-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.8-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.9-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.9-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.9-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.9-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.9-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.9-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.9-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx_1.9.9-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~squeeze_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~squeeze_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.0-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~squeeze_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~squeeze_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.1-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~squeeze_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~squeeze_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.2-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.3-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.3-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.3-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.3-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.3-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.3-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.3-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.4-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.4-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.4-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.4-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.4-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.4-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.4-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.4-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.5-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.5-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.5-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.5-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.5-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.5-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.5-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.6-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.6-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.6-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.6-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.6-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.6-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.7-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.7-1~jessie_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.7-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.7-1~jessie_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.7-1~wheezy_amd64.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.7-1~wheezy_i386.deb

http://nginx.org/packages/mainline/debian/pool/nginx/n/nginx/nginx-debug_1.9.7-1~wheezy_i386.deb

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.3.15-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.3.16-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.0-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.10-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.11-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.1-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.12-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.13-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.2-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.3-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.6-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.7-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.8-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.5.9-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.7.0-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.7.1-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.7.2-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.7.3-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.7.4-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.7.5-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/i586/RPMS/nginx-debug-1.7.6-1opensuse12.1.ngx.i586.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.3.15-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.3.16-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.0-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.10-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.11-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.1-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.12-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.13-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.2-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.3-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.5-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.6-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.7-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.8-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.5.9-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.7.0-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.7.1-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.7.2-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.7.3-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.7.4-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.7.5-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/opensuse/12.1/x86_64/RPMS/nginx-debug-1.7.6-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.10-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.10-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.8-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.8-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.9-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.11.9-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.10-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.10-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.11-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.11-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.12-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.12-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.13-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.13-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.14-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.14-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.15-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.15-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.8-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.8-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.9-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-1.9.9-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/i386/RPMS/nginx-debug-1.9.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.10-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.10-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.8-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.8-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.9-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.11.9-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.10-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.10-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.11-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.11-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.12-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.12-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.13-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.13-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.14-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.14-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.15-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.15-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.8-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.8-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.9-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-1.9.9-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/5/x86_64/RPMS/nginx-debug-1.9.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.10-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.10-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.8-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.8-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.9-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.11.9-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.10-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.10-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.11-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.11-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.12-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.12-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.13-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.13-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.14-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.14-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.15-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.15-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.8-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.8-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.9-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-1.9.9-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/i386/RPMS/nginx-debug-1.9.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.10-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.10-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.8-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.8-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.9-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.11.9-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.10-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.10-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.11-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.11-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.12-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.12-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.13-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.13-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.14-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.14-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.15-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.15-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.8-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.8-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.9-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-1.9.9-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/rhel/6/x86_64/RPMS/nginx-debug-1.9.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.10-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.10-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.1-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.1-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.2-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.2-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.3-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.3-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.4-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.4-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.5-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.5-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.6-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.6-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.7-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.7-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.8-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.8-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.9-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.11.9-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.10-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.10-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.1-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.11-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.11-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.12-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.12-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.13-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.13-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.14-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.14-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.15-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.15-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.2-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.3-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.4-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.5-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.6-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.7-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.8-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.8-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.9-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-1.9.9-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.1-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.1-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.2-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.3-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.3-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.4-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.4-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.5-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.5-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.6-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.6-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.7-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/sles/12/x86_64/RPMS/nginx-debug-1.9.7-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.0-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.10-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.1-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.2-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.3-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.4-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.5-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.6-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.7-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.8-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~yakkety_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.11.9-1~yakkety_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.0-1~lucid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.0-1~lucid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.0-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.0-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.0-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.0-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.0-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.0-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.10-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.1-1~lucid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.1-1~lucid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.1-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.1-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.1-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.1-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.1-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.1-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.11-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.12-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.13-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.14-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~xenial_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.15-1~xenial_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.2-1~lucid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.2-1~lucid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.2-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.2-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.2-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.2-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.2-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.2-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.3-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.3-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.3-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.3-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.3-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.3-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.3-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.3-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.4-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.4-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.4-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.4-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.4-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.4-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.5-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.5-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.5-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.5-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.5-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.5-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.6-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.6-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.6-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.6-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.6-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.6-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.6-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.6-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.7-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.7-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.7-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.7-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.7-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.7-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.7-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.7-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.8-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx_1.9.9-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.5.12-1~raring_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.5.12-1~raring_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.5.12-1~raring_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.5.7-1~oneiric_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.5.7-1~oneiric_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.7.2-1~quantal_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.7.2-1~quantal_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.7.3-1~saucy_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.7.3-1~saucy_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.7.3-1~saucy_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.7.3-1~saucy_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~lucid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~lucid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.0-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~lucid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~lucid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~lucid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~lucid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.1-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~lucid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~lucid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.2-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~utopic_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~utopic_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.3-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.4-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.5-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.6-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~precise_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~precise_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~trusty_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~trusty_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~vivid_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~vivid_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~wily_amd64.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~wily_i386.deb

http://nginx.org/packages/mainline/ubuntu/pool/nginx/n/nginx/nginx-debug_1.9.7-1~wily_i386.deb

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.6.0-2.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.6.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.6.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.6.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.10-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.11-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.12-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.4-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.5-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.6-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.7-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.8-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-1.7.9-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.6.0-2.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.6.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.6.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.6.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.10-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.11-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.12-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.4-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.5-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.6-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.7-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.8-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/centos/7/x86_64/nginx-debug-1.7.9-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.5.13-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.6.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.6.0-2.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.6.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.6.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.6.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.10-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.11-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.12-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.8-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-1.7.9-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.2.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.2.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.2.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.2.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.2.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.2.8-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.3.15-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.3.16-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.4.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.4.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.4.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.4.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.4.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.4.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.4.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.4.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.10-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.11-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.12-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.13-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.8-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.5.9-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.6.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.6.0-2.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.6.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.6.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.6.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.10-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.11-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.12-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.4-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.5-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.6-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.7-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.8-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/i386/nginx-debug-1.7.9-1.el5.ngx.i386.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.5.13-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.6.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.6.0-2.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.6.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.6.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.6.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.10-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.11-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.12-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.8-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-1.7.9-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.2.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.2.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.2.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.2.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.2.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.2.8-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.3.15-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.3.16-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.4.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.4.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.4.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.4.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.4.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.4.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.4.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.4.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.10-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.11-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.12-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.13-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.8-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.5.9-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.6.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.6.0-2.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.6.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.6.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.6.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.10-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.11-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.12-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.4-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.5-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.6-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.7-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.8-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/5/x86_64/nginx-debug-1.7.9-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.5.13-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.6.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.6.0-2.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.6.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.6.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.6.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.10-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.11-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.12-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.8-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-1.7.9-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.2.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.2.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.2.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.2.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.2.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.2.8-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.3.15-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.3.16-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.4.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.4.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.4.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.4.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.4.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.4.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.4.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.4.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.10-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.11-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.12-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.13-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.8-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.5.9-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.6.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.6.0-2.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.6.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.6.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.6.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.10-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.11-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.12-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.4-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.5-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.6-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.7-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.8-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/i386/nginx-debug-1.7.9-1.el6.ngx.i386.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.5.13-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.6.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.6.0-2.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.6.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.6.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.6.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.10-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.11-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.12-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.8-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-1.7.9-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.2.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.2.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.2.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.2.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.2.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.2.8-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.3.15-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.3.16-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.4.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.4.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.4.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.4.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.4.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.4.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.4.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.4.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.10-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.11-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.12-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.13-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.8-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.5.9-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.6.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.6.0-2.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.6.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.6.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.6.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.10-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.11-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.12-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.4-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.5-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.6-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.7-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.8-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/6/x86_64/nginx-debug-1.7.9-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/7/x86_64/nginx-debug-1.6.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/7/x86_64/nginx-debug-1.6.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/old/rhel/7/x86_64/nginx-debug-1.7.11-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.0.14-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.0.15-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.2.0-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.2.1-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.2.2-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.2.4-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.2.5-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.2.6-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.2.8-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.4.0-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.4.2-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.4.3-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.4.4-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.4.5-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.4.6-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.4.7-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.6.0-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.6.0-2opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.6.1-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/opensuse/12.1/x86_64/RPMS/nginx-debug-1.6.2-1opensuse12.1.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.10.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.10.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.10.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.10.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.10.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.10.2-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.10.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.10.3-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.8.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-1.8.1-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/i386/RPMS/nginx-debug-1.8.0-1.el5.ngx.i386.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.10.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.10.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.10.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.10.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.10.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.10.2-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.10.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.10.3-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.8.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-1.8.1-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/5/x86_64/RPMS/nginx-debug-1.8.0-1.el5.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.10.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.10.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.10.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.10.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.10.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.10.2-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.10.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.10.3-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.8.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-1.8.1-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-debug-1.8.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/i386/RPMS/nginx-debug-1.8.0-1.el6.ngx.i386.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.2-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.10.3-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.8.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-1.8.1-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/6/x86_64/RPMS/nginx-debug-1.8.0-1.el6.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.10.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.10.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.10.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.10.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.10.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.10.2-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.10.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.10.3-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.8.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-1.8.1-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/rhel/7/x86_64/RPMS/nginx-debug-1.8.0-1.el7.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.10.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.10.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.10.1-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.10.1-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.10.2-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.10.2-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.10.3-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.10.3-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.8.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-1.8.1-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-debug-1.8.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/sles/12/x86_64/RPMS/nginx-debug-1.8.0-1.sles12.ngx.x86_64.rpm

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~wily_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~wily_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~wily_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~wily_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~xenial_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~xenial_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~xenial_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.0-1~xenial_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~wily_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~wily_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~wily_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~wily_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~xenial_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~xenial_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~xenial_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.1-1~xenial_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~xenial_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~xenial_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~xenial_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.2-1~xenial_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~xenial_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~xenial_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~xenial_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~xenial_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~yakkety_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~yakkety_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~yakkety_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.10.3-1~yakkety_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~lucid_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~lucid_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~utopic_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~utopic_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~vivid_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~vivid_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~wily_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.0-1~wily_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.1-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.1-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.1-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.1-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.1-1~vivid_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.1-1~vivid_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.1-1~wily_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx_1.8.1-1~wily_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.4.4-1~oneiric_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.4.4-1~oneiric_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.4.4-1~oneiric_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.4.4-1~oneiric_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.4.7-1~raring_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.4.7-1~raring_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.4.7-1~raring_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.4.7-1~raring_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.6.0-1~quantal_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.6.0-1~quantal_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.6.0-1~quantal_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.6.0-1~saucy_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.6.0-1~saucy_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.6.0-1~saucy_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.6.0-1~saucy_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~lucid_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~lucid_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~lucid_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~precise_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~precise_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~trusty_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~trusty_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~utopic_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~utopic_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~utopic_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~vivid_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~vivid_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~vivid_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~vivid_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~wily_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~wily_amd64.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~wily_i386.deb

http://nginx.org/packages/ubuntu/pool/nginx/n/nginx/nginx-debug_1.8.0-1~wily_i386.deb

http://packages.eu-west-1.amazonaws.com/2015.03/main/20150301f40d/x86_64/Packages/nginx-1.6.2-1.23.amzn1.x86_64.rpm

http://packages.eu-west-1.amazonaws.com/2015.09/main/201509419456/x86_64/Packages/nginx-1.8.0-10.25.amzn1.x86_64.rpm

http://packages.eu-west-1.amazonaws.com/2016.03/main/20160301f525/x86_64/Packages/nginx-1.8.1-1.26.amzn1.x86_64.rpm

http://packages.eu-west-1.amazonaws.com/2016.03/updates/0f1bdc3765e6/x86_64/Packages/nginx-1.8.1-3.27.amzn1.x86_64.rpm

http://packages.eu-west-1.amazonaws.com/2016.09/main/4c53375a3f86/x86_64/Packages/nginx-1.10.1-1.28.amzn1.x86_64.rpm

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.1-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.2-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.3-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.3-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.3-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.3-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.3-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.3-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.3-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.11.3-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.3.12-0ubuntu0ppa2~maverick_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.3.12-0ubuntu0ppa2~maverick_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.5.0-1~ppa1~natty_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.5.0-1~ppa1~natty_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.5.0-1~ppa1~oneiric_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.5.0-1~ppa1~oneiric_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.5.9-1~raring0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.5.9-1~raring0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.7.0-1+quantal0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.7.0-1+quantal0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.7.1-1+saucy0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.7.1-1+saucy0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.10-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.11-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.11-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.11-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.11-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.11-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.11-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.11-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.11-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.12-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.12-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.12-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.12-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.12-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.12-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.12-0+xenial1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.12-0+xenial1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.13-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.13-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.13-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.13-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.13-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.13-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.13-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.13-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.14-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.14-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.14-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.14-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.14-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.14-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.14-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.14-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+utopic0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+utopic0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.3-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.4-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.4-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.4-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.4-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.4-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.4-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.4-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.4-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+precise1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+precise1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+precise2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+precise2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+trusty1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+trusty1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+trusty2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+trusty2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+vivid1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+vivid1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+vivid2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+vivid2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+wily1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+wily1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+wily2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+wily2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+xenial2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.6-1+xenial2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.7-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-extras_1.9.9-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.1-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.2-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.3-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.3-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.3-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.3-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.3-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.3-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.3-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.11.3-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.3.12-0ubuntu0ppa2~maverick_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.3.12-0ubuntu0ppa2~maverick_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.5.0-1~ppa1~natty_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.5.0-1~ppa1~natty_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.5.0-1~ppa1~oneiric_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.5.0-1~ppa1~oneiric_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.5.9-1~raring0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.5.9-1~raring0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.7.0-1+quantal0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.7.0-1+quantal0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.7.1-1+saucy0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.7.1-1+saucy0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.10-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.11-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.11-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.11-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.11-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.11-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.11-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.11-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.11-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.12-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.12-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.12-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.12-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.12-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.12-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.12-0+xenial1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.12-0+xenial1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.13-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.13-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.13-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.13-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.13-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.13-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.13-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.13-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.14-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.14-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.14-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.14-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.14-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.14-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.14-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.14-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+utopic0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+utopic0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.3-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.4-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.4-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.4-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.4-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.4-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.4-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.4-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.4-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+precise1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+precise1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+precise2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+precise2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+trusty1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+trusty1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+trusty2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+trusty2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+vivid1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+vivid1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+vivid2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+vivid2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+wily1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+wily1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+wily2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+wily2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+xenial2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.6-1+xenial2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.7-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-full_1.9.9-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.1-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.2-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.3-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.3-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.3-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.3-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.3-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.3-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.3-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.11.3-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.3.12-0ubuntu0ppa2~maverick_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.3.12-0ubuntu0ppa2~maverick_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.5.0-1~ppa1~natty_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.5.0-1~ppa1~natty_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.5.0-1~ppa1~oneiric_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.5.0-1~ppa1~oneiric_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.5.9-1~raring0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.5.9-1~raring0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.7.0-1+quantal0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.7.0-1+quantal0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.7.1-1+saucy0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.7.1-1+saucy0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.10-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.11-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.11-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.11-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.11-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.11-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.11-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.11-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.11-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.12-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.12-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.12-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.12-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.12-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.12-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.12-0+xenial1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.12-0+xenial1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.13-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.13-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.13-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.13-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.13-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.13-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.13-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.13-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.14-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.14-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.14-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.14-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.14-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.14-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.14-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.14-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+utopic0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+utopic0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.3-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.4-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.4-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.4-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.4-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.4-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.4-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.4-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.4-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+precise1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+precise1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+precise2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+precise2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+trusty1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+trusty1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+trusty2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+trusty2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+vivid1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+vivid1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+vivid2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+vivid2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+wily1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+wily1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+wily2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+wily2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+xenial2_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.6-1+xenial2_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.7-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-light_1.9.9-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.3.12-0ubuntu0ppa2~maverick_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.3.12-0ubuntu0ppa2~maverick_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.5.0-1~ppa1~natty_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.5.0-1~ppa1~natty_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.5.0-1~ppa1~oneiric_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.5.0-1~ppa1~oneiric_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.5.9-1~raring0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.5.9-1~raring0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.0-1+quantal0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.0-1+quantal0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.1-1+saucy0_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.1-1+saucy0_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.5-1+precise1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.5-1+precise1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.5-1+trusty1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.5-1+trusty1_i386.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.5-1+utopic1.1_amd64.deb

http://ppa.launchpad.net/nginx/development/ubuntu/pool/main/n/nginx/nginx-naxsi_1.7.5-1+utopic1.1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.0-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.10.1-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.2.7-0ubuntu0ppa1~maverick_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.2.7-0ubuntu0ppa1~maverick_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.4.1-1ppa1~natty_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.4.1-1ppa1~natty_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.4.1-1ppa1~oneiric_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.4.1-1ppa1~oneiric_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.4.4-4~raring_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.4.4-4~raring_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.6.0-1+quantal0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.6.0-1+quantal0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.6.0-1+saucy0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.6.0-1+saucy0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.0-1+precise1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.0-1+precise1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.0-1+trusty1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.0-1+trusty1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.0-1+utopic1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.0-1+utopic1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.0-1+vivid1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.0-1+vivid1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-extras_1.8.1-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.0-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.10.1-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.2.7-0ubuntu0ppa1~maverick_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.2.7-0ubuntu0ppa1~maverick_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.4.1-1ppa1~natty_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.4.1-1ppa1~natty_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.4.1-1ppa1~oneiric_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.4.1-1ppa1~oneiric_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.4.4-4~raring_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.4.4-4~raring_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.6.0-1+quantal0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.6.0-1+quantal0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.6.0-1+saucy0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.6.0-1+saucy0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.0-1+precise1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.0-1+precise1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.0-1+trusty1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.0-1+trusty1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.0-1+utopic1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.0-1+utopic1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.0-1+vivid1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.0-1+vivid1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-full_1.8.1-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.0-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+yakkety0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.10.1-0+yakkety0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.2.7-0ubuntu0ppa1~maverick_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.2.7-0ubuntu0ppa1~maverick_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.4.1-1ppa1~natty_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.4.1-1ppa1~natty_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.4.1-1ppa1~oneiric_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.4.1-1ppa1~oneiric_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.4.4-4~raring_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.4.4-4~raring_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.6.0-1+quantal0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.6.0-1+quantal0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.6.0-1+saucy0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.6.0-1+saucy0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.0-1+precise1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.0-1+precise1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.0-1+trusty1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.0-1+trusty1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.0-1+utopic1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.0-1+utopic1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.0-1+vivid1_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.0-1+vivid1_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+vivid0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+vivid0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+wily0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+wily0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+xenial0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-light_1.8.1-1+xenial0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.2.7-0ubuntu0ppa1~maverick_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.2.7-0ubuntu0ppa1~maverick_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.4.1-1ppa1~natty_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.4.1-1ppa1~natty_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.4.1-1ppa1~oneiric_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.4.1-1ppa1~oneiric_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.4.4-4~raring_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.4.4-4~raring_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.0-1+quantal0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.0-1+quantal0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.0-1+saucy0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.0-1+saucy0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.2-1+precise0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.2-1+precise0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.2-1+trusty0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.2-1+trusty0_i386.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.2-1+utopic0_amd64.deb

http://ppa.launchpad.net/nginx/stable/ubuntu/pool/main/n/nginx/nginx-naxsi_1.6.2-1+utopic0_i386.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-extras_1.2.1-2.2+wheezy4_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-extras_1.6.2-5+deb8u2_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-extras_1.6.2-5+deb8u3_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-extras_1.6.2-5+deb8u3_i386.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-extras_1.6.2-5+deb8u4_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-full_1.2.1-2.2+wheezy4_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-full_1.6.2-5+deb8u2_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-full_1.6.2-5+deb8u3_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-full_1.6.2-5+deb8u3_i386.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-full_1.6.2-5+deb8u4_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-light_1.2.1-2.2+wheezy4_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-light_1.2.1-2.2+wheezy4_i386.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-light_1.6.2-5+deb8u2_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-light_1.6.2-5+deb8u3_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-light_1.6.2-5+deb8u3_i386.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-light_1.6.2-5+deb8u4_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-light_1.6.2-5+deb8u4_i386.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-naxsi_1.2.1-2.2+wheezy4_amd64.deb

http://security.debian.org/debian-security/pool/updates/main/n/nginx/nginx-naxsi_1.2.1-2.2+wheezy4_i386.deb

http://www.atomicorp.com/channels/atomic/centos/6/x86_64/RPMS/nginx-1.6.0-21.el6.art.x86_64.rpm

http://www.atomicorp.com/channels/atomic/centos/6/x86_64/RPMS/nginx-1.6.1-22.el6.art.x86_64.rpm

http://www.atomicorp.com/channels/atomic/centos/6/x86_64/RPMS/nginx-1.6.2-23.el6.art.x86_64.rpm

http://www.atomicorp.com/channels/atomic/centos/7/x86_64/RPMS/nginx-1.6.1-22.el7.art.x86_64.rpm

http://www.atomicorp.com/channels/atomic/centos/7/x86_64/RPMS/nginx-1.6.2-23.el7.art.x86_64.rpm

https://archives.dotdeb.org/dists/wheezy/nginx/1.6.2/binary-amd64/nginx-extras_1.6.2-1~dotdeb.1_amd64.deb

https://archives.dotdeb.org/dists/wheezy/nginx/1.6.2/binary-amd64/nginx-full_1.6.2-1~dotdeb.1_amd64.deb

https://archives.dotdeb.org/dists/wheezy/nginx/1.6.2/binary-amd64/nginx-light_1.6.2-1~dotdeb.1_amd64.deb

https://archives.dotdeb.org/dists/wheezy/nginx/1.6.2/binary-amd64/nginx-naxsi_1.6.2-1~dotdeb.1_amd64.deb

https://copr-be.cloud.fedoraproject.org/results/openresty/openresty/epel-5-x86_64/00446691-openresty/openresty-1.11.2.1-3.el5.centos.x86_64.rpm

https://copr-be.cloud.fedoraproject.org/results/openresty/openresty/epel-6-x86_64/00446691-openresty/openresty-1.11.2.1-3.el6.x86_64.rpm

https://copr-be.cloud.fedoraproject.org/results/openresty/openresty/epel-7-x86_64/00446691-openresty/openresty-1.11.2.1-3.el7.centos.x86_64.rpm

https://copr-be.cloud.fedoraproject.org/results/openresty/openresty/fedora-24-x86_64/00446691-openresty/openresty-1.11.2.1-3.fc24.x86_64.rpm

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/lucid/main/n/nginx/nginx-extras_1.6.3-8.5.0.7~lucid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/lucid/main/n/nginx/nginx-extras_1.6.3-8.5.0.7~lucid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.0-8.5.0.28~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.0-8.5.0.28~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.1-8.5.0.29~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.1-8.5.0.29~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.1-8.5.0.30~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.1-8.5.0.30~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.2-8.5.1.0~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.2-8.5.1.0~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.2-8.5.1.1~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.2-8.5.1.1~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.2-8.5.1.2~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.10.2-8.5.1.2~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.6.3-8.5.0.7~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.6.3-8.5.0.7~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.6.3-8.5.0.8~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.6.3-8.5.0.8~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.10~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.10~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.11~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.11~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.13~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.13~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.14~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.14~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.15~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.15~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.16~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.16~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.18~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.18~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.19~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.19~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.20~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.20~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.21~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.21~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.22~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.22~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.23~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.23~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.9~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.0-8.5.0.9~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.1-8.5.0.24~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.1-8.5.0.24~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.1-8.5.0.25~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.1-8.5.0.25~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.1-8.5.0.26~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.1-8.5.0.26~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.1-8.5.0.27~precise1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/precise/main/n/nginx/nginx-extras_1.8.1-8.5.0.27~precise1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.0-8.5.0.28~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.0-8.5.0.28~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.1-8.5.0.29~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.1-8.5.0.29~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.1-8.5.0.30~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.1-8.5.0.30~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.2-8.5.1.0~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.2-8.5.1.0~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.2-8.5.1.1~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.2-8.5.1.1~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.2-8.5.1.2~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.10.2-8.5.1.2~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.6.3-8.5.0.7~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.6.3-8.5.0.7~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.6.3-8.5.0.8~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.6.3-8.5.0.8~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.10~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.10~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.11~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.11~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.13~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.13~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.14~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.14~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.15~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.15~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.16~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.16~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.18~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.18~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.19~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.19~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.20~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.20~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.21~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.21~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.22~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.22~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.23~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.23~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.9~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.0-8.5.0.9~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.1-8.5.0.24~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.1-8.5.0.24~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.1-8.5.0.25~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.1-8.5.0.25~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.1-8.5.0.26~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.1-8.5.0.26~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.1-8.5.0.27~trusty1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/trusty/main/n/nginx/nginx-extras_1.8.1-8.5.0.27~trusty1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.6.3-8.5.0.8~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.6.3-8.5.0.8~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.10~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.10~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.11~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.11~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.13~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.13~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.14~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.14~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.15~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.15~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.16~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.16~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.18~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.18~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.19~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.19~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.20~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.20~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.21~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.21~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.22~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.22~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.23~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.23~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.9~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.0-8.5.0.9~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.1-8.5.0.24~vivid1_amd64.deb

https://oss-binaries.phusionpassenger.com/apt/passenger/pool/vivid/main/n/nginx/nginx-extras_1.8.1-8.5.0.24~vivid1_i386.deb

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.10.0-8.p5.0.28.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.10.1-8.p5.0.29.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.10.1-8.p5.0.30.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.6.3-8.p5.0.8.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.10.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.11.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.13.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.14.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.15.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.16.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.17.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.18.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.19.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.20.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.21.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.22.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.23.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.0-8.p5.0.9.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.1-8.p5.0.24.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.1-8.p5.0.25.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.1-8.p5.0.26.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/i386/nginx-1.8.1-8.p5.0.27.el6.i686.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.10.0-8.p5.0.28.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.10.1-8.p5.0.29.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.10.1-8.p5.0.30.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.6.3-8.p5.0.8.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.10.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.11.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.13.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.14.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.15.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.16.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.17.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.18.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.19.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.20.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.21.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.22.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.23.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.0-8.p5.0.9.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.1-8.p5.0.24.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.1-8.p5.0.25.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.1-8.p5.0.26.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/6/x86_64/nginx-1.8.1-8.p5.0.27.el6.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.10.0-8.p5.0.28.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.10.1-8.p5.0.29.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.10.1-8.p5.0.30.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.6.3-8.p5.0.8.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.10.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.11.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.13.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.14.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.15.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.16.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.17.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.18.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.19.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.20.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.21.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.22.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.23.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.0-8.p5.0.9.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.1-8.p5.0.24.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.1-8.p5.0.25.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.1-8.p5.0.26.el7.x86_64.rpm

https://oss-binaries.phusionpassenger.com/yum/passenger/el/7/x86_64/nginx-1.8.1-8.p5.0.27.el7.x86_64.rpm

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.3_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.3_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.3_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B7.3_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.1-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2Bhttp2%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.2-2~dotdeb%2Bhttp2%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.3-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.3-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.3-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.3-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.3-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.10.3-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.0-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-extras_1.8.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.3_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.3_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.3_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B7.3_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.1-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2Bhttp2%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.2-2~dotdeb%2Bhttp2%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.3-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.3-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.3-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.3-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.3-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.10.3-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.0-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-full_1.8.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.3_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.3_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.3_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B7.3_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.1-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2Bhttp2%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.2-2~dotdeb%2Bhttp2%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.3-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.3-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.3-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.3-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.3-1~dotdeb%2Bhttp2%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.10.3-1~dotdeb%2Bhttp2%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.0-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-light_1.8.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B7.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B7.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B8.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.0-1~dotdeb%2B8.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B6.2_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B6.2_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B7.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B7.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B8.1_amd64.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B8.1_i386.deb

https://packages.dotdeb.org/pool/all/n/nginx/nginx-naxsi_1.8.1-1~dotdeb%2B8.1_i386.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.0~omnibus.1-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.0~omnibus.2-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.0~omnibus.3-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.0~omnibus.4-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.0~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.1~omnibus.2-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.1~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.4~omnibus.1-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.4~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.10.5~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.11.0~omnibus.1-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.11.0~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.11.1~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.11.2~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.11.3~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.11.4~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.12.0~omnibus.1-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.12.0~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.12.1~omnibus.1-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.12.1~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.12.2~omnibus.1-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.12.2~omnibus-1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.13.0-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.13.1-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.13.2-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.13.3-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.13.4-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.13.5-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.14.0-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.14.1-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.14.2-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.14.2-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.14.3-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_7.14.3-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.0.0-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.0.1-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.0.1-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.0.2-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.0.2-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.0.3-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.0.4-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.0.5-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.1.0-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.1.0-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.1.1-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.1.1-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.1.2-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.1.3-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.1.4-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.2.0-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.2.1-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.2.2-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.2.3-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.3.0-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.3.1-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.3.2-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.3.3-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.3.3-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.3.4-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.4.0-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.4.1-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.4.2-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.4.3-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.4.4-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.4.5-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.5.0-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.5.0-ce.1_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.5.1-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.5.2-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.5.3-ce.0_amd64.deb

https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/pool/trusty/main/g/gitlab-ce/gitlab-ce_8.5.4-ce.0_amd64.deb

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.11.3-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.11.3-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.11.5-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.11.5-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.13-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.13-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.13-2.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.13-2.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.9-2.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.9-2.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.9-3.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.9-3.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.9-4.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-1.9.9-4.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.11.3-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.11.3-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.13-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.13-1.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.13-2.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.13-2.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.9-2.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.9-2.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.9-3.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.9-3.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.9-4.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/amzn/2017.03/x86_64/RPMS/nginx-plus-extras-1.9.9-4.amzn1.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.11.3-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.11.3-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.11.5-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.11.5-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.7.11-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.7.11-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.7.3-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.7.3-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.7.7-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.7.7-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.7.7-3.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.7.7-4.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.13-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.13-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.13-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.13-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.4-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.4-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.4-3.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.9-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.9-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.9-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.9-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.9-3.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.9-3.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.9-4.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-1.9.9-4.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.12-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.12-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.3-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.3-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.3-4.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.3-5.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.7-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.7-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.7-3.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.5.7-4.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.11-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.11-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.3-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.3-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.7-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.7-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.7-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.7-3.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.7.7-4.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.9.4-1.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.9.4-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.9.4-2.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/i386/RPMS/nginx-plus-debug-1.9.4-3.el5.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.11.3-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.11.3-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.11.5-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.11.5-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.7.11-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.7.11-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.7.3-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.7.3-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.7.7-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.7.7-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.7.7-3.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.7.7-4.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.13-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.13-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.13-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.13-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.4-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.4-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.4-3.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.9-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.9-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.9-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.9-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.9-3.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.9-3.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.9-4.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-1.9.9-4.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.0-2.ngx.el5.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.0-2.ngx.el5.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.12-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.3-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.3-4.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.3-5.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.3-5.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.7-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.7-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.7-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.7-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.7-3.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.5.7-4.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.11-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.11-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.3-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.3-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.3-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.3-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.7-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.7-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.7-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.7-3.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.7.7-4.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.9.4-1.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.9.4-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.9.4-2.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/5/x86_64/RPMS/nginx-plus-debug-1.9.4-3.el5.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.11.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.11.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.11.5-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.11.5-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.7.11-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.7.11-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.7.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.7.3-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.7.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.7.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.7.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.7.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.13-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.13-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.13-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.13-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.9-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.9-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.9-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.9-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.9-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.9-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.9-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-1.9.9-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.12-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.12-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.3-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.3-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.3-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.3-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.3-5.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.5.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.7.11-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.7.11-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.7.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.7.3-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.7.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.7.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.7.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.7.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-debug-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.11.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.11.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.5.12-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.7.11-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.7.11-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.7.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.7.3-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.7.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.7.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.7.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.7.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.13-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.13-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.13-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.13-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.9-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.9-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.9-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.9-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.9-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.9-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.9-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-1.9.9-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.5.12-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.5.12-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.11-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.11-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.3-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.7.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-extras-debug-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-http2-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-http2-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-http2-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-http2-debug-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-http2-debug-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-http2-debug-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.7.11-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.7.11-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.7.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.7.3-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.7.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.7.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.7.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.7.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.5.12-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.5.12-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.5.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.5.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.5.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.5.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.5.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.5.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.11-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.11-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.3-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.3-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.3-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.7-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.7-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.7-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.7.7-4.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.9.4-1.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.9.4-2.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/i386/RPMS/nginx-plus-lua-debug-1.9.4-3.el6.ngx.i386.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.11.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.11.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.11.5-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.11.5-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.7.11-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.7.11-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.7.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.7.3-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.7.7-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.7.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.7.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.7.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.13-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.13-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.13-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.13-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.9-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.9-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.9-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.9-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.9-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.9-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.9-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-1.9.9-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.0-2.ngx.el6.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.12-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.12-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.3-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.3-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.3-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.3-5.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.7-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.5.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.11-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.11-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.3-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.7-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.7.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-debug-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.11.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.11.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.5.12-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.7.11-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.7.11-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.7.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.7.3-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.7.7-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.7.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.7.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.7.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.13-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.13-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.13-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.13-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.9-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.9-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.9-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.9-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.9-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.9-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.9-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-1.9.9-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.5.12-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.11-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.11-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.11-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.3-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-extras-debug-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-http2-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-http2-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-http2-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-http2-debug-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-http2-debug-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-http2-debug-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-http2-debug-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.7.11-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.7.11-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.7.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.7.3-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.7.7-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.7.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.7.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.7.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.5.12-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.5.7-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.5.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.5.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.5.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.5.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.11-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.11-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.11-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.3-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.3-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.3-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.7-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.7-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.7-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.7.7-4.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-1.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-2.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/6/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-3.el6.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.11.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.11.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.11.5-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.11.5-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.7.11-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.7.11-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.7.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.7.3-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.7.7-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.7.7-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.7.7-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.7.7-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.13-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.13-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.13-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.13-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.4-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.9-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.9-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.9-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.9-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.9-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.9-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.9-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-1.9.9-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.11-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.11-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.11-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.3-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.3-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.7-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.7-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.7-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.7-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.7-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.7.7-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.9.4-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-debug-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.11.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.11.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.7.11-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.7.11-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.7.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.7.3-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.7.7-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.7.7-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.7.7-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.7.7-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.13-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.13-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.13-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.13-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.4-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.9-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.9-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.9-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.9-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.9-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.9-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.9-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-1.9.9-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.11-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.11-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.11-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.3-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.7.7-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.9.4-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-extras-debug-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-http2-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-http2-1.9.4-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-http2-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-http2-debug-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-http2-debug-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-http2-debug-1.9.4-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-http2-debug-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.7.11-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.7.11-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.7.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.7.3-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.7.7-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.7.7-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.7.7-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.7.7-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.9.4-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.11-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.11-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.11-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.3-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.3-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.3-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.7-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.7-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.7-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.7.7-4.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-1.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-2.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/centos/7/x86_64/RPMS/nginx-plus-lua-debug-1.9.4-3.el7.ngx.x86_64.rpm

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~squeeze_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~squeeze_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~wheezy_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~jessie_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~jessie_i386.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~wheezy_amd64.deb

https://plus-pkgs.nginx.com/debian/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~wheezy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.3-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~yakkety_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~yakkety_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~yakkety_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.11.5-1~yakkety_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.11-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-3~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.7.7-4~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.13-2~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.4-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-2~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-3~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus_1.9.9-4~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.12-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.3-5~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-1~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-2~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-3~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.5.7-4~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.11-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.3-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-3~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.7.7-4~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-debug_1.9.4-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.11.3-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.11-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-3~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.7.7-4~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-1~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~xenial_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.13-2~xenial_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.4-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-2~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-3~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~wily_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras_1.9.9-4~wily_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.5.12-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.11-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.3-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-3~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.7.7-4~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-extras-debug_1.9.4-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2_1.9.4-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-http2-debug_1.9.4-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.11-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-3~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.7.7-4~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua_1.9.4-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.12-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-3~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~quantal_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~quantal_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~raring_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~raring_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~saucy_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.5.7-4~saucy_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.11-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.3-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-1~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-2~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-3~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~lucid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~lucid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~utopic_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.7.7-4~utopic_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-1~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-2~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~precise_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~precise_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~trusty_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~trusty_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~vivid_amd64.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~vivid_i386.deb

https://plus-pkgs.nginx.com/ubuntu/pool/nginx-plus/n/nginx-plus/nginx-plus-lua-debug_1.9.4-3~vivid_i386.deb

https://pulp.inuits.eu/passenger/rhel/7/x86_64/passenger-5.0.10-8.el7.x86_64.rpm

https://pulp.inuits.eu/passenger/rhel/7/x86_64/passenger-5.0.11-8.el7.x86_64.rpm

https://pulp.inuits.eu/passenger/rhel/7/x86_64/passenger-5.0.8-8.el7.x86_64.rpm

https://pulp.inuits.eu/passenger/rhel/7/x86_64/passenger-5.0.9-8.el7.x86_64.rpm

https://repo.webtatic.com/yum/el7/x86_64/RPMS/nginx16-1.6.3-1.w7.x86_64.rpm

https://repo.webtatic.com/yum/el7/x86_64/RPMS/nginx18-1.8.1-1.w7.x86_64.rpm

Web Server Master Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Supported web servers and platforms

Watch and learn

Managing agents

Note

This page describes configuration of the Web Server master agent, thus relates to Classic Agent only. For information about technology-specific configurations, see one the following pages:

- Apache Agent Configuration.
- IIS Agent Configuration.
- NGINX Agent Configuration.

Note, that these Agents don't communicate with the Collector directly, but via the master Agent.

The Web Server Agent executable `dtwsagent[.exe]` is in `<DT_HOME>/agent/lib[64]`.

On Windows, the Agent is a service configured to auto-start at boot time.

On Linux and UNIX, the Agent should be auto-started as a daemon. The Agent starts as a web server process. See the isMasterAgentServiceInstalled setting below for information on how to configure the Web Server Agent as a daemon. You should copy the `<DT_HOME>/init.d/dynaTraceWebServerAgent` shell script to or reference it (as a link) in `/etc/init.d`. Add the shell script to the system boot in `chkconfig` or `update-rc.d`, depending on your distribution.

Configure the Web Server Agents and Modules by adding settings in `<DT_HOME>/agent/conf/dtwsagent.ini`.

General settings

Setting	Default Value	Description
`Port`	`8001`	The UDP port on which the master Web Server Agent shoul packets of the slave Web Server Agent. Change the port nu default port (8001) is used by a different application. This is which the Agent connects to the AppMon Collector.
`Name`	`dtwsagent`	The name of the Web Server Agent, used for mapping the A System Profile on the AppMon Server.
`Server[:port] or Collector[:port]`	`localhost:9998`	The AppMon Collector to which this Agent should connect. T form `CollectorName[:port]`, for example `[exampleCollector:9997]` `(http://exampleCollector:9998)` This setting because in earlier releases the Collector was embedded in The embedded Collector is still available, but the standalon used in all situations except for testing.
`Loglevel`	`info`	The log level. Valid values are `0` through `8`, or `finest | | config | info | warning | severe |`
`ConsoleLoglevel`	`none`	The console log level. Valid values are `0` through `8`, or `fi fine | config | info | warning | seve` Depending on this setting, log messages write to the conso output) as well as to the Agent log file. For Apache, the log selected level is also in the `error.log` file.
`LogfilePath`	`<DT_HOME>/log`	The path to the directory that holds the log files.
`LogfileSize`	`10 MiB`	The maximum log file size. As soon as a log file exceeds this created and the old one is saved. If no space is left on the stopped until space is available. The maximum log file size 64 KB. Log files are created for the Web Server Agent and for each of the web server worker processes. At any given t different log files exist. Specify the value as `<size> [Ki`
`SharedMemoryFileName`	`<DT_HOME>/agent/conf /dynaTraceWebServerSharedMemory`	The path to the file used for shared memory communicatior specified, the Agent attempts to use the shared memory file `<DT_HOME>/agent/conf/dynaTraceWebServ` This option should only be altered if there are Agent installa directories and it is required that these Agents communicate

`RemoveTag false`	Remove the AppMon request header. The Web Server Age keep track of the PurePaths. If you set the `RemoveTag` v path correlation no longer works. For example, use this set environments, where different AppMon versions are running you want to ensure that they do not interfere with each othe
`NoBootstrap <true|not set=false> not set / commented out`	If set to `true`, the Agent does not download the newest Ag objects from the AppMon Collector, but instead uses the Ag `<dynatrace install dir>/agent/lib[64]`
`usessl=true|false false`	Enables SSL encryption between the Agent and the Collect communication uses the same port as non-encrypted. To e set this parameter as true before launching the Agent.

Compression of injected content and response overhead

If you get too much overhead through compression, switch it off with `CompressInjectedResponse false` in `dtwsagent.ini`. Compression may be handled by a load balancer.

For ASP.NET Applications also have a look at `IISInjectionSpeedup` in `twsagent.ini`. (description also in IIS Settings below) Changing this setting can reduce overhead for compression that you may otherwise see in the IIS.

Setting Default Value	Description
`CompressInjectedResponse true`	Use this setting to handle compression of injected content (responses). If `CompressInjectedResponse` is set to `true`, then every response is compressed depending on the `Accept-Encoding` request header. If it is set to `false`, the injected content is never compressed, regardless of the `Accept-Encoding` request header. Turning this option off might yield better response times and lower CPU load, but the uncompressed content increases network traffic. Possible values are `true` and `false`. You should instrument applications as early as possible. If you have an application server, you should inject the Agent there if possible, and send uncompressed data if your web server or any tier in front (for example, a load balancer) compresses the outgoing content.

IIS settings

Setting Default value	Description

`IISInjectionSpeedup htm:always,html:always,aspx:never`	The JavaScript Agent injection can be made faster for compressed requests by adapting request headers to postpone the compression until after the injection. The downside is a marginally slower performance for requests not yielding an injectable HTML. This can be selectively enabled or disabled per extension or a heuristic can be applied to automatically decide for which extensions the speedup makes sense. Arguments: a comma-separated list of `<extension>:<setting>` `<setting>` being one of `always, never` or `auto`. For unspecified extensions `auto` is the default. `aspx:auto` is not recommended as it's known to cause problems with script resources. `aspx:always` is known to be incompatible with certain SharePoint components. If you don't use SharePoint you might want to consider setting `aspx:always` for a better performance of other aspx applications. Example when you do not use SharePoint: `IISInjectionSpeedup aspx: always`

*NIX settings

Setting Default value	Description
`isMasterAgentServiceInstalled false`	**On Linux or UNIX platforms**, control how the master Agent is started. The default is `false` or not set: The master Web Server Agent is started as a process by the web server. It is recommended to start the Web Server Agent as a daemon. Set `isMasterAgentServiceInstalled` to `true` in `<DT_HOME>/agent/conf/dtwsagent.ini`. It signals that the master Agent was started and does not need to be started by the web server.
`SharedMemoryUnixFilePermissions 666`	On Unix platforms, use this directive to change the default permissions of the shared memory file. Use an octal value to specify the permissions (see man chmod(1)). The default is 666, which means read and write rights for the owner, the group, and everyone. This setting has no relevance on Windows platforms.

Restart master agents

You can restart the Web Server master Agents from the Client's **Agent Overview / Status dashboard > Agent** context menu.

Multiple instrumented web server instances on one machine

If you instrument several web server instances on the same machine, the slave Agents of the web server instances connect to the same master Agent service (Windows) or process/daemon (Linux or UNIX), stemming from the same directory structure and configuration in `<DT_HOME>/agent/conf/dtwsagent.ini`. They share the UDP and `agent/conf/dynaTraceWebServerSharedMemory`.
There are two variants on Linux/UNIX:

- Deprecated legacy: The master is started by the slave. The *n*\th slave tries to start another master with the same configuration and reverts to the first master after 30 seconds when it sees that the UDP port is already taken.
- Preferred: The master Agent is started as a daemon and the slaves connect on its UDP.

If your web server instances run different web sites and you want to have data for each web site in a separate System Profile:

1. For every additional, differently named master Web Server Agent, copy the AppMon Web Server Agent installation directory `<DT_HOME>/agent` to `<DT_HOME><InstanceName> /agent`.
2. Make sure that the read/write permissions are the same for each folder after this step.
3. Configure a unique UDP port (very first item in the `.ini`) and Agent name (and possibly a port on that the Collector listens) for each master Web Server Agent instance in `<DT_HOME>InstanceName/agent/conf/dtwsagent.ini`.

Notes

- How an Agent maps to an Agent Group in a System Profile defaults to a “Starts” comparison. The default Agent name in `dtwsagent.ini` is `dtwsagent`. If you call the second master Web Server Agent `dtwsagent2` and leave the comparison at “Starts”, the Agent maps into the same group.
- The correct syntax for the Collector name and port is Server <CollectorName>[:port]. Supply a name, even if it´s `localhost` and only the port is different.

1. **Apache or IHS**: In the `httpd.conf` file on Apache, load the correct module from the corresponding installation directory.
2. **IIS**: Configure the native module for the web application, choosing the `<DT_HOME>InstanceName/agent/lib[64]/dtagent.dll` file from the corresponding installation directory.

3. **Windows**: Install the additional Web Server Agent as an auto-starting service with `<DT_HOME>InstanceName/agent/lib/dtwsagent -service install -instance InstanceName`
4. **Linux or UNIX**:
   - **Deprecated legacy**: The slave tries to start a master with the Agent name and UDP from its `agent/conf/dtwsagent.ini`, and reverts to the master with the specific UDP after 30 seconds.
   - **Preferred**: Install the additional Web Server Agent as an auto-starting daemon:
     `cp <DT_HOME>/init.d/dynaTraceWebServerAgent /etc/init.d/dynaTraceWebServerAgentInstanceName`
     In the shell script, adapt `<DT_HOME>` to point to the instance directory structure and add it to boot with `chkconfig --add dynaTraceWebServerAgentInstanceName`. Change the `<DT_PIDFILE>` variable in each of the copied shell scripts to point to different locations for the PID file so that one shell script does not overwrite the PID file of the other shell script.

Node.js Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Agent installation, update

Upgrade

Licensing

Related pages

Node.js Agent

Node.js Agent troubleshooting

Supported Node.js environments

Click here for troubleshooting tips.

Agent installation, update

Defining Agent group/tier

After installing AppMon, be sure to define an Agent Group/tier for Node.js. To do so:

1. Open the the **Application Environment Configuration** wizard, by clicking the **Configure** button on the initial Welcome screen, or later, by clicking the **Configure Environment** item in the gear menu in the top right corner of the **Monitoring** dashboard.
2. At the step 1 of the wizard, select **Other**.
3. At the step 2 of the wizard, click **Add** under **Java / .Net / Native**.

4. Select **Node.js** in the **Application Server** category in the next step. The wizard lets you select the location and bitness of your Node.js installation, gives you an alternative download, the commands to un-tar or un-zip it and the resulting Agent connection string that you need to put at the beginning of your main JS file.

Integration

To integrate the Node.js Agent with your installation:

1. Download the **Node.js Agent** archive for your operating system. The archive contains both 32 and 64 bit versions of the Agent.

2. **Linux**

   In a command prompt, change to the directory where the AppMon folder is located and unpack the `.tgz`, For example:

   ```
   cd /opt
   tar -xzf ~/Downloads/dynatrace-one-agent-nodejs-<version>-linux-x86.tgz
   ```

   This puts the `onenodeloader.js` file from the `.tgz` into the respective AppMon installation locations (`<DT_HOME>/agent/bin/any/`).

   Windows

   Extract the downloaded `.zip` into the `<DT_HOME>` directory.

3. Include the following lines at the top of the main JavaScript file of your application.

   > **Note**
   >
   > This is the Agent connection string, and these lines **must** be the first lines at the beginning of the file to capture PurePaths.

```
try
   {
       require('<DT_HOME>/agent/bin/any/onenodeloader.js')({
         server: 'http://<CollectorName>:8042',
         agentName: '<AgentName>'
       });
   } catch (err) {
       console.error(err.toString());
   }
```

The port after `<CollectorName>` is not optional as with other Agents that use TCP, because with HTTP it defaults to 80.

For example, if all AppMon components are installed to `/opt/dynatrace-7.0` and the `Node.js` Agent - Agent Group/tier mapping is `Node.js_Monitoring`, then the main JavaScript must start with the following lines:

```
try
   {
       require('/opt/dynatrace-7.0/agent/bin/any/onenodeloader.js')({
         server: 'http://localhost:8042',
         agentName: 'Node.js_Monitoring'
       });
   } catch (err) {
       console.error(err.toString());
   }
```

Upgrade

It is strongly recommended to start with a new installation (new folder), when upgrading the Node.js Agent from any older version to 7.0.

Licensing

Licensing is per process on the server side. For example, if you have a license for 100 `Node.js` Agents you can concurrently run 100 instrumented `Node.js` processes. Refer to `Node.js` documentation to learn how to tweak the clustering.

Node.js Agent troubleshooting

5. In the browser, go to `[http://localhost:1337](http://localhost:1337)` to trigger a "/" PurePath in the **Cockpits > Monitoring System Profile > Diagnose Performance > PurePath dashlet > overview** and a WebRequestHandler() detail.

4. Enter the following to launch an instrumented `Node.js` web server.

```
cd js
~/node-0.12.4/bin/node serverHello.js
```

   You should now see the Agent in the **Cockpits sidebar > Agents Overview**.

   If you do not have a `Node.js` tier in your System Profile, such as the Monitoring system profile, "Agent did not match System Profile" appears to the left of the Agent. For a new AppMon installation, click the **Configure** button. Otherwise right-click the System Profile of choice and select **Add Tier**. Then in the wizard, select **Application Type > Other**, **Add Java, .NET, Native** and select `Node.js` from the application server category.

> **Tip**
>
> You can ignore any yellow alert triangle version alerts at this point.

3. Enter `sudo chown -R <username> /opt/[dynatrace-release]` for the permissions.

2. Use the commands similar to the example below to install the `Node.js` Agent. As seen in this example, AppMon 6.5 is installed in `/opt`.

```
cd /opt
sudo tar -xzf ~/Downloads/dynatrace-one-agent-nodejs-6.5.0.1289-linux-x86.tgz
```

1. Put the most basic `Node.js` in `~/js/serverHello.js`, then reference the actual `Node.js` Agent installation directory in the first line and adapt to the versions that you are using. For example:

```
try
    {
      require('/opt/dynatrace-6.5/agent/bin/any/onenodeloader.js')({
        server: 'http://localhost:8042',
        agentName: 'Node.js_Monitoring'
      });
    } catch (err) {
        console.error(err.toString());
    }

var http = require('http');
http.createServer(function (req, res)
    {
        res.writeHead(200, {'Content-Type': 'text/plain'});
        res.end('Hello localhost\n');
    }
    ).listen(1337, '127.0.0.1');
     console.log('Server running at http://127.0.0.1:1337/');
```

To test `Node.js`, download and unzip it to an appropriate location. For example, unzip `Node.js` 0.12.4 to `~/node-0.12.4`.

Basic Node.js testing setup

> **Tip**
>
> For AppMon 6.3 and later, the HTTP port can be changed at startup with the argument `-dtangport <port>` to the launcher (`dtserver` or `dtcollector`) and is persisted in `.config.xml`. This only needs to be applied once. The HTTPS port is HTTP+1. You can also change a port with `-Dcom.dynatrace.diagnostics.dtangport=<port>` in the respective `.ini` files in `<DT_HOME>`.

There are two `<dtang>` entries in the AppMon Server settings.The first entry applies to the embedded Collector, and is only used for testing. Its settings should be the same as in `collector.config.xml` of the stand-alone Collector(s), which by default is 8042 / 8043 for HTTP / HTTPS. The second occurrence is the port the Server listens for HTTP / HTTPS communications from the Collector(s),

which by default is 8040 / 8041).

Additional Collector instances on one machine listen on the port with the offset specified at instance creation. For example 8142 / 8143.

Look for the `<dtang>` sub-tree in the settings. For the Collector, the Agent ports are 8042 / 8043 (HTTP / HTTPS) by default.

- Server - Collector in `<DT_HOME>/server/conf/server.config.xml`
- Collector - Agents in `<DT_HOME>/collector/conf/collector.config.xml`

If conflicting with other applications, you can change the HTTP(S) communication ports. For AppMon 6.3 and later, both configuration locations changed.

Resolve port conflicts

When you do not see Agent-loading feedback when you start your application, you may be using an unsupported `Node.js` version. Use a supported version of `Node.js`.

No Agent-loading feedback at application startup

Troubleshooting

Database Agent configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Grant necessary permissions for DB monitoring

Configuration

Related pages

Database Agent configuration - DB2

Database Agent

Database Agent troubleshooting

Database monitoring

Watch and learn

Database diagnostics use cases with Dynatrace

Forum posts

Dynatrace Forums Home

As stated on the Database Monitoring main page, it is important for configuration to differentiate between the database monitoring on the DBMS instance level, overarching a whole AppMon installation described here, and the JDBC, ADO.Net and PHP DB sensors on the System Profile / app level that capture SQL statements and bind values and their count and timing, and come with the Java, .NET and PHP Agents respectively.



See also Database Agent Troubleshooting in case of issues.

Grant necessary permissions for DB monitoring

The user that connects to the DB instance needs to be granted the following permissions:

Database Platform	Permissions needed
Oracle	• CREATE SESSION • SELECT_CATALOG_ROLE
MS SQL Server	Database login needs the following permissions: • CONNECT SQL • VIEW SERVER STATE • VIEW ANY DATABASE (to see tablespaces) User in the *master* database needs the following permissions: • CONNECT • VIEW DATABASE STATE (to see locks, tablespaces, wait states) For each database that should be included in the Live View tablespaces report and execution plans for its queries: • CONNECT • VIEW DATABASE STATE To view execution plans for stored procedures user needs access to stored procedures (at least VIEW DEFINITION). This can be granted on the whole database scope, server, schema or per individual stored procedure. See example
MySQL	• SHOW DATABASES • PROCESS • SELECT • EXECUTE
DB2	• SQLADM (or wider: DBADM), • SYSMON (or wider: SYSMAINT, SYSCTRL, SYSADM), • DATAACCESS (or per object permissions, see DB2 section below)

Refer to the release notes for more details about supported versions.

SQL Server 2012 & 2014

```
CREATE LOGIN <dtagent> WITH PASSWORD = 'password';
GRANT CONNECT SQL TO <dtagent>;
GRANT VIEW SERVER STATE TO <dtagent>;
GRANT VIEW ANY DATABASE TO <dtagent>;

USE master;
CREATE USER <dtagent> FOR LOGIN <dtagent>;
GRANT CONNECT TO <dtagent>;
GRANT VIEW DATABASE STATE TO <dtagent>;

-- for each monitored database
USE <monitored_database>;
CREATE USER <dtagent> FOR LOGIN <dtagent>;
GRANT CONNECT TO <dtagent>;
GRANT VIEW DATABASE STATE TO <dtagent>;
GRANT VIEW DEFINITION TO <dtagent>;
```

Oracle Database 11g/12c

```
CREATE USER dtagent IDENTIFIED BY password
default tablespace users
temporary tablespace temp;

GRANT CREATE SESSION, SELECT_CATALOG_ROLE TO <dtagent>;
```

> DBMS_LOB package with granted EXECUTE permission is required to fetch queries' explain plans.

> Access to `Dynamic Performance Views` need to be granted for Oracle monitoring.

MySQL 5.6/5.7

```
CREATE USER 'dtagent'@'%' IDENTIFIED BY 'password';
GRANT SELECT, EXECUTE, PROCESS, SHOW DATABASES on *.* to 'dtagent'@'%';
```

MySQL User created for the database agent must use default password encryption (`mysql_native` authentication plugin). Other password encryption methods, like SHA256 (`sha256_password` authentication plugin) are not supported.

To view SQL Execution plans when using MySQL db agent, the user must also have the SELECT privilege on the schema containing tables specified in the query

To use the database agent with MySQL releases prior to 5.7.7, the `sys` schema (documentation) must be installed, see sys schema project page for installation instructions. Additionally, Performance Schema (documentation) must be enabled.

DB2 9.7.0.11, 10.1, 10.5, 11.1

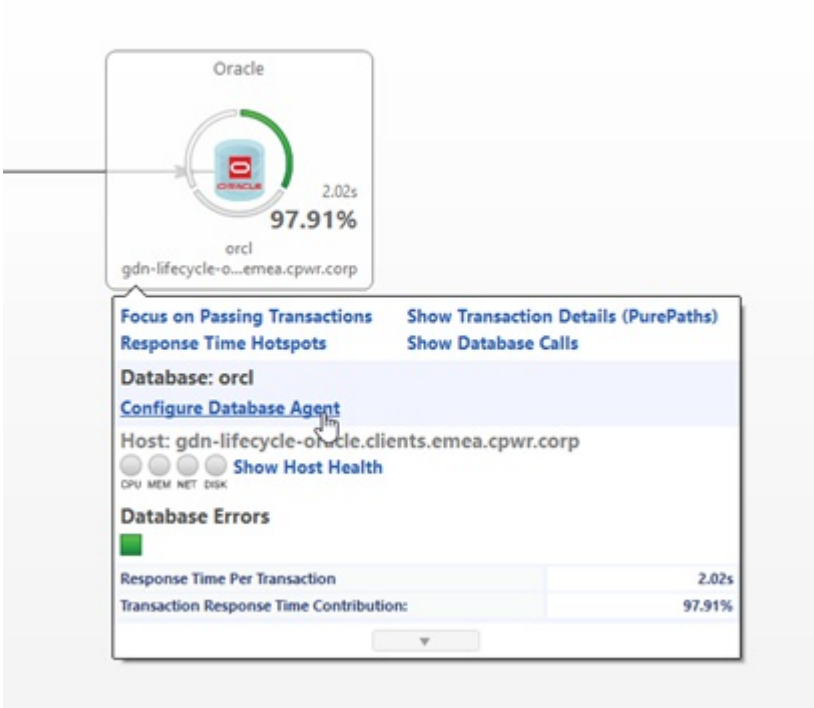
See Database Agent configuration - DB2

Configuration

Configure database agents for database monitoring in the Client by selecting **Settings > Dynatrace Server > Databases** and clicking **Create** or **Edit**.

> **Tip**
>
> You can also configure JDBC, ADO.Net and PHP DB sensors in **tiers > Sensor Configuration** for a specific System Profile sensor.

Initially create Database Agents using the **Application Environment Configuration** wizard. In the AppMon Client, click the gear icon in the top right corner of the **Monitoring** dashboard, and select **Configure Environment** to access the wizard. Then when monitoring applications, you can access the wizard directly from the context menu of an automatically detected database tier in the Transaction Flow.



You can use automatic cluster discovery to configure Oracle Databases with Real Application Clusters (**Oracle RAC**). To do this, select **Oracle Database with Real Application Clusters (RAC)** in the **Database type** field of in any Create Database Agent window and fill other data for any of your RAC node. After testing connection, confirm any discovered nodes.

For any issues with cluster automatic discovery, you can discover all cluster nodes one by one, as they are correlated with each other automatically. After adding AppMon Agents for cluster, they are visible as separate agents in all configuration screens and can be edited manually.

Oracle RAC 12c leaf nodes

With Oracle Database with Real Application Clusters (RAC) **12c**, new type of base platform was introduced: Oracle Flex Clusters. It contains two types of nodes: Hub Nodes and Leaf Nodes. AppMon is able to monitor only nodes, where actual database instance is running - and it can run only on Hub nodes. Because of that, monitoring Leaf nodes with Database Agent is not possible. However, Host Agent can be installed on machine, in order to collect data like memory & CPU usage and more.

Database Agent troubleshooting

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Frequently asked questions / issues

Dynatrace support

Related pages

Database Agent configuration

Database monitoring

Database health monitoring

SQL execution plan

Watch and learn

Forum posts

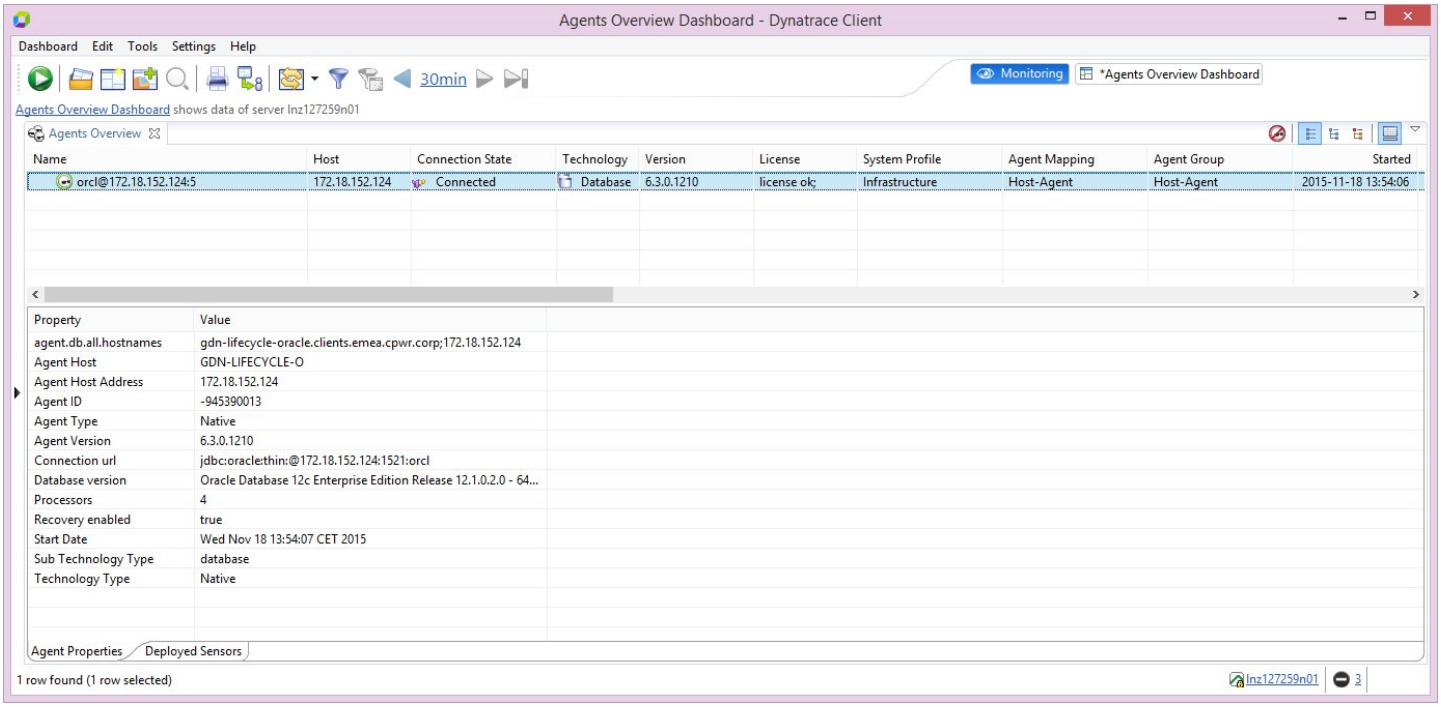
Dynatrace Forums Home

The following page describes various techniques for troubleshooting and for collecting the necessary data for support.

Frequently asked questions / issues

How can I check the status of the Database Agent?

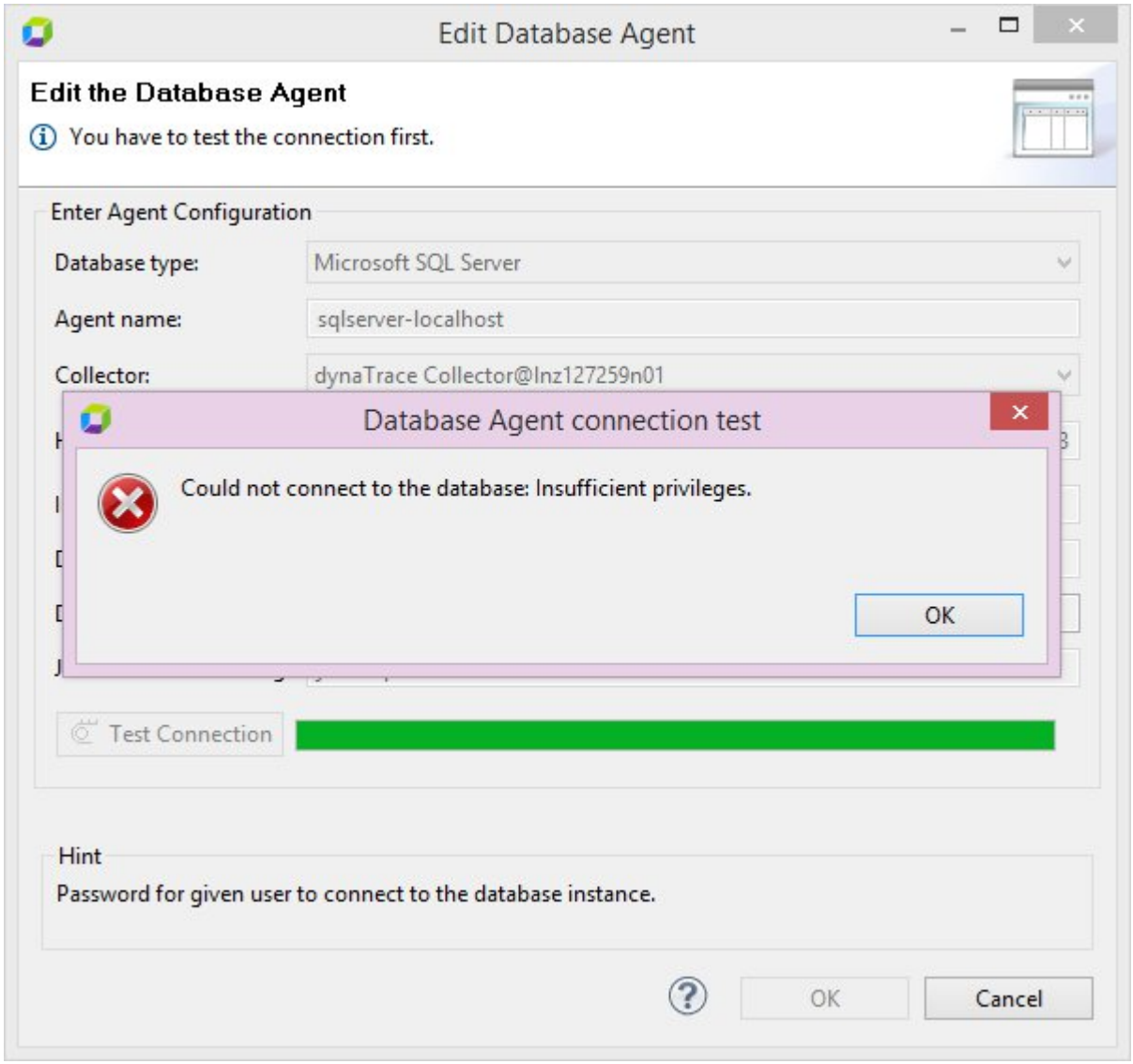
As with other Agents the status of this one is displayed in the Agents Overview Dashlet Dashboard. Make sure to check if it is connected and its license status.



In case of issue, make sure to check the status of the Collector hosting the Agent as well the connection to the database. (Settings menu > Dynatrace Server… > Databases).

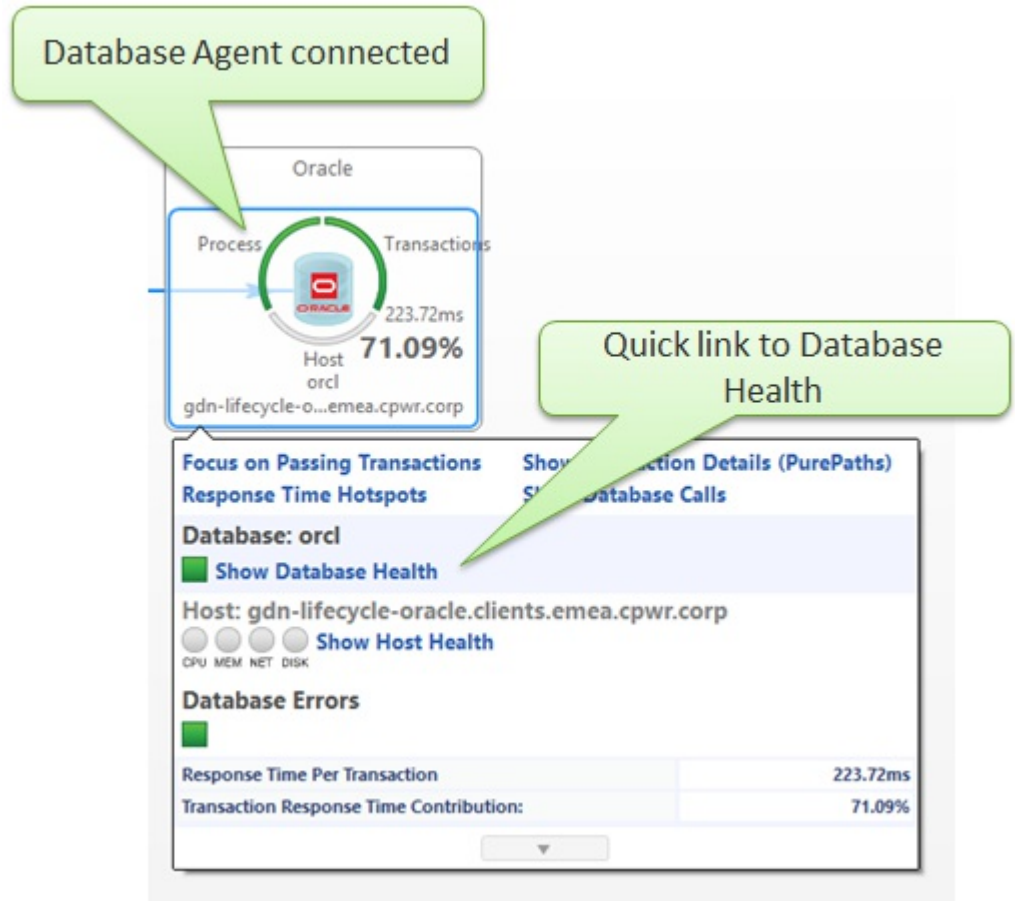
Which permissions are needed to connect the Database Agent? how can I test it?

The permissions needed for the Database Agent and the respective SQL statements are listed in Database Agent Configuration > Permissions.
After configuring the Agent, click on Test Connection. The following error appears if the configured user doesn't have the needed permissions:

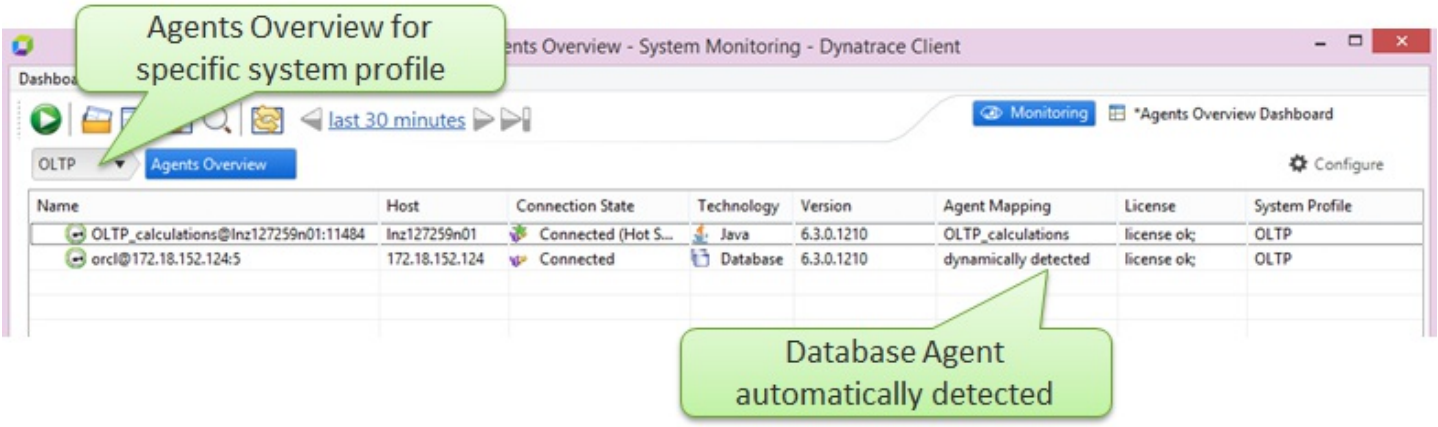


How does the mapping of global monitoring data into System Profiles work?

Databases monitoring data are mapped into System Profiles dynamically if a database tier is detected in a PurePath and a DB Agent is connected to that same database.



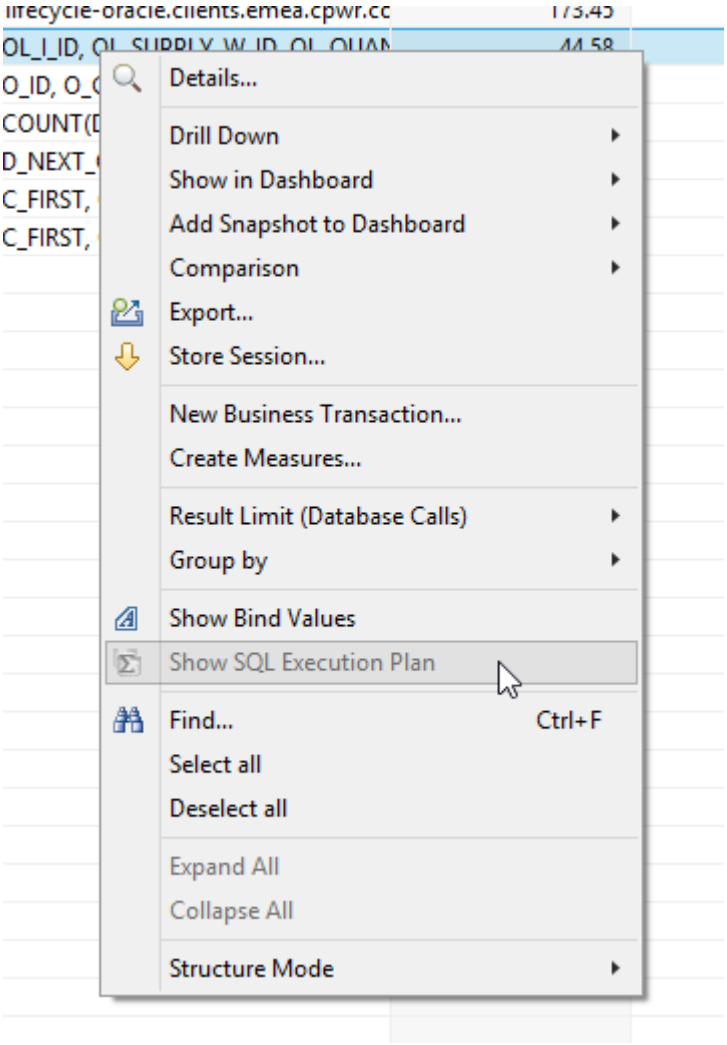
The dynamically mapping can be checked in the **Agent Overview** of a specific system profile:



Why is the option 'Show SQL Execution Plan' disabled?

It is disabled if:

- No Database Agent has been configured for that particular database or it is currently not connected.
- The Database Agent has not been automatically detected for that System Profile - see below “How does the mapping with the System Profile work?”
- The database call is a stored procedure, which is currently not supported.



Where are the log files stored?

They are stored with the log files of the Collector on which the Database Agent has been configured. Search for DbAgent.

Is there an automatic fail-over for Database Agents?

No, not currently.

Is Oracle RAC supported?

We currently offer limited support for RAC. You can either connect each of the RAC nodes separately to have a real visibility into the cluster OR connect the cluster as one single DB instance (but in that case the Database Agent will only monitor the node to which it is connected).

How is the Database Agent licensed?

A license is consumed for each DB instance with a maximum of eight CPU cores per license. Every other instance or eight cores consume another license. See also Licensing Details.

The **CPU cores for Oracle** are retrieved by querying the v$parameter view:

```
SELECT value FROM v$parameter WHERE name LIKE 'cpu_count'
```

The **CPU cores for SQL Server** are retrieved by querying the system view dm_os_sys_info:

```
SELECT cpu_count FROM sys.dm_os_sys_info
```

MySQL: a license is consumed for each DB instance - no limitation on the number of CPU cores.

Are all (Oracle and MS) DBMS editions (enterprise, standard...)

Yes, the Database Agent doesn't require any specific edition. All Database Agent features are available with all editions.

Is Windows authentication supported for SQL Server?

Yes, since AppMon 6.5.

DB2 - Explain tables are not present in the database

Invoke the following to create explain tables in the SYSTOOLS schema:

```
CALL SYSPROC.SYSINSTALLOBJECTS('EXPLAIN', 'C', CAST (NULL AS VARCHAR(128)), CAST (NULL AS
```

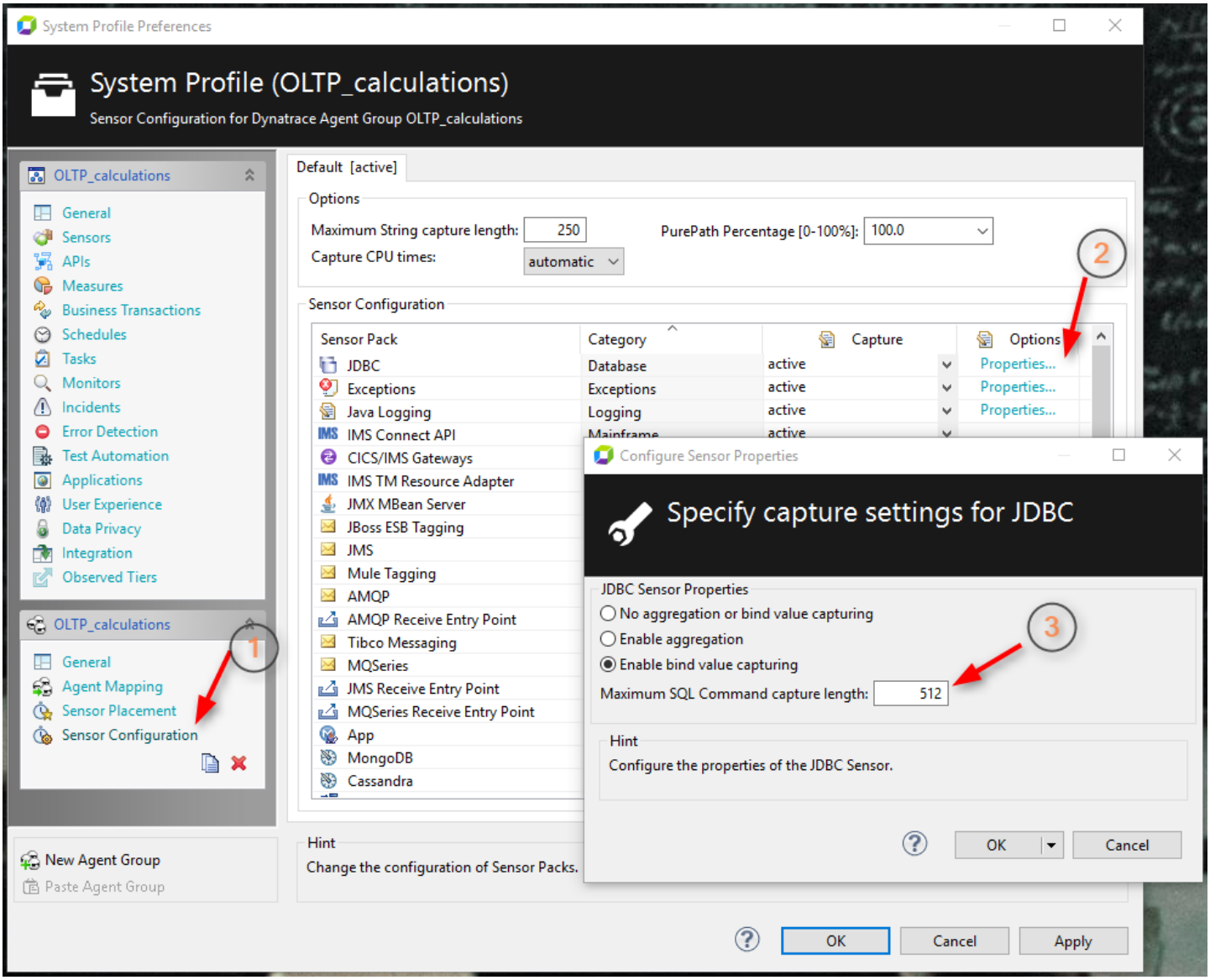
DB2 explain facility works by populating explain tables. These are not created by default. IBM documentation

Alternatively the database agent DB2 user may not have all required permissions:

```
grant all on SYSTOOLS.EXPLAIN_ACTUALS to <dtagent>
grant all on SYSTOOLS.EXPLAIN_ARGUMENT to <dtagent>
grant all on SYSTOOLS.EXPLAIN_DIAGNOSTIC to <dtagent>
grant all on SYSTOOLS.EXPLAIN_DIAGNOSTIC_DATA to <dtagent>
grant all on SYSTOOLS.EXPLAIN_INSTANCE to <dtagent>
grant all on SYSTOOLS.EXPLAIN_OBJECT to <dtagent>
grant all on SYSTOOLS.EXPLAIN_OPERATOR to <dtagent>
grant all on SYSTOOLS.EXPLAIN_PREDICATE to <dtagent>
grant all on SYSTOOLS.EXPLAIN_STATEMENT to <dtagent>
grant all on SYSTOOLS.EXPLAIN_STREAM to <dtagent>
```

Why does SQL appear truncated?

For performance reasons SQL gathered by agents is truncated to 512 characters. This can be modified by editing agent sensor properties for database category sensors (JDBC, ADO.net, PHP…):



Why is my explain plan not retrieved?

Please refer to SQL execution plan limitations section

Dynatrace support

If you need further assistance, go to our **support** page and make sure to provide a session export and a support archive.

Database Agent configuration - DB2

On this page

Related pages

Database Agent configuration

Database Agent

Database Agent troubleshooting

Database monitoring

Watch and learn

Database diagnostics use cases with Dynatrace

Forum posts

Dynatrace Forums Home

DB2 9.7.0.11, 10.1, 10.5, 11.1

Please note that DB2 9.8 is not supported.

The *sysmon* authority assigned to the group specified by the sysmon_group configuration parameter (run db2 get database manager cfg on the DB2 machine to check). If the parameter is null it defaults to the administrators group (Windows) or the primary group of the DB2 instance owner (Linux/Unix).

```
grant sqladm on database to <dtagent>
grant dataaccess on database to <dtagent> -- see "Replacement for DATAACCESS authority" b
CALL SYSPROC.SYSINSTALLOBJECTS('EXPLAIN', 'C', CAST (NULL AS VARCHAR(128)), CAST (NULL AS
```

Replacement for DATAACCESS authority (AppMon 7.0.2):

```
      grant execute on function SYSPROC.COMPILATION_ENV to <dtagent>
        grant execute on function SYSPROC.AUTH_LIST_GROUPS_FOR_AUTHID to <dtagent>
        grant execute on function SYSPROC.MON_FORMAT_LOCK_NAME to <dtagent>
        grant execute on function SYSPROC.MON_GET_CONNECTION to <dtagent>
        grant execute on function SYSPROC.MON_GET_CONTAINER to <dtagent>
        grant execute on function SYSPROC.MON_GET_PKG_CACHE_STMT to <dtagent>
        grant execute on function SYSPROC.MON_GET_SERVICE_SUBCLASS to <dtagent>
        grant execute on function SYSPROC.MON_GET_TABLESPACE to <dtagent>
        grant execute on function SYSPROC.MON_GET_WORKLOAD to <dtagent>
        grant execute on function SYSPROC.MON_GET_LOCKS to <dtagent>

        grant select on SYSIBMADM.ENV_INST_INFO to <dtagent>
        grant select on SYSIBMADM.ENV_SYS_INFO to <dtagent>
        grant select on SYSIBMADM.ENV_SYS_RESOURCES to <dtagent>
        grant select on SYSIBMADM.TBSP_UTILIZATION to <dtagent>

        grant all on SYSTOOLS.EXPLAIN_ACTUALS to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_ARGUMENT to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_DIAGNOSTIC to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_DIAGNOSTIC_DATA to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_INSTANCE to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_OBJECT to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_OPERATOR to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_PREDICATE to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_STATEMENT to <dtagent>
        grant all on SYSTOOLS.EXPLAIN_STREAM to <dtagent>

-- execute in addition to the above for DB2 10.5 and up:

        grant execute on function SYSPROC.MON_GET_INSTANCE to <dtagent>
        grant execute on function SYSPROC.MON_GET_DATABASE to <dtagent>
```

(System monitor authority (SYSMON) IBM documentation) (sysmon_group IBM documentation) (SQLADM IBM documentation) (DATAACCESS IBM documentation)

The SYSPROC.SYSINTERNALOBJECT invocation creates explain plan tables, which need to exist for the explain plan functionality to work. This call creates the explain tables under the SYSTOOLS schema, which is the location assumed by Dynatrace AppMon. (Explain tables IBM documentation)

OneAgent configuration

On this page

Port Configuration

Related pages

OneAgent

OneAgent rollout

System requirements

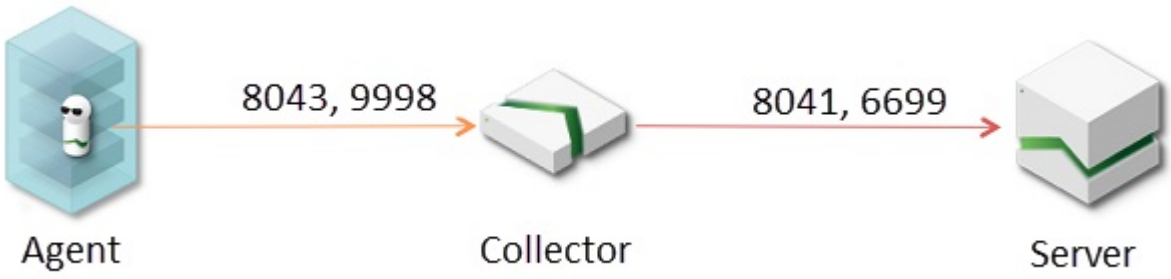
Set up communication connections

AppMon 2017 May Agents are always installed as Classic Agents and use the AppMon platform. To use the OneAgent platform you need to rollout your Agents to it. See OneAgent rollout to learn how.

Port Configuration

If you want to use the OneAgent, you need to open the following HTTPS communication ports for it:

- 8043, 9998 for the Agent-Collector connection
- 8041, 6699 for the Collector-AppMon Server connection



Notes

- Requires Bootstrap Agent v6.x, or higher.
- No further changes of the Agent setup is required. Agent configuration is automatically merged.

FAQ

Q: Are the Collector Groups and load balancing supported with the OneAgent?

A: Yes

Q: Can I connect Classic Agents and OneAgents to the same collector ?

A: Yes. Just make sure all required ports are open.

OneAgent rollout

Related pages

OneAgent

OneAgent configuration

System requirements

Set up communication connections

For easy migration from existing installations, every 6.x, v7.x Bootstrap Agent is capable to rollout the OneAgent as well as switch back to the Classic Agent without the need to re-configure the Agent injection. The switch of the Agent platforms can be simply triggered via the AppMon Client.

How to migrate from Classic Agent to OneAgent for AppMon

Before you migrate your production environment, make sure that:

- The OneAgent meets your existing requirements and supports your technology.
- The required ports are open.
- Your BootStrap Agent is version 6.x or later.

You need to switch your AppMon Client to the Debug Mode.

> **Important**
>
> OneAgent is a BETA feature for some Agents. You should **not** use it in your production environment. The aim of BETA is to test it in pre-production environment, and see how the new Agent affects your custom dashboards/configurations you use to monitor your production environment.

1. In the AppMon Client right click your System Profile and select **Edit System Profile > General**.

2. Click **Agent Platform**.
3. In the **Agent Platform** dialog, select the required technology from the list at the top.
4. Select the required Agent Groups, and click **Move to the right**.
5. Apply your changes.
6. Restart your application and the affected Agents.

If needed, use the same procedure to switch back to the Classic Agent.

> **Note**
>
> Newly connected Agents of this group will also use the OneAgent platform.

Set up the AppMon Client

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What to know before setup

Starting the Client

Related pages

Set up the Webstart Client

AppMon Clients for Windows and *NIX are 64-bit, which has several advantages:

- Standalone clients are included in Windows and Linux full installation packages, so separate installations are no longer necessary.
- For Linux 64-bit, there is no need to install extra libraries.
- Bigger data sets for analysis, coinciding nicely with the introduction of the Frontend Server serving these.

What to know before setup

- The AppMon 32-bit Client is no longer supported with AppMon 6.5 and later.
- The Linux installation process requires a Java Runtime Environment (JRE) 1.8 or later.
- AppMon Server, Collector and Client standalone installations run only on operating systems with Java 8. On Windows and Linux, the AppMon package comes with a private JRE 1.8 for run time.
- The Client on the Mac requires JRE 1.8 for run time.
- Download and install the AppMon Client on Windows, Linux or MacOS X (Intel architecture only).

See System Requirements > AppMon Client for exact supported OS versions.

- Memory requirements: 512MB minimum, 1GB recommended.

- If you want a locally installed Client, for example to connect to several same-version AppMon Servers (no matter what bitness and on which OS) you either install the Client with a full installer together with other components or on Windows do a custom installation – or install separately from a Client-only installer.
  Sample *NIX Client-only installation command: `java -jar <path>/dynatrace-client-<version>-<platform>.jar`
  The Linux Client installation file is named `dynatrace-client-<version>-linux-x86.jar` contains 64-bit client only.

To change the used memory for the AppMon Client, choose **Settings > Dynatrace Client** and set the **Maximum heap memory** to a reasonable size.

Use the **Dynatrace Client Preferences** dialog to configuration the following:

- **General settings**: Maximum heap memory, check for Client update, dashboard change, reset warning dialogs and layouts, UI full or simple.
- **Services**: Enable REST Web services and port, allow non-SSL connections and set port, configure proxy for Client.
- **Directories**: Configure where to store reports, dashboards, logs, etc.
- **External Tools**: .NET reflector and text editor location and Java decompiler.

Starting the Client

Windows

Start the Client in one of the following ways:

- From the **Start** menu.
- From the command line with `%DT_HOME%\dtclient.exe`, where `%DT_HOME%` for a default full installation is `C:\Program Files\dynaTrace\Dynatrace 7.0`.

Command Line Start Examples

Installation package	AppMon bitness	CD to default installation <DT_HOME> copy/paste example
Full	64	`cd C:\Program Files\dynaTrace\Dynatrace 7.0`
Client	64	`cd C:\Program Files\dynaTrace\Dynatrace Client 7.0`

So `cd <DT_HOME>` and `dtclient.exe`.

Linux

If you accept the default directory and install to the recommended `/opt`, you can invoke the Client with the following:

```
cd /opt/dynatrace-7.0 ./dtclient
```

Mac OS X

Copy the AppMon Client from the disk image to the installation location on your hard drive. To start it, double-click the Client icon (JRE 1.8 required).

To import settings from a previous installation:

1. Open the old and new Client application package by choosing **Show Package Content** from the context menu.
2. Copy the files listed below from inside the old Client to the corresponding directories in the **Users** home directory. The new AppMon Client creates these after its first start.

Copy from:

- `Contents/MacOS/client/sessions/*`
- `Contents/MacOS/client/conf/dashboards/*`
- `Contents/MacOS/client/conf/client.config.xml`
- `Contents/MacOS/client/conf/client.properties.xml`

Paste to:

- `<userhome>/.dynaTrace/dynaTrace <major.minorVersion>/client/sessions/*`
- `<userhome>/.dynaTrace/dynaTrace <major.minorVersion>/client/conf/dashboards/*`
- `<userhome>/.dynaTrace/dynaTrace <major.minorVersion>/client/conf/client.config.xml`
- `<userhome>/.dynaTrace/dynaTrace <major.minorVersion>/client/conf/client.properties.xml`

Set up the Webstart Client

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

How does it work?

Requirements

Run the client using webstart

Run the client using JNLP file

Configuration

Working directory

Migration of stored sessions and dashboards

Frequently asked questions

Related pages

Set up the AppMon Client

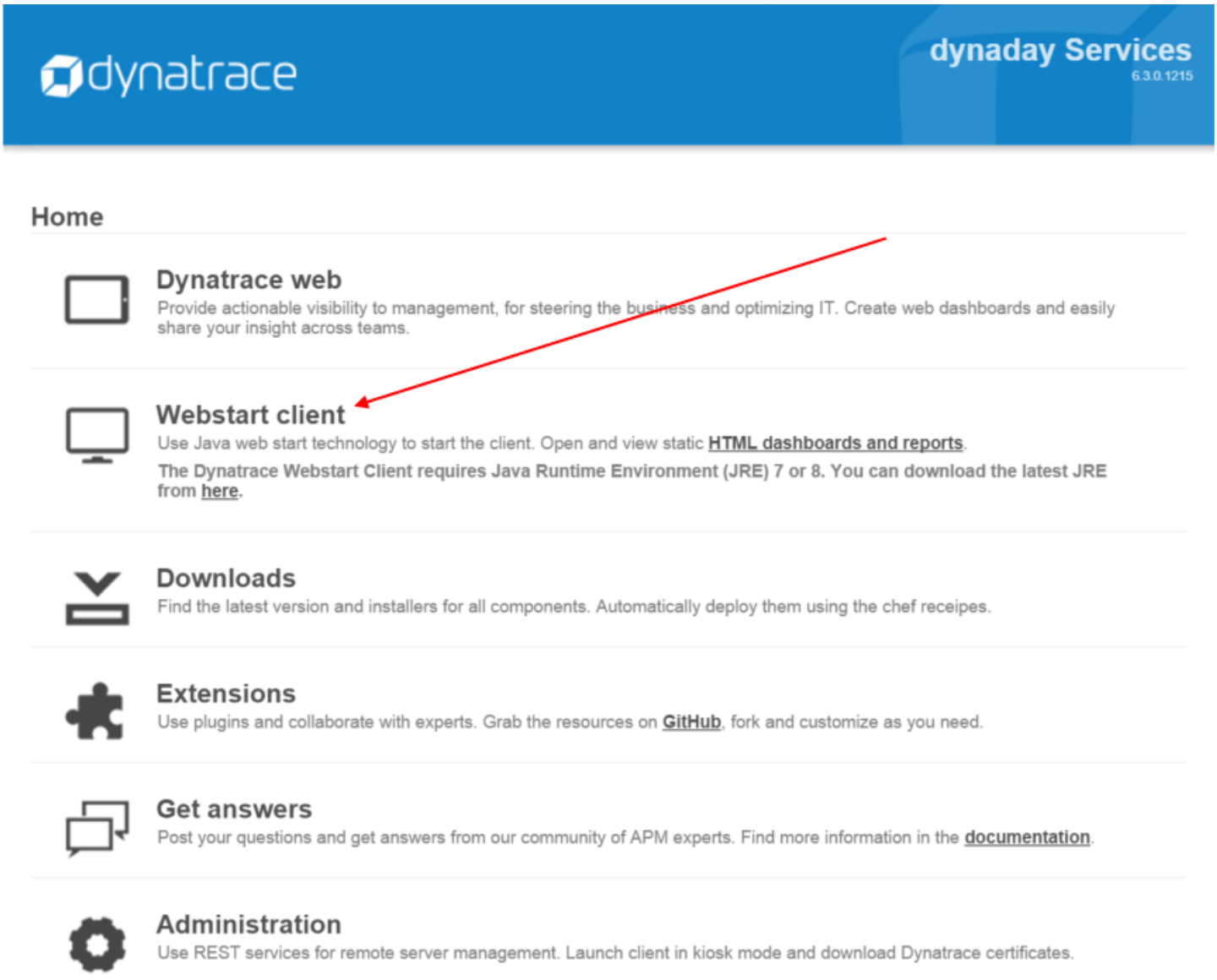
The AppMon Webstart Client is a fully functional AppMon Client that uses the Java Web Start Technology to minimize administrative effort. To access the AppMon Webstart Client, you open a link in a web browser. You don't install an AppMon Client on the local machine. When the AppMon Server updates, all AppMon Webstart Clients update automatically at start up. The Webstart Client can only connect to a single AppMon Server at a time. Consequently, the **Add Dynatrace Server** button does not appear in the Client's **Settings > Dynatrace Client** dialog.

How does it work?

By default, the AppMon Server ships with an AppMon Webstart plugin. The AppMon Server uses the plugin to display a download link on the Server home page. The default HTTP settings are at `https://<servername>:8021`.

The link **Webstart client** references a JNLP file that you need to open with a Java Web Start Launcher. The Java Web Start Launcher downloads all required resources and starts up the Webstart Client. The Java Network Launching Protocol (JNLP) is specified in JSR 056 by the Java Community Process.

Launch the Webstart Client from the AppMon Server Home Page



Requirements

See system requirements.

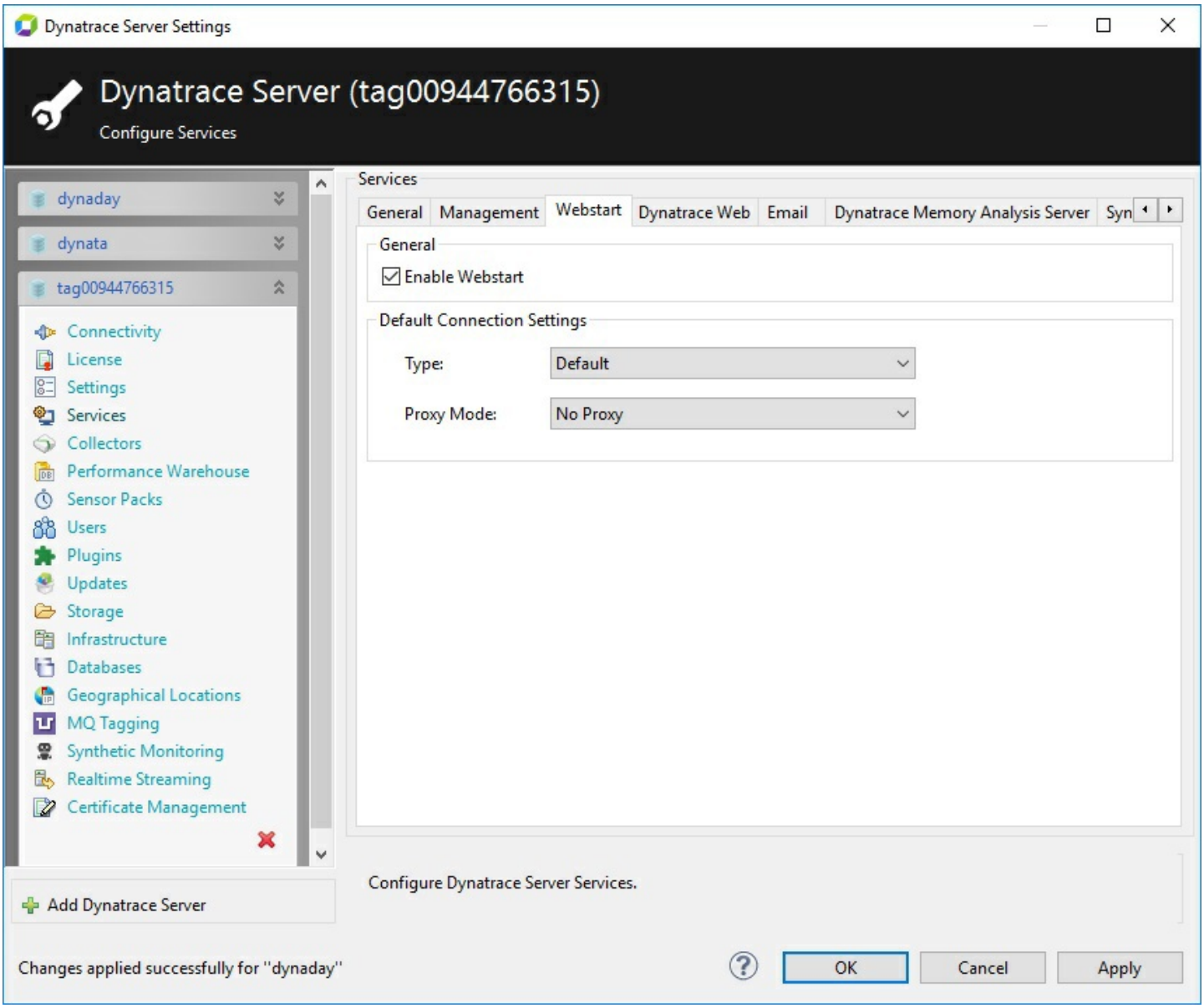
Run the client using webstart

Open a web browser, navigate to `https://<servername>:8021/webstart/Client/client.jnlp` and click `Webstart client` link. The browser may ask for confirmation to run `client.jnlp` file.

Run the client using JNLP file

If your browser does not support webstart, you can save the JNLP file, and use it to launch the client. Refer to Java documentation for the detailed intructions.

Configuration



The default installation is pre-configured so the AppMon Webstart Client is immediately available.

- **Enable Webstart**: Enables or disables the AppMon Webstart Client.
- **VM Arguments**: Shows the JVM arguments used by the AppMon Webstart Client. Due to security restrictions in the Oracle Java Runtime environment this settings cannot be edited.
- **Default Connection Settings**: Specifies the default Connectivity settings of AppMon Webstart Clients.

Working directory

Webstart Client can connect to its host server only. This is the only difference from a locally installed Client, which supports connections to multiple servers. Additionally, configuration files of the Webstart Client are stored in server-based working directories to prevent incompatibility issues.

The working directory of Webstart Clients is `/.dynaTrace/Dynatrace Webstart Client <servername>`. You can find it in the user working directory of your system.

The user working directory depends on the operating system:

- **Windows**: `/Users/<username>/.dynaTrace/Dynatrace Webstart Client <servername>`.
- **Linux**: The main working directory is hidden in `/home/<username>/.dynaTrace/Dynatrace Webstart Client <servername>` and you must enable hidden system folders to display in your file browser.

Migration of stored sessions and dashboards

Follow these steps to migrate **locally Stored Sessions** such as PurePath sessions or memory dumps from an older AppMon Webstart Client:

1. Run the older AppMon Webstart Client first.
2. Export all necessary AppMon stored sessions.
3. Run the newer AppMon Webstart Client.
4. Import the AppMon stored sessions.

Follow these steps to migrate dashboards:

1. Run the older AppMon Webstart Client first.

2. Store all necessary dashboards at an AppMon Server.
3. Run the newer AppMon Webstart Client.
4. Open dashboards from the AppMon Server.
5. Optionally, store dashboards locally.

Webstart Client kiosk mode

Go to `https://<servername>:8021` with your web browser. Under `Administration` you find the `Kiosk mode` link to launch the Webstart Client in kiosk mode. The default interval of dashboard switches is 30 seconds. You can customize the interval by applying for instance a 10 sec interval argument in the Webstart launch URL:

```
https://<servername>:8021/webstart/Client/client.jnlp?argument=-kiosk&argument=10
```

Webstart Client URL equivalent to command line arguments

The following is a locally installed Client's command line argument example in Client Commandline Arguments:

```
<DT_HOME>/dtclient -dashboard "online://<servername>/Incident Dashboard?filter=tf:Last7d"
```

The Webstart Client URL equivalent:

```
http://<servername>:8021/webstart/Client/client.jnlp?argument=-dashboard&argument=online:
```

After the basic Webstart URL `http://<servername>:8021/webstart/Client/client.jnlp`, the query starts with the usual ? and the arguments are separated by &. The space in *Incident dashboard* is escaped with a + and can also be escaped with %23. Refer to w3schools.com > HTML URL Encoding Reference for a complete table.

Note

In the `online://` argument, you must use the AppMon Server's name, which is typically the lowercased host name and not localhost.

With the additional `argument=-reuse`, the dashboard opens in a running Client.

```
http://<servername>:8021/webstart/Client/client.jnlp?argument=-reuse&argument=-dashboard&
```

The following example only works if you define a PurePath Comparison dashboard and two stored sessions. The & is for argument separation and the `source`. The `compare` part uses the same character, and must be escaped with %26.

```
http://<servername>:8021/webstart/Client/client.jnlp?argument=-dashboard&argument=online:
```

See Client Commandline Arguments and REST Filters for more argument descriptions.

Frequently asked questions

Can I launch the Webstart Client even offline when the server is unavailable?

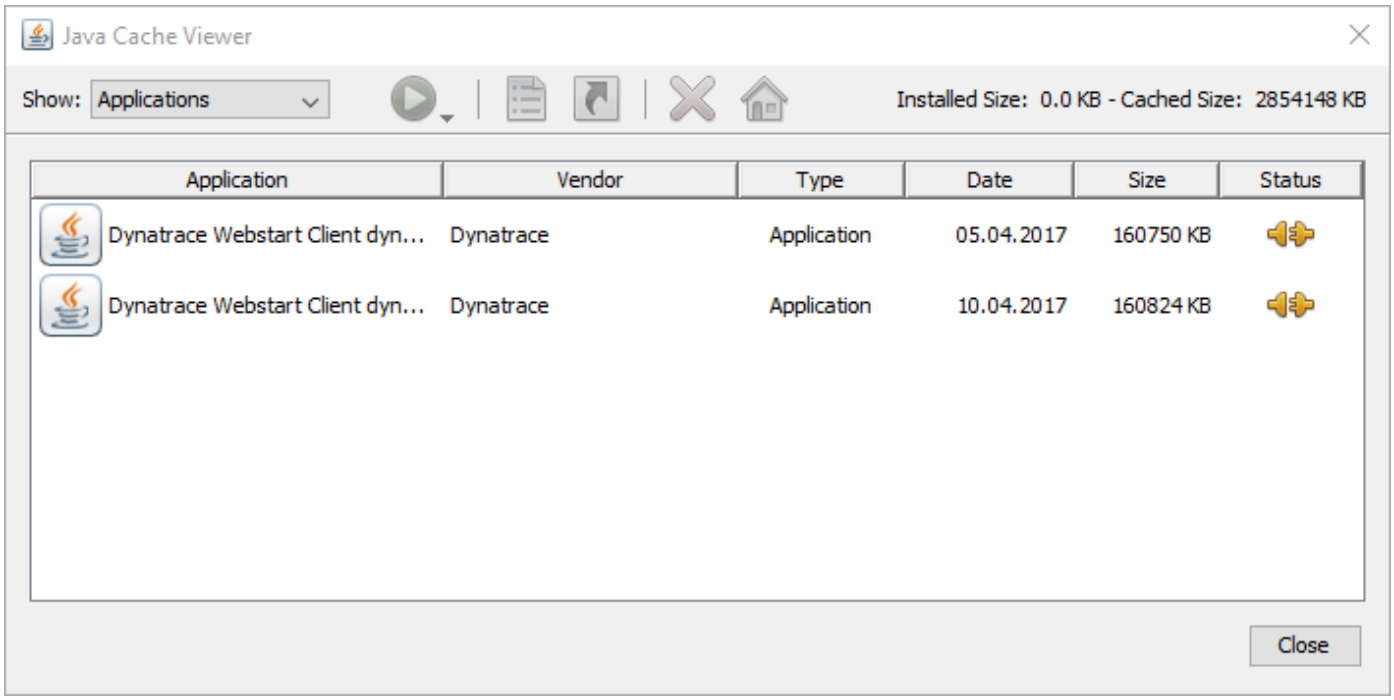
On Windows, use the **Start** menu shortcut that that appears after the first start of the AppMon Webstart Client. Java automatically tries to connect to the AppMon Server to check for updates. If the AppMon Server is not available, the AppMon Client launches the available version.

Is it possible to place a Webstart Client shortcut on the desktop?

Yes. Follow these steps:

1. If you have not launched a AppMon Webstart Client, run it now.
2. To open the Java Cache Viewer, run the command `javaws -viewer`.

3. The following dialog box appears (Java 1.6):



4. Select the Webstart Client and click the corresponding button to create a desktop shortcut.

How can I remove the Webstart Client from my computer?

1. Open the Java Cache Viewer. To do this, run the command `javaws -viewer`.
2. Select the Webstart Client and click the corresponding button to remove the application.

Can I control where the cache for the webstart files is located, e.g. if the user-home-drive is located on a shared folder?

You can control this in the Java Control Panel, which is available on Windows under **Control Panel > Java**. If the button to choose the folder is disabled, you can adjust a property file. This has the same effect:

```
<user-home>\<username>\AppData\LocalLow\Sun\Java\Deployment\deployment.properties
```

Create the following property:

```
`deployment.user.cachedir=<location_to_webstart_cache>`
```

See the Oracle page for Deployment Configuration File and Properties for more information.

Application integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Apache Tomcat

Application Environment Configuration

Cassandra

CICS transaction gateway

Fujitsu Interstage Application Server

GlassFish Server Open Source Edition

Java applets

JBoss, Wildfly Application Server

Microsoft SharePoint

MongoDB

Mule ESB

Oracle Application Server

SAP Netweaver

WebLogic Application Server

WebMethods Integration Server

WebSphere Application Server

WebSphere Liberty Profile

WebSphere Message Broker

Windows Services

This section explains how to integrate the AppMon Agent into applications and instrument the

applications to run the AppMon Agent when the application runs. Integration with these applications enables the AppMon Server to collect PurePath and performance statistics about transactions as they are monitored. Integration of the AppMon Agent with applications only associates the AppMon Agent with the application. The AppMon Agent does not do anything until it is invoked by the AppMon Server as a part of specific AppMon Sessions.

This section is organized to provide specific platform instructions for supported application servers, and then platform instructions for instrumenting Java applications and running the AppMon Agent to diagnose applications implemented as Windows services.

Note

On Linux and UNIX, the Agent library typically needs permissions 644, while on zOS and HP-UX, it needs permissions 755.

Application Environment Configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

LAMPP Apache and PHP

XAMPP Apache and PHP

XAMPP Apache-only

IIS

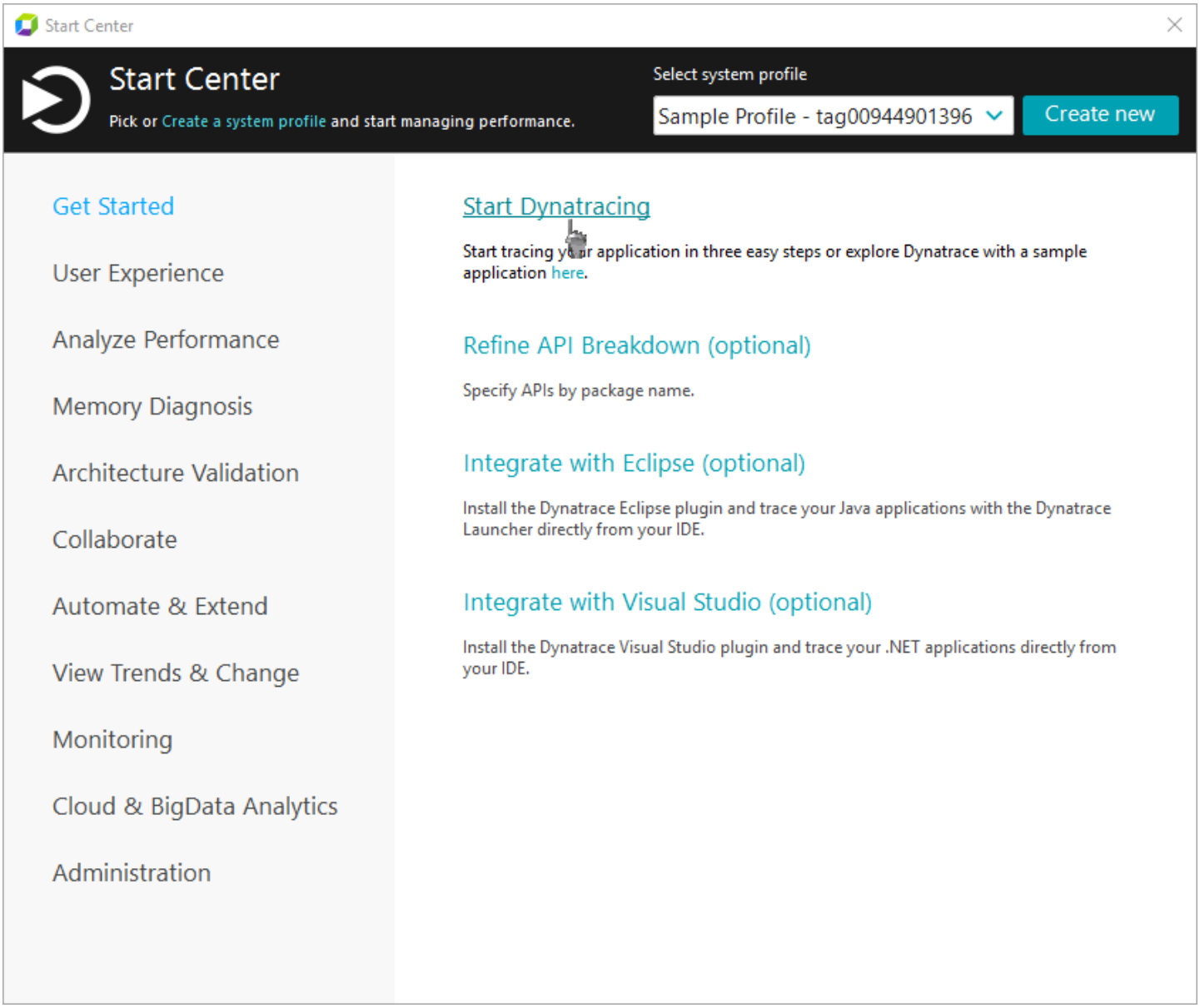
Application server / Tomcat / Alfresco

Application server / IIS/ASP.NET / DotNetNuke

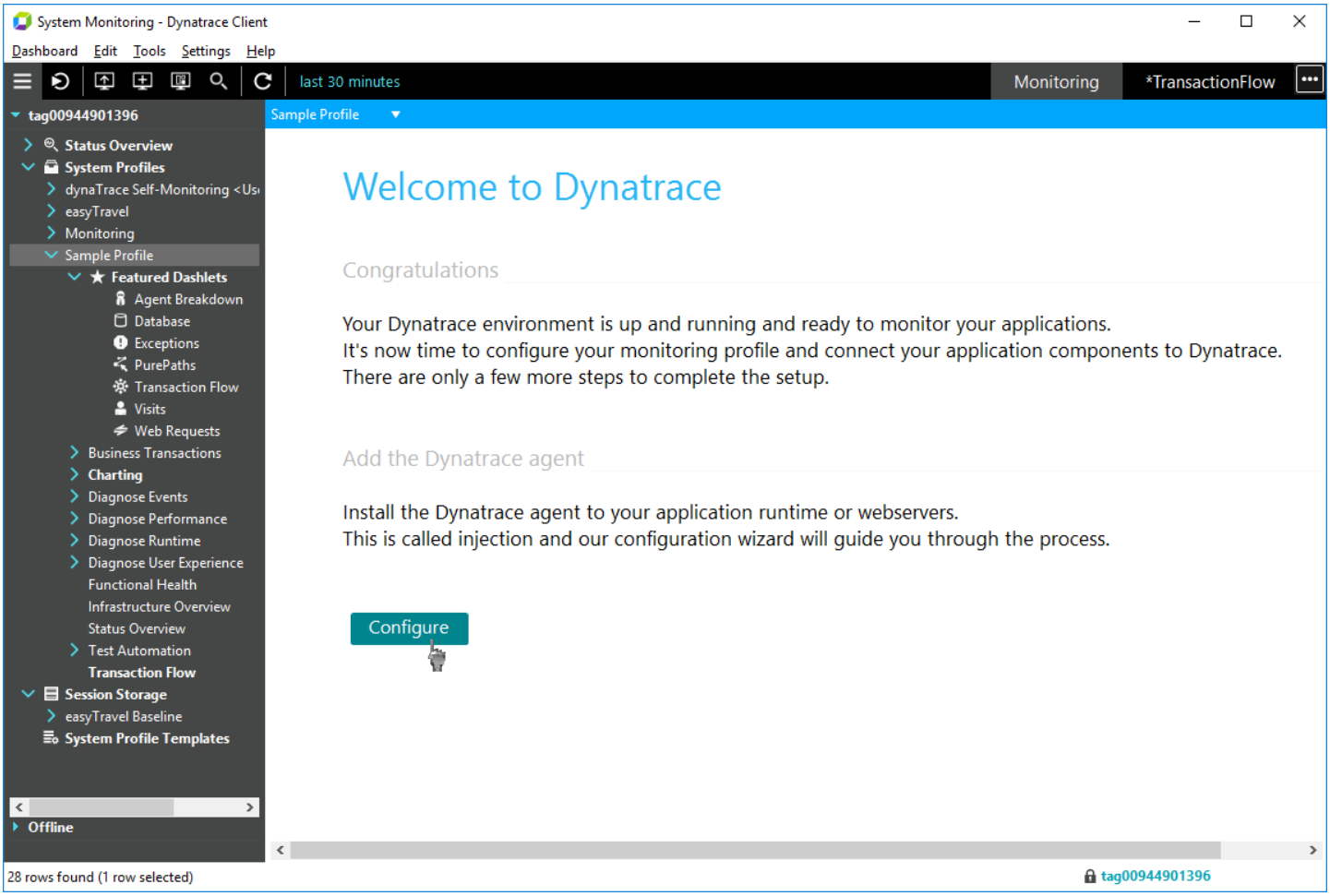
To access the **Application Environment Configuration** dialog, click the **Configure** button on the **Welcome** screen of a new AppMon installation or click **Start Dynatracing** in the **Get Started** section of the **Start Center** dialog box.

Note

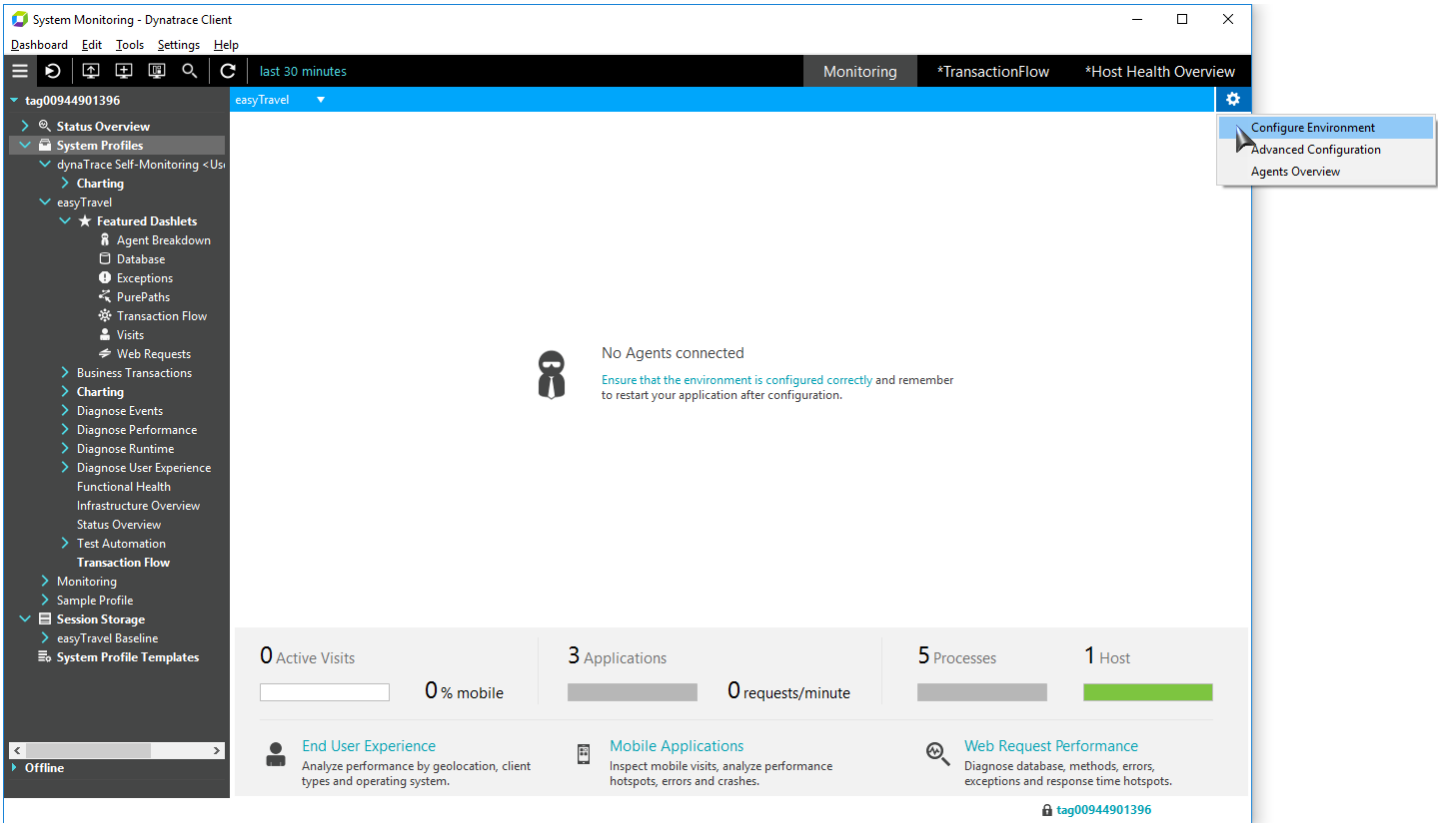
After data from a successfully instrumented app appears in your **System Profile**, the **Welcome** screen will no longer appear when starting the Client. You can, however, continue to access the **Application Environment Configuration** dialog from **Start Center** dialog box, or from the **Configuration** menu.



Access to the Application Environment Configuration dialog box via the Start Center



Access to the Application Environment Configuration dialog box via the Welcome Screen



Access to the Application Environment Configuration dialog box via the Configuration menu

In the **Application Environment Configuration** dialog, select the application type, and tier type you want to add. The **Add Tier** wizard appears, displaying a menu specific to each tier type.

For examples using the **Application Environment Configuration** dialog box and **Add Tier** wizard to configure specific application environments, refer to the following sample instrumentation instructions.

LAMPP Apache and PHP

See LAMPP Apache and PHP Sample Instrumentation .

XAMPP Apache and PHP

See XAMPP Apache and PHP sample instrumentation .

XAMPP Apache-only

See XAMPP Apache sample instrumentation.

IIS

See IIS sample instrumentation.

Application server / Tomcat / Alfresco

See Application server - Tomcat - Alfresco sample instrumentation.

Application server / IIS/ASP.NET / DotNetNuke

See Application server - ASP.NET - DotNetNuke sample instrumentation.

LAMPP Apache and PHP sample instrumentation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Other sample instrumentations

XAMPP Apache and PHP sample instrumentation

XAMPP Apache sample instrumentation

IIS sample instrumentation

Application server - Tomcat - Alfresco sample instrumentation

Application server - ASP.NET - DotNetNuke sample instrumentation

If you use XAMPP / LAMPP stack and the Apache Web server and PHP contained, then the entire instrumentation process takes seconds.

Download XAMPP/LAMPP and install to `/opt`, unless you already have them.

To perform the configuration:

1. Open the **Application Environment Configuration** dialog.

   Note

   JavaScript Agent is injected automatically to HTML code of your website, so you don't have to perform additional configuration for the web application UEM, this is enabled by default. Make sure your license has some UEM volume for this to work.

2. Select the application type as PHP.

3. In the **PHP Engine on Webserver** step, click **Add**. The **Configure PHP Engine on Webserver** dialog appears.

4. Select **PHP Engine on Apache**. Keep the default name for it.

5. Select the machine, where the Apache installed and its architecture.

6. Replace the `Name` and `Server` argument values in the `/%DT_HOME%/agent/conf/dtwsagent.ini` file with the text from the dialog, to change the Agent name and Collector to connect to.

7. Add the lines from the dialog to the end of the `httpd.conf` and `php.ini` files in the `/opt/lampp/etc` directory.

8. Restart the Dynatrace Web Server Agent and the Web server (e.g. from XAMPP Control) and navigate to the site served by the latter.

> **Note**
>
> For a Web server with PHP instrumentation you need three components: a running Web Server (master) Agent *and* the actual Agent LoadModule for the Web server and an Agent extension hooked into PHP.
>
> If the feedback screen says “Successfully connected 3 Agents” it means instrumentation should work.
>
> If it says “Successfully connected 1 Agent” then only the Web Server Agent runs. Something went wrong with the Apache LoadModule and the PHP extension.
>
> If the feedback screen doesn´t report any success then there is a problem with the Web Server Agent and as a consequence the state of the Apache LoadModule / PHP extension injection is unknown.

9. Close the summary dialog.

0. Browse the site to make sure that components are actually loaded.

XAMPP Apache sample instrumentation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Other sample instrumentations

LAMPP Apache and PHP sample instrumentation

XAMPP Apache and PHP sample instrumentation

IIS sample instrumentation

Application server - Tomcat - Alfresco sample instrumentation

Application server - ASP.NET - DotNetNuke sample instrumentation

Instrumenting an Apache Web server is a matter of seconds. You can use the just unzip XAMPP stack on Windows.

Download the applicable 32-bit 7Zip archive, open it and drag the `XAMPP` folder to the `C:` drive.

Note

You can set the environment variable `%DT_HOME%`, so you can reference it in paths. For example, set a default 64-bit AppMon installation with `setx DT_HOME "C:\Program Files\Dynatrace\DynaTrace 7.0"`. You can also do it interactively in the System control panel.

To perform the configuration:

1. Open the **Application Environment Configuration** dialog.

> **Note**
>
> JavaScript Agent is injected automatically to HTML code of your website, so you don't have to perform additional configuration for the web application UEM, this is enabled by default. Make sure your license has some UEM volume for this to work.

2. Select the application type as Web / Java / .NET.
3. In the **Webserver or PHP Engine** step, click **Add**. The **Configure Webserver or PHP Engine** dialog appears.
4. In the **Web Server** list, select **Apache**. Keep the default name for it.
5. Select the machine, where the Apache installed and its architecture.
6. Replace the `Name` and `Server` argument values in the `/%DT_HOME%/agent/conf/dtwsagent.ini` file with the text from the dialog, to change the Agent name and Collector to connect to.
7. Add the lines from the dialog to the end of the `xampp/apache/conf/httpd.conf file`.
8. Restart the Dynatrace Web Server Agent and the web server (e.g. from XAMPP Control).

Note

For a Web server with PHP instrumentation you need three components: a running Web Server (master) Agent *and* the actual Agent LoadModule for the Web server and an Agent extension hooked into PHP.

If the feedback screen says “Successfully connected 3 Agents” it means instrumentation should work.

If it says “Successfully connected 1 Agent” then only the Web Server Agent runs. Something went wrong with the Apache LoadModule and the PHP extension.

If the feedback screen doesn´t report any success then there is a problem with the Web Server Agent and as a consequence the state of the Apache LoadModule / PHP extension injection is unknown.

9. Close the summary dialog.

0. Browse the site to make sure that components are actually loaded.

XAMPP Apache and PHP sample instrumentation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Other sample instrumentations

LAMPP Apache and PHP sample instrumentation

XAMPP Apache sample instrumentation

IIS sample instrumentation

Application server - Tomcat - Alfresco sample instrumentation

Application server - ASP.NET - DotNetNuke sample instrumentation

Instrumenting an Apache Web server, including PHP, is a matter of seconds. You can use the just-unzip XAMPP stack on Windows.

Download the applicable 32-bit 7Zip archive, open it and drag the XAMPP folder to the C: drive.

Note

You can set the environment variable `%DT_HOME%`, so you can reference it in paths. For example, set a default 64-bit AppMon installation with `setx DT_HOME "C:\Program Files\Dynatrace\Dynatrace 7.0"`. You can also do it interactively in the System control panel.

To perform the configuration:

1. Open the **Application Environment Configuration** dialog.

> **Note**
>
> JavaScript Agent is injected automatically to HTML code of your website, so you don't have to perform additional configuration for the web application UEM, this is enabled by default. Make sure your license has some UEM volume for this to work.

2. Select the application type as PHP.
3. In the **PHP Engine on Webserver** step, click **Add** . The **Configure PHP Engine on Webserver** dialog appears.
4. Select **PHP Engine on Apache** . Keep the default name for it.
5. Select the machine, where the Apache installed and its architecture.
6. Replace the `Name` and `Server` argument values in the `/%DT_HOME%/agent/conf/dtwsagent.ini` file with the text from the dialog, to change the Agent name and Collector to connect to.
7. Add the line from the **Apache Web Server Module Configuration** pane as last module in the `xampp/apache/conf/httpd.conf` file, and add the line from the **PHP Extension Configuration** pane as last extension or even at the end of the `xampp/php/php.ini` file.
8. Restart the Dynatrace Web Server Agent and the Web server (e.g. from XAMPP Control) and navigate to the site served by the latter.

Note

For a Web server with PHP instrumentation you need three components: a running Web Server (master) Agent *and* the actual Agent LoadModule for the Web server and an Agent extension hooked into PHP.

If the feedback screen says “Successfully connected 3 Agents” it means instrumentation should work.

If it says “Successfully connected 1 Agent” then only the Web Server Agent runs. Something went wrong with the Apache LoadModule and the PHP extension.

If the feedback screen doesn´t report any success then there is a problem with the Web Server Agent and as a consequence the state of the Apache LoadModule / PHP extension injection is unknown.

9. Close the summary dialog.

0. Browse the site to make sure that components are actually loaded.

IIS sample instrumentation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Other sample instrumentations

LAMPP Apache and PHP sample instrumentation

XAMPP Apache and PHP sample instrumentation

XAMPP Apache sample instrumentation

Application server - Tomcat - Alfresco sample instrumentation

Application server - ASP.NET - DotNetNuke sample instrumentation

Going through Instrumenting an IIS Manager is a matter of seconds.

Note

You can set the environment variable `%DT_HOME%`, so you can reference it in paths. For example, for a default 64 bit AppMon installation, it will be `setx DT_HOME "C:\Program Files\dynaTrace\dynaTrace 6.5"`. You can also do it interactively in the System control panel.

To perform the configuration:

1. Open the **Application Environment Configuration** dialog.

> **Note**
>
> JavaScript Agent is injected automatically to HTML code of your website, so you don't have to perform additional configuration for the web application UEM, this is enabled by default. Make sure your license has some UEM volume for this to work.

2. Select the application type as Web / Java / .NET.
3. In the **Webserver or PHP Engine** step, click **Add** . The **Configure Webserver or PHP Engine dialog** appears.
4. In the **Web Server** list, select **Microsoft IIS**. Keep the default name for it.
5. Select the machine, where the IIS installed and its architecture.
6. Replace the values for the `Name` and `Server` arguments in the `%DT_HOME%/agent/conf/dtwsagent.ini` file with the text from the dialog, to change the name of the Agent and Collector to connect to.

   Now, activate Dynatrace web server module for yours site.
7. Open the IIS Manager.
8. In the left panel, select the site you want to monitor.
9. In the **IIS** section of the middle panel, click **Modules**.
0. Check for the Dynatrace modules. If they are missing, in the right panel, click **Configure Native Modules**. If they are here, go to the step 12.
1. In the **Configure Native Modules** dialog, select Dynatrace modules. You can select both 32- and 64-bit select Agents.
2. Restart the Dynatrace Web Server Agent service in Windows Services.
3. Restart or Recycle your IIS Application Pool in the IIS Manager.

Note

For a Web server instrumentation you need *two* components: a running Web Server (master) Agent service *and* the actual Agent module loaded into the Web server.

If the feedback screen says “Successfully connected 2 Agents” it means instrumentation should work.

If it says “Successfully connected 1 Agent” then only the Web Server Agent service runs. Something went wrong with the LoadModule.

If the feedback screen doesn´t report any success then there is a problem with the Web Server Agent service and as a consequence the state of the LoadModule injection (“Agent 2”) is unknown.

4. Close the summary dialog.

5. Browse the site to make sure that components are actually loaded.

Application server - Tomcat - Alfresco sample instrumentation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Other sample instrumentations

LAMPP Apache and PHP sample instrumentation

XAMPP Apache and PHP sample instrumentation

XAMPP Apache sample instrumentation

IIS sample instrumentation

Application server - ASP.NET - DotNetNuke sample instrumentation

If you have a default setup, the entire instrumentation process takes seconds.

Add the configuration string to the end of `JAVA_OPTS` in the `<ALFRESCO_HOME>\alfresco.bat` file. See the example below. Note that this is **not** a copy/paste example, but only a fragment with a sample configuration string!

```
set JAVA_OPTS=%JAVA_OPTS% -XX:CompileCommand=exclude, ... -agentpath:"C:\Program Files\dy
```

To perform the configuration:

1. Open the **Application Environment Configuration** dialog.

> **Note**
>
> JavaScript Agent is injected automatically to HTML code of your website, so you don't have to perform additional configuration for the web application UEM, this is enabled by default. Make sure your license has some UEM volume for this to work.

2. Select the application type as Web / Java / .NET.
3. In the **Application Servers** step, click **Add**. The **Configure Application Servers** dialog appears.
4. From the **Application Server** list, select **Apache Tomcat 5+**, and set its name as **Alfresco**.
5. Select the machine, where the Alfresco is installed, and its environment.
6. Add the configuration string from the dialog to the end of the set `JAVA_OPTS=%JAVA_OPTS% ... -agentpath...` in the `<ALFRESCO_HOME>\alfresco.bat` file.
7. Restart Alfresco to connect to the AppMon.
8. Close the summary dialog.

Application server - ASP.NET - DotNetNuke sample instrumentation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Other sample instrumentations

LAMPP Apache and PHP sample instrumentation

XAMPP Apache and PHP sample instrumentation

XAMPP Apache sample instrumentation

IIS sample instrumentation

Application server - Tomcat - Alfresco sample instrumentation

Going through instrumenting a .NET is a matter of seconds.

To perform the configuration:

1. Open the **Application Environment Configuration** dialog.

   Note

   JavaScript Agent is injected automatically to HTML code of your website, so you don't have to perform additional configuration for the web application UEM, this is enabled by default. Make sure your license has some UEM volume for this to work.

2. Select the application type as Web / Java / .NET.

3. In the **Application Servers** step, click **Add**. The **Configure Application Servers** dialog appears.

4. From the **Application Server** list, select **Microsoft IIS/ASP.NET 6+**, and set its name as **DotNetNuke**.

5. Select the machine, where the DotNetNuke installed and its architecture, and click **Next**.

   Now, add values from the dialog to Dynatrace Agent Configuration for .NET.

6. Open your site in the browser to make sure the application is started.

7. Launch the **Dynatrace .NET Agent Configuration** application.

8. In the **Dynatrace .NET Agent Configuration** application, click **+**. The **Create .NET Agent Configuration** dialog appears.

9. Select the DNN w3wp.

0. Enter the values from the Configure Application Servers dialog.

   DNN will appear in the lower pane, but will not be instrumented until you restart the DNN application pool.

1. Switch back to the **Configure Application Servers** dialog, and click **Next**.

2. Restart the DNN application to instrument the DNN.

   > **Note**
   >
   > If the feedback screen says “Successfully connected 1 Agent” it means instrumentation should work.

3. Close the summary dialog.

4. Browse the site to make sure that components are actually loaded.

Cassandra

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installation

Usage

Supported client and server versions

Related pages

Cassandra sensor

System requirements

Java Agent configuration

Transaction Flow dashlet

Database dashlet

Database Hotspots dashlet

Downloads

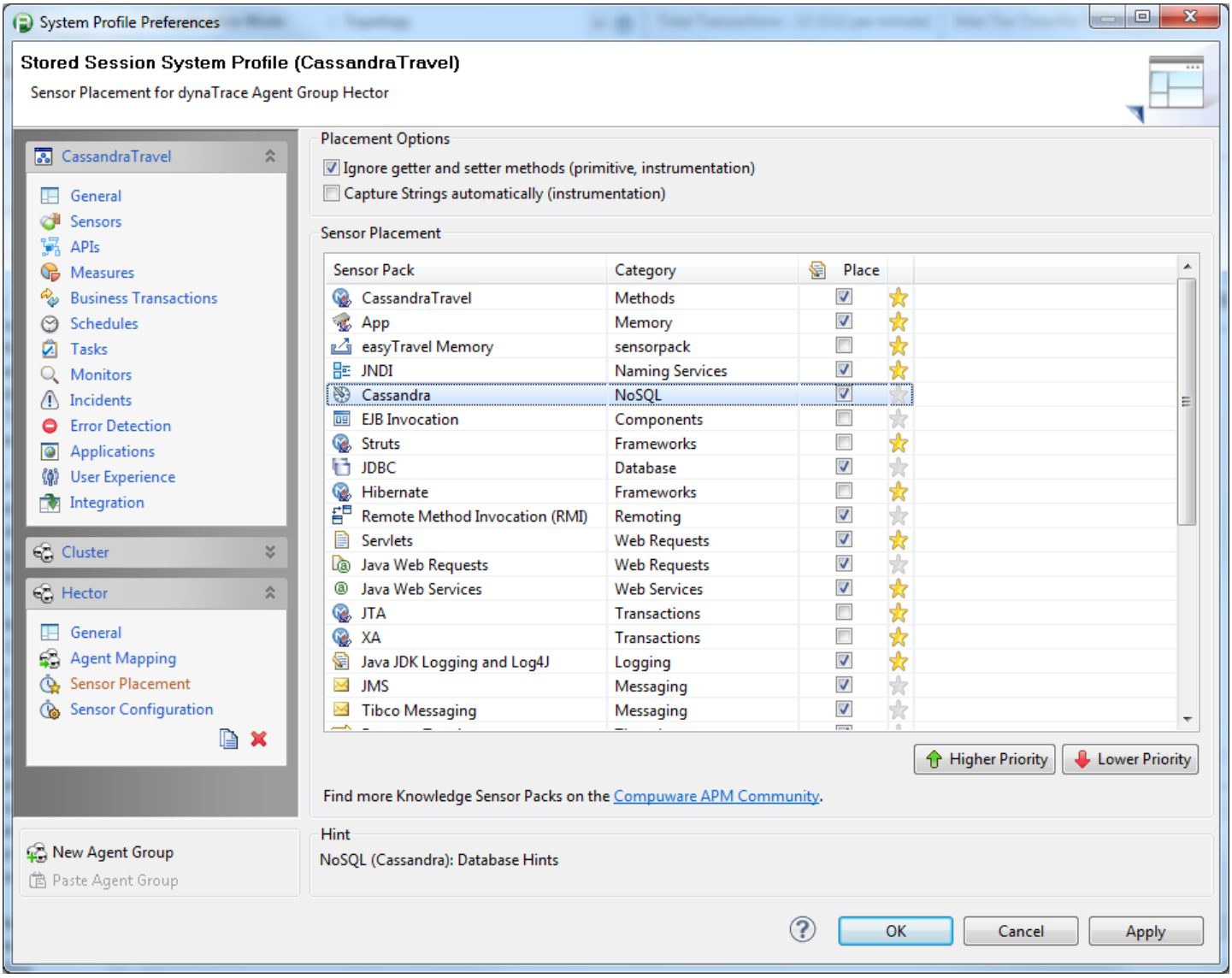
Apache Cassandra FastPack

Installation

Adapt your System Profile

For instrumentation of the Cassandra client API, you have to inject the AppMon Agent, and the Cassandra Sensor must be placed in those Agent Groups that make calls to Cassandra. The Sensor needs to be explicitly placed in new Agent Groups as shown in the following figure.

Cassandra Sensor Placement



Cassandra client

For analysis of your Cassandra client code, instrument your application by injecting the AppMon Agent (see Java Agent Configuration).

Cassandra server

To trace Cassandra calls from client to server, add a new Agent Group for the Cassandra server node. Make sure that this Agent Group has the Cassandra Sensor placed. As Cassandra does a lot of threading, you may want to disable the Thread Activation Sensor to reduce the amount of data collected. However, in order to be able to see Cassandra CQL3 protocol based statement execution, the **Executor Tagging** sensor needs to be active.

Then add the AppMon Agent parameter to the JVM_OPTS in the Cassandra server configuration file and restart your nodes.

Usage

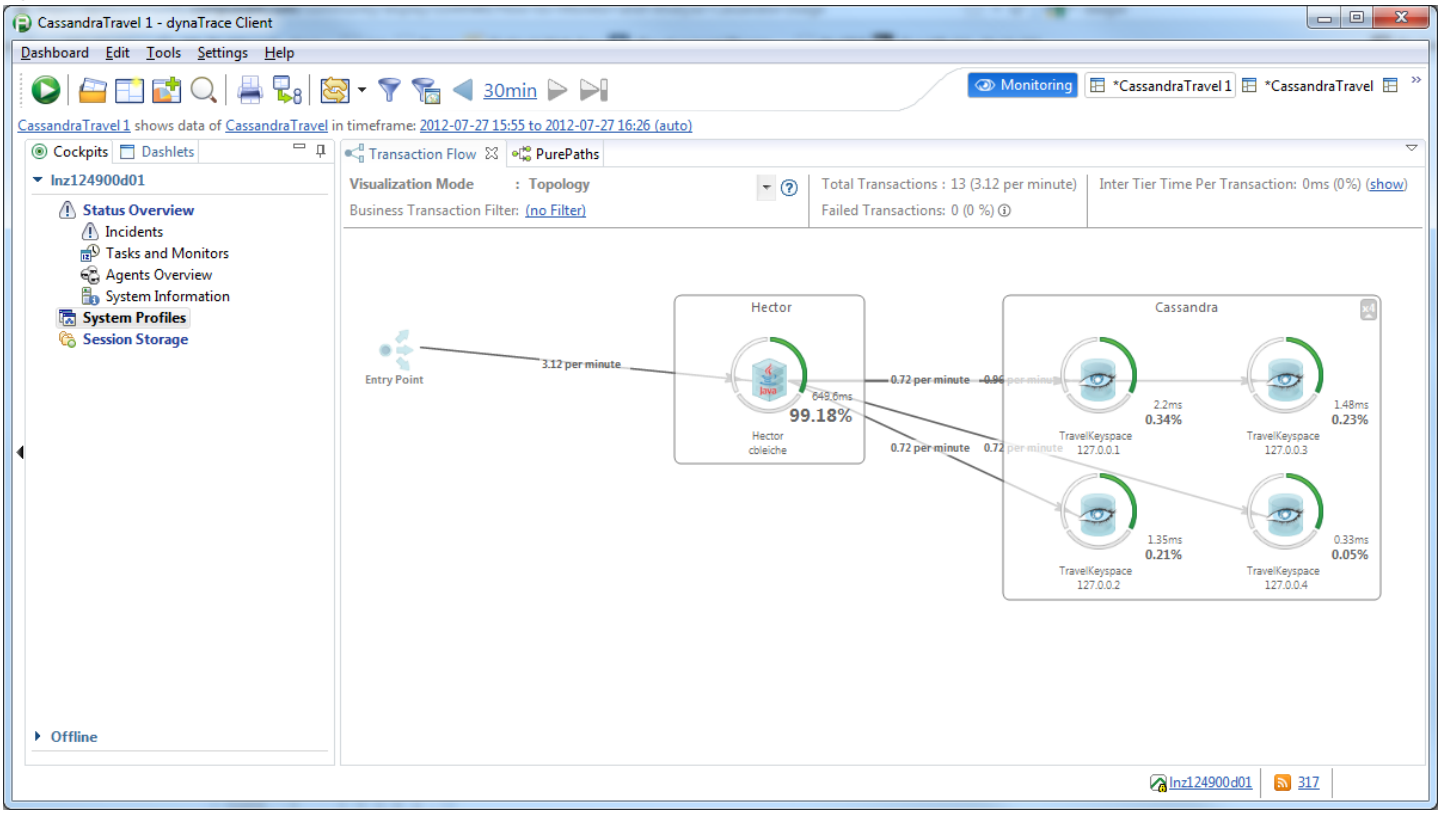
Transaction Flow

In the Transaction Flow Dashlet Dashlet, Cassandra calls are visualized like database calls. In contrast to traditional databases, Cassandra will nearly always run in a cluster (meaning multiple server nodes). Consequently, you will see multiple database nodes with the name of the Keyspace that you execute against and the host/port.

- Execution time contribution
- Number of calls (in response time mode)
- Number of calls/min (in topology mode)

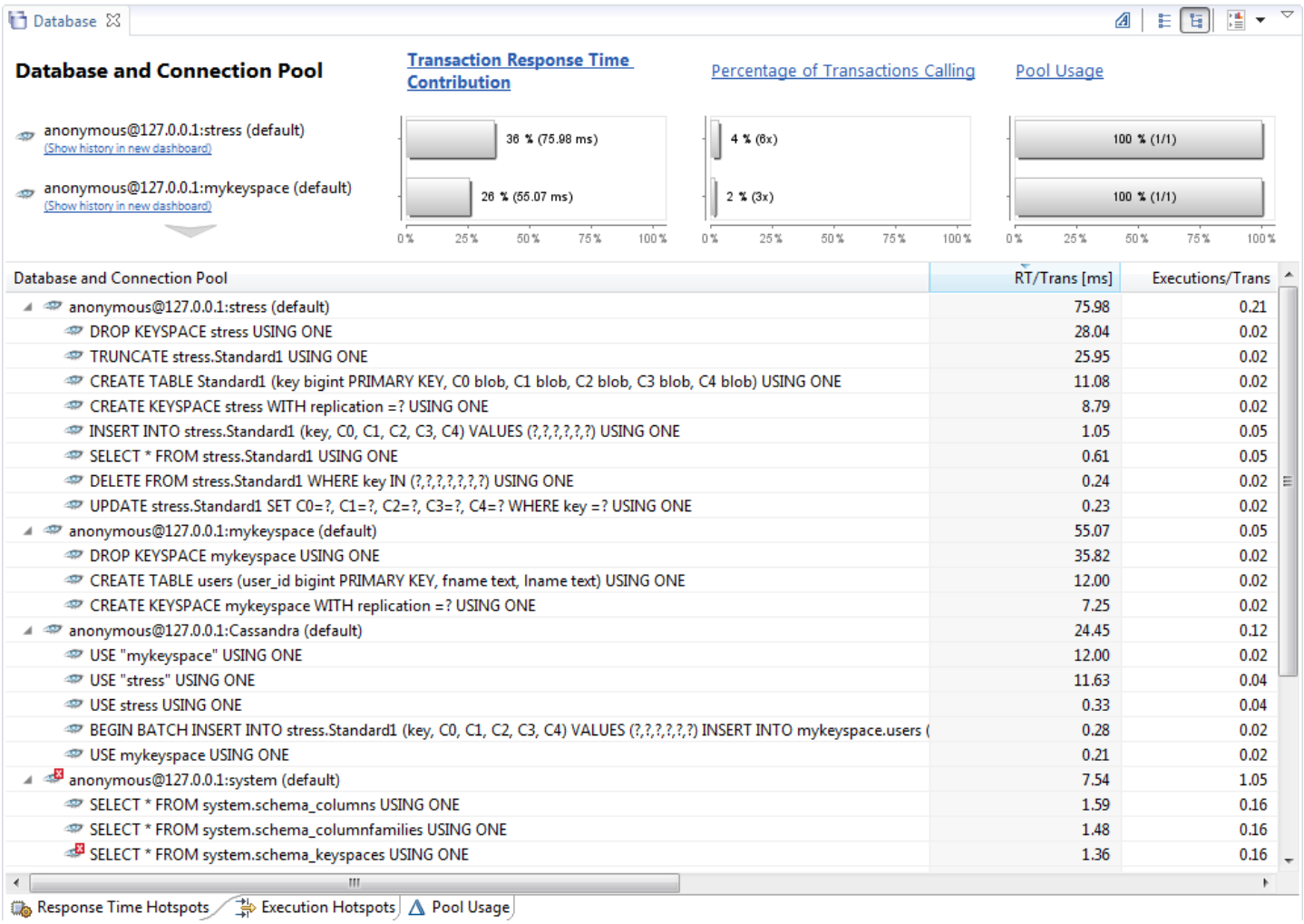
When you hover over a Cassandra database node, you can also look at the specific Cassandra calls. You can do the same by drilling down from a particular JVM to the database.

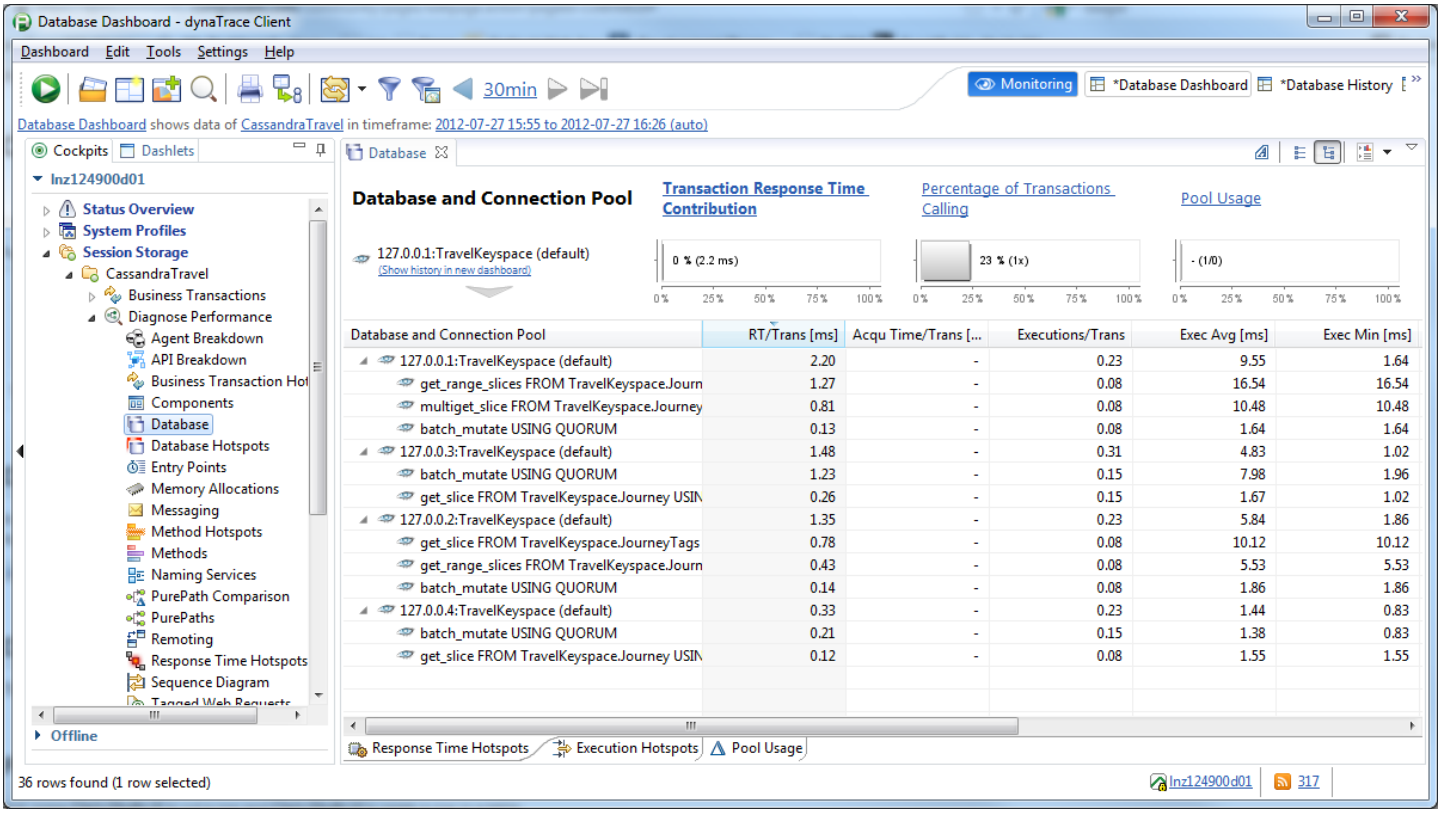
Typical Cassandra Transaction Flow



Database

In the Database Dashlet Dashlet, Cassandra calls based on the Thrift protocol are displayed in a “SQL-like” form. Cassandra calls, based on the native CQL3 protocol, directly visualize CQL statements. Each CassandraNode/Keyspace combination is visualized as a separate Database Pool, making it easy to check, e.g., Access Distribution. For suspicious statements, the actual execution and the attached details can be reviewed.


Database
Database and Connection Pool
Transaction Response Time Contribution
Percentage of Transactions Calling
Pool Usage
anonymous@127.0.0.1:stress (default)
(Show history in new dashboard)
anonymous@127.0.0.1:mykeyspace (default)
(Show history in new dashboard)
36 % (75.98 ms)
26 % (55.07 ms)
4 % (6x)
2 % (3x)
100 % (1/1)
100 % (1/1)
0 % 25 % 50 % 75 % 100 %
Database and Connection Pool
RT/Trans [ms]
Executions/Trans
anonymous@127.0.0.1:stress (default) 75.98 0.21
DROP KEYSPACE stress USING ONE 28.04 0.02
TRUNCATE stress.Standard1 USING ONE 25.95 0.02
CREATE TABLE Standard1 (key bigint PRIMARY KEY, C0 blob, C1 blob, C2 blob, C3 blob, C4 blob) USING ONE 11.08 0.02
CREATE KEYSPACE stress WITH replication =? USING ONE 8.79 0.02
INSERT INTO stress.Standard1 (key, C0, C1, C2, C3, C4) VALUES (?,?,?,?,?,?) USING ONE 1.05 0.05
SELECT * FROM stress.Standard1 USING ONE 0.61 0.05
DELETE FROM stress.Standard1 WHERE key IN (?,?,?,?,?,?,?) USING ONE 0.24 0.02
UPDATE stress.Standard1 SET C0=?, C1=?, C2=?, C3=?, C4=? WHERE key =? USING ONE 0.23 0.02
anonymous@127.0.0.1:mykeyspace (default) 55.07 0.05
DROP KEYSPACE mykeyspace USING ONE 35.82 0.02
CREATE TABLE users (user_id bigint PRIMARY KEY, fname text, lname text) USING ONE 12.00 0.02
CREATE KEYSPACE mykeyspace WITH replication =? USING ONE 7.25 0.02
anonymous@127.0.0.1:Cassandra (default) 24.45 0.12
USE "mykeyspace" USING ONE 12.00 0.02
USE "stress" USING ONE 11.63 0.04
USE stress USING ONE 0.33 0.04
BEGIN BATCH INSERT INTO stress.Standard1 (key, C0, C1, C2, C3, C4) VALUES (?,?,?,?,?,?) INSERT INTO mykeyspace.users (0.28 0.02
USE mykeyspace USING ONE 0.21 0.02
anonymous@127.0.0.1:system (default) 7.54 1.05
SELECT * FROM system.schema_columns USING ONE 1.59 0.16
SELECT * FROM system.schema_columnfamilies USING ONE 1.48 0.16
SELECT * FROM system.schema_keyspaces USING ONE 1.36 0.16
Response Time Hotspots
Execution Hotspots
Pool Usage


Database Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*Database Dashboard
*Database History
Database Dashboard shows data of CassandraTravel in timeframe: 2012-07-27 15:55 to 2012-07-27 16:26 (auto)
Cockpits
Dashlets
lnz124900d01
Status Overview
System Profiles
Session Storage
CassandraTravel
Business Transactions
Diagnose Performance
Agent Breakdown
API Breakdown
Business Transaction Hot
Components
Database
Database Hotspots
Entry Points
Memory Allocations
Messaging
Method Hotspots
Methods
Naming Services
PurePath Comparison
PurePaths
Remoting
Response Time Hotspots
Sequence Diagram
Offline
Database
Database and Connection Pool
Transaction Response Time Contribution
Percentage of Transactions Calling
Pool Usage
127.0.0.1:TravelKeyspace (default)
(Show history in new dashboard)
0 % (2.2 ms)
23 % (1x)
- (1/0)
Database and Connection Pool
RT/Trans [ms]
Acqu Time/Trans [...
Executions/Trans
Exec Avg [ms]
Exec Min [ms]
127.0.0.1:TravelKeyspace (default) 2.20 - 0.23 9.55 1.64
get_range_slices FROM TravelKeyspace.Journ 1.27 - 0.08 16.54 16.54
multiget_slice FROM TravelKeyspace.Journey 0.81 - 0.08 10.48 10.48
batch_mutate USING QUORUM 0.13 - 0.08 1.64 1.64
127.0.0.3:TravelKeyspace (default) 1.48 - 0.31 4.83 1.02
batch_mutate USING QUORUM 1.23 - 0.15 7.98 1.96
get_slice FROM TravelKeyspace.Journey USIN 0.26 - 0.15 1.67 1.02
127.0.0.2:TravelKeyspace (default) 1.35 - 0.23 5.84 1.86
get_slice FROM TravelKeyspace.JourneyTags 0.78 - 0.08 10.12 10.12
get_range_slices FROM TravelKeyspace.Journ 0.43 - 0.08 5.53 5.53
batch_mutate USING QUORUM 0.14 - 0.08 1.86 1.86
127.0.0.4:TravelKeyspace (default) 0.33 - 0.23 1.44 0.83
batch_mutate USING QUORUM 0.21 - 0.15 1.38 0.83
get_slice FROM TravelKeyspace.Journey USIN 0.12 - 0.08 1.55 1.55
Response Time Hotspots
Execution Hotspots
Pool Usage
36 rows found (1 row selected)
lnz124900d01
317

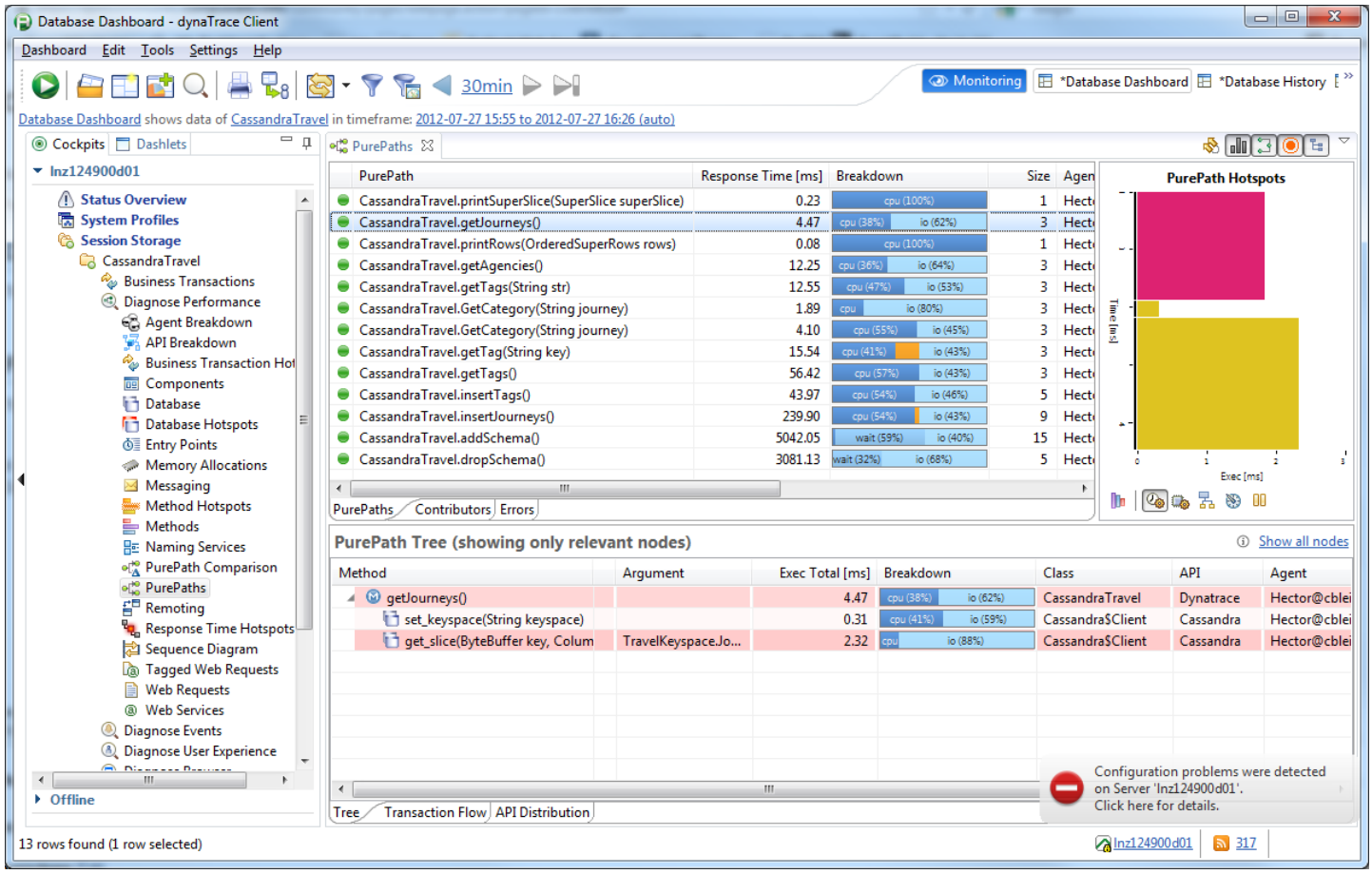
Database Hotspots

The Database Hotspots Dashlet Dashlet shows where your Cassandra calls originate.

PurePaths - thrift based

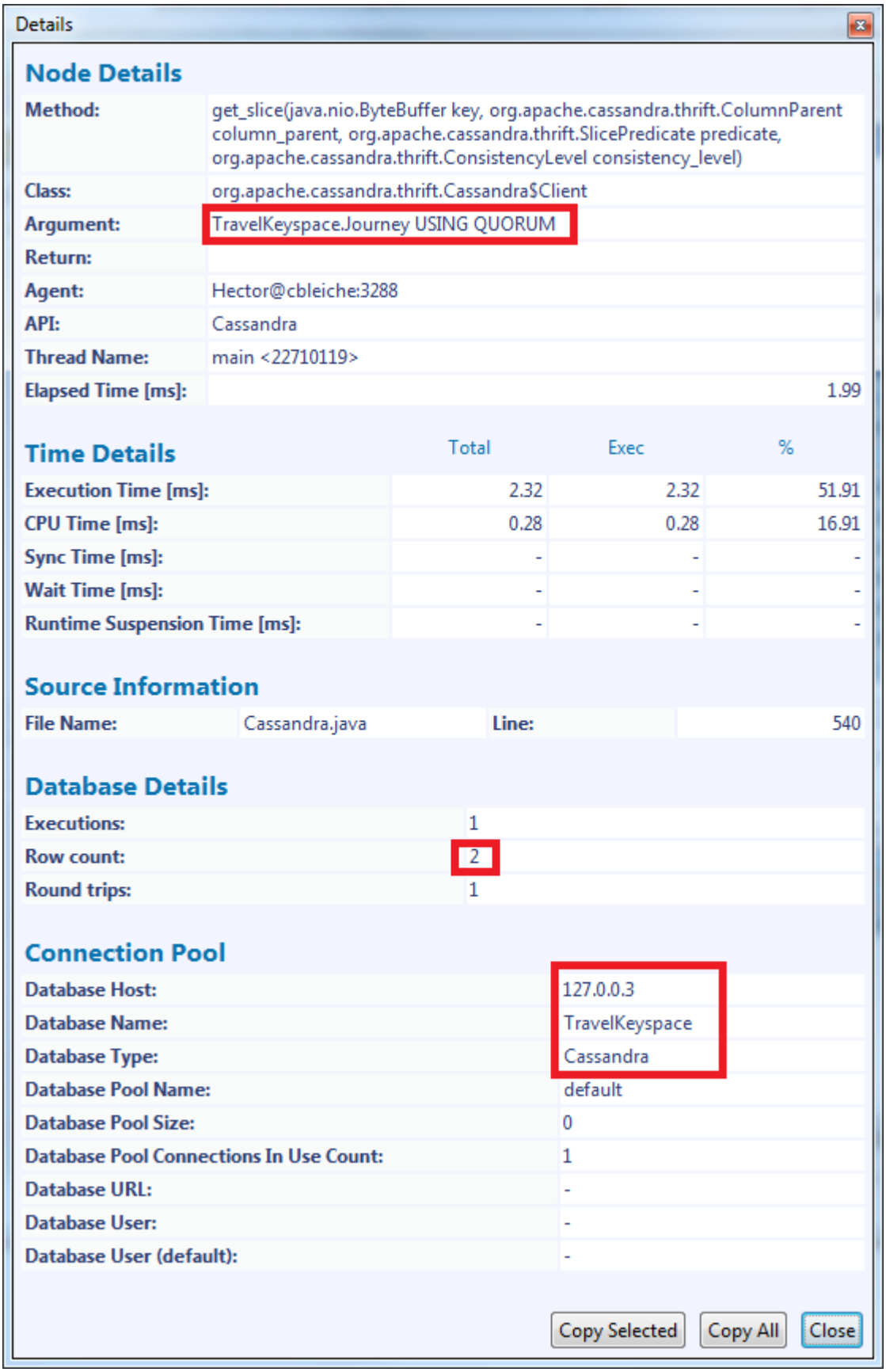
The PurePath shows each Cassandra call by its client method. The **Argument** column shows details like Keyspace, ColumnFamily, and ConsistencyLevel.

Cassandra PurePaths



Statement details (right-click on a statement to view details) provide further information like the number of manipulated rows (row count).

Cassandra Statement Details



Depending on the executed Cassandra method, this value contains (among others):

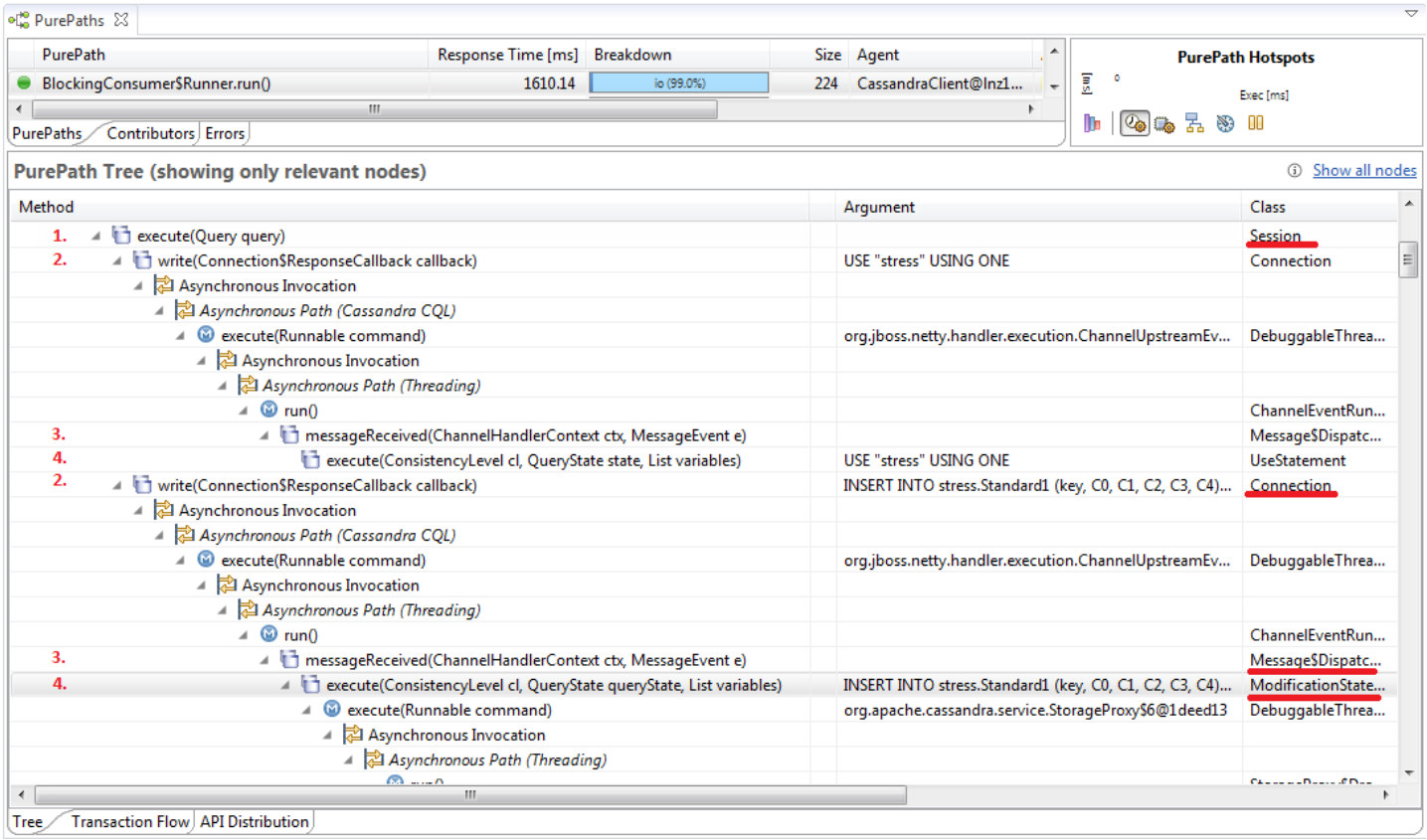
Operation	Comment	Example Visualization in Database Dashlet
batch_mutate	Number of mutations sent to the server	batch_mutate USING QUORUM
get	number of returned rows (0…1)	get FROM TravelKeyspace.Journey USING QUORUM
get_count	returned value	get_count FROM TravelKeyspace.Journey USING QUORUM
get_range_slices	number of returned slices	get_range_slices FROM TravelKeyspace.JourneyTags USING QUORUM
get_slice	number of returned columns	get_slice FROM TravelKeyspace.Journey USING QUORUM
multiget_count	size of returned map	multiget_count FROM TravelKeyspace.Journey USING QUORUM
multiget_size	size of returned map	multiget_slice FROM TravelKeyspace.JourneyTags USING QUORUM

PurePaths - CQL3 binary protocol-based

Depending on the injected agents, client- and/or server-side communication is visualized on the PurePath.

If both sides (client and server) are instrumented, the client method node will start the PurePath on which the database communication takes place (client-side Cassandra CQL3 protocol calls do not automatically start PurePaths). The following figure shows a sample PurePath with client and server instrumented.

CQL3 PurePath - Client and Server Instrumented



The PurePath nodes marked with 2 and 3 specify the CQL3 protocol communication endpoints: methods where CQL3 protocol messages are sent from the client (2 - Connection.write) and received on the server (3 - Message$Dispatcher.messageReceived). All CQL3 protocol messages (such as server startup and statement execution) can be traced via this mechanism.

CQL3 statements may be processed within a Session (node marked with 1) or may be sent directly to Cassandra via Connection.write (nodes marked with 2). CQL statement data is available on both the client (on node Connection.write - 2) and server side (on the respective Statement.execute node - 4). However, the available data on the client slightly differs from that on the server side. The following table shows what data is available on the client and the server and what data is used for further processing in the database dashlet:

Data	Client (1)	Server (1)	Database dashlet uses data from
Connection Pool			
Database Host:	X	X	Server
Database Name:	may not be set	X (2)	Server

Database Type:	Cassandra by default	Cassandra by default	Server
Database Pool Name:	default	default	Client
Database Pool Size:	X	-	Client
Database User (default):	-	X	Server
Database Details			
Bind values	X	-	Client
Row count:	• (3)	only available for SELECT queries	Server
Cassandra specific			
Consistency Level	X	X	Server
Query string	X	X	Server

(1) Client: Datastax 1.0.2 library Server: Cassandra 1.2

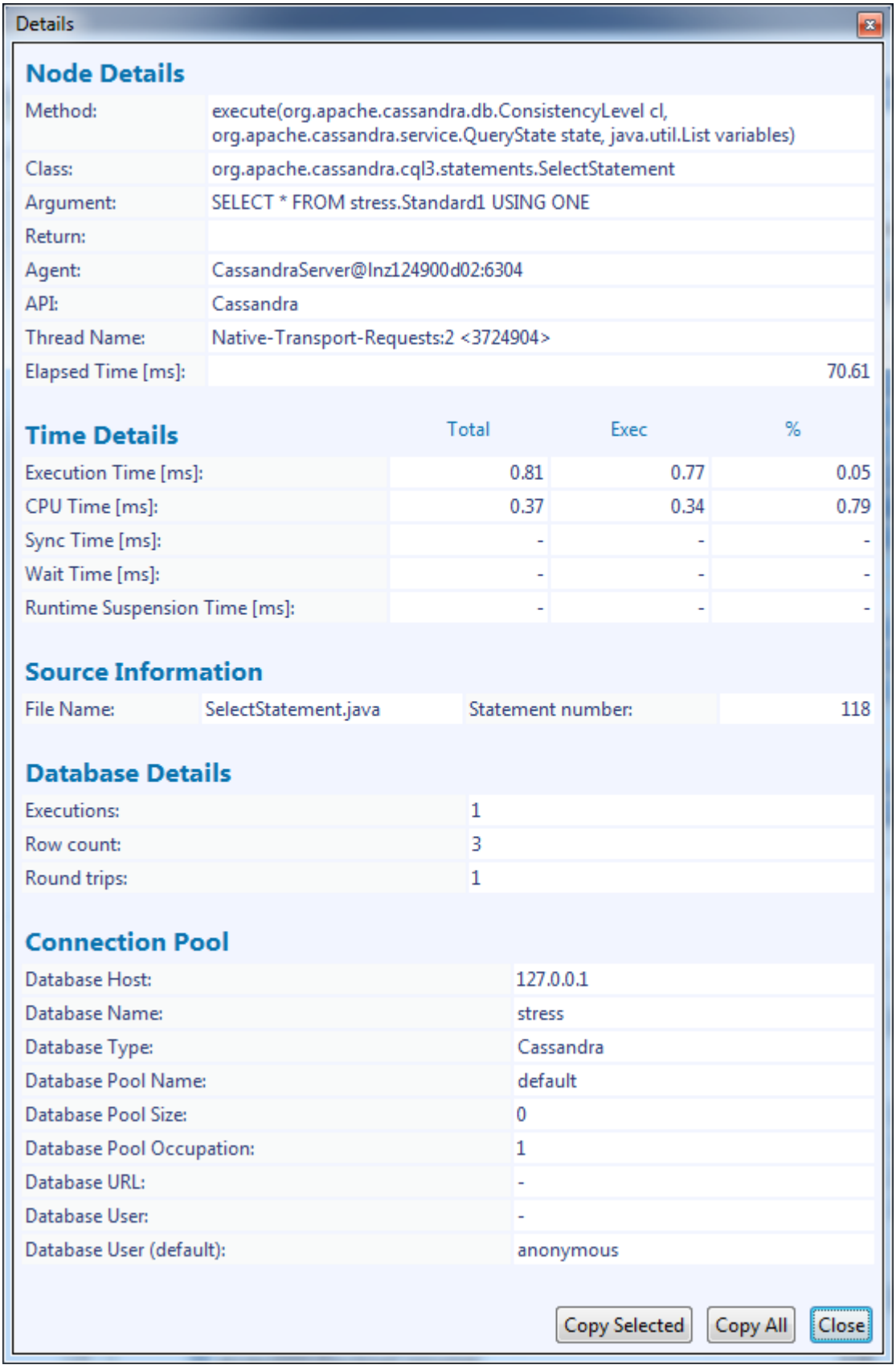
(2) The keyspace is not available for BATCH statements as the statements within a batch may be executed on different keyspaces.

(3) As CQL3 protocol communication is asynchronous by default, record count information is only available if the server side is instrumented.

If only the client or the server side is instrumented, the available data will be used for further processing. Note that this especially affects the execution time of CQL statements, especially when using asynchronous clients as asynchronous callbacks are currently not supported.

The following figures show the data available on client and server nodes respectively.


Details
Node Details
Method: write(com.datastax.driver.core.Connection$ResponseCallback callback)
Class: com.datastax.driver.core.Connection
Argument: INSERT INTO stress.Standard1 (key, C0, C1, C2, C3, C4) VALUES (?,?,?,?,?,?) USING ONE
Return:
Agent: CassandraClient@lnz124900d02:7784
API: Cassandra
Thread Name: Consumer Threads <23977007>
Elapsed Time [ms]: 11.49
Time Details
Total
Exec
%
Execution Time [ms]: 8.18 8.18 0.51
CPU Time [ms]: 0.72 0.72 1.71
Sync Time [ms]: - - -
Wait Time [ms]: - - -
Runtime Suspension Time [ms]: - - -
Source Information
File Name: Connection.java
Statement number: 258
Database Details
Executions: 1
Row count: -
Round trips: 1
Connection Pool
Database Host: 127.0.0.1
Database Name: stress
Database Type: Cassandra
Database Pool Name: default
Database Pool Size: 1
Database Pool Occupation: 1
Database URL: -
Database User: -
Database User (default): -
Copy Selected
Copy All
Close

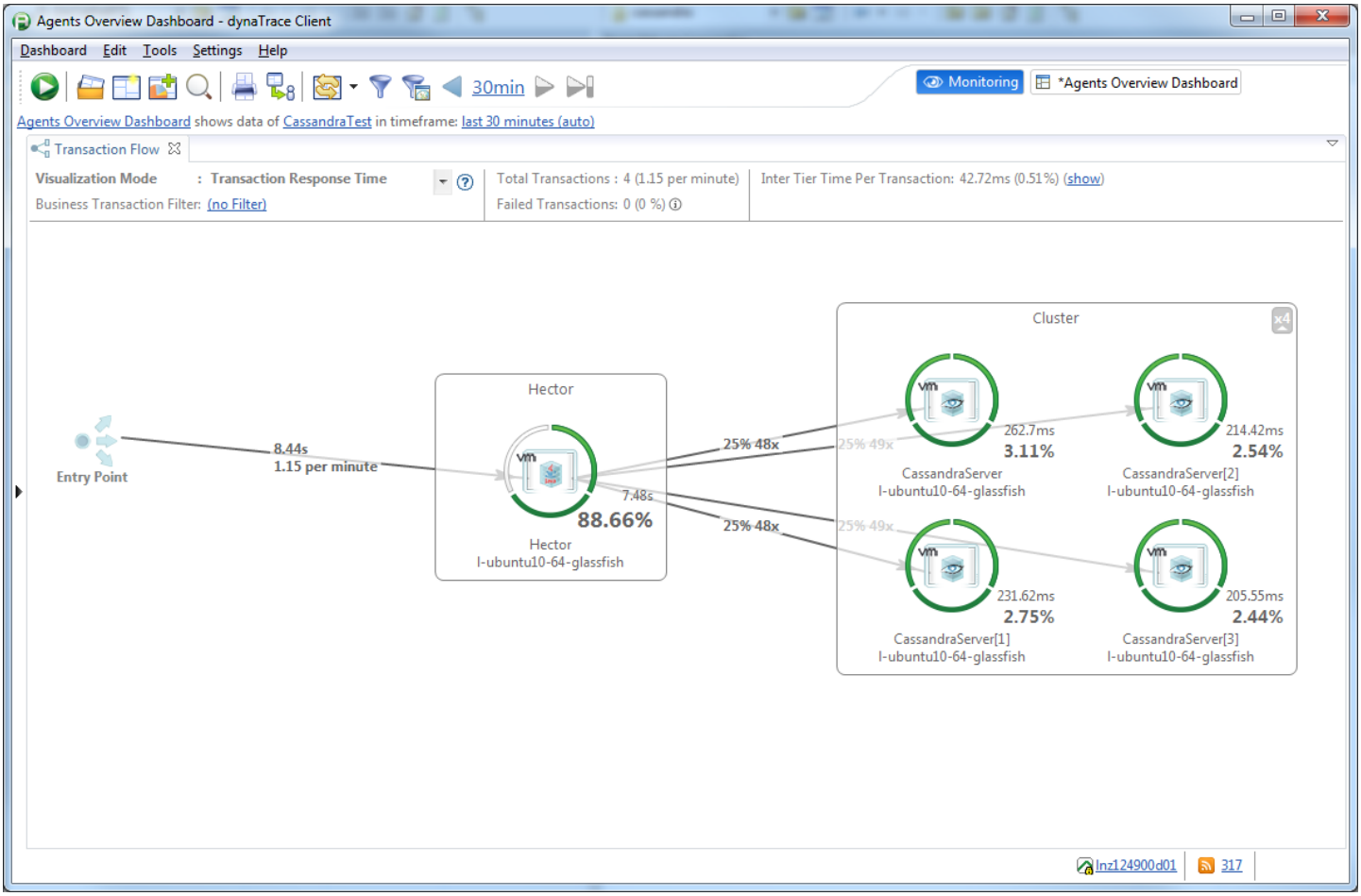


Tracing into the Cassandra server

In addition to understanding Cassandra calls, AppMon can also trace each call into the Cassandra server, making it possible to understand why certain statements were slow. This includes problems such as server-side garbage collection, CPU, disk, and memory problems.

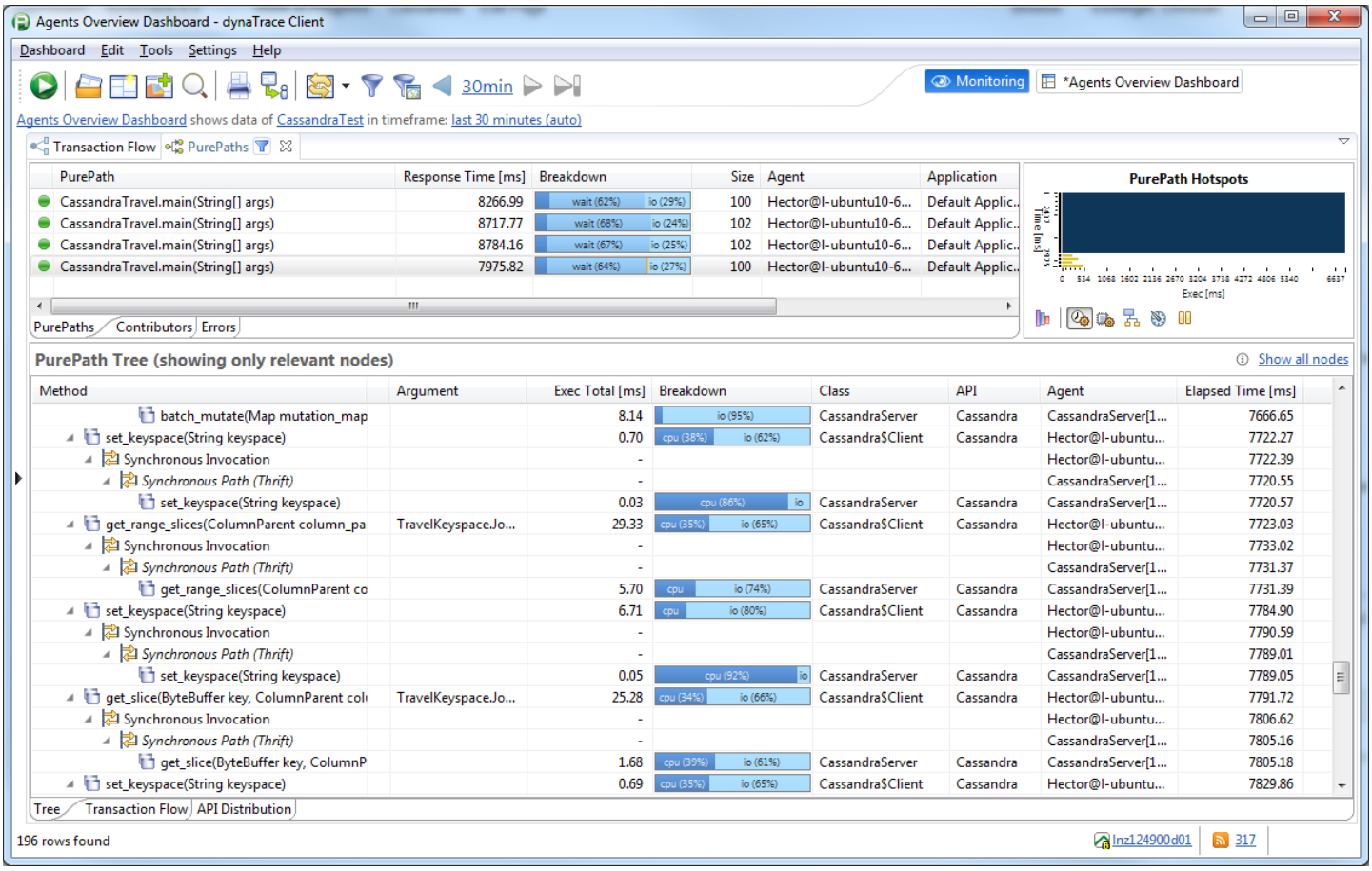
In the Transaction Flow, you will see which applications communicate with your Cassandra cluster.

Client and Server Instrumented



Notice that the Cassandra server nodes will now show up as JVMs with communication coming from your application. You will automatically get host and JVM health and be able to look at Method Hotspot Levels of a particular node right from the Transaction Flow. When you drill down to the PurePath, notice that each call to Cassandra is followed.

Drilldown to PurePath



If you look closely at the Elapsed Time column on the right of the screenshot, you'll notice a surprising ~-2ms difference between Synchronous Invocation and Synchronous Path (Thrift) methods. These slight inaccuracies can happen when timings come from different Agents.

You can now understand the behavior of each call, as well as the impact of GC suspension or CPU problems. You also can see the latency between your application and Cassandra which may heavily influence performance.

Monitoring Cassandra server

In addition to the details shown here, there is an Apache Cassandra Fastpack. It features a new Measure Group specifically for Cassandra server nodes and out-of-the-box Dashboards for monitoring of Cassandra.

Supported client and server versions

Currently, the sensors shipped with AppMon support the following Hector and Cassandra versions:

	Supported Versions Thrift	Supported Versions CQL3 binary
Client	Hector 1.0-1 1.0-3 1.0-5 1.1-2 Astyanax 1.56.44 1.56.48	Datastax 1.0.2 2.1
Cassandra	1.0 1.1 1.2	1.2 2.0

Indirectly Supported Clients

- Firebrand, an ORM Client that uses Hector underneath.

Although the sensors are specifically written for Hector, all other clients supporting the Apache Thrift protocol (0.6.x, 0.7.x) also will show Cassandra calls and tracing will work. Information about the Keyspace or the Cassandra host that it is executed against is available only for Hector clients.

Partly Supported Clients

- PlayOrm uses Astyanax underneath
- Astyanax
- Kundera

Fujitsu Interstage Application Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Interstage Application Server 9.0

Troubleshooting

Related pages

Set up agents

System Requirements

Note

The instructions on this page use `<InterstageAppServerHome>` to denote your Interstage Application Server installation folder.

See Set up Agents for information on installing the agent.

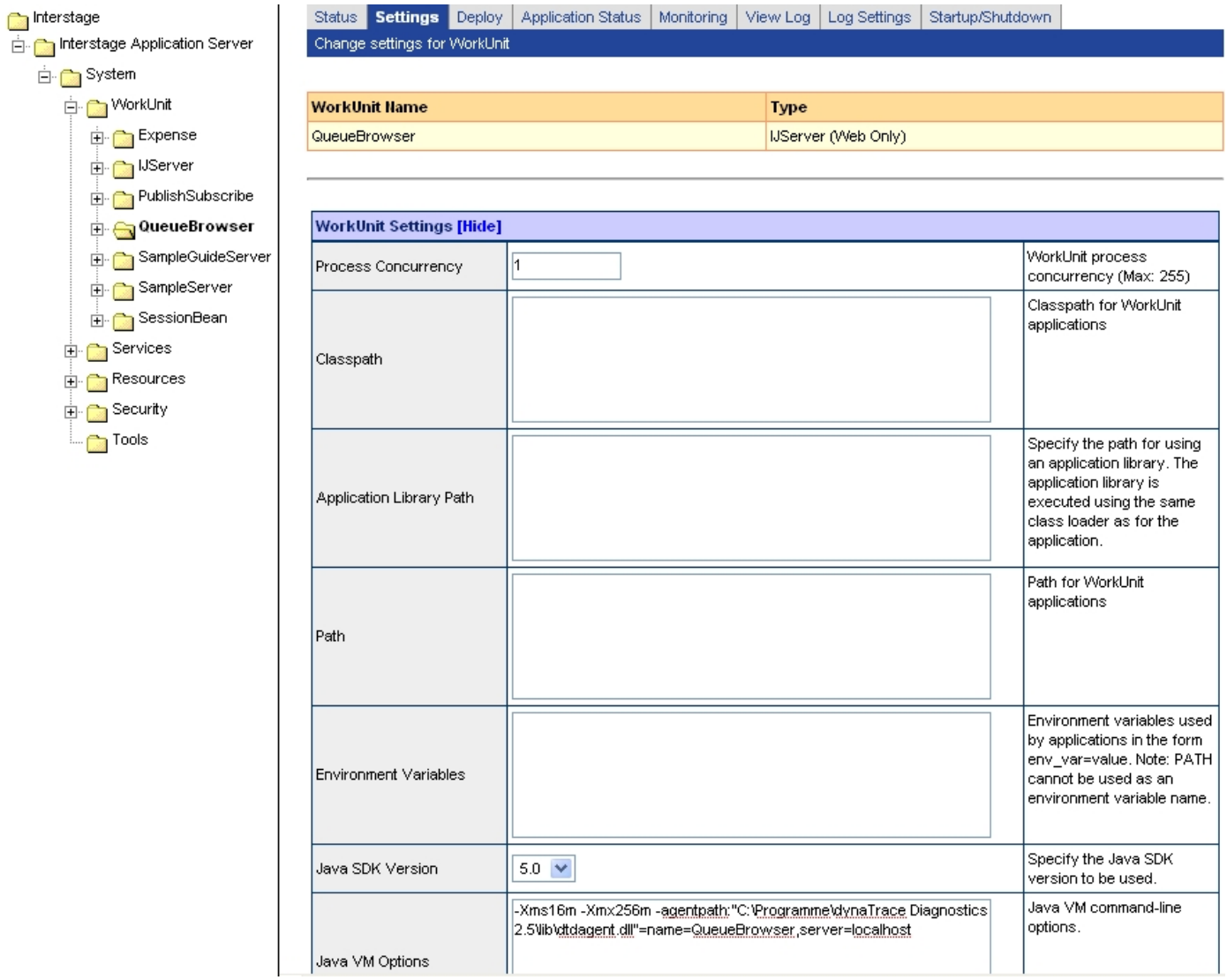
Interstage Application Server 9.0

Windows

The following section describes how to configure AppMon to diagnose applications on Interstage Application Server running on Windows using J2SE 5.0.

J2SE 5.0

1. After starting your Interstage Application Server, use http://localhost:12000/IsAdmin/ to open the Interstage Web Console. When accessing the server remotely, specify the machine´s hostname rather than localhost.
2. In the Interstage Web Console, go to **Open Interstage > Interstage App Server > System > Work Unit** and select the Work Unit of interest box and stop the Work Unit to change its settings.
3. Click the desired Work Unit and select the **Settings** tab. Expand the Work Unit settings by choosing **[Show]**.
4. Step to the Java VM options and add or append the following option to the text field: `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`.



5. Update the Work Unit´s configuration. Select the **Status** tab and start the Work Unit.

Troubleshooting

Configuration files:

Interstage Application Server running on Windows:

```
<InterstageAppServerHome>\td\var\td001\def\wu\yourWorkUnit.wu.
```

Log files:

Interstage Application Server running on Windows:

```
<InterstageAppServerHome>\J2EE\var\deployment\ijserver\yourWorkUnit\log\
```

The log files can also be accessed via the Interstage Web Console: **Interstage > Interstage Application Server > System > Work Unit** on the **View Log** tab.

GlassFish Server Open Source Edition

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installation

GlassFish 3.1 / 4.0

Related pages

Set up agents

System Requirements

Installation

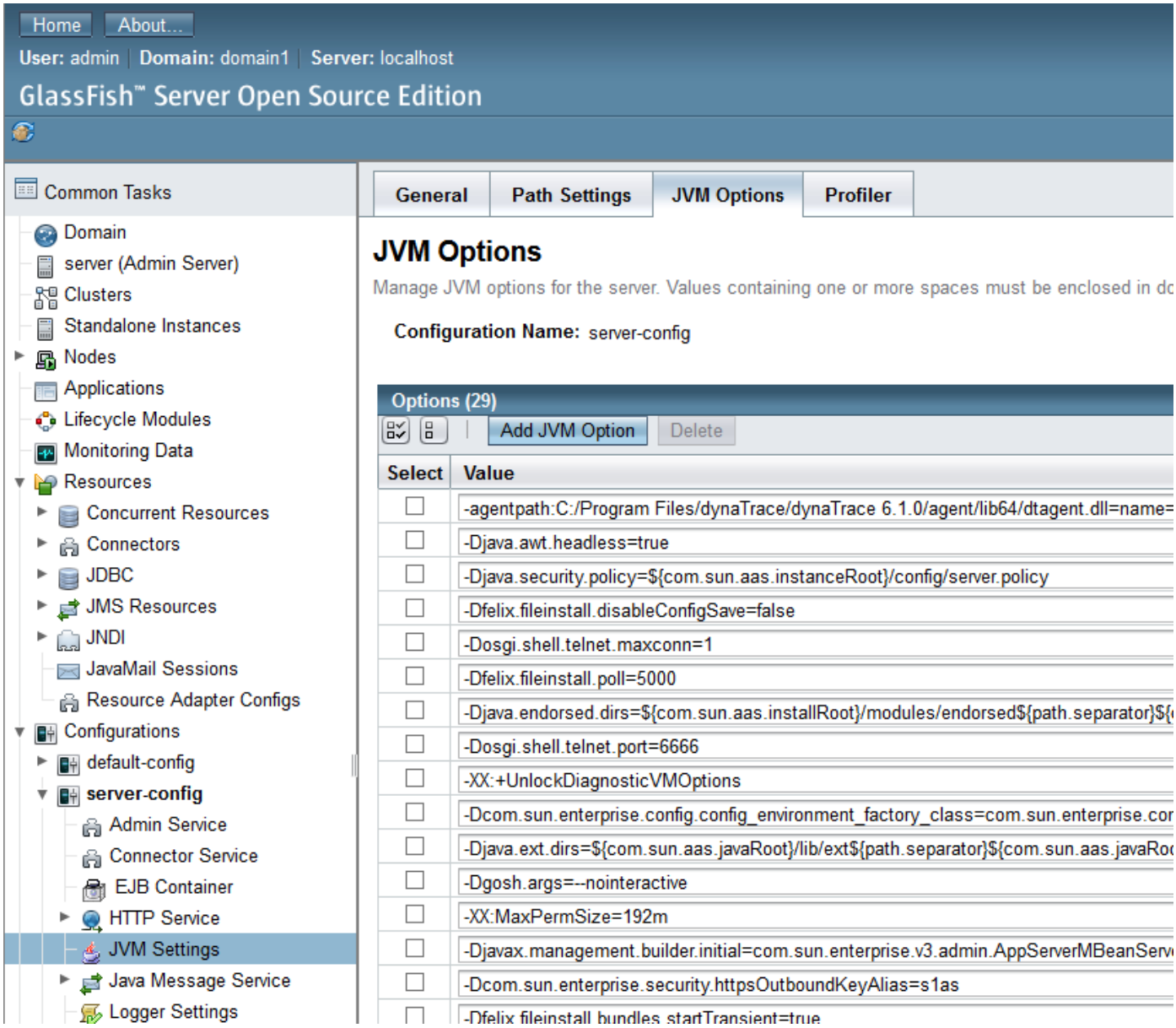
For installing the agent, please see Set up Agents.

> **Note**
>
> Instructions on this page use *<GlassFishHome>* to denote your GlassFish installation folder.

GlassFish 3.1 / 4.0

The following steps describe how to configure AppMon to diagnose applications on Oracle GlassFish 3 (using Java 1.6) and GlassFish 4 (using Java 1.7).

1. Start the Oracle GlassFish using the Graphical User Interface or the command line.
   Linux command line example: `<GlassFishHome>/bin/asadmin start-domain domain1`
   Windows command line example: `<GlassFishHome>/bin/asadmin.bat start-domain domain1`
2. Open the Admin Console use the URL http://localhost:4848/. When accessing the server remotely, specify the machine's hostname rather than localhost.
3. Enter your user name and password, and then log in.
4. Navigate to **Configurations > [server-config] > JVM Settings > JVM Option > Add JVM Option**. *Oracle GlassFish Administration Console*



5. In the resulting empty text field, enter one of the following:
   For Linux: `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`
   For Windows: `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`

6. Save the configuration changes and restart the server.

You can also set `domain.xml` as (`<GlassFishHome>\domains\<domainname>\config\domain.xml`) as shown in the following examples:

GlassFish 3.1 - domain.xml

```
<java-config classpath-suffix="" system-classpath="" debug-options="-Xdebug -Xrunjdwp:tra
-->     <jvm-options>-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server
        <jvm-options>-XX:MaxPermSize=192m</jvm-options>
        <jvm-options>-XX:PermSize=64m</jvm-options>
        <jvm-options>-client</jvm-options>
...
```

GlassFish 4.0 - domain.xml

```
<java-config classpath-suffix="" system-classpath="" debug-options="-agentlib:jdwp=transp
-->     <jvm-options>-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server
        <jvm-options>-XX:MaxPermSize=192m</jvm-options>
        <jvm-options>-client</jvm-options>
        <jvm-options>-Djava.awt.headless=true</jvm-options>
...
```

Java applets

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Applet viewer

Web browser: Java control panel

Web browser: alternate agent deployment for JAVA_TOOL_OPTIONS

Related pages

System Requirements

See Set up Agents for instructions on installing the agent.

Applet viewer

*NIX

AppMon Agent deployment steps for Java 1.5 or higher

1. Locate the AppletViewer within the JDK package installed on your machine. For example: `/usr/java/jdk1.5.0_06/bin`.
2. Execute the Applet Viewer with the following parameters: `./appletviewer URL -J-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`.

```
[labuser@labmachine bin]$ ./appletviewer
http://java.sun.com/applets/jdk/1.4/demo/applets/TicTacToe/example1.html
-J-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=
<dynaTraceCollectorName>.
```

AppMon Agent deployment steps for Java 1.4

1. Append `<DT_HOME>/agent/lib` to the system environment variable `LD_LIBRARY_PATH`.
2. Locate the AppletViewer within the JDK package installed on your machine. For example: `/usr/java/jdk1.4.2_05/bin`.
3. Execute the Applet Viewer with the following parameters: `./appletviewer URL -J-Xrundtagent:name=<agentname>,server=<serverhostname>`.

For example:
```
[labuser@labmachine bin]$ export
LD_LIBRARY_PATH=/home/user/dynatrace-6.5/agent/lib:$LD_LIBRARY_PATH
[labuser@labmachine bin]$ ./appletviewer
http://java.sun.com/applets/jdk/1.4/demo/applets/TicTacToe/example1.html
-J-Xrundtagent:name=<agentname>,server=<serverhostname>
```

Windows

AppMon Agent deployment steps for J2SE 5.0

1. Locate the AppletViewer within the JDK package installed on your machine. For example: `C:\devtools\jdk1.5.0_06\bin`.
2. Execute the Applet Viewer with the following parameters: `appletviewer.exe URL -J-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`.

For example:
```
C:\devtools\jdk1.5.0_06\bin>appletviewer.exe
http://java.sun.com/applets/jdk/1.4/demo/applets/TicTacToe/example1.html
-J-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=
<dynatraceCollectorName>
```

AppMon Agent deployment steps for J2SE 1.4

1. Append `<DT_HOME>\agent\lib` to the system environment variable PATH.

2. Locate the AppletViewer within the JDK package installed on your machine. For example: `C:\devtools\jdk1.4.2_05\bin`.
3. Execute the Applet Viewer with the following parameters: `./appletviewer URL -J-Xrundtagent:name=<agentname>,server=<serverhostname>`.

For example: `C:\devtools\jdk1.4.2_05\bin>appletviewer.exe http://java.sun.com/applets/jdk/1.4/demo/applets/TicTacToe/example1.html -J-Xrundtagent:name=<agentname>,server=<serverhostname>`

Web browser: Java control panel

*NIX

AppMon Agent deployment steps for Java 1.5 or higher

1. Locate the Java Control Panel within the JDK package installed on your machine. For example: `/usr/java/jdk1.5.0_06/jre/bin`.
2. Start the Control Panel.
3. Add the following option to **Java > Java Applet Runtime Settings > Java Runtime Parameters**: `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`.

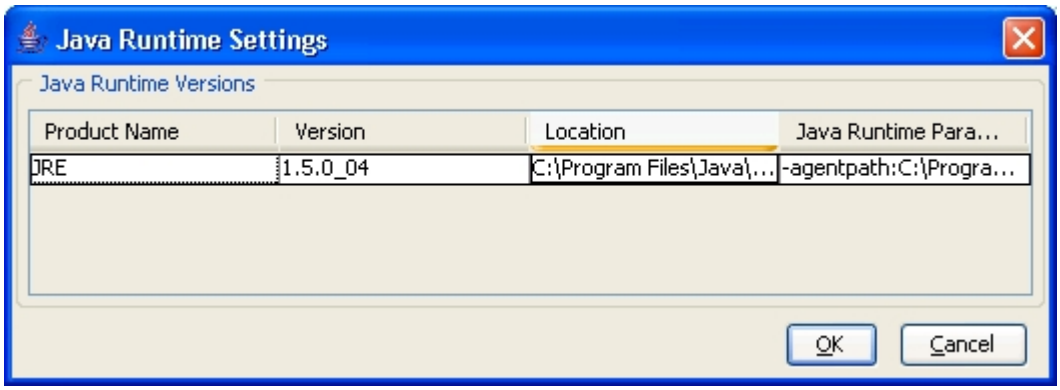
AppMon Agent deployment steps for Java 1.4:

1. Append `<DT_HOME>/agent/lib` to the system environment variable `LD_LIBRARY_PATH.`
2. Locate the Java Control Panel within the JDK package installed on your machine. For example: `/usr/java/jdk1.4.2_05/jre/bin`. Make sure that the `JAVA_HOME` system environment variable is set to this JDK package.
3. Start the Control Panel.
4. Add the following option to **Advanced > Java Runtime Parameters**: `-Xrundtagent:name=<agentname>,server=<serverhostname>`.

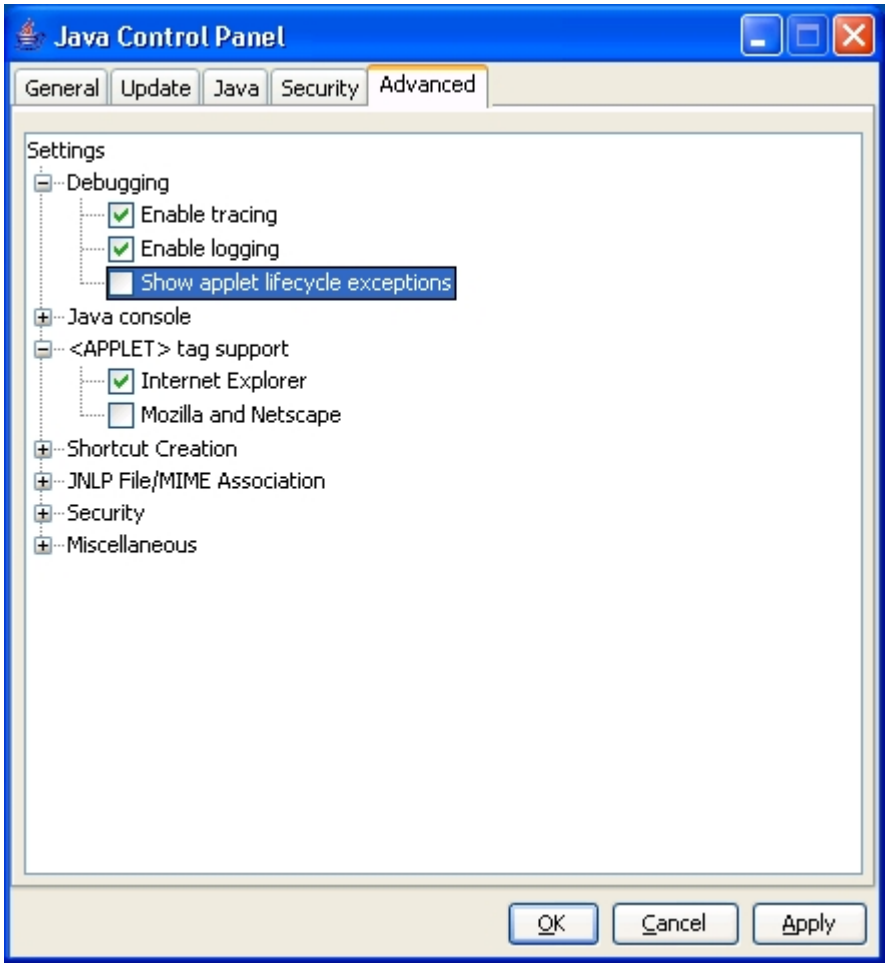
Windows

Deployment steps for Java 1.5 or higher

1. Open and start the Java Control Panel, **Start > Control Panel**.
2. Add the following option to **Java > Java Applet Runtime Settings > Java Runtime Parameters**: `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`.



Additionally, the following debug options should be set:



AppMon Agent deployment steps for Java 1.4

1. Append `<DT_HOME>\agent\lib` to the system environment variable PATH.
2. Open and start the Java Control Panel: **Start >Control Panel**.
   Make sure that the `JAVA_HOME` system environment variable is set to this JDK package.

3. Add the following option to **Advanced > Java Runtime Parameters**: `-Xrundtagent:name=<agentname>,server=<serverhostname>`.

Web browser: alternate agent deployment for JAVA_TOOL_OPTIONS

Since the command-line cannot always be accessed or modified (for example, in embedded VMs or in VMs launched deep within scripts), a JAVA_TOOL_OPTIONS variable is provided so that the AppMon Agent can be launched in these cases. Platforms that support environment variables or other named strings may support the JAVA_TOOL_OPTIONS variable. This variable will be broken into options at white-space boundaries. White-space characters include space, tab, carriage-return, new-line, vertical-tab, and form-feed. Sequences of white-space characters are considered equivalent to a single white-space character. No white space is included in the options unless quoted. Quoting is as follows:

- All characters enclosed within single quotation (´ ´), except quotation marks themselves, are quoted.
- Double-quotation characters have no special meaning inside a pair of single quotation marks.
- All characters enclosed between a pair of double-quotation marks (“ “), except a double-quotation, are quoted.
- Single-quotation characters have no special meaning inside a pair of double-quotation marks.
- A quoted part can start or end anywhere in the variable.
- White-space characters have no special meaning when quoted – they are included in the option like any other character and do not mark white-space boundaries.
- The pair of quotation marks is not included in the option.

Set JAVA_TOOL_OPTIONS=`-agentpath:<DT_HOME>\agent\lib\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`. Then start your Web Browser from the command line. For example: `C:\Program Files\Internet Explorer>IEXPLORE.EXE`.

JBoss, Wildfly Application Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installation

Wildfly 8.x - 10.0.0 / JBoss 7

JBoss 6

JBoss 5

Related pages

Set up agents

System Requirements

External links

JBoss domain mode doc page

Installation

For general Agent installation instructions, see Set up Agents.

32-bit Agent example

```
-agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=name=<AgentName>,server=<collectorName>
-agentpath:"<DT_HOME>/agent/lib/libdtagent.so"=name=<AgentName>,server=<collectorName>
```

If you're running a 64-bit JBoss, use `lib64` in the path instead of `lib`.

Note

For Windows, the unconventional nested double quotes set in “JAVA_OPTS=…-agentpath:”<pathWithSpace>…“…”” (`set "JAVA_OPTS=...-agentpath:"<pathWithSpace>..."..."`) are correct. A known bug in JBoss versions 8.1.0 and 8.2.0 prevents usage of spaces in the JAVA_OPTS environment variable. To work around this issue:

- Use the latest version 8 (8.2.1) if possible.
- If you can install to a space-less path you can get along with 8.1.0 or 8.2.0.
- You can apply a special treatment to / edit `standalone.bat`.

standalone.bat workaround

Find the following piece of code in `standalone.bat` and place the Agent string between `"%JAVA%"` and `%JAVA_OPTS%`.

```
:RESTART
rem if x%XLOGGC% == x (
  "%JAVA%" -agentpath:"C:\Program Files\Dynatrace\Dynatrace 7.0\agent\lib\dtage
   "-Dorg.jboss.boot.log.file=%JBOSS_LOG_DIR%\server.log" ^
   "-Dlogging.configuration=file:%JBOSS_CONFIG_DIR%/logging.properties" ^
      -jar "%JBOSS_HOME%\jboss-modules.jar" ^
      -mp "%JBOSS_MODULEPATH%" ^
      -jaxpmodule "javax.xml.jaxp-provider" ^
       org.jboss.as.standalone ^
      "-Djboss.home.dir=%JBOSS_HOME%" ^
       %SERVER_OPTS%
```

The example is for AppMon standard installation path, JBoss 8.2.0 and the Collector on the same machine (`server=localhost`). Replace as necessary.

Wildfly 8.x - 10.0.0 / JBoss 7

> **Note**
>
> WildFly 10 does not deliver any pool statistics if the data source has not set the “enable statistics” flag.

Windows

Do one of the following depending on whether Wildfly 8 / JBoss 7 runs in standalone or domain mode.

Standalone mode

1. Extend the environment variable `JAVA_OPTS`. For example, `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory.

2. Edit `<JBossHome>\bin\standalone.conf.bat` as shown in the `set` line below.

```
...
rem # Use JBoss Modules lockless mode
rem set "JAVA_OPTS=%JAVA_OPTS% -Djboss.modules.lockless=true"

rem # dynaTrace Agent Injection
set "JAVA_OPTS=%JAVA_OPTS% -agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=name=<AgentNam

:JAVA_OPTS_SET
```

Domain mode

1. Set the Agent options per server group or every individual server instance.

2. To set the Agent options per server group, modify `<JBossHome>\domain\configuration\domain.xml` as shown below.

```
...
    <server-groups>
        <server-group name="main-server-group" profile="full">
            <jvm name="default">
                <heap size="64m" max-size="512m"/>
                    <jvm-options>
                        <option value="-agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=na
                    </jvm-options>
                </jvm>
            <socket-binding-group ref="full-sockets"/>
        </server-group>
    </server-groups>
</domain>
```

3. To set the Agent options for every individual server, edit `<JBossHome>\domain\configuration\host.xml` as shown below.

```
...
    <server name="server-one" group="main-server-group">
        <jvm name="default">
            <jvm-options>
                <option value="-agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=name=<Ager
            </jvm-options>
        </jvm>
    </server>
...
```

Linux

Do one of the following depending on whether Wildfly 8 / JBoss 7 runs in standalone or domain mode.

Standalone mode

1. Extend the environment variable `JAVA_OPTS`. For example, `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory.

2. Edit `<JBossHome>\bin\standalone.conf` as shown in the `JAVA_OPTS=` line below.

```
# when you get "epoll_create function not implemented" message in dmesg output
# JAVA_OPTS="$JAVA_OPTS -Djava.nio.channels.spi.SelectorProvider=sun.nio.ch.PollSelect

# dynaTrace Agent Injection
JAVA_OPTS="$JAVA_OPTS -agentpath:"<DT_HOME>/agent/lib/libdtagent.so"=name=<AgentName>,
```

Domain mode

1. Set the Agent options per server group or every individual server instance. To do this, modify `<JBossHome>\domain\configuration\domain.xml` as shown below. This adds all server instances that are member of that server group with the same settings.

```
...
    <server-groups>
        <server-group name="main-server-group" profile="full">
            <jvm name="default">
                <heap size="64m" max-size="512m"/>
                    <jvm-options>
                        <option value="-agentpath:"<DT_HOME>/agent/lib/libdtagent.so"=
                    </jvm-options>
            </jvm>
            <socket-binding-group ref="full-sockets"/>
        </server-group>
    </server-groups>
</domain>
```

2. Set the Agent options for every individual server. To do this, edit `<JBossHome>\domain\configuration\host.xml` as shown below.

```
...
    <server name="server-one" group="main-server-group">
        <jvm name="default">
            <jvm-options>
                <option value="-agentpath:"<DT_HOME>/agent/lib/libdtagent.so"=name=<Ag
            </jvm-options>
        </jvm>
    </server>
...
```

JBoss 6

Windows

1. Extend the environment variable `JAVA_OPTS`. For example, `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory.
2. Modify `<JBossHome>\bin\run.conf.bat` as shown in the `set` line below.

```
# Sample JPDA settings for remote socket debuging
# JAVA_OPTS="$JAVA_OPTS -Xrunjdwp:transport=dt_socket,address=8787,server=y,suspend=n"

# Sample JPDA settings for shared memory debugging
# JAVA_OPTS="$JAVA_OPTS -Xrunjdwp:transport=dt_shmem,server=y,suspend=n,address=jboss"

rem # dynaTrace Agent Injection
set "JAVA_OPTS=%JAVA_OPTS% -agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=name=<AgentName>,

:JAVA_OPTS_SET
```

Linux

1. Extend the environment variable `JAVA_OPTS`. For example, `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory.
2. Modify `<JBossHome>\bin\run.conf` as shown in the `JAVA_OPTS=` line below.

```
# Sample JPDA settings for remote socket debuging
# JAVA_OPTS="$JAVA_OPTS -Xrunjdwp:transport=dt_socket,address=8787,server=y,suspend=n"

# Sample JPDA settings for shared memory debugging
# JAVA_OPTS="$JAVA_OPTS -Xrunjdwp:transport=dt_shmem,server=y,suspend=n,address=jboss"

# dynaTrace Agent Injection
JAVA_OPTS="$JAVA_OPTS -agentpath:"<DT_HOME>/agent/lib/libdtagent.so"=name=<AgentName>,ser
```

JBoss 5

Windows

1. Extend the environment variable `JAVA_OPTS`. For example, `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory.
2. Modify `<JBossHome>\bin\run.bat` as shown in the `set JAVA_OPTS` line below.

```
...
rem Setup the java endorsed dirs
set JBOSS_ENDORSED_DIRS=%JBOSS_HOME%\lib\endorsed
set JAVA_OPTS=%JAVA_OPTS% -agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=name=<AgentName>,s

echo ===============================================================================
echo .
echo JBoss Bootstrap Environment
echo .
echo JBOSS_HOME: %JBOSS_HOME%
echo .
echo JAVA: %JAVA%
echo .
echo JAVA_OPTS: %JAVA_OPTS%
echo .
echo CLASSPATH: %JBOSS_CLASSPATH%
echo .
echo ===============================================================================
echo .

:RESTART
"%JAVA%" %JAVA_OPTS% "-Djava.endorsed.dirs=%JBOSS_ENDORSED_DIRS%" -classpath "%JBOSS_CLAS
IF ERRORLEVEL 10 GOTO RESTART

:END
if "%NOPAUSE%" == "" pause

:END_NO_PAUSE
```

Linux

1. Extend the environment variable `JAVA_OPTS`. For example, `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory.
2. Modify `<JBossHome>\bin\run.conf` as shown in the `JAVA_OPTS` line below.

```
...
# Sample JPDA settings for remote socket debuging
# JAVA_OPTS="$JAVA_OPTS -Xrunjdwp:transport=dt_socket,address=8787,server=y,suspend=n"

# Sample JPDA settings for shared memory debugging
# JAVA_OPTS="$JAVA_OPTS -Xrunjdwp:transport=dt_shmem,server=y,suspend=n,address=jboss"

# dynaTrace Agent Injection
JAVA_OPTS="$JAVA_OPTS -agentpath:"<DT_HOME>/agent/lib/libdtagent.so"=name=<AgentName>,ser
```

See the external page Enable Monitoring in JBoss Domain Modes for more details about JBoss Domain Mode.

Microsoft SharePoint

You can also watch the Video Tutorial on YouTube: SharePoint Performance with Dynatrace in 15 Minutes.
The latest Knowledge Sensor Pack and documentation for Microsoft Windows SharePoint Services and Microsoft Office SharePoint Server is available on the AppMon SharePoint FastPack download page.

Windows Services

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Java 1.4

Java 1.5 or higher

Related pages

Set up agents

System Requirements

See Set up Agents for information on installing the agent.

If the System under Diagnosis is started as a Windows service and you don't have the opportunity to inject the Diagnostics Agent using a configuration file, you must follow the steps described below for Java 1.4 or Java 1.5 or higher:

Java 1.4

1. Edit the Path system environment variable. Prepend `<DT_HOME>\agent\lib` to the path of the environment variable. `<DT_HOME>` represents the AppMon installation directory. For example, `<DT_HOME>\agent\lib`.
2. Edit the application's `ImagePath` in the system registry. For example, `HKEY_LOCAL_MACHINE\System\ControlSet001\Services\<ApplicationEntry>\ImagePath`.
3. Add the following Java options to the `ImagePath`: `-Xrundtagent:name=<agentname>,server=<serverhostname>`.
4. Restart the computer to change the system environment variable.

Java 1.5 or higher

1. Edit the `Path` system environment variable. Prepend `<DT_HOME>\agent\lib` to the path of the environment variable. `<DT_HOME>` represents the AppMon installation directory. For example, `<DT_HOME>\agent\lib`.
2. Edit the application's `ImagePath` in the system registry. For example, `HKEY_LOCAL_MACHINE\System\ControlSet001\Services\<ApplicationEntry>\ImagePath`.
3. Add the following Java options to the `ImagePath`: `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`.
4. Restart the computer to change the system environment variable.

Mule ESB

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Mule ESB 1.x or later

Related pages

System Requirements

Downloads

Tanuki service wrapper

See Set up Agents for information on installing the agent.

Mule ESB 1.x or later

Mule ESB uses the Tanuki Service Wrapper for starting the Java processes. Edit the file `<MULE_HOME>/conf/wrapper.conf` and add an additional java option. For a new Mule ESB installation, this is typically number 4.

On Windows, add this line:

```
wrapper.java.additional.4=-agentpath:
<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=
<dynatraceCollectorName>
```

On *NIX add this line:

```
wrapper.java.additional.4=-agentpath:
<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=
<dynaTraceCollectorName>
```

Oracle Application Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Oracle 10.1.3+

Oracle 10g

Troubleshooting

Related pages

Set up agents

System Requirements

See Set up Agents for information on installing the agent.

Oracle 10.1.3+

Unix

To configure AppMon to diagnose applications on Oracle 10.1.3+ running on Unix using Java 1.5+:

1. Open the Oracle Enterprise Manager (for example, http://localhost:1810/). Then select the instance subject to introspection and follow the links **Administration** and **Server Properties**.
2. In the section **Command Line Option** of the **Server Properties** page, add the following to the Java Options: `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`.
3. Restart the Oracle instance.

Windows

To configure AppMon to diagnose applications on Oracle 10.1.3+ running on Windows using Java 1.5+:

1. Open the Oracle Enterprise Manager (for example, http://localhost:1810/). Then select the instance that is subject to introspection and then follow the links **Administration** and **Server Properties**.
2. In the section **Command Line Option** of the **Server Properties page**, add the following to the Java Options: `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`.
3. Restart the Oracle instance.

Alternative deployment

The `opmn.xml` file is in the `<OraHome>/opmn/conf` directory. This file contains the startup information and thus the startup parameters for the Oracle Application Server (OAS) instances. If you are running the OAS on Windows, replace the Linux/Unix agent string with the Windows string.

```
<ias-component id="default_group">
        <process-type id="home" module-id="OC4J" status="enabled">
        <module-data>
          <category id="start-parameters">
                 <data id="java-options" value="-server -mx1024M -ms512M -XX:MaxPermSize=
          </category>
          <category id="stop-parameters">
                 <data id="java-options" value="-Djava.security.policy=$ORACLE_HOME/j2ee/
          </category>
        </module-data>
        <start timeout="600" retry="2"/>
        <stop timeout="120"/>
        <restart timeout="720" retry="2"/>
        <port id="default-web-site" range="8888" protocol="http"/>
        <port id="rmi" range="12401-12500"/>
        <port id="rmis" range="12701-12800"/>
        <port id="jms" range="12601-12700"/>
        <process-set id="default_group" numprocs="1"/>
        </process-type>
        <process-type id="backend" module-id="OC4J" status="enabled">
        <module-data>
          <category id="start-parameters">
                 <data id="java-options" value="-server -Djava.security.policy=$ORACLE_HO
          </category>
          <category id="stop-parameters">
                 <data id="java-options" value="-Djava.security.policy=$ORACLE_HOME/j2ee/
          </category>
        </module-data>
        <start timeout="600" retry="2"/>
        <stop timeout="120"/>
        <restart timeout="720" retry="2"/>
        <port id="default-web-site" range="12501-12600" protocol="http"/>
        <port id="rmi" range="12401-12500"/>
        <port id="rmis" range="12701-12800"/>
        <port id="jms" range="12601-12700"/>
        <process-set id="default_group" numprocs="1"/>
        </process-type>
</ias-component>
```

Oracle 10g

`<OraHome>` is the directory of your Oracle Installation (for example, `c:\OraHome` on Windows platforms).
`<UniqueAgentName>` is a unique and descriptive name for the Agent. This name is further displayed in AppMon reports.

Unix

To configure AppMon to diagnose applications on Oracle 10g running on Unix using J2SE 1.4:

1. Copy `libdtagent.so` from `<dthome>/lib/` to `<OraHome>/lib/`.
2. Open the Oracle Enterprise Manager (for example, http://localhost:1810/). Then select the instance subject to introspection and then follow the links **Administration** and **Server Properties**.
3. In the section **Command Line Option** of the **Server Properties** page, add the following to the Java Options: `-Xrundtagent:name=<UniqueAgentName>,server=<Diagnostics Server>`.
4. Restart the Oracle instance.

Windows

To configure AppMon to diagnose applications on Oracle 10g running on Windows using J2SE 1.4:

1. Copy `dtagent.dll` from `<dthome>\lib\` to `<OraHome>\lib\`.
2. Open the Oracle Enterprise Manager (for example, http://localhost:1810/). Then select the instance that is subject to introspection and then follow the links **Administration** and **Server Properties**.
3. In the section **Command Line Option** of the **Server Properties** page, add the following to the Java Options: `-Xrundtagent:name=<UniqueAgentName>,server=<Diagnostics Server>`.
4. In the section **Environment Variables**, add a variable (`PATH`) that points to the full path of the `dtagent.dll` file (for example, `<OraHome>\lib\dtagent.dll`).
5. Restart the instance that is subject to introspection.

The following example configures the AppMon Agent in the Oracle 10g Server Properties page. `-Xrundtagent:name=Agent1,server=lab1`

Troubleshooting

Use the configuration file and log file to aid troubleshooting. The Windows configuration file is `<OraHome>/opmn/conf/opmn.xml`. The Windows log files are in the Log section of the OAS Administration Console.

SAP Netweaver

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

SAP Netweaver CE 7.3 (Java)

Related pages

Set up agents

System Requirements

See Set up Agents for information on installing the agent.

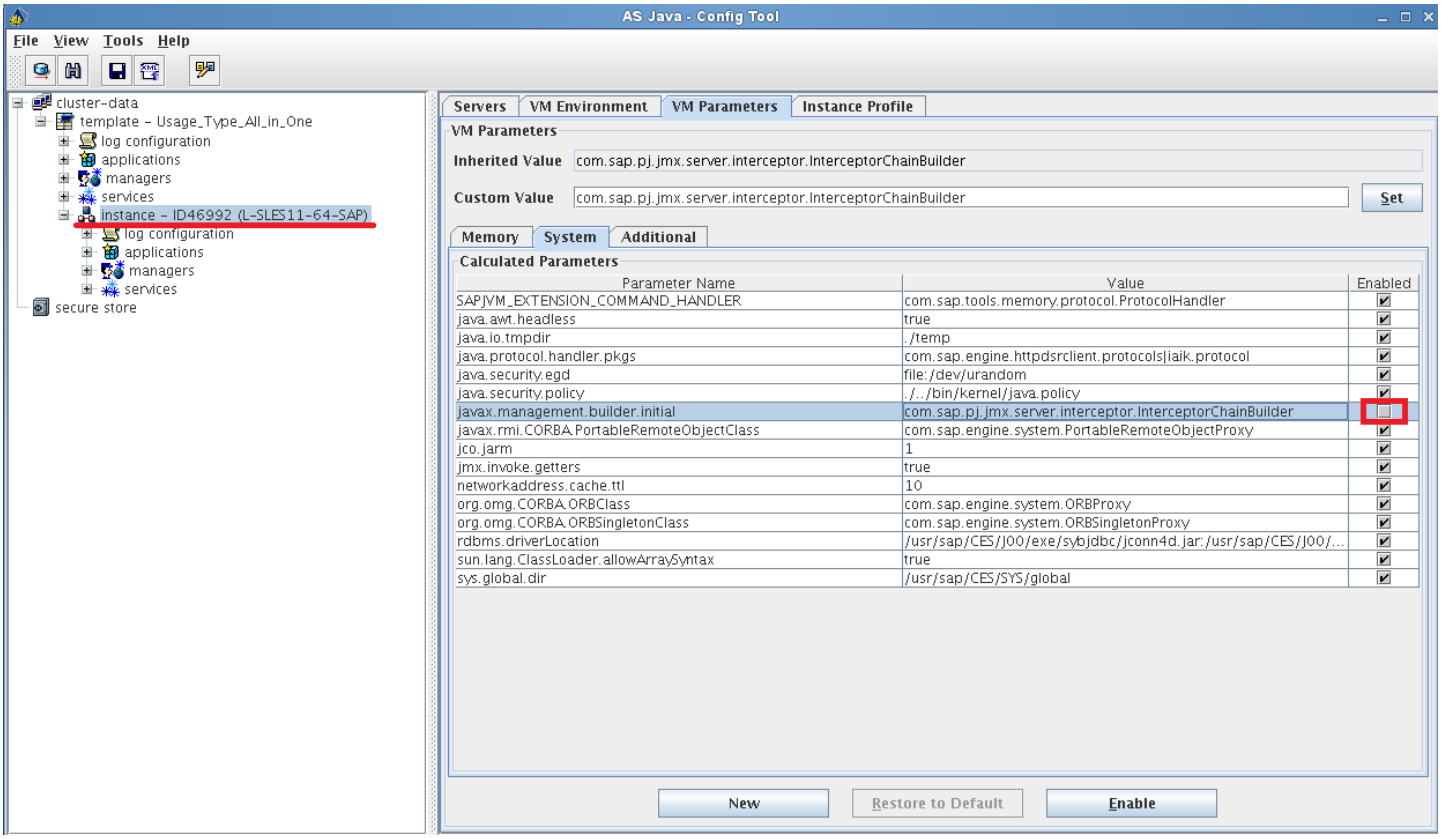
SAP Netweaver CE 7.3 (Java)

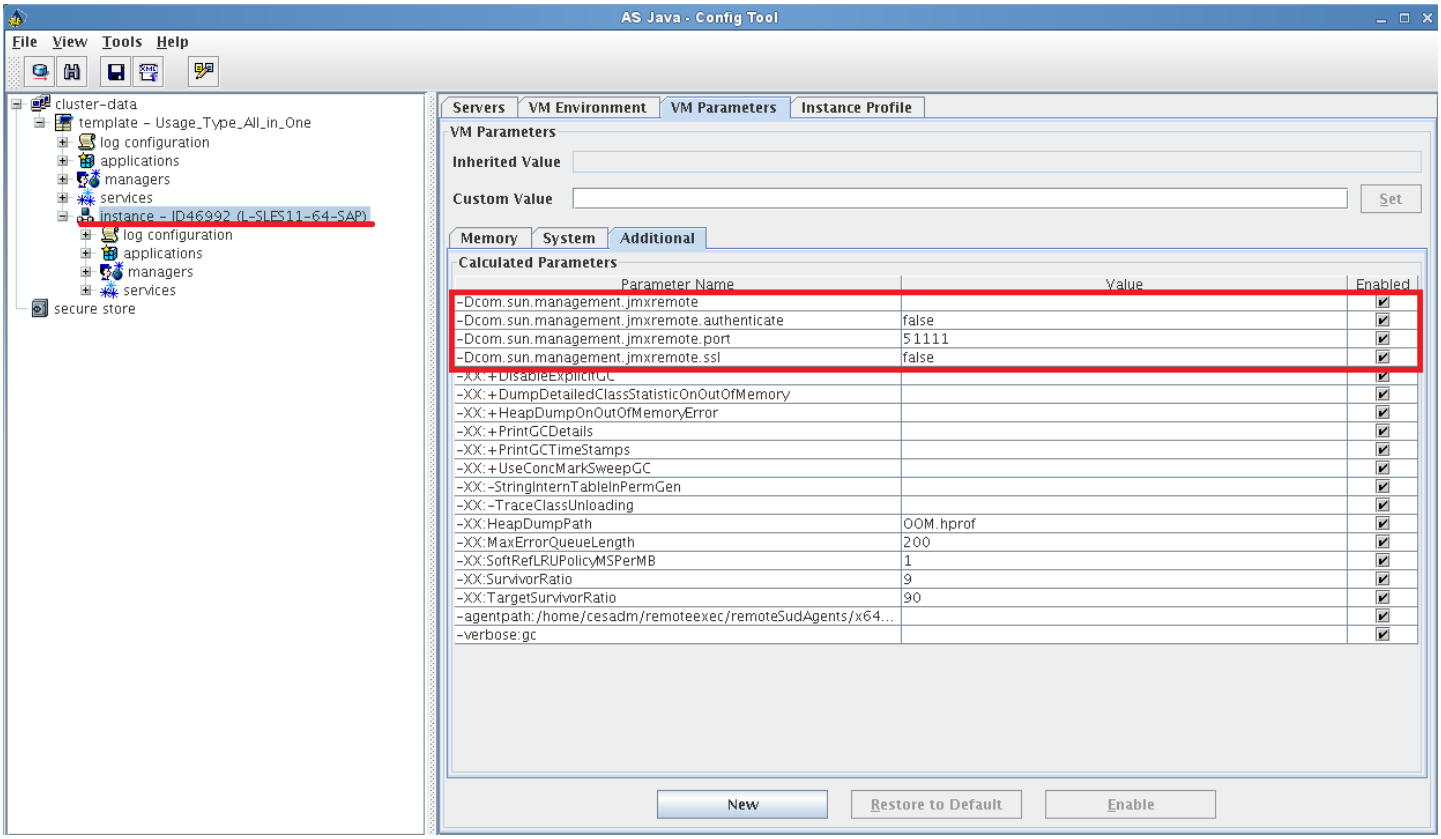
Set the VM parameters using the SAP Config Tool located at `<SAP_install_dir>/<system_name>/<instance_name>/j2ee/configtool/configtc` (or `/sap/scripts/RunConfigTool.sh` on *NIX).

This SAP page describes the procedure for setting the parameters. Add the parameter `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>` and leave the value empty.

JMX support

Before being able to connect to SAP AS Java MBeans, the Server has to be configured as shown below.





Troubleshooting

Use the log file to aid troubleshooting. The Windows log file is in the Application Server working directory (for example, `C:\usr\sap\J2E\JC00\work\std_server0`).

WebLogic Application Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

WebLogic 9 - WebLogic 12

Alternatives for all supported Operating Systems

WebLogic Agent naming

Related pages

System Requirements

See Set up Agents for information on installing the agent.

WebLogic 9 - WebLogic 12

Windows

Perform these steps to configure AppMon to diagnose applications on WebLogic 9 or 10 running on Windows using Java 1.5 or higher. WebLogic 12 using Java 7 is also supported.

1. Extend the environment variable `JAVA_OPTIONS` to include the following: `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>` where `<DT_HOME>` represents the AppMon installation directory.

2. Modify `JAVA_OPTIONS` in the WebLogic startup script (`startWebLogic.bat`) as shown in the following.
   The example path shows an AppMon 64-bit installation.

```
@REM START WEBLOGIC

*set JAVA_OPTIONS={dtAgentString_win} %JAVA_OPTIONS%*

echo starting weblogic with Java version:

%JAVA_HOME%\bin\java %JAVA_VM% -version

if "%WLS_REDIRECT_LOG%"=="" (
echo Starting WLS with line:
echo %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% {color:blue}*%JAVA_OPTIONS%*{color}
     -Dweblogic.Name=%SERVER_NAME% -Djava.security.policy=%WL_HOME%\server\lib\weblogi
     %PROXY_SETTINGS% %SERVER_CLASS%
%JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
     -Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy %PROXY_SETTINGS% %SEF
) else (
echo Redirecting output from WLS window to %WLS_REDIRECT_LOG%
%JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% {color:blue}*%JAVA_OPTIONS%*{color} -Dweblog
     -Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy %PROXY_SETTINGS%
     %SERVER_CLASS% >"%WLS_REDIRECT_LOG%" 2>&1
)
```

*NIX

Perform the following steps to configure AppMon to diagnose applications on WebLogic 9 or 10 running on Linux using J2SE 5.0. WebLogic 12 using Java 7 is also supported.

1. Extend the environment variable JAVA_OPTIONS as follows:
   `JAVA_OPTIONS=-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>${JAVA_OPTIONS}`.
2. Modify `JAVA_OPTIONS` in the WebLogic startup script (`startWebLogic.sh`)as shown below.

```
*JAVA_OPTIONS={dtAgentString_unix} {env:JAVA_OPTIONS}"*

echo "starting weblogic with Java version:"

{env:JAVA_HOME}/bin/java {env:JAVA_VM} -version

if [ "{env:WLS_REDIRECT_LOG}" = "" ] ; then
    echo "Starting WLS with line:"
echo "{env:JAVA_HOME}/bin/java {env:JAVA_VM} {env:MEM_ARGS} {env:JAVA_OPTIONS} -Dweblogic
          -Djava.security.policy={env:WL_HOME}/server/lib/weblogic.policy {env:PROXY_SETT

    {env:JAVA_HOME}/bin/java {env:JAVA_VM} {env:MEM_ARGS} {env:JAVA_OPTIONS} -Dweblogic.N
          -Djava.security.policy={env:WL_HOME}/server/lib/weblogic.policy {env:PROXY_SETT

else
    echo "Redirecting output from WLS window to {env:WLS_REDIRECT_LOG}"

    {env:JAVA_HOME}/bin/java {env:JAVA_VM} {env:MEM_ARGS} {env:JAVA_OPTIONS} -Dweblogic.N
          -Djava.security.policy={env:WL_HOME}/server/lib/weblogic.policy {env:PROXY_SETT
          {env:SERVER_CLASS} 2>&1 >"{env:WLS_REDIRECT_LOG}"

fi
```

Alternatives for all supported Operating Systems

If using the browser-based Console or Node Manager to control your WebLogic instances, you have other configuration options. You can configure the Node Manager to use start scripts. According to Oracle docs:

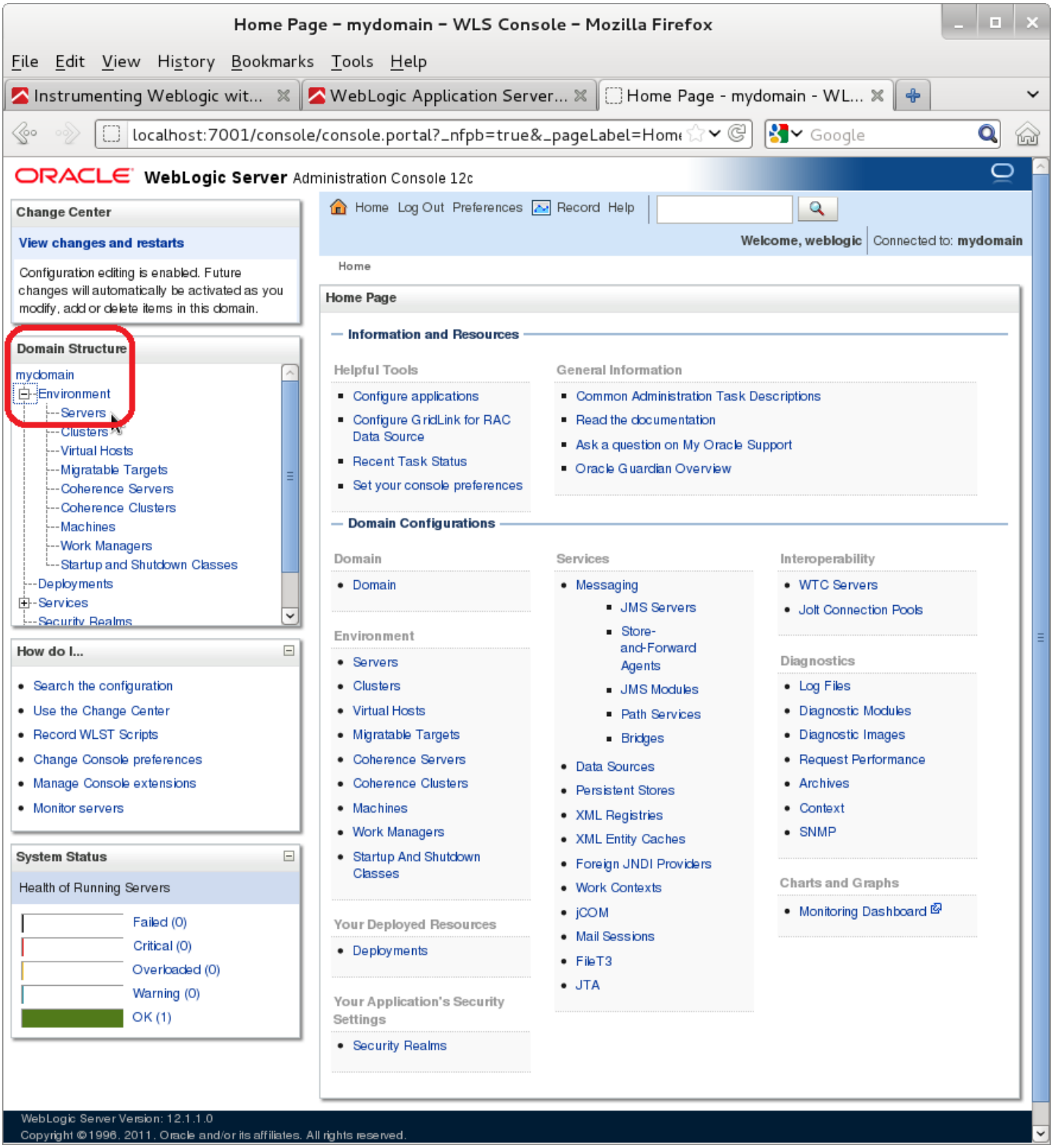
When you start Node Manager, it reads the nodemanager.properties file with the StartScriptEnabled=true property, and uses the start scripts when it subsequently starts Managed Servers. Note that you need to run the setNMProps script only once.

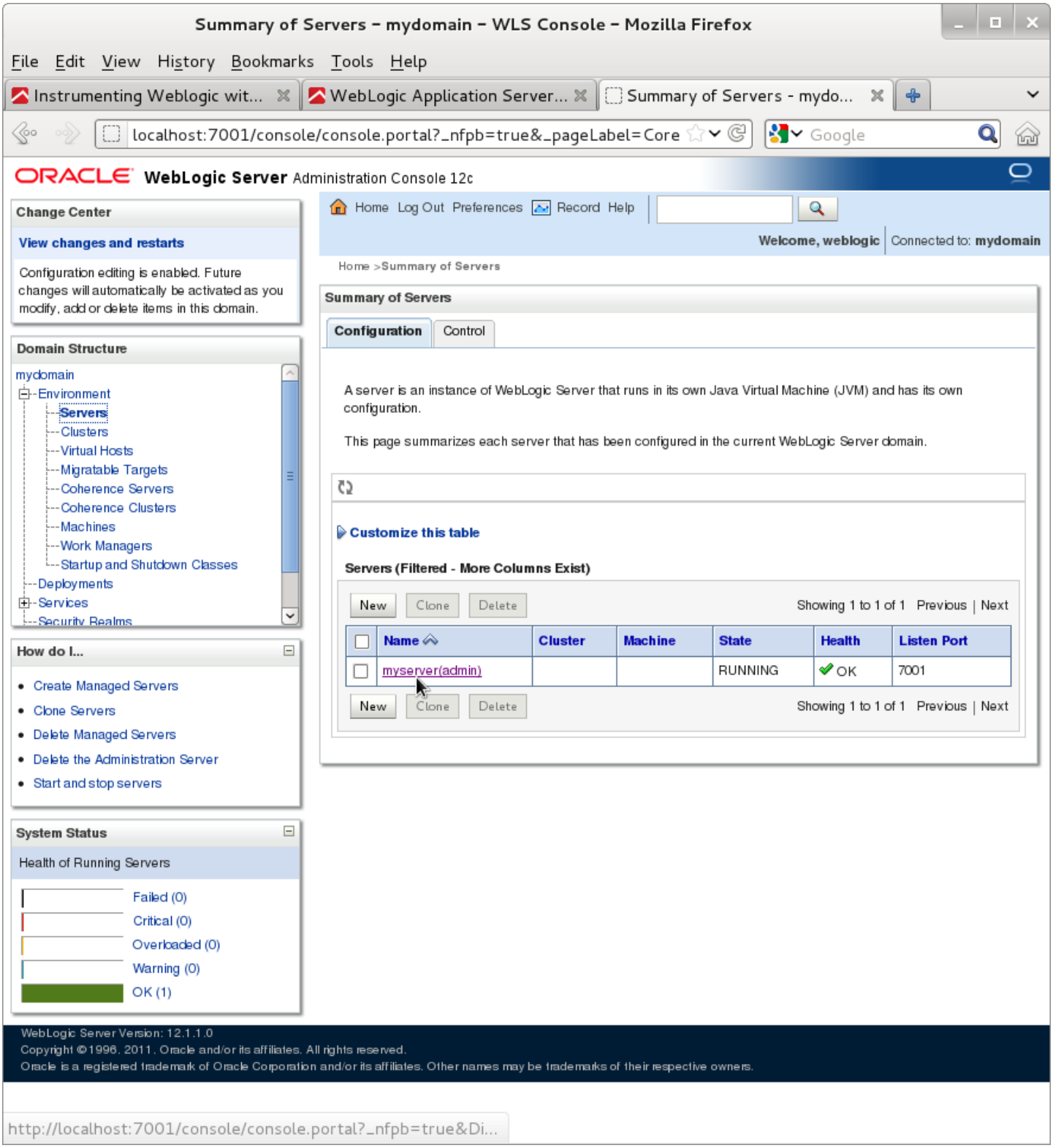
If you do not want to use start scripts, you can do the following, as illustrated by the figures below:

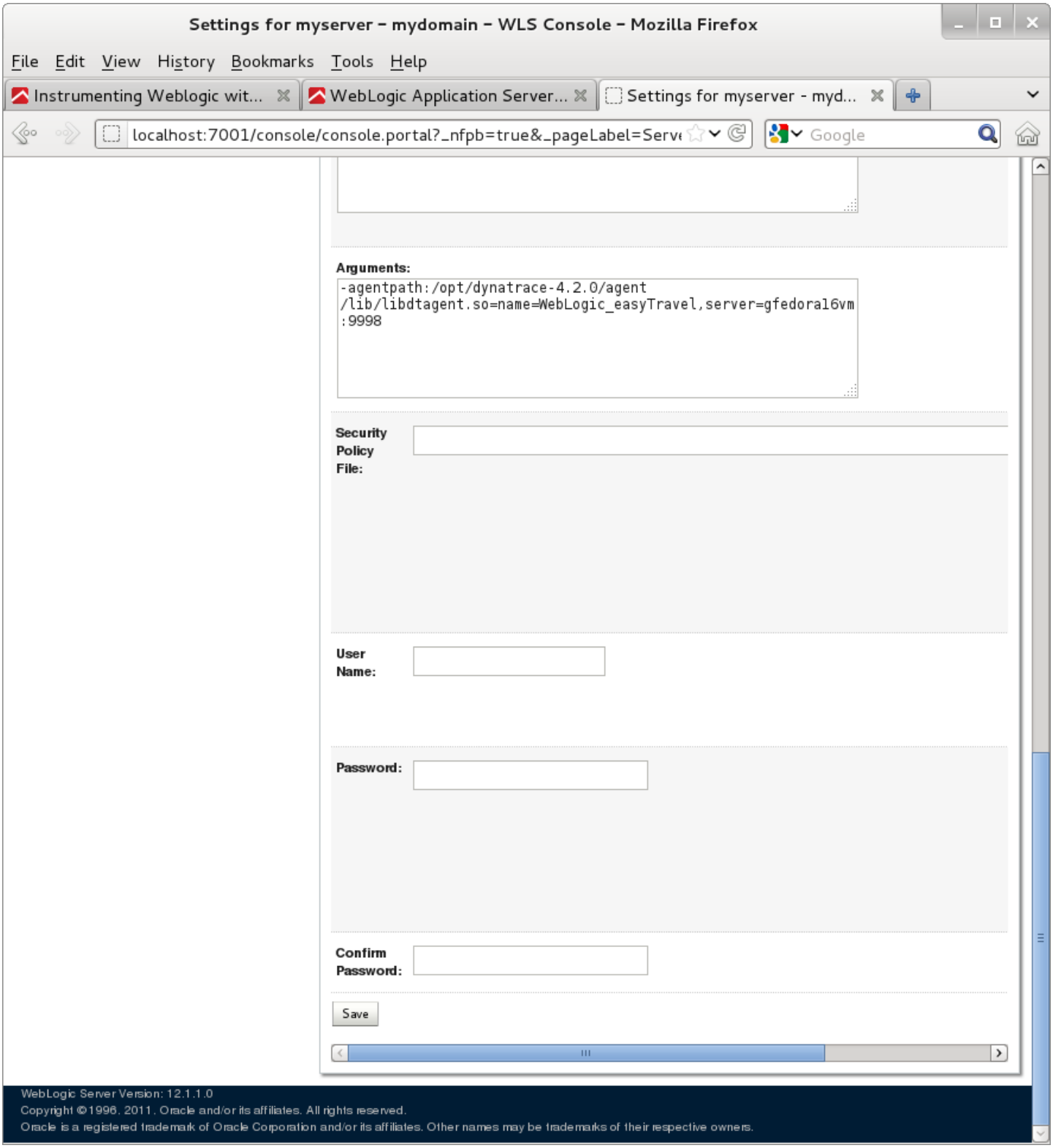
1. Start your WebLogic Admin instance.
2. Start the Administration Console by entering the `<host>:<port>`, for example `localhost:7001`.

3. Through the **Domain Structure** pane, navigate to **<base_domain> > Environment > Servers > <yourManagedServer>**.
4. In the Servers table on the **Configuration** tab in the **Summary of Servers**, select the server.
5. In the **Arguments** field, enter the **Add Tier** wizard´s Agent string as in the following example, where `WebLogic_easyTravel` is an example for `<technology_name>_<SystemProfileName>` and `[gfedora16vm:9998](http://gfedora16vm:9998)` is an example for `<Collector or embedded_in_Server_Collector>:<port>`.

```
[-agentpath:/home/user/dynatrace-]
()7.0/agent/lib/[libdtagent.so]
(http://libdtagent.so)=name=WebLogic_easyTravel,server=gfedora16vm:9
```


Home Page – mydomain – WLS Console – Mozilla Firefox
File Edit View History Bookmarks Tools Help
Instrumenting Weblogic wit...
WebLogic Application Server...
Home Page - mydomain - WL...
localhost:7001/console/console.portal?_nfpb=true&_pageLabel=Hom
Google
ORACLE WebLogic Server Administration Console 12c
Change Center
View changes and restarts
Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.
Domain Structure
mydomain
Environment
Servers
Clusters
Virtual Hosts
Migratable Targets
Coherence Servers
Coherence Clusters
Machines
Work Managers
Startup and Shutdown Classes
Deployments
Services
Security Realms
How do I...
Search the configuration
Use the Change Center
Record WLST Scripts
Change Console preferences
Manage Console extensions
Monitor servers
System Status
Health of Running Servers
Failed (0)
Critical (0)
Overloaded (0)
Warning (0)
OK (1)
Home Log Out Preferences Record Help
Welcome, weblogic Connected to: mydomain
Home
Home Page
Information and Resources
Helpful Tools
Configure applications
Configure GridLink for RAC Data Source
Recent Task Status
Set your console preferences
General Information
Common Administration Task Descriptions
Read the documentation
Ask a question on My Oracle Support
Oracle Guardian Overview
Domain Configurations
Domain
Domain
Environment
Servers
Clusters
Virtual Hosts
Migratable Targets
Coherence Servers
Coherence Clusters
Machines
Work Managers
Startup And Shutdown Classes
Your Deployed Resources
Deployments
Your Application's Security Settings
Security Realms
Services
Messaging
JMS Servers
Store-and-Forward Agents
JMS Modules
Path Services
Bridges
Data Sources
Persistent Stores
XML Registries
XML Entity Caches
Foreign JNDI Providers
Work Contexts
jCOM
Mail Sessions
FileT3
JTA
Interoperability
WTC Servers
Jolt Connection Pools
Diagnostics
Log Files
Diagnostic Modules
Diagnostic Images
Request Performance
Archives
Context
SNMP
Charts and Graphs
Monitoring Dashboard
WebLogic Server Version: 12.1.1.0
Copyright © 1996, 2011, Oracle and/or its affiliates. All rights reserved.


Summary of Servers – mydomain – WLS Console – Mozilla Firefox
File Edit View History Bookmarks Tools Help
Instrumenting Weblogic wit...
WebLogic Application Server...
Summary of Servers - mydo...
localhost:7001/console/console.portal?_nfpb=true&_pageLabel=Core
Google
ORACLE WebLogic Server Administration Console 12c
Change Center
View changes and restarts
Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.
Domain Structure
mydomain
Environment
Servers
Clusters
Virtual Hosts
Migratable Targets
Coherence Servers
Coherence Clusters
Machines
Work Managers
Startup and Shutdown Classes
Deployments
Services
Security Realms
How do I...
Create Managed Servers
Clone Servers
Delete Managed Servers
Delete the Administration Server
Start and stop servers
System Status
Health of Running Servers
Failed (0)
Critical (0)
Overloaded (0)
Warning (0)
OK (1)
Home Log Out Preferences Record Help
Welcome, weblogic Connected to: mydomain
Home >Summary of Servers
Summary of Servers
Configuration Control
A server is an instance of WebLogic Server that runs in its own Java Virtual Machine (JVM) and has its own configuration.
This page summarizes each server that has been configured in the current WebLogic Server domain.
Customize this table
Servers (Filtered - More Columns Exist)
New Clone Delete
Showing 1 to 1 of 1 Previous | Next
Name Cluster Machine State Health Listen Port
myserver(admin) RUNNING OK 7001
New Clone Delete
Showing 1 to 1 of 1 Previous | Next
WebLogic Server Version: 12.1.1.0
Copyright ©1996, 2011, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.
http://localhost:7001/console/console.portal?_nfpb=true&Di...



WebLogic Agent naming

For agents injected to WebLogic, the server name is appended in square brackets right after the agent name, where `serverName` is defined by command line argument `-Dweblogic.Name`.

WebMethods Integration Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

WebMethods Integration Server 9.0+

WebMethods Integration Server 8.2

Related pages

System Requirements

See Set up Agents for information on installing the agent.

32 and 64-bit environments

If you're running a 64-bit WebMethods Integration Server, all `agentpath` parameters must be changed from `.../lib/...` to `.../lib64/...`

32-bit

```
-agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=name=<AgentName>,server=<collectorName>
-agentpath:"<DT_HOME>/agent/lib/libdtagent.so"=name=<AgentName>,server=<collectorName>
```

64-bit

```
-agentpath:"<DT_HOME>\agent\lib64\dtagent.dll"=name=<AgentName>,server=<collectorName>
-agentpath:"<DT_HOME>/agent/lib64/libdtagent.so"=name=<AgentName>,server=<collectorName>
```

WebMethods Integration Server 9.0+

Windows

Modify `<WebMethodsHome>\IntegrationServer\bin\setenv.bat` to include the Agent configuration `-agentpath:<DT_HOME>\agent\lib\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory, as shown below.

```
...
rem ##########################################################################
rem
rem Enter your custom Java options in the variable JAVA_CUSTOM_OPTS
rem e.g. JAVA_CUSTOM_OPTS="-Dmy.prop1=value1 -Dmy.prop2=value2"
rem ##########################################################################

set JAVA_CUSTOM_OPTS=-agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=name=<AgentName>,server

...
```

Unix/Linux

Modify `<WebMethodsHome>/IntegrationServer/bin/setenv.sh` to include the Agent configuration `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory, as shown below.

```
...
##############################################################################
#
# Enter your custom Java options in the variable JAVA_CUSTOM_OPTS
# e.g. JAVA_CUSTOM_OPTS="-Dmy.prop1=value1 -Dmy.prop2=value2"
##############################################################################

JAVA_CUSTOM_OPTS=-agentpath:"<DT_HOME>/agent/lib/libdtagent.so"=name=<AgentName>,server=<

...
```

WebMethods Integration Server 8.2

Windows

Modify `<WebMethodsHome>\IntegrationServer\bin\server.bat` to include the Agent configuration `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`, where `<DT_HOME>` represents the AppMon installation directory, as shown below.

```
...
SET DEBUG_JAVA_OPTS=%DEBUG_JAVA_OPTS% -agentpath:"<DT_HOME>\agent\lib\dtagent.dll"=name=<

SET JAVA_OPTS=-Dsun.lang.ClassLoader.allowArraySyntax=true -Dwatt.server.prepend.classes=

...
```

WebSphere Application Server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Websphere 7.0 – 8.5.5

Troubleshooting

Agent naming for WebSphere

Related pages

Set up agents

System Requirements

See Set up Agents for information on installing the agent.

Websphere 7.0 – 8.5.5

AIX

Do the following to configure AppMon to diagnose applications on WebSphere running on AIX:

1. Start the WebSphere server using the graphical user interface or the command line.
   For example: `/opt/ibm/WebSphere<version>/AppServer/bin/sh startServer.sh server1`.
2. Open the Administrative Console through the graphical user interface, or enter the URL in your web browser: `http://localhost:9060/ibm/console`.
   When accessing the server remotely, specify the machine's hostname rather than `localhost`.
3. Enter your user ID and password, and then log in.

4. Navigate to **Server > Application servers > server1 > Java and Process Management > Process Definition > Java Virtual Machine**.
5. In the **Generic JVM arguments** text field, enter `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`.
6. Apply the changes and save the configuration.
7. Restart the server.

HP-UX

Do the following to configure AppMon to diagnose applications on WebSphere running on HP-UX:

1. Start the WebSphere server using the graphical user interface, or use the command line.
   For example: `/opt/ibm/WebSphere<version>/AppServer/bin/sh startServer.sh server1`.
2. Open the Administrative Console through the graphical user interface, or enter the URL in your web browser: `[http://localhost:9060/ibm/console](http://localhost:9060/ibm/console)`.
   When accessing the server remotely, specify the machine's hostname rather than `localhost`.
3. Enter your user ID and password, and then log in.
4. Navigate to **Server > Application servers > server1 > Java and Process Management > Process Definition > Java Virtual Machine**.
5. In the **Generic JVM arguments** text field, enter:
   - For PA-RISC: `-agentpath:<DT_HOME>/agent/lib/libdtagent.sl=name=<AgentName>,server=<ServerAddress>`.
   - For IA64: `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`.
6. Apply the changes and save the configuration.
7. Restart the server.

z/OS

Do the following to configure AppMon to diagnose applications on WebSphere running on z/OS:

1. Start the WebSphere server using the graphical user interface, or use the command line.
   For example: `/opt/ibm/WebSphere<version>/AppServer/bin/sh startServer.sh server1`.

2. Open the Administrative Console through the graphical user interface or enter the URL in your web browser: `[http://localhost:9060/ibm/console] (http://localhost:9060/ibm/console)`.
   When accessing the server remotely, specify the machine's hostname rather than `localhost`.
3. Enter your user ID and password, and then log in.
4. Navigate to **Server > Application servers > server1 > Java and Process Management > Process Definition > Servant > Java Virtual Machine**.
5. In the **Generic JVM arguments** text field, enter: `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName> -Dcom.ibm.CORBA.NoLocalInterceptors=true`
6. Navigate back to **Servant** and open **Environment Entries**.
7. Add or append the `<DT_HOME>/agent/lib` to the `LIBPATH` variable.
8. Apply changes, save the configuration, and restart the server.

> **Note**
>
> The `-Dcom.ibm.CORBA.NoLocalInterceptors=true` option is recommended because WebSphere for z/OS does not support portable interceptors.

Linux and z/Linux

Do the following to configure AppMon to diagnose applications on WebSphere running on Linux:

1. Start the WebSphere server using the graphical user interface, or use the command line.
   For example: `/opt/ibm/WebSphere<version>/AppServer/bin/sh startServer.sh server1`
2. Open the Administrative Console through the graphical user interface or enter the URL in your web browser: `[http://localhost:9060/ibm/console] (http://localhost:9060/ibm/console)`.
   When accessing the server remotely, specify the machine's hostname rather than `localhost`.
3. Enter your user ID and password, and then log in.
4. Navigate to **Server > Application servers > server1 > Java and Process Management > Process Definition > Java Virtual Machine**.
5. In the **Generic JVM arguments** text field, enter `-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>`.

6. Apply the changes and save the configuration.
7. Restart the server.

Windows

Do the following to configure AppMon to diagnose applications on WebSphere running on Windows:

1. Start the WebSphere server using the Windows **Start** menu, or use the command line.
   For example: `c:\WebSphere<version>\AppServer\bin\startserver server1`.
2. Open the Administrative Console through the graphical user interface, or enter the URL in your web browser: `[http://localhost:9060/ibm/console] (http://localhost:9060/ibm/console)`.
   When accessing the server remotely, specify the machine's hostname rather than `localhost`.
3. Enter your user ID and password, and then log in.
4. Navigate to **Server > Application servers > server1 > Java and Process Management > Process Definition > Java Virtual Machine**.
5. In the **Generic JVM arguments** text field, enter `-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>`.
6. Apply the changes and save the configuration.
7. Restart the server.

Troubleshooting

Configuration files

For WebSphere 6.1 and later:

`<WebSphereHome>/profiles/yourInstanceName/config/cells/yourNodeCell/node`

Log files

For Windows:
`<WebSphereHome>/logs/yourServer/search for native_*.log`

For WebSphere 6.1 and later running on Linux, z/Linux, AIX, HP-UX:
`<WebSphereHome>/profiles/yourInstanceName/logs/yourServer/*.log`

For z/OS:
The logs can be found in the appropriate IBM 3270 terminal job log.

Agent naming for WebSphere

For agents which are injected to WebSphere an additional name is shown in square brackets right after the agent name.

This name has the format [<cellName>-<nodeName>-<serverName>]. The values are either taken from command line, or defined by following environment variables if the command line could not be parsed.

Values taken from the command line are:
`cellName = Websphere cell name` (third of last argument of command line).
`nodeName = Websphere server name` (last argument of command line).

Values taken from environment variables are:
`cell_name`, `node_name`, `server_specific_name`.

The serverName may be optional if not cannot be determined.

WebSphere Liberty Profile

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

WebSphere Liberty Profile 8.5.5.5 - 16.0.0.2

Related pages

System Requirements

Monitor a WAS liberty profile server

See Set up Agents for information on installing the agent.

WebSphere Liberty Profile 8.5.5.5 - 16.0.0.2

Go to `${server.config.dir}` and add the following line to the `jvm.options` file based on whether you're using WWindows or Linux. The `jvm.options` file must be created if it does not exist.

Windows

```
-agentpath:<DT_HOME>\agent\lib\\dtagent.dll=name=<AgentName>,server=<dynatraceCollectorName>
```

Linux

```
-agentpath:<DT_HOME>/agent/lib/libdtagent.so=name=<AgentName>,server=<dynaTraceCollectorName>
```

See Directory locations and properties of WebSphere Liberty for more information.

WebSphere Message Broker

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installation

WebSphere Message Broker 2017 May and higher / WebSphere integration bus 9

Related pages

Related pages

System Requirements

Installation

Install the AppMon Agent package using the appropriate command (based on your OS) to the machine running the System under Diagnosis (SUD).

- On *NIX, use `java -jar packagename.jar`. You are prompted for the target directory, which defaults to your current one.
- On Windows, select the installation type **WebSphere MB** to install only the WebSphere Message Broker Agent. With the full Windows installation package, choose installation type **Custom** and ensure that the installation of the WebSphere Message Broker Agent is enabled.

Make sure that the AppMon Server is running before deploying the AppMon Agent.

WebSphere Message Broker 2017 May and higher / WebSphere integration bus 9

The WebSphere Message Broker Agent is integrated using a *user exit* library (`.lel`) file that must be registered in the WebSphere Message Broker installation.

The default Agent name is **Websphere_MB[<qmgr>-<broker>-< execgrp>]**. To change the prefix to something other than **Websphere_MB**, set the environment variable `DT_AGENTNAME`.

AIX and z/Linux

1. Create the file `{$MQSI_WORKPATH}/common/profiles/dtwsmbagent.sh` with the following contents and change `<server-address>` and `<dthome>` as needed to fit your environment:

```
# !/usr/bin/sh

export DT_SERVER=<server-address>
export DT_HOME=<dthome>
# If not set then DT_AGENTNAME defaults to "WebSphere_MB"
#export DT_AGENTNAME=<agentname-prefix>

# Per default we store the PurePath id in variables/dynaTrace-tag
#To store it in the Message Broker's environment, set variable name/path with e.g.:
#export DT_WSMBTAGPATH=dynaTrace/pathId

if [ "$MQSI_VERSION_V" = "6" ]
then
   export MQSI_USER_EXIT_PATH=${MQSI_USER_EXIT_PATH}:${DT_HOME}/agent/lib
   export MQSI_USER_EXIT_PATH64=${MQSI_USER_EXIT_PATH64}:${DT_HOME}/agent/lib64
else
   export MQSI_USER_EXIT_PATH=${MQSI_USER_EXIT_PATH}:${DT_HOME}/agent/lib64
   export MQSI_USER_EXIT_PATH32=${MQSI_USER_EXIT_PATH32}:${DT_HOME}/agent/lib
fi
```

2. Open a terminal session as user with WebSphere MB administration privileges and ensure that the Message Broker environment was set up (mqsiprofile was executed).

3. Stop WebSphere MB:

```
mqsistop <brokername>
```

4. Activate the Agent in WebSphere MB:

```
mqsichangebroker <brokername> -e dtwsmbagent
```

5. Start WebSphere MB:

```
mqsistart <brokername>
```

Windows

1. Create the file `%MQSI_WORKPATH%\common\profiles\dtwsmbagent.cmd` with the following contents and change `<server-address>` and `<dthome>` for your environment:

```
@echo off
set DT_SERVER=<server-address>
set DT_HOME=<dthome>
rem If not set, DT_AGENTNAME defaults to "WebSphere_MB"
rem SET DT_AGENTNAME=<agentname-prefix>

rem Per default we store the PurePath id in variables/dynaTrace-tag
rem To store it in the Message Broker's environment, set variable name/path with e.g.:
rem set DT_WSMBTAGPATH=dynaTrace/pathId

set MQSI_USER_EXIT_PATH=%MQSI_USER_EXIT_PATH%;%DT_HOME%\agent\lib
```

2. Open the WebSphere MB Command Console (**Start > Programs > IBM WebSphere Message Broker 2017 May > Command Console**).

3. Stop WebSphere MB:

```
mqsistop <brokername>
```

4. Activate the Agent in WebSphere MB:

```
mqsichangebroker <brokername> -e dtwsmbagent
```

5. Start WebSphere MB:

```
mqsistart <brokername>
```

Instrumenting a specific Execution Group only

If you want to implement the Agent on just the execution group and not the broker, use the syntax provided below.

> **Note**
>
> Follow the steps above and only use this if you want to instrument the execution group only.

Postscript: set broker active on a certain execution group by using `mqsichangeflowuserexits`:

```
mqsichangebroker <broker> [-x <path to libdtagent.lel>]
mqsichangeflowuserexits <broker> -e <ExecutionGroup> -a dtwsmbagent
```

Removing the Agent

To remove the Agent from a Message Broker installation, first disable the **dtwsmbagent** user exit. This can be done by setting a list of active user exits that does not contain **dtwsmbagent**.
To disable all user exits, specify an empty list as follows (it has to be executed while the broker is not running):

```
mqsichangebroker <brokername> -e ""
```

After disabling the user exit, the Agent is not used anymore but is still loaded by the Message Broker. To avoid this, remove the Agent from the user exit path by removing the `dtwsmbagent.sh` script (on *NIX) or `.cmd` script (on Windows).

Supported protocols

Tagging requests is supported for the following protocols / node types:

- HTTP: on the node types HTTPInput, HTTPReply, HTTPRequest, SOAPInput, SOAPReply, SOAPRequest, SOAPAsyncRequest, and SOAPAsyncResponse
- JMS: on the node types JMSInput, JMSOutput, JMSReply, MQInput, and MQOutput
- MQ: on the node types MQInput and MQOutput. Currently MQ tagging is only supported within the WS-MB-Agent. Only MQ messages that are put into a queue by WS-MB and are read by WS-MB are tagged.

Message broker PurePath

When looking at a Message Broker PurePath, the PurePath nodes reflect the following information:

- Node Type = Class Name
- Node Name = Method Name
- Message Flow = API

MongoDB

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installation

Usage

Configuration

Supported versions

Related pages

System Requirements

Transaction Flow dashlet

Database dashlet

Database Hotspots dashlet

PurePaths dashlet

Installation

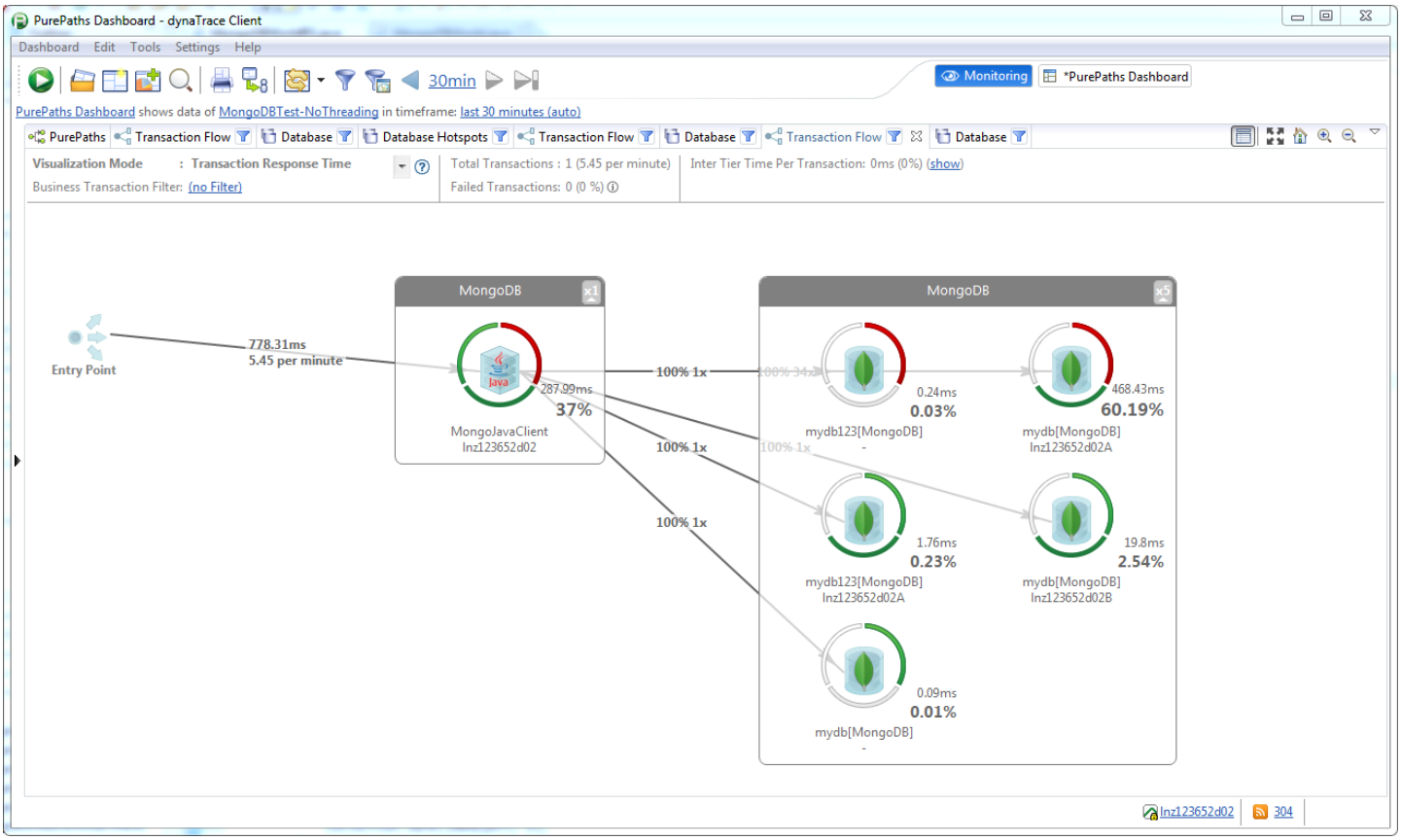
To instrument the MongoDB Client Java API, inject the AppMon Agent and make sure the MongoDB Sensor is placed and active. In new System Profiles, this Sensor is placed and active by default.

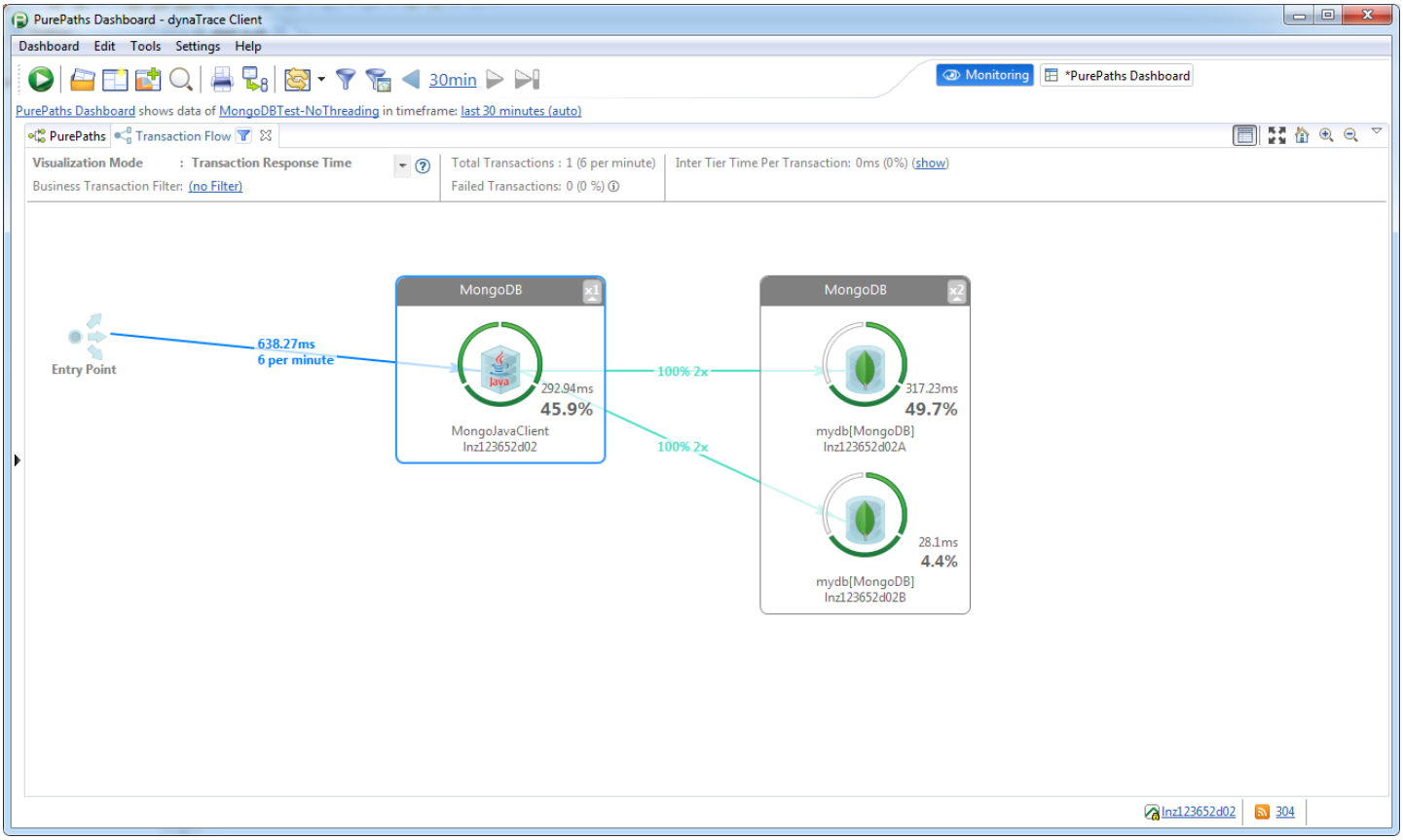
Usage

Transaction Flow

MongoDB calls are generally treated and visualized like database calls. In the Transaction Flow dashlet, every MongoDB instance is rendered as a node.

In the following figures, the time consumed for fetching data from MongoDB is displayed in the MongoDB node to the right.

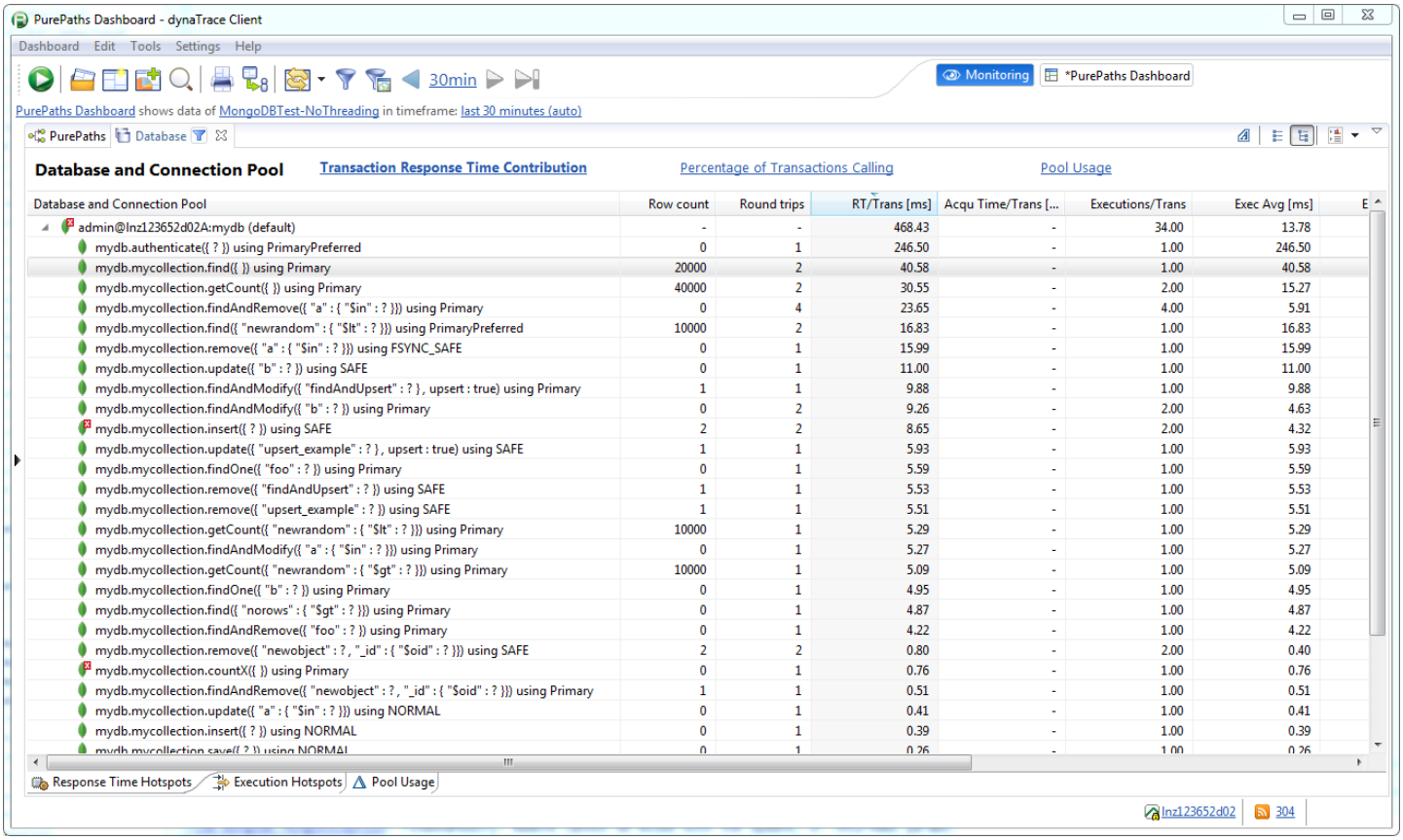


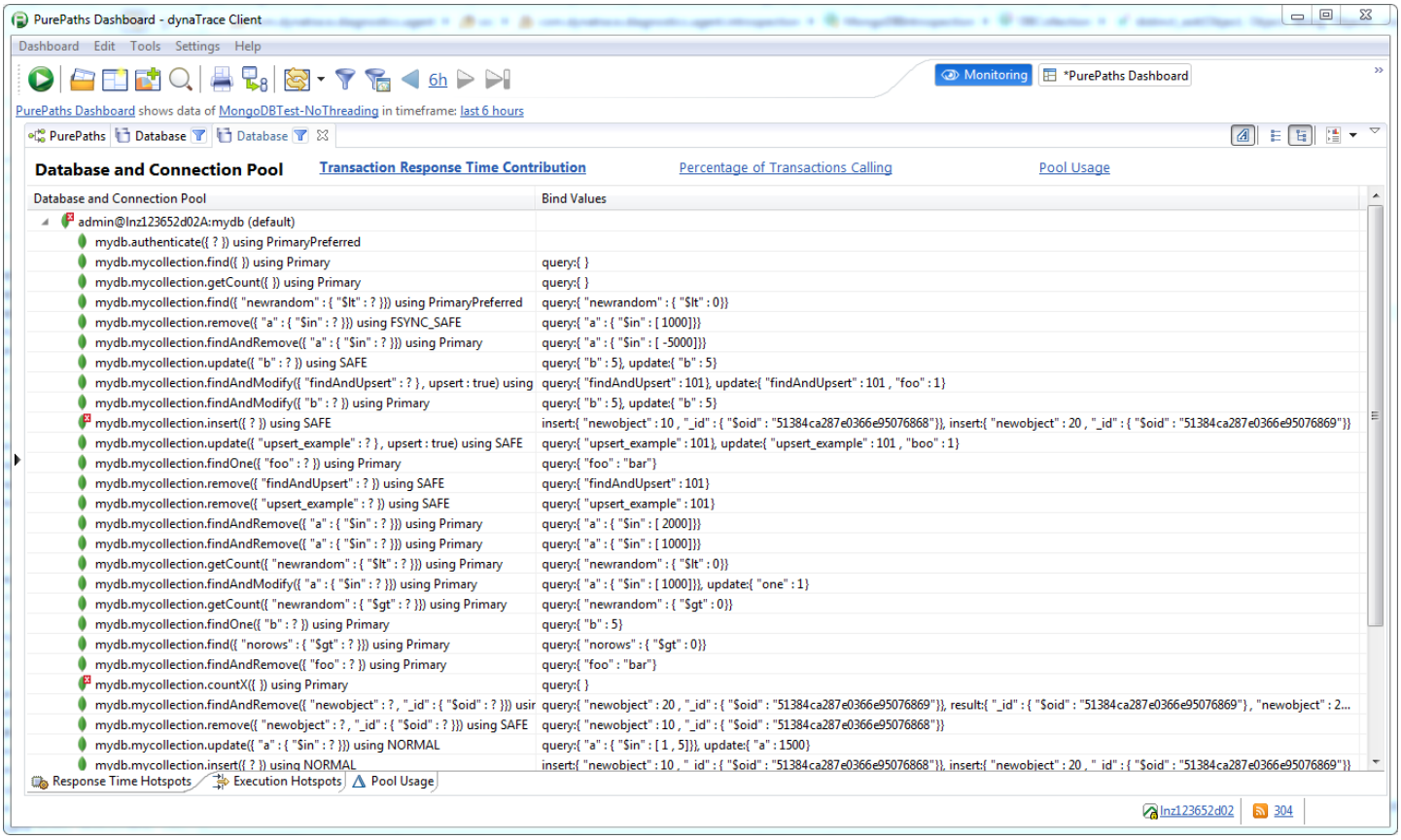


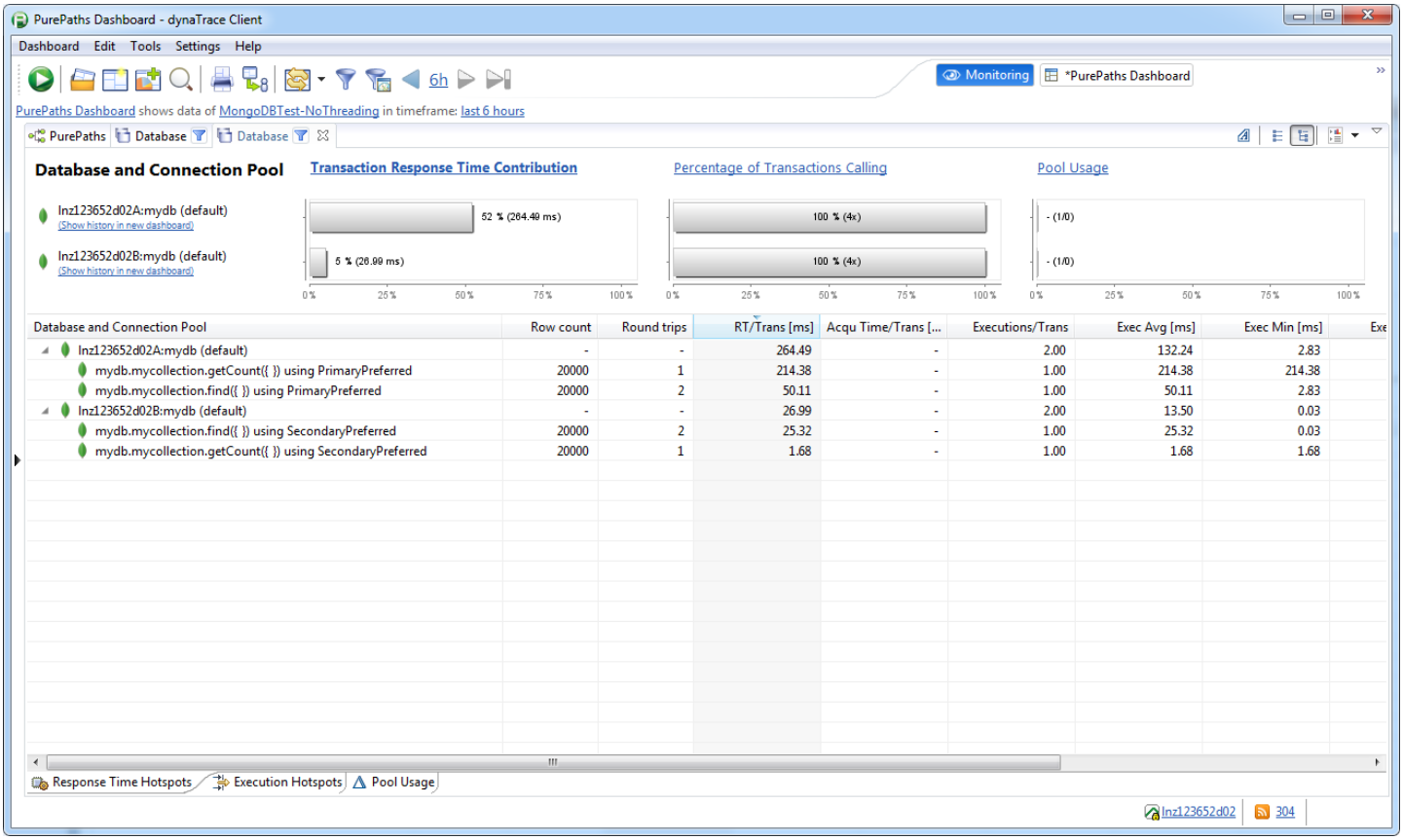
Database

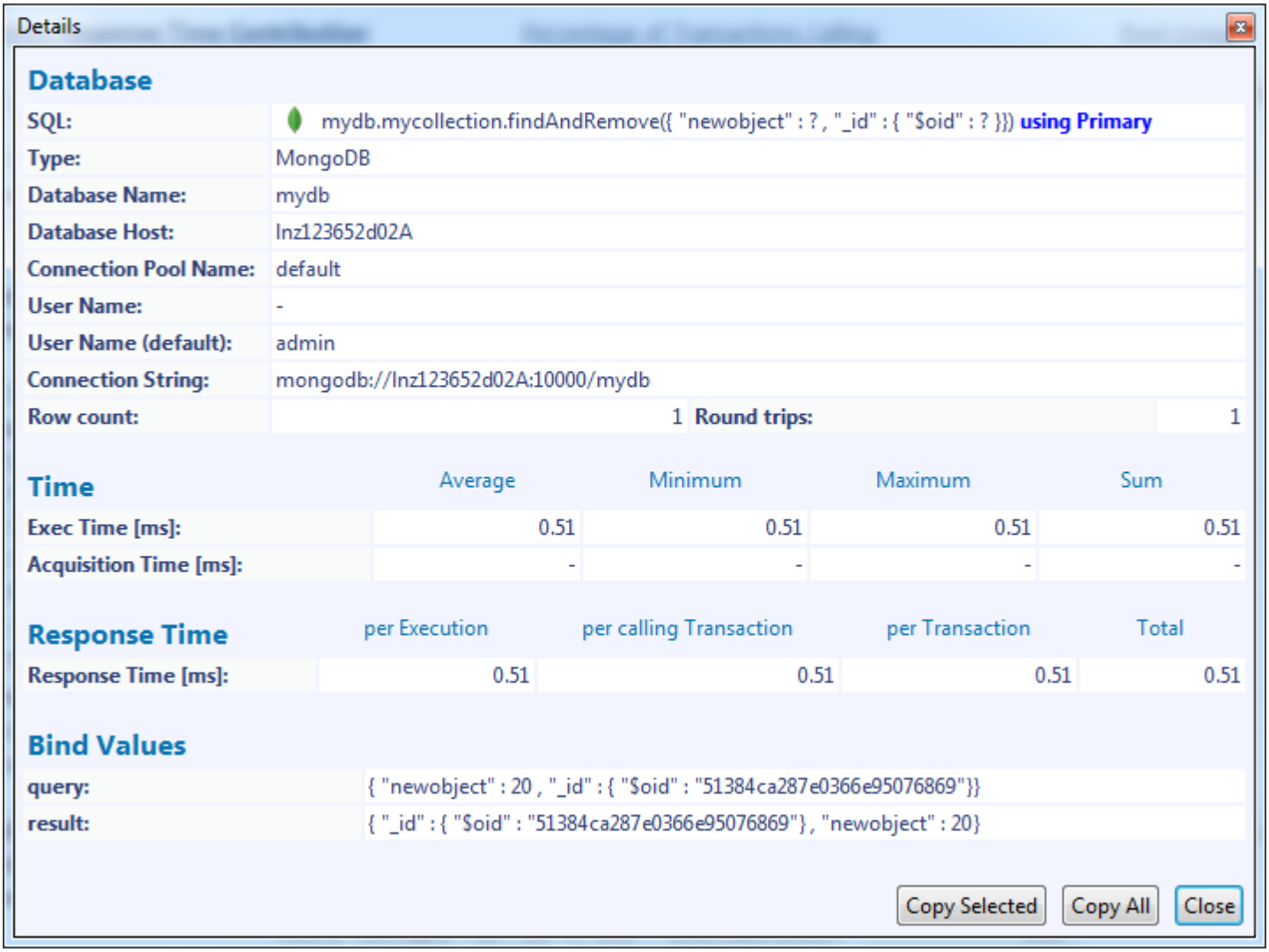
In the Database dashlet, MongoDB calls are displayed in a way very similar to its client (JavaScript) shell. Each connection pool corresponds to a MongoDB instance that groups all calls that were issued against it. This grouping allows analysis if the load is equally distributed among your MongoDB instances in a replica set scenario.

The **Row Count** and **Round Trips** columns display further detail, as shown in the first figure. The second figure shows bind values displayed. See Configuration for more information.


PurePaths Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*PurePaths Dashboard
PurePaths Dashboard shows data of MongoDBTest-NoThreading in timeframe: last 30 minutes (auto)
PurePaths
Database
Database and Connection Pool
Transaction Response Time Contribution
Percentage of Transactions Calling
Pool Usage
Database and Connection Pool
Row count
Round trips
RT/Trans [ms]
Acqu Time/Trans [...
Executions/Trans
Exec Avg [ms]
admin@lnz123652d02A:mydb (default) - - 468.43 - 34.00 13.78
mydb.authenticate({ ? }) using PrimaryPreferred 0 1 246.50 - 1.00 246.50
mydb.mycollection.find({ }) using Primary 20000 2 40.58 - 1.00 40.58
mydb.mycollection.getCount({ }) using Primary 40000 2 30.55 - 2.00 15.27
mydb.mycollection.findAndRemove({ "a" : { "$in" : ? }}) using Primary 0 4 23.65 - 4.00 5.91
mydb.mycollection.find({ "newrandom" : { "$lt" : ? }}) using PrimaryPreferred 10000 2 16.83 - 1.00 16.83
mydb.mycollection.remove({ "a" : { "$in" : ? }}) using FSYNC_SAFE 0 1 15.99 - 1.00 15.99
mydb.mycollection.update({ "b" : ? }) using SAFE 0 1 11.00 - 1.00 11.00
mydb.mycollection.findAndModify({ "findAndUpsert" : ? }, upsert : true) using Primary 1 1 9.88 - 1.00 9.88
mydb.mycollection.findAndModify({ "b" : ? }) using Primary 0 2 9.26 - 2.00 4.63
mydb.mycollection.insert({ ? }) using SAFE 2 2 8.65 - 2.00 4.32
mydb.mycollection.update({ "upsert_example" : ? }, upsert : true) using SAFE 1 1 5.93 - 1.00 5.93
mydb.mycollection.findOne({ "foo" : ? }) using Primary 0 1 5.59 - 1.00 5.59
mydb.mycollection.remove({ "findAndUpsert" : ? }) using SAFE 1 1 5.53 - 1.00 5.53
mydb.mycollection.remove({ "upsert_example" : ? }) using SAFE 1 1 5.51 - 1.00 5.51
mydb.mycollection.getCount({ "newrandom" : { "$lt" : ? }}) using Primary 10000 1 5.29 - 1.00 5.29
mydb.mycollection.findAndModify({ "a" : { "$in" : ? }}) using Primary 0 1 5.27 - 1.00 5.27
mydb.mycollection.getCount({ "newrandom" : { "$gt" : ? }}) using Primary 10000 1 5.09 - 1.00 5.09
mydb.mycollection.findOne({ "b" : ? }) using Primary 0 1 4.95 - 1.00 4.95
mydb.mycollection.find({ "norows" : { "$gt" : ? }}) using Primary 0 1 4.87 - 1.00 4.87
mydb.mycollection.findAndRemove({ "foo" : ? }) using Primary 0 1 4.22 - 1.00 4.22
mydb.mycollection.remove({ "newobject" : ?, "_id" : { "$oid" : ? }}) using SAFE 2 2 0.80 - 2.00 0.40
mydb.mycollection.countX({ }) using Primary 0 1 0.76 - 1.00 0.76
mydb.mycollection.findAndRemove({ "newobject" : ?, "_id" : { "$oid" : ? }}) using Primary 1 1 0.51 - 1.00 0.51
mydb.mycollection.update({ "a" : { "$in" : ? }}) using NORMAL 0 1 0.41 - 1.00 0.41
mydb.mycollection.insert({ ? }) using NORMAL 0 1 0.39 - 1.00 0.39
Response Time Hotspots
Execution Hotspots
Pool Usage
lnz123652d02
304


PurePaths Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
6h
Monitoring
*PurePaths Dashboard
PurePaths Dashboard shows data of MongoDBTest-NoThreading in timeframe: last 6 hours
PurePaths
Database
Database
Database and Connection Pool
Transaction Response Time Contribution
Percentage of Transactions Calling
Pool Usage
Database and Connection Pool
Bind Values
admin@lnz123652d02A:mydb (default)
mydb.authenticate({ ? }) using PrimaryPreferred
mydb.mycollection.find({ }) using Primary query:{ }
mydb.mycollection.getCount({ }) using Primary query:{ }
mydb.mycollection.find({ "newrandom" : { "$lt" : ? }}) using PrimaryPreferred query:{ "newrandom" : { "$lt" : 0}}
mydb.mycollection.remove({ "a" : { "$in" : ? }}) using FSYNC_SAFE query:{ "a" : { "$in" : [1000]}}
mydb.mycollection.findAndRemove({ "a" : { "$in" : ? }}) using Primary query:{ "a" : { "$in" : [-5000]}}
mydb.mycollection.update({ "b" : ? }) using SAFE query:{ "b" : 5}, update:{ "b" : 5}
mydb.mycollection.findAndModify({ "findAndUpsert" : ? }, upsert : true) using query:{ "findAndUpsert" : 101}, update:{ "findAndUpsert" : 101 , "foo" : 1}
mydb.mycollection.findAndModify({ "b" : ? }) using Primary query:{ "b" : 5}, update:{ "b" : 5}
mydb.mycollection.insert({ ? }) using SAFE insert:{ "newobject" : 10 , "_id" : { "$oid" : "51384ca287e0366e95076868"}}, insert:{ "newobject" : 20 , "_id" : { "$oid" : "51384ca287e0366e95076869"}}
mydb.mycollection.update({ "upsert_example" : ? }, upsert : true) using SAFE query:{ "upsert_example" : 101}, update:{ "upsert_example" : 101 , "boo" : 1}
mydb.mycollection.findOne({ "foo" : ? }) using Primary query:{ "foo" : "bar"}
mydb.mycollection.remove({ "findAndUpsert" : ? }) using SAFE query:{ "findAndUpsert" : 101}
mydb.mycollection.remove({ "upsert_example" : ? }) using SAFE query:{ "upsert_example" : 101}
mydb.mycollection.findAndRemove({ "a" : { "$in" : ? }}) using Primary query:{ "a" : { "$in" : [2000]}}
mydb.mycollection.findAndRemove({ "a" : { "$in" : ? }}) using Primary query:{ "a" : { "$in" : [1000]}}
mydb.mycollection.getCount({ "newrandom" : { "$lt" : ? }}) using Primary query:{ "newrandom" : { "$lt" : 0}}
mydb.mycollection.findAndModify({ "a" : { "$in" : ? }}) using Primary query:{ "a" : { "$in" : [1000]}}, update:{ "one" : 1}
mydb.mycollection.getCount({ "newrandom" : { "$gt" : ? }}) using Primary query:{ "newrandom" : { "$gt" : 0}}
mydb.mycollection.findOne({ "b" : ? }) using Primary query:{ "b" : 5}
mydb.mycollection.find({ "norows" : { "$gt" : ? }}) using Primary query:{ "norows" : { "$gt" : 0}}
mydb.mycollection.findAndRemove({ "foo" : ? }) using Primary query:{ "foo" : "bar"}
mydb.mycollection.countX({ }) using Primary query:{ }
mydb.mycollection.findAndRemove({ "newobject" : ?, "_id" : { "$oid" : ? }}) usir query:{ "newobject" : 20 , "_id" : { "$oid" : "51384ca287e0366e95076869"}}, result:{ "_id" : { "$oid" : "51384ca287e0366e95076869"} , "newobject" : 2...
mydb.mycollection.remove({ "newobject" : ?, "_id" : { "$oid" : ? }}) using SAFE query:{ "newobject" : 10 , "_id" : { "$oid" : "51384ca287e0366e95076868"}}
mydb.mycollection.update({ "a" : { "$in" : ? }}) using NORMAL query:{ "a" : { "$in" : [1 , 5]}}, update:{ "a" : 1500}
mydb.mycollection.insert({ ? }) using NORMAL insert:{ "newobject" : 10 , "_id" : { "$oid" : "51384ca287e0366e95076868"}}, insert:{ "newobject" : 20 , "_id" : { "$oid" : "51384ca287e0366e95076869"}}
Response Time Hotspots
Execution Hotspots
Pool Usage
lnz123652d02
304


PurePaths Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
Monitoring
*PurePaths Dashboard
PurePaths Dashboard shows data of MongoDBTest-NoThreading in timeframe: last 6 hours
PurePaths Database Database
Database and Connection Pool
Transaction Response Time Contribution
Percentage of Transactions Calling
Pool Usage
lnz123652d02A:mydb (default)
(Show history in new dashboard)
lnz123652d02B:mydb (default)
(Show history in new dashboard)
52 % (264.49 ms)
5 % (26.99 ms)
100 % (4x)
100 % (4x)
- (1/0)
- (1/0)
Database and Connection Pool
Row count
Round trips
RT/Trans [ms]
Acqu Time/Trans [...
Executions/Trans
Exec Avg [ms]
Exec Min [ms]
lnz123652d02A:mydb (default) - - 264.49 - 2.00 132.24 2.83
mydb.mycollection.getCount({ }) using PrimaryPreferred 20000 1 214.38 - 1.00 214.38 214.38
mydb.mycollection.find({ }) using PrimaryPreferred 20000 2 50.11 - 1.00 50.11 2.83
lnz123652d02B:mydb (default) - - 26.99 - 2.00 13.50 0.03
mydb.mycollection.find({ }) using SecondaryPreferred 20000 2 25.32 - 1.00 25.32 0.03
mydb.mycollection.getCount({ }) using SecondaryPreferred 20000 1 1.68 - 1.00 1.68 1.68
Response Time Hotspots
Execution Hotspots
Pool Usage
lnz123652d02
304



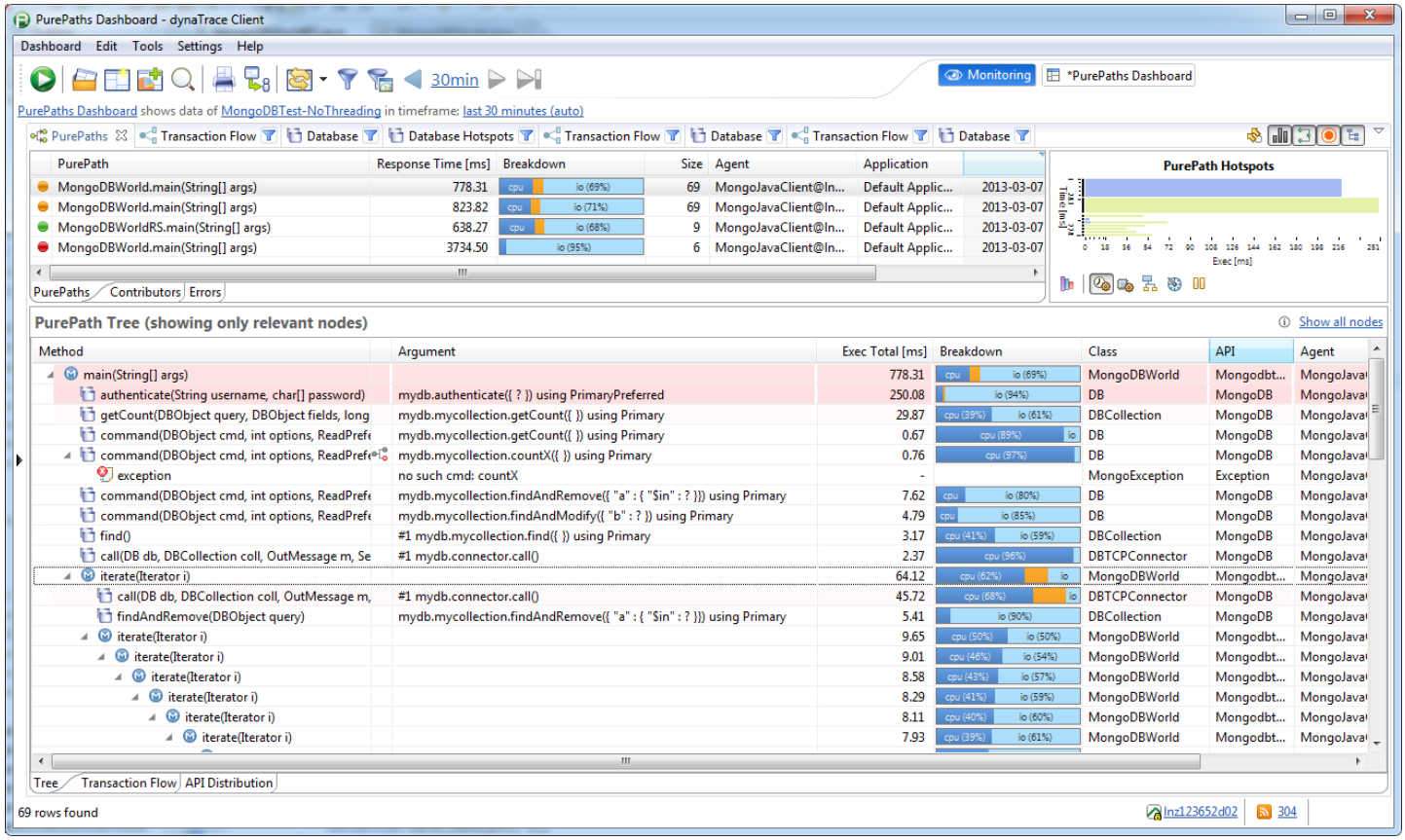
The Database Hotspots dashlet shows where your MongoDB calls are being made.

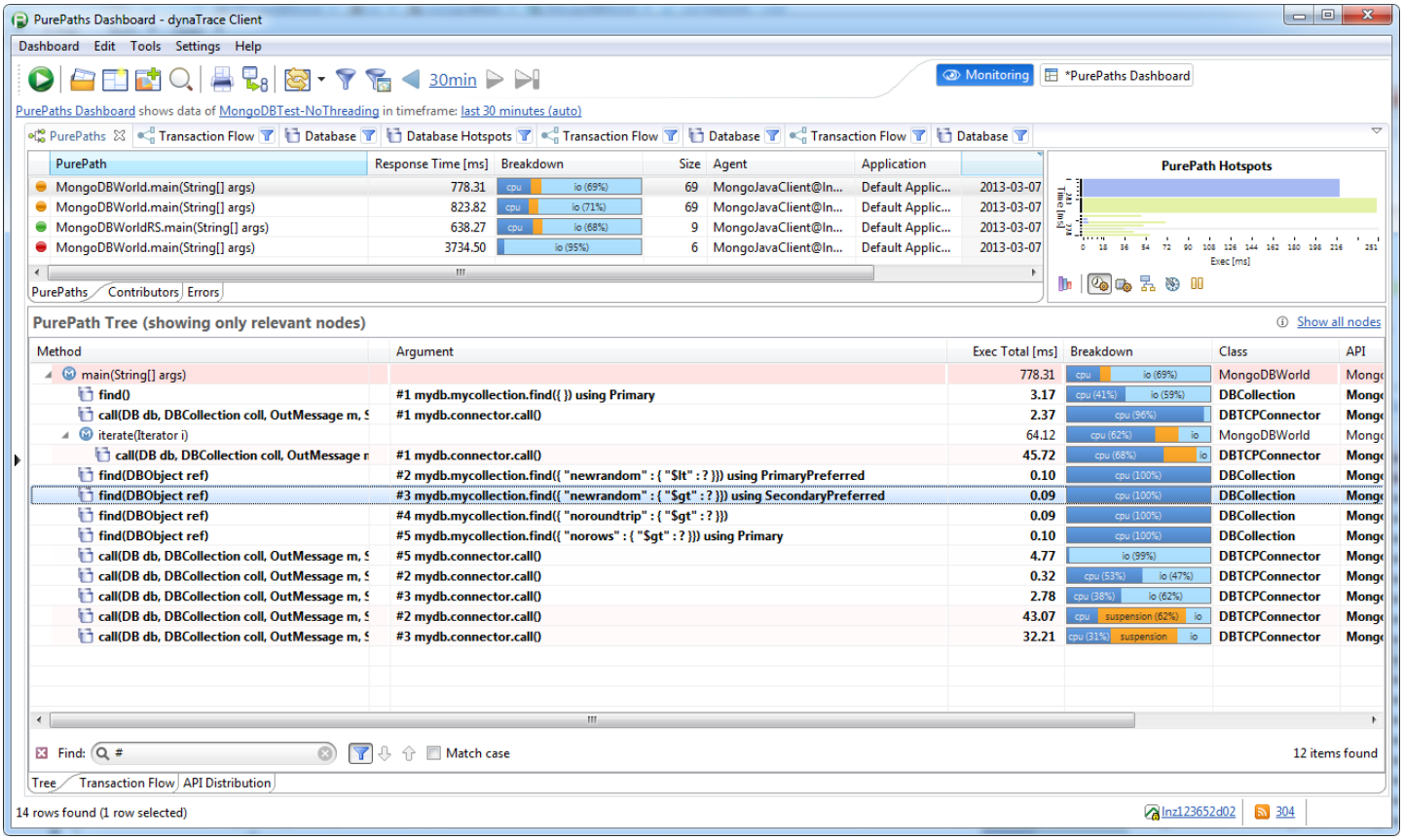
PurePaths

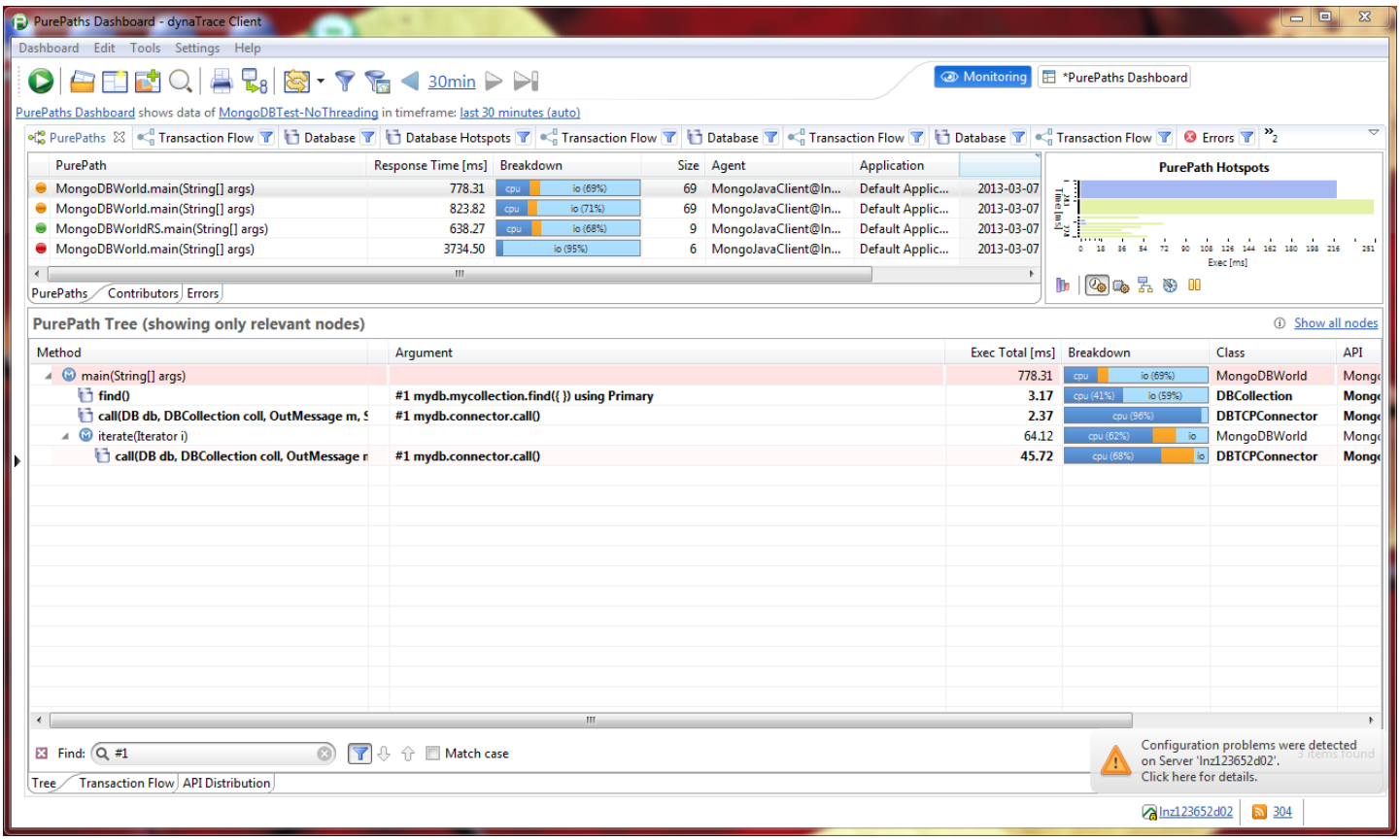
You can view MongoDB calls in the PurePaths dashlet similar to other database calls. The `find()` call is treated as a special case, because it does not actually send or retrieve anything from the MongoDB instance. Sending and receiving only happen after the application code accesses the result set — when the `call()` method is executed (the actual database fetch) and retrieves a part of the result set from the database.

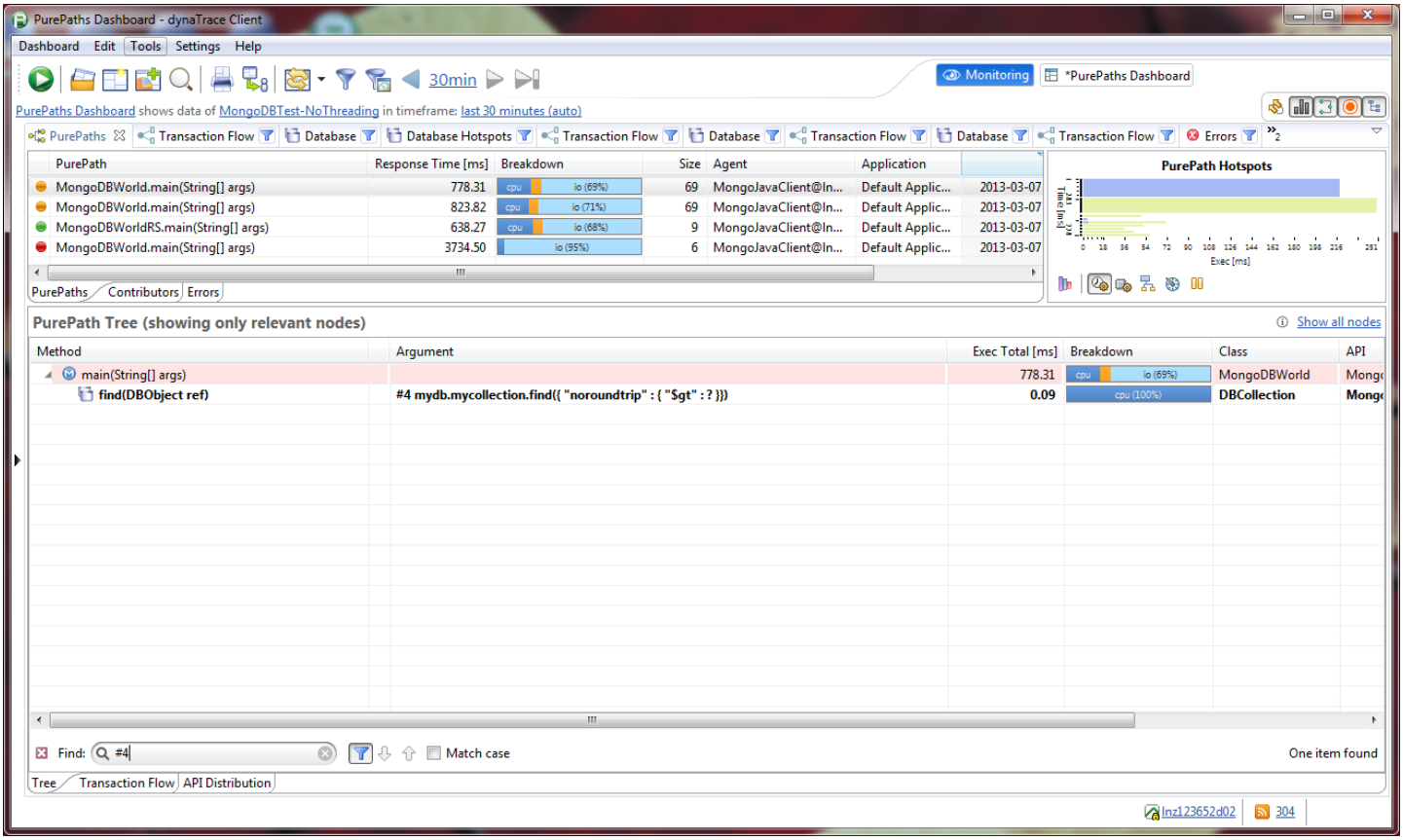
One `find()` call can result in multiple `call()` calls (multiple fetches/round trips). Each `find()` is assigned a number, starting from 1, in the **Argument** column. The `call()` node's **Argument** column features the number that corresponds to the `find()`. You can use this number to identify which calls belong to which `find()`.

The time consumed by one `find()` operation can be determined by adding all execution times, for example, of `find`/`call` nodes annotated with number assigned to the `find`, and subtracting the suspension time. The result is the database time. This is exactly the time that is displayed in the **Database** dashlet. All other MongoDB operations are simple method invocations and are displayed as such, with a database icon.




PurePaths Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*PurePaths Dashboard
PurePaths Dashboard shows data of MongoDBTest-NoThreading in timeframe: last 30 minutes (auto)
PurePaths Transaction Flow Database Database Hotspots Transaction Flow Database Transaction Flow Database
PurePath Response Time [ms] Breakdown Size Agent Application
MongoDBWorld.main(String[] args) 778.31 69 MongoJavaClient@In... Default Applic... 2013-03-07
MongoDBWorld.main(String[] args) 823.82 69 MongoJavaClient@In... Default Applic... 2013-03-07
MongoDBWorldRS.main(String[] args) 638.27 9 MongoJavaClient@In... Default Applic... 2013-03-07
MongoDBWorld.main(String[] args) 3734.50 6 MongoJavaClient@In... Default Applic... 2013-03-07
PurePath Hotspots
PurePaths Contributors Errors
PurePath Tree (showing only relevant nodes)
Show all nodes
Method Argument Exec Total [ms] Breakdown Class API
main(String[] args) 778.31 MongoDBWorld
find() #1 mydb.mycollection.find({ }) using Primary 3.17 DBCollection
call(DB db, DBCollection coll, OutMessage m, S #1 mydb.connector.call() 2.37 DBTCPConnector
iterate(Iterator i) 64.12 MongoDBWorld
call(DB db, DBCollection coll, OutMessage r #1 mydb.connector.call() 45.72 DBTCPConnector
find(DBObject ref) #2 mydb.mycollection.find({ "newrandom" : { "$lt" : ? }}) using PrimaryPreferred 0.10 DBCollection
find(DBObject ref) #3 mydb.mycollection.find({ "newrandom" : { "$gt" : ? }}) using SecondaryPreferred 0.09 DBCollection
find(DBObject ref) #4 mydb.mycollection.find({ "noroundtrip" : { "$gt" : ? }}) 0.09 DBCollection
find(DBObject ref) #5 mydb.mycollection.find({ "norows" : { "$gt" : ? }}) using Primary 0.10 DBCollection
call(DB db, DBCollection coll, OutMessage m, S #5 mydb.connector.call() 4.77 DBTCPConnector
call(DB db, DBCollection coll, OutMessage m, S #2 mydb.connector.call() 0.32 DBTCPConnector
call(DB db, DBCollection coll, OutMessage m, S #3 mydb.connector.call() 2.78 DBTCPConnector
call(DB db, DBCollection coll, OutMessage m, S #2 mydb.connector.call() 43.07 DBTCPConnector
call(DB db, DBCollection coll, OutMessage m, S #3 mydb.connector.call() 32.21 DBTCPConnector
Find: #
Match case
12 items found
Tree Transaction Flow API Distribution
14 rows found (1 row selected)
lnz123652d02
304

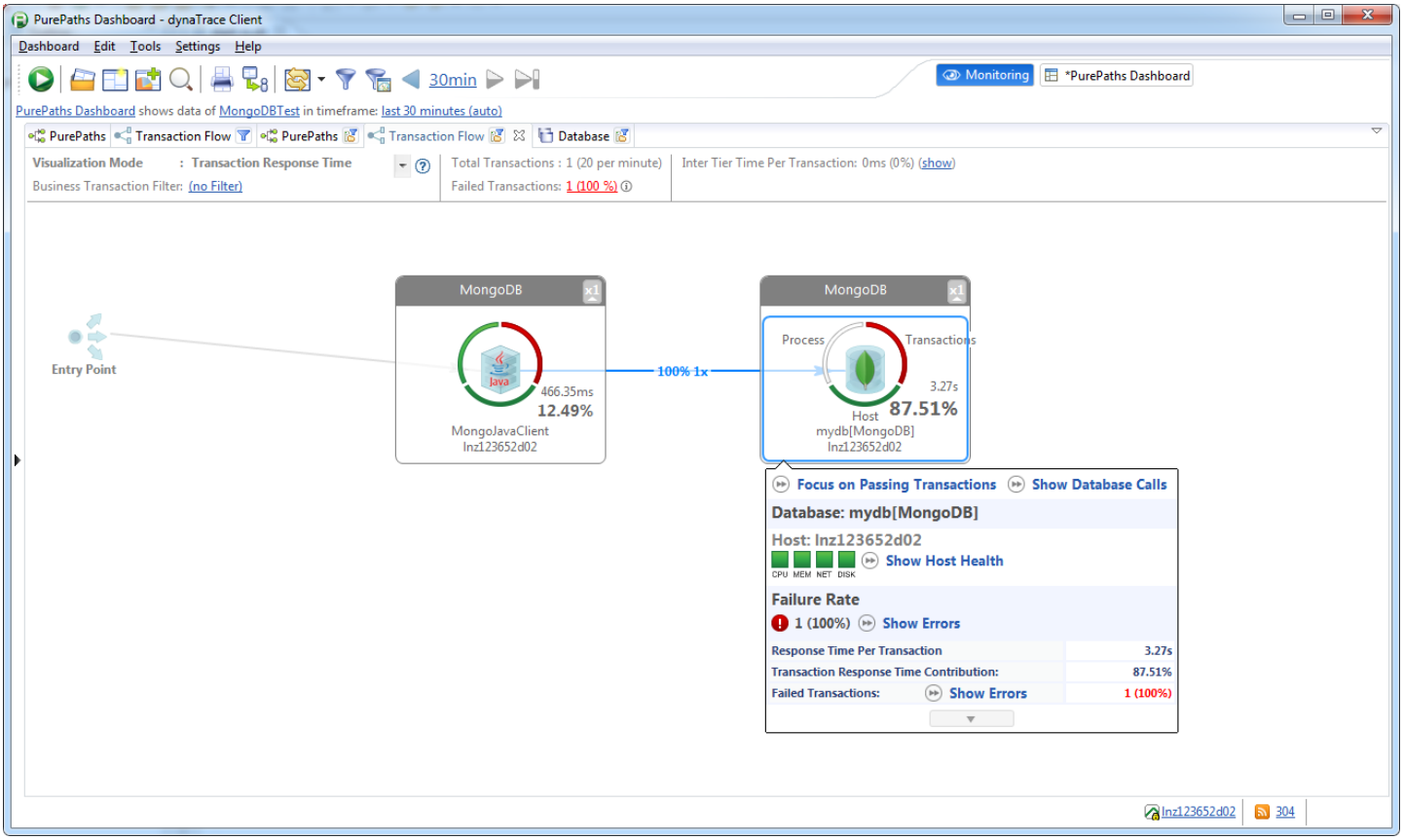

PurePaths Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*PurePaths Dashboard
PurePaths Dashboard shows data of MongoDBTest-NoThreading in timeframe: last 30 minutes (auto)
PurePaths Transaction Flow Database Database Hotspots Transaction Flow Database Transaction Flow Database Transaction Flow Errors
PurePath Response Time [ms] Breakdown Size Agent Application
MongoDBWorld.main(String[] args) 778.31 69 MongoJavaClient@In... Default Applic... 2013-03-07
MongoDBWorld.main(String[] args) 823.82 69 MongoJavaClient@In... Default Applic... 2013-03-07
MongoDBWorldRS.main(String[] args) 638.27 9 MongoJavaClient@In... Default Applic... 2013-03-07
MongoDBWorld.main(String[] args) 3734.50 6 MongoJavaClient@In... Default Applic... 2013-03-07
PurePath Hotspots
PurePaths Contributors Errors
PurePath Tree (showing only relevant nodes)
Show all nodes
Method Argument Exec Total [ms] Breakdown Class API
main(String[] args) 778.31 MongoDBWorld
find() #1 mydb.mycollection.find({ }) using Primary 3.17 DBCollection
call(DB db, DBCollection coll, OutMessage m, S #1 mydb.connector.call() 2.37 DBTCPConnector
iterate(Iterator i) 64.12 MongoDBWorld
call(DB db, DBCollection coll, OutMessage r #1 mydb.connector.call() 45.72 DBTCPConnector
Find: #1
Match case
Configuration problems were detected on Server 'lnz123652d02'.
Click here for details.
Tree Transaction Flow API Distribution
lnz123652d02
304

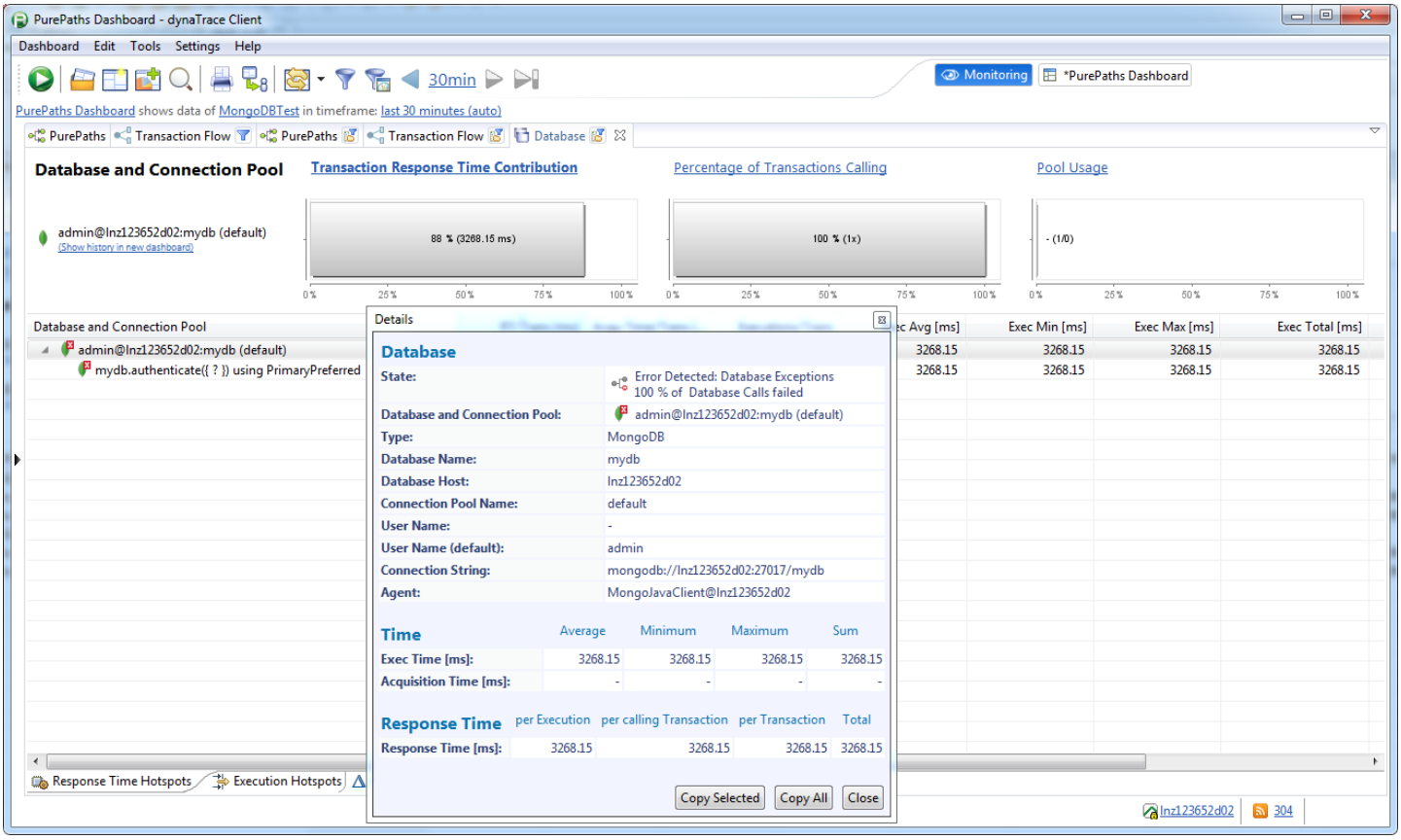


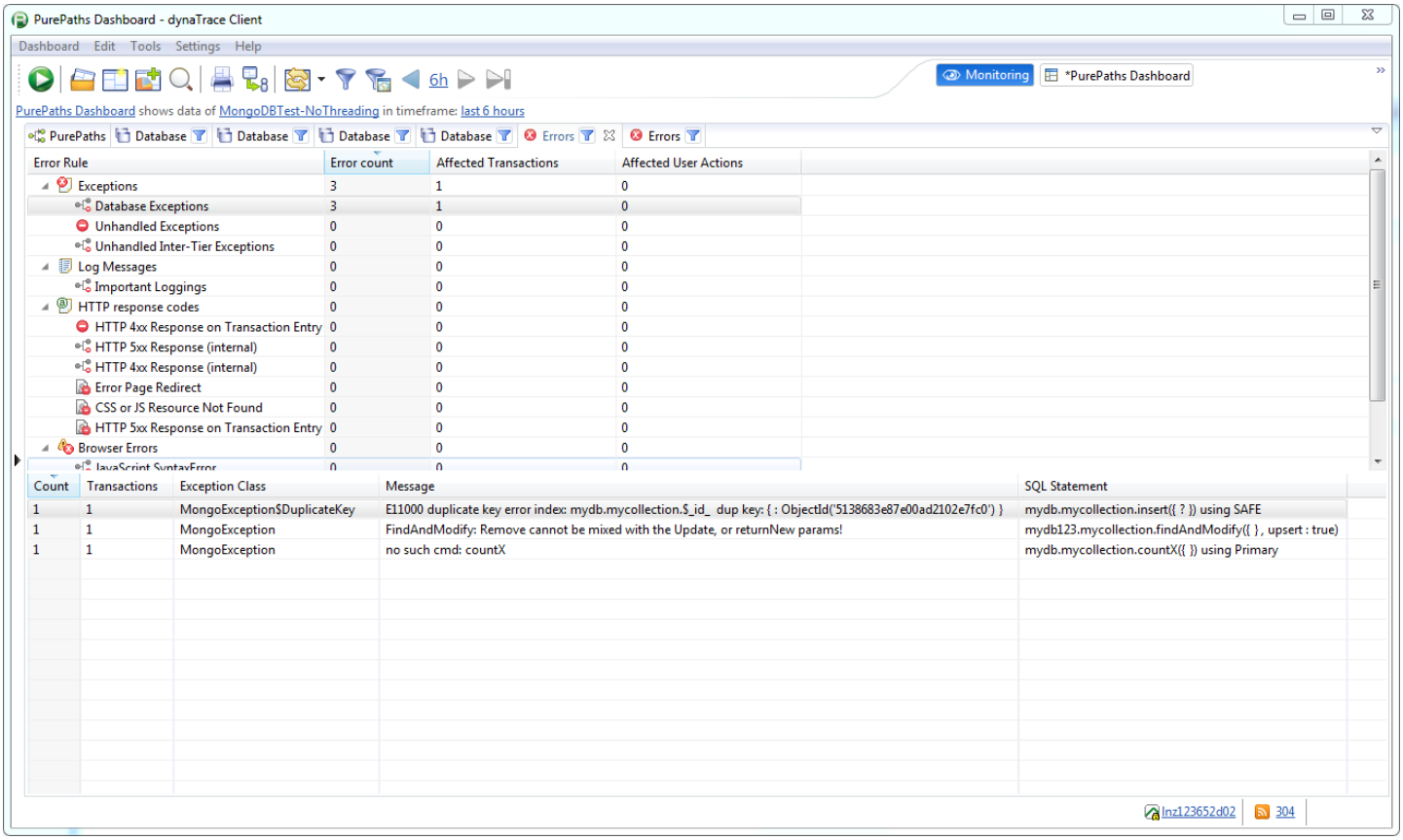
Error detection

To be able to detect all errors, make sure the Java Exception Sensor is placed and active. Errors are visualized in all the dashlets described above. You can drill down to the **Errors** dashlet from each location.

The following figures demonstrate a drilldown from the **Transaction Flow** and **Database** dashlets to the **Errors** dashlet.


PurePaths Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*PurePaths Dashboard
PurePaths Dashboard shows data of MongoDBTest in timeframe: last 30 minutes (auto)
PurePaths Transaction Flow PurePaths Transaction Flow Database
Visualization Mode : Transaction Response Time
Business Transaction Filter: (no Filter)
Total Transactions : 1 (20 per minute)
Failed Transactions: 1 (100 %)
Inter Tier Time Per Transaction: 0ms (0%) (show)
Entry Point
MongoDB
466.35ms
12.49%
MongoJavaClient
lnz123652d02
100% 1x
MongoDB
Process
Transactions
3.27s
Host
87.51%
mydb[MongoDB]
lnz123652d02
Focus on Passing Transactions
Show Database Calls
Database: mydb[MongoDB]
Host: lnz123652d02
CPU MEM NET DISK
Show Host Health
Failure Rate
1 (100%)
Show Errors
Response Time Per Transaction 3.27s
Transaction Response Time Contribution: 87.51%
Failed Transactions: Show Errors 1 (100%)
lnz123652d02
304


PurePaths Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*PurePaths Dashboard
PurePaths Dashboard shows data of MongoDBTest in timeframe: last 30 minutes (auto)
PurePaths Transaction Flow PurePaths Transaction Flow Database
Database and Connection Pool
Transaction Response Time Contribution
Percentage of Transactions Calling
Pool Usage
admin@lnz123652d02:mydb (default)
(Show history in new dashboard)
88 % (3268.15 ms)
100 % (1x)
- (1/0)
Database and Connection Pool
admin@lnz123652d02:mydb (default)
mydb.authenticate({ ? }) using PrimaryPreferred
Exec Avg [ms] Exec Min [ms] Exec Max [ms] Exec Total [ms]
3268.15 3268.15 3268.15 3268.15
3268.15 3268.15 3268.15 3268.15
Details
Database
State: Error Detected: Database Exceptions
100 % of Database Calls failed
Database and Connection Pool: admin@lnz123652d02:mydb (default)
Type: MongoDB
Database Name: mydb
Database Host: lnz123652d02
Connection Pool Name: default
User Name: -
User Name (default): admin
Connection String: mongodb://lnz123652d02:27017/mydb
Agent: MongoJavaClient@lnz123652d02
Time Average Minimum Maximum Sum
Exec Time [ms]: 3268.15 3268.15 3268.15 3268.15
Acquisition Time [ms]: - - - -
Response Time per Execution per calling Transaction per Transaction Total
Response Time [ms]: 3268.15 3268.15 3268.15 3268.15
Copy Selected
Copy All
Close
Response Time Hotspots
Execution Hotspots
lnz123652d02
304



Configuration

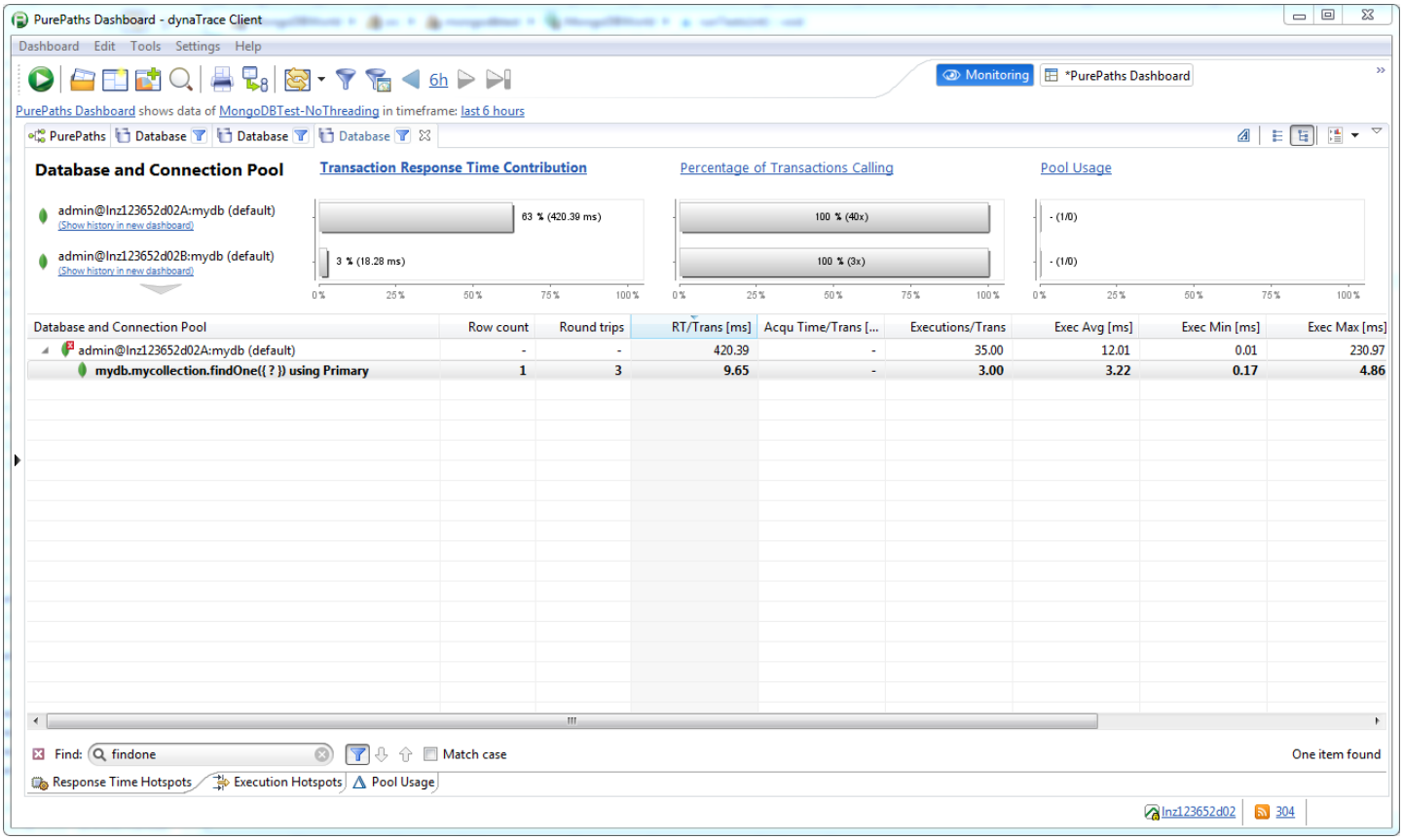
To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **MongoDB** item in the **Sensor Configuration** list.

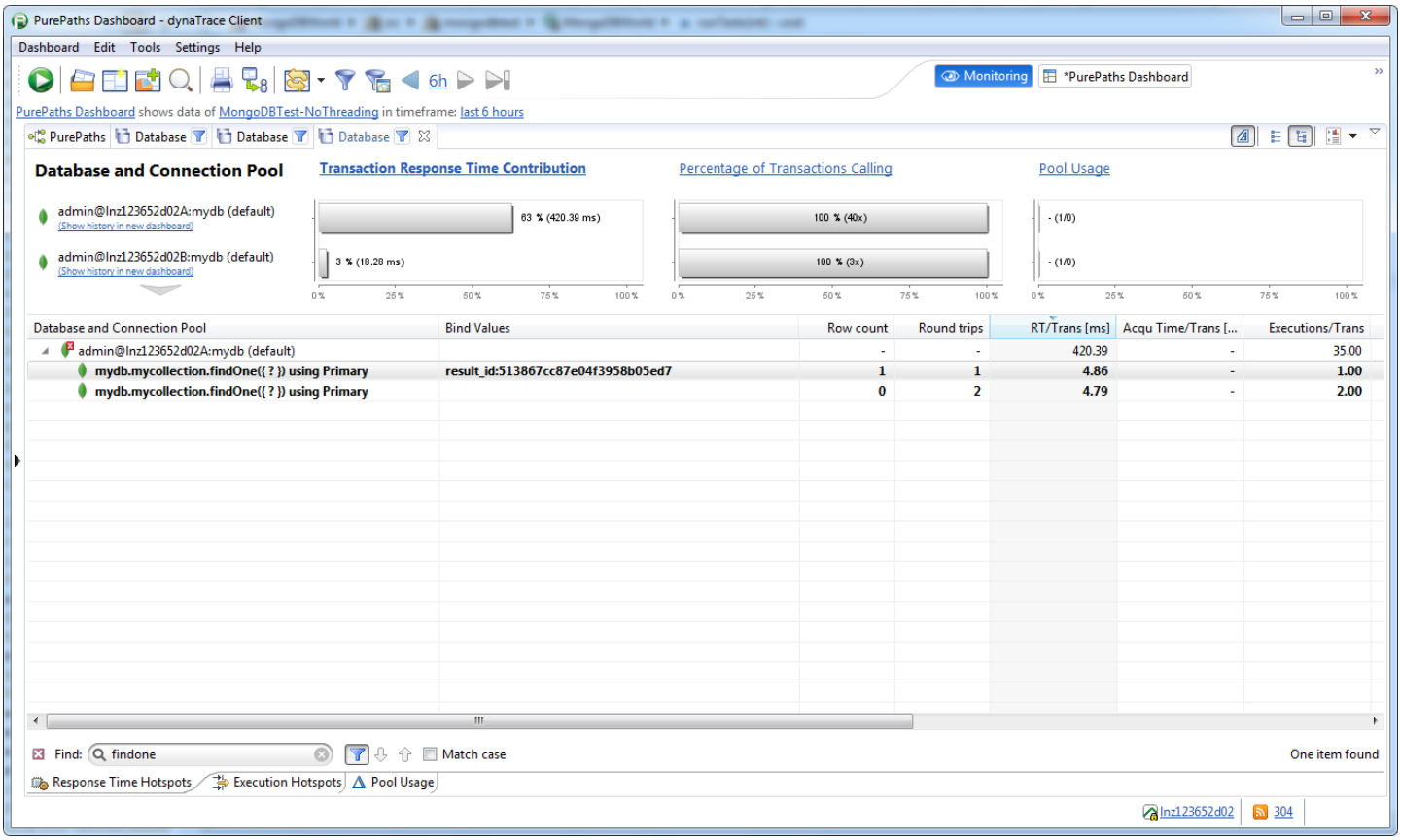
- **Enable extended capturing**: Enables capturing of detailed information about JSON query strings or document IDs.
- **Capture only document id(s)**: Capture document IDs where available.
- **Capture full JSON objects(s) (queries and documents)**: Capture all JSON objects as serialized strings.
- **Maximum string length**: Limits the maximum length of the captured JSON strings. Exceeding symbols are cut.

In either case, you can view the details of a node in the **Details** popup windows of the **PurePaths** and **Database** dashlets. To display the details in the **Database** dashlet, you have to enable bind values grouping in the appropriate Sensor. The captured data may also be called **Bind Values**.

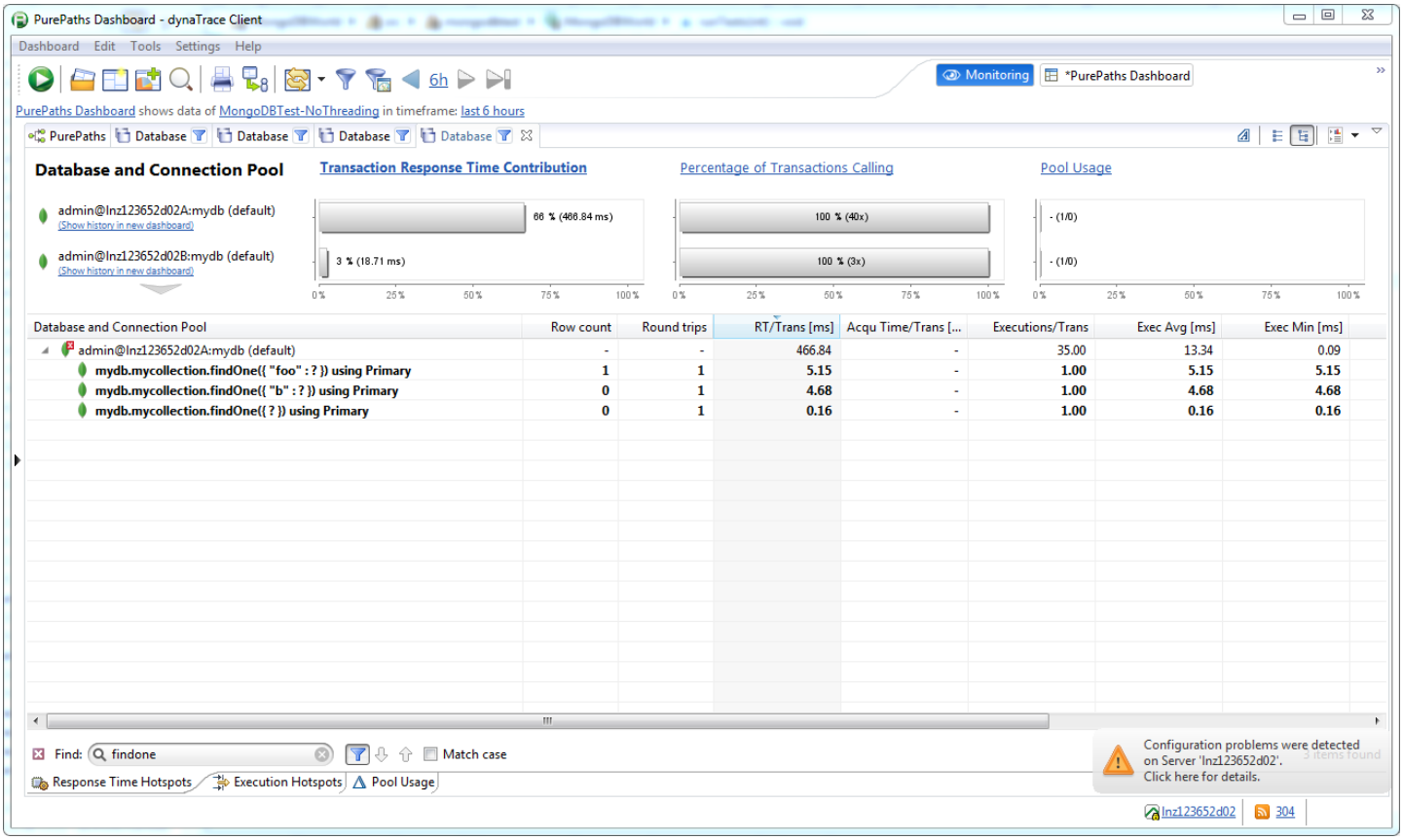
The following figures show a simple `findOne()` operation in different configurations.

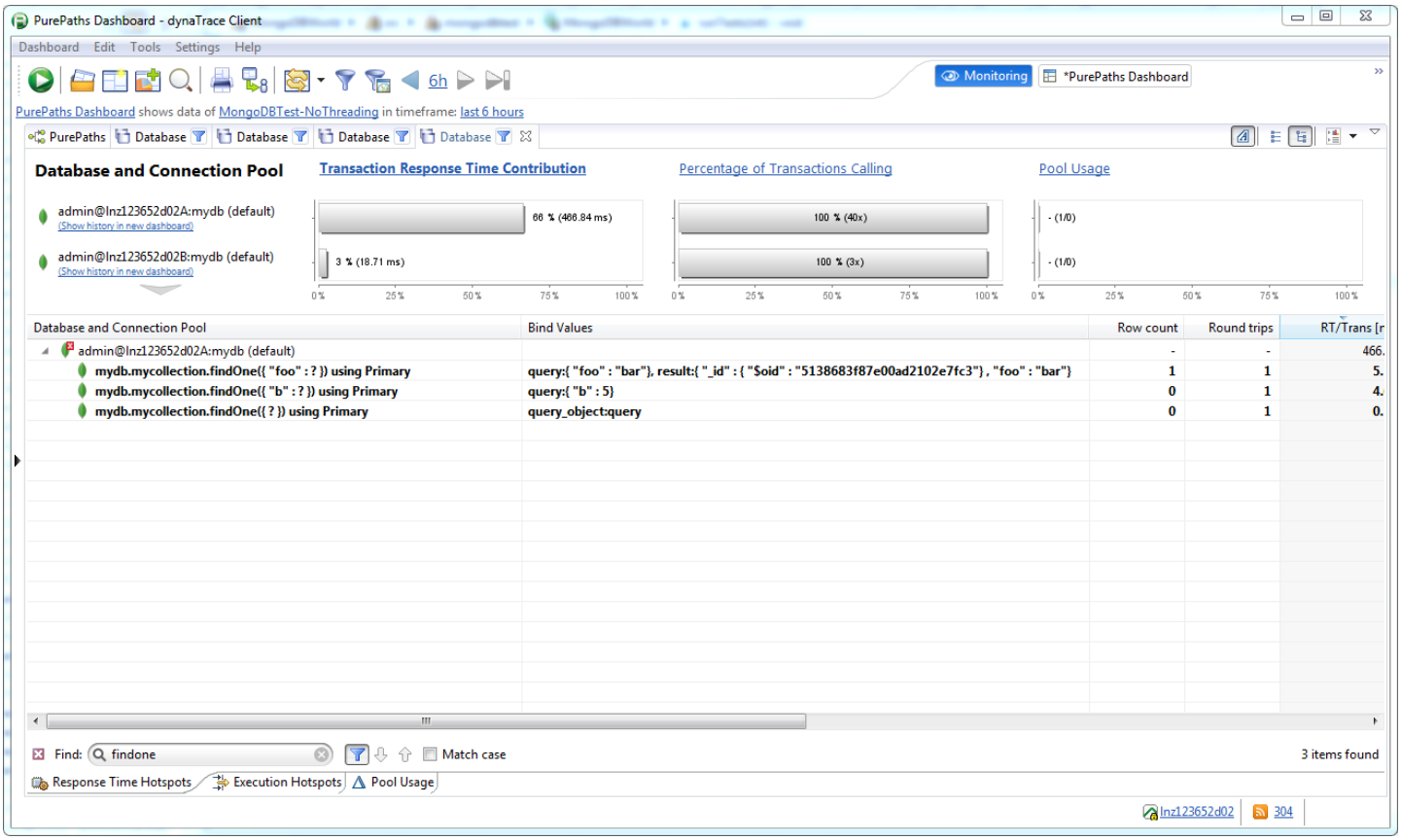
In the second figure, notice that the result's document ID displays.





In the second figure, notice that the bind values are part of the statement string.





If full JSON objects are captured, the JSON query strings become part of the database statement strings, as described in the table.

Statement string	Meaning
`find()`	No query is specified (find all: limited/document ID capturing).
`find({ })`	No query is specified (find all: extended/JSON string capturing).
`find({ ? })`	A query is specified: limited/document ID capturing.
`find({ mykey: ?, ... })`	A query is specified: extended/JSON string capturing.

Values are replaced with `?` to avoid explosion of statement strings. The actual query string (for example, `find({ mykey: "myvalue", ... })`) can be found in the bind values.

Supported versions

- Supported server versions: 2.0.*x* , 2.2.*x* , and 2.4.*x*
- Supported Java driver versions: 2.3 through to 2.11.3

Testing an installation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Watch and learn

Installing, configuring, and using easyTravel

Use the Demo Application

Download and run the *easyTravel* demo application to test your AppMon installation. This realistic multi-tier web application contains Java, .NET, and native components. The download package also includes a preconfigured System Profile and dashboards to help you quickly configure and use AppMon to monitor easyTravel. For additional assistance, view our walkthroughs, which provide real-world guidance in using AppMon and contain many use case examples based on easyTravel.

For more information:

- Download and Install easyTravel
- View walkthroughs that show how to use dynaTrace

Updates

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is a product update?

Related pages

Automatic agent updates

Install an update

Roll back or remove an update

Update a standalone Memory Analysis Server

Update Scenarios

User experience management data files

Downloads

Dynatrace Product Downloads

What is a product update?

An update may contain defect fixes, performance improvements, or usability and feature enhancements. The process for installing an update includes uploading the update to a running AppMon Server and managing the installation of the update using the **Update Wizard**. This wizard is accessible from the AppMon Client. It lets you completely update all AppMon components or perform a controlled update rollout in which only selected Agents in specified Agent Groups or on specific hosts are updated.

An update version is determined by the build and revision number in its versioning scheme, which includes major, minor, and revision identifiers as well as the build number. For example, the revision and build number **1.1027** identifies update 6.2.**1.1027**. The major and minor identifiers of the update must match the version of the AppMon system on which it is installed. For example, you can install update 6.2.**1**.**1027** on an AppMon **6.2**.0.1239 system, but not on version 6.**1**.0.8154.

Only **one** update can be active on an AppMon system at a given time. Therefore, activating an update deactivates an older one if present. Update files have the extension `.dtf` and are usually distributed in a `.zip` archive with a checksum file and release notes.

What's the difference between an update, plugin, FastPack, and product upgrade?

Plugins are supplemental product libraries that extend AppMon with features and functions beyond the functionality in the installed product. Plugins are imported into the product from **Dynatrace Server Settings** dialog box, which is accessible from the Client. See Plugins for more details.

FastPacks (formerly known as ResourcePacks) are downloadable packages that can contain resources such as preconfigured plugins, sensor packs, dashboards, and System Profiles. A FastPack bundles these different resources into one installation file that can be imported and installed into AppMon just like a plugin. See Building FastPacks for more details.

A **product upgrade** requires a completely different installation process from an update, plugin, or FastPack. A product upgrade must be installed to a separate directory structure and settings and data needs to be migrated from the old to the new version to persist your data. See Upgrade and Migration Guide for more details.

Update availability

A push notification issues a download link in the Client when updates are available, and describes major update details.

- **Updates** are available to all customers on the AppMon product download site. Updates are cumulative, which means they contain the fixes of updates released earlier. For example, AppMon update 7.0.2 contains all changes from 7.0.1.
- **Private updates** address customer and installation-specific issues. They are not cumulative, which means that they may not contain the fixes of updates released earlier, so they should only be applied in consultation with support.

Update management

The primary update management tasks are:

- Install – The update installation process requires using the **Update Wizard**, which is accessed from the Client UI, to upload the update file to the Server and guide you through the remaining configuration and deployment steps of the installation process. Additional installation options include using the AppMon REST API and selectively installing an update on specified Agents.
- Rollback – Use the update rollback feature to revert to a previously installed update or restore the originally installed version of AppMon.
- Remove – Use the update remove feature to delete an inactive update from the Server machine on which it was originally uploaded.

Required permissions

Your AppMon user account must have the following permissions enabled to install, roll back, or remove an update: **Manage Updates**,**Shutdown or Restart**, and **Shutdown or Restart Collector**. See User Permissions and Authentication for more information.

Update Scenarios

On this page

Maintaining large enterprise environments

Reasons to keep an old version of an agent

Upgrade in multiple steps

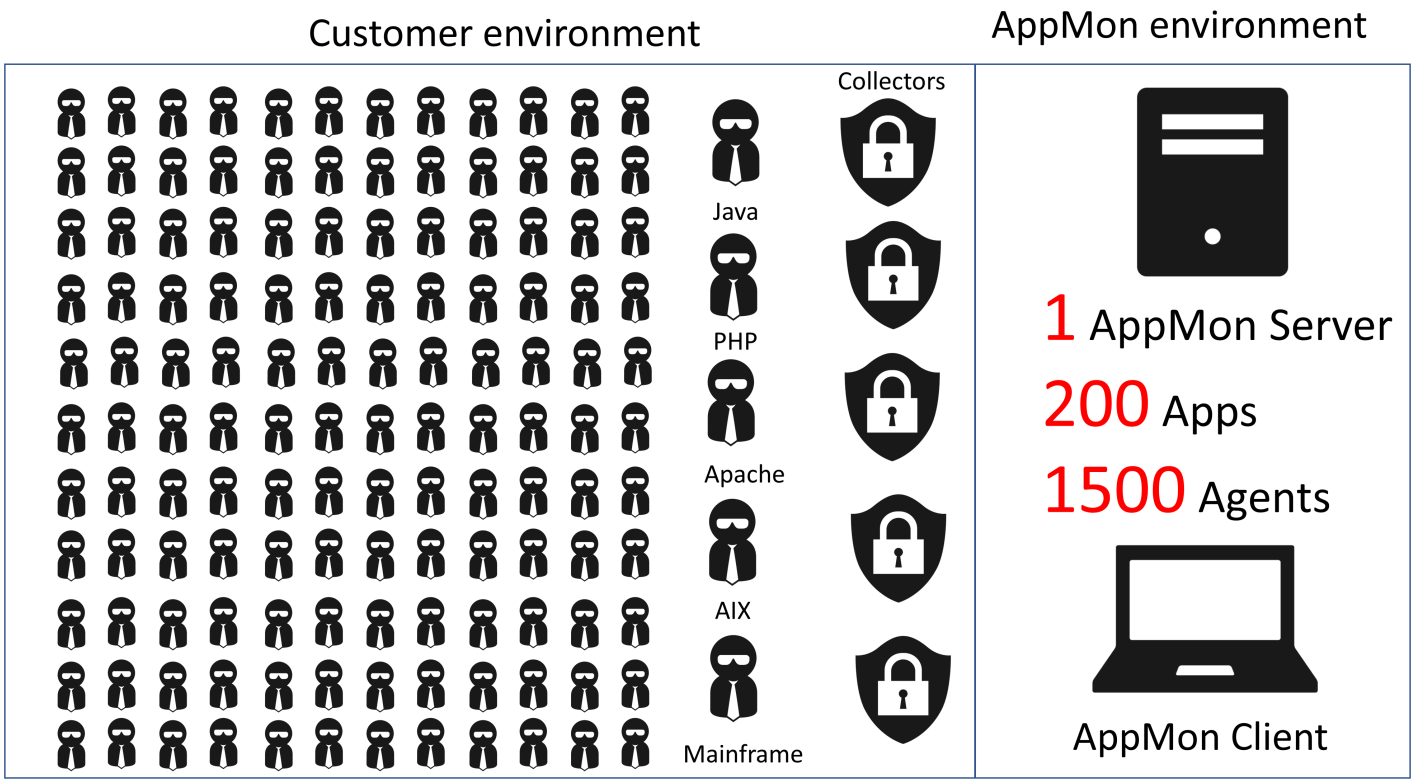
Agent version pinning

FAQ

Maintaining large enterprise environments

Work in Progress

This page is work in progress, please ignore.



How does a rollout of a new AppMon version looks like in a big enterprise environment?

1. Upgrade of AppMon Server
2. If you try to connect an AppMon Client to an AppMon Server which has a newer version you are asked if you want to upgrade the client. Please keep in mind that AppMon Client can only connect to AppMon Server which has the same version.
3. Upgrade of collector (optional step): For a description on how to upgrade the collector see. You don't have to immediately upgrade all collectors because collectors are compatible with newer versions of AppMon server (See Agent collector compatibility)
4. Upgrade of agents: Bootstrapped agents are automatically upgraded when the collector the agent is connected to has a newer version and the instrumented application is restarted. See "How to find out if my agent is a bootstrapped agent" in FAQ. Non-Bootstrapped agents need to be upgraded manually

Reasons to keep an old version of an agent

Sometimes it's not possible to upgrade an agent because it's monitoring a technology which is no longer supported by it's vendor and is therefore no longer supported in the current AppMon version (e.g. IIS 6). See Support levels and end of life policy

The good thing is that it's not always necessary (although highly recommended) to upgrade all agents immediately. See agent collector compatibility and the following section.

Upgrade in multiple steps

The following graphics show how a typical step by step upgrade can be executed:

Agent version pinning

The Agent version can be pinned to prevent its version being changed by different methods:

Controlled Update rollout

- Upgrade + Update: Controlled rollout via tech / hostgroups in Update rollout wizard UI - have to finish complete rollout before going to next update (with controlled rollout agent)- major drawback: need complete rollout before able to rollout next update version - min. bootstrap version required

Controlled update rollout helps you to determine which agents should be updated and which not.

Requirements

To successfully install an update on an agent when performing a controlled update rollout, the agent must be a bootstrap agent for a supported Appmon release.

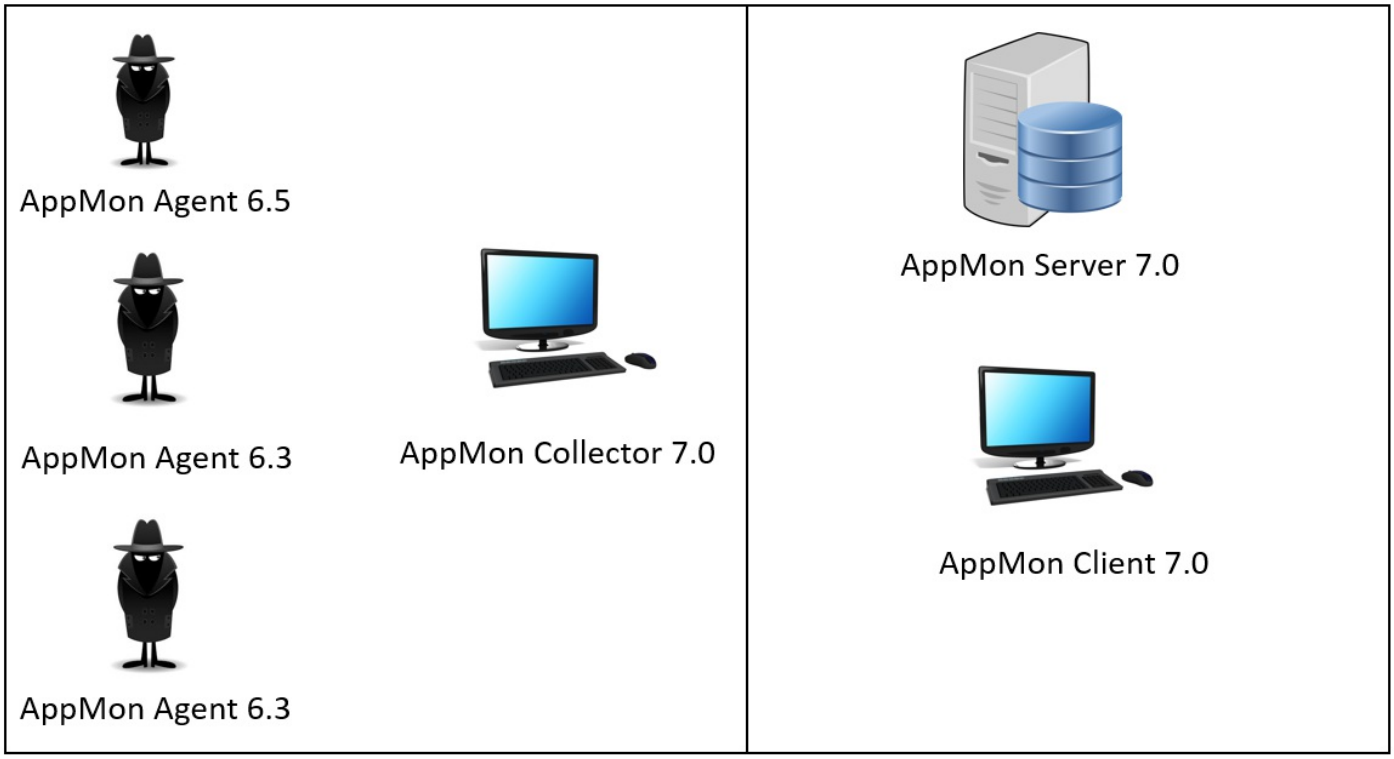
Drawback

Need complete rollout before able to rollout next update version

Disable Automatic Agent Updates

Prevents Updates and upgrades

Don’t update agents in Update-UI in Client (global checkbox for all agents) - can only be done globally - minimum version of bootstrapper required



Disable Automatic agent updates

Requirements

A bootstrap Agent with version equal to or later than 6.3.
The bootstrap Agent must have been connected to a Collector with the version you want to use.

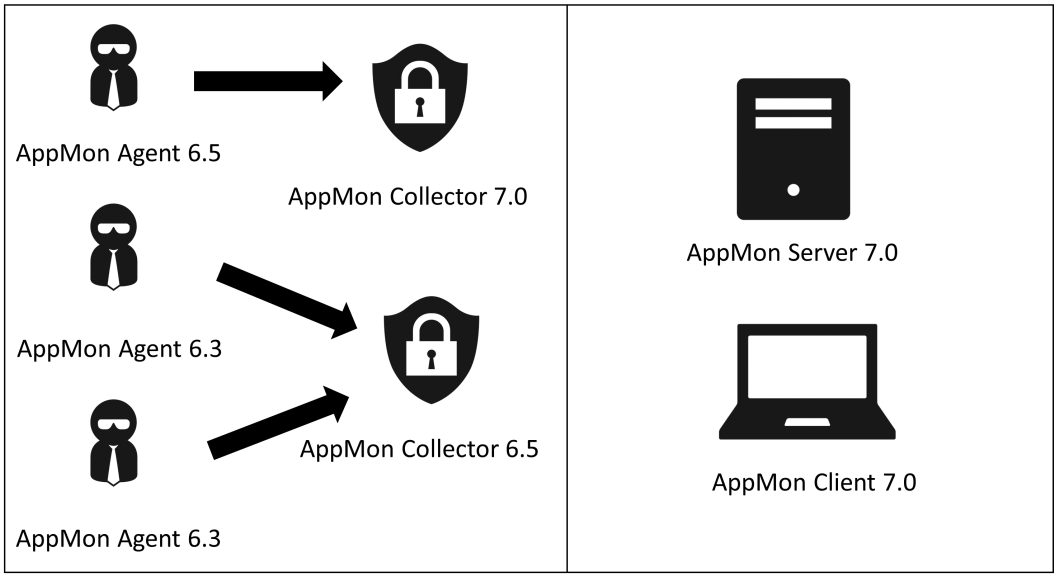
Using an old Collector

Prevents only upgrades.

Agent Overview:

- Old agent version due to old collector

Keep old version via old collector: and through this old agent version: drawbacks: no update installation directly possible (7.0 updates don’t contains latest 6.5 updates/agents/collectors)



Collector is not updated to new version therefore also the agents which are connected to this collector will keep their old version.

> **Drawback**
>
> No update installation directly possible. For example, no automatic update from 7.0 to 7.0.1

Disable bootstrapping

Prevents Updates and upgrades.

FAQ

How do I find out if my agent is bootstrapped? In the AppMon Client in “Agent Overview”-Dashlet the column “Bootstrap Version” will show “Not bootstrapped”.

Can I connect with an AppMon Client to an AppMon Server with a newer version? No, AppMonClient can only connect to AppMon Server which has the same version

Can I connect with an AppMon Client to an AppMon Server with an older version? No, AppMonClient can only connect to AppMon Server which has the same version

What is the difference between Update and Upgrade? Upgrade: major version change, e.g. 6.3 => 6.5, go through Upgrade guide Update: minor version change, e.g. 6.5.1 => 6.5.2 Use Update UI

How to I keep my enterprise environment up-to-date? Install an Update

Automatic agent updates

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Disable automatic updates

Requirements

Enforce selective agent update

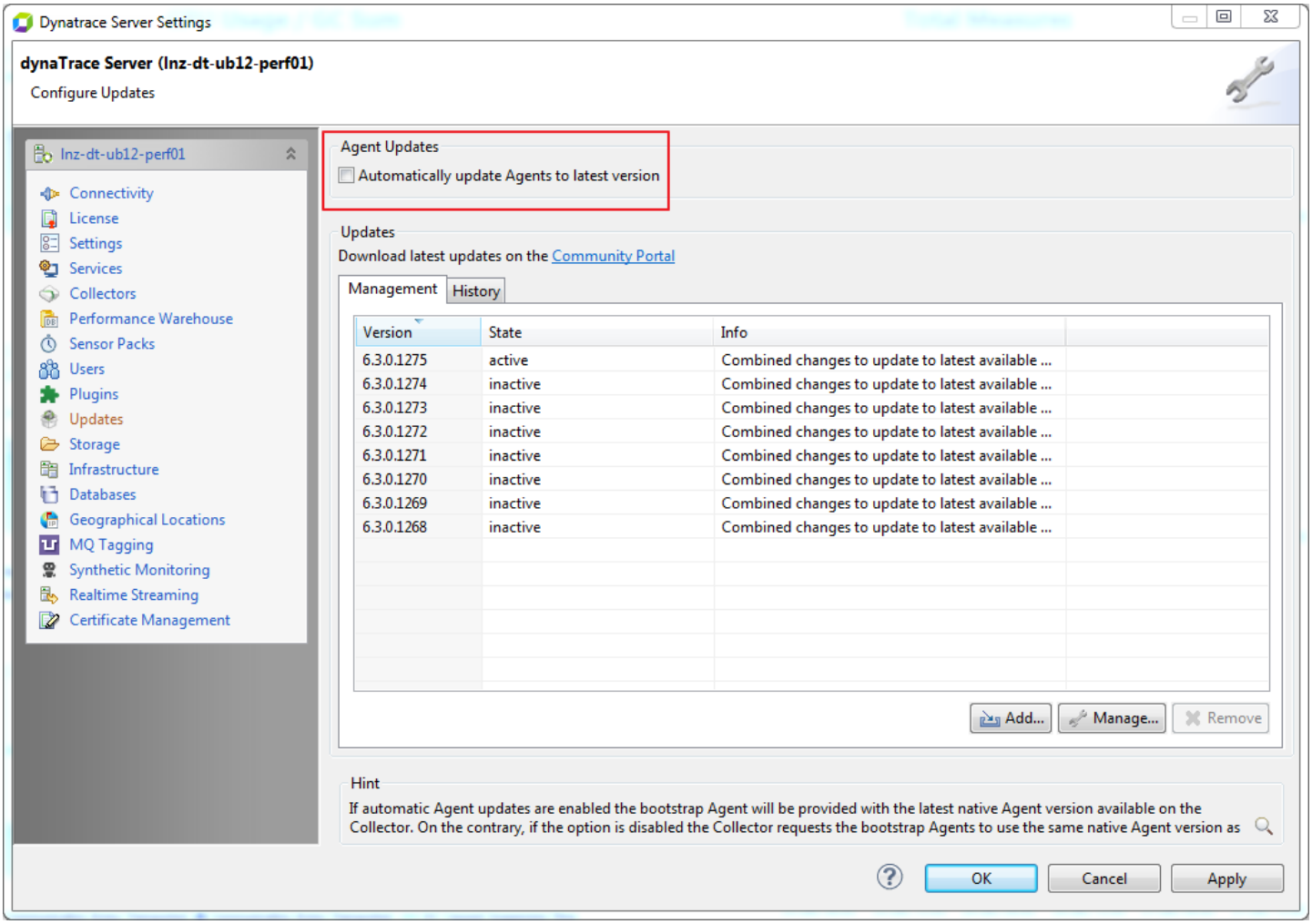
Bootstrapped Agents update to the most recent version when connecting to the AppMon Collector by default. For AppMon 6.3 and later, you can disable these automatic Agent updates. If you disable automatic Agent updates, the bootstrapped Agents no longer uses the Agent version provided by the Collector. Instead they stay on their current version.

Note

It is strongly recommend to keep the automatic Agent updates enabled unless you have regulatory or policy requirements which make the disablement necessary. After disabling automatic Agent updates, the Agents no longer gets the latest bug fixes. The supported application environments and technologies are bound to the AppMon Server version. Although discontinued technologies might still work, they are no longer supported. Moreover, new AppMon features might not work.

Disable automatic updates

You can disable automatic Agent updates in the Client under **Tools > Manage Updates**. The **Dynatrace Server Settings** dialog box appears and displays the **Update** pane. Clear **Automatically update Agents to latest version** check box.



Requirements

- A bootstrap Agent with version equal to or later than **6.3**.
- The bootstrap Agent must have been connected to a Collector with the version you want to use.

Enforce selective agent update

To update a certain Agent although automatic Agent updates are disabled, install an update using the controlled rollout and explicitly select the Agent you want to update in the Agent selection dialog box. See Controlled Update Rollout for more information.

Install an update

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Requirements

Installation process overview

Install an update

REST API installation

Controlled update rollout

Troubleshooting

Related pages

Updates

Roll back or remove an update

Plugin management (REST)

Downloads

Latest update

Requirements

Important

If you want to install a version where major and minor version do not match (for instance 6.5 on top of 6.3), you cannot use the update mechanism. Refer to the Upgrade and Migration Guide.

The following requirements must be met to successfully install an update:

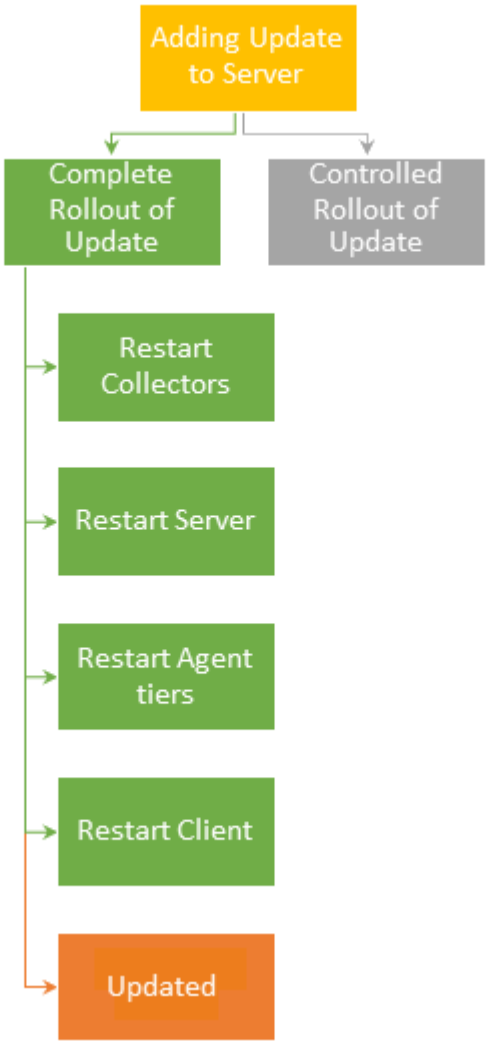
- Disable virus scanning software the AppMon Server and Client machines.
- Your AppMon user account must have the following permissions enabled: **Manage Updates**,**Shutdown or Restart**, and **Shutdown or Restart Collectors**.
- You must be able to restart all Servers and Collectors immediately after installing the update.
- A pending restart must not be not currently logged against the Server or any connected Collector. Check **Warnings** in the Start Center to determine if a pending restart is required.
- You have verified that the update is of a **higher** version than any previously installed update.
- If you have a private update installed, you must check out with the support that the update you're going to install contains the fixes of your previously installed private updates. If the update you want to install overwrites the fixes of the active update, the warning message will appear, listing the fixes you may lose.

Installing an update on a server or collector instance

When installing an update on a Server or Collector instance, only the instance updates. The update is not installed on any other Server or Collector instances. If installing an update on an AppMon Server or Collector that is not run as an instance, then any Server/ Collector instances that run on the same system do not update.

Installation process overview

The following diagram shows the necessary steps for installing an update.



Install an update

1. Download the update from the AppMon product download site. Extract the contents to local temp folder on the Client machine.
2. Start the Client and choose **Tools > Manage Updates**. The **Dynatrace Server Settings** dialog box appears and displays the **Update** pane.
   The **Manage Updates** menu command is only enabled if the AppMon Client is connected to an AppMon Server. For the standalone Client, make sure to select the desired AppMon Server on which the update should be installed. For the Webstart Client, the Server from which you started the Webstart Client is the only Server available.
3. Click **Add** to start the **Update Wizard**.
4. Click the **Select File** button, and then select the `update*.dtf` you want to install. Alternatively, in the Windows file explorer, double-click the `update*.dtf` file to open the wizard with the file already selected.
5. Click **Next** and **OK**. The update uploads to the Server machine and is ready to install.

6. Click **Next**. The **Configure Rollout Action** page appears.
7. Choose **Complete Install** and click **Next**. To selectively install the update on specified Agents, choose **Controlled Install**. For more details, see Controlled Update Rollout.
8. Apply the changes to deploy the update to the Collectors. This step may take a while (especially for large environments).
9. For the remaining steps in the wizard, choose when to restart the Collectors and Server.
   If an AppMon Collector is part of a Collector group, you have to restart the whole group, which makes the single Collector unselectable.
0. On the **Restart Agent tiers** page, the bootstrapped AppMon Agents that must be restarted are shown. To update bootstrapped Agents, you must restart the tier that is monitored by the Agent. It may take a few minutes until the first Agents appear in the restart overview. The overview updates every 15 seconds.
1. Click **Finish** and **Yes** when prompted to restart the Client and complete the update installation process.

REST API installation

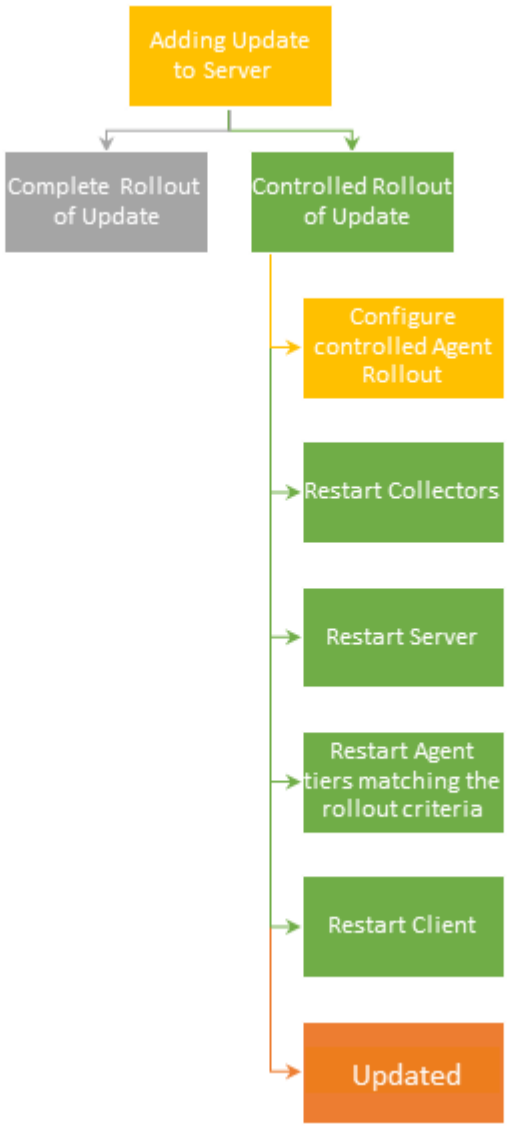
You can install an update using the AppMon REST API. See Plugins and Updates (REST) for more information.

Controlled update rollout

In a controlled update rollout, you choose specific Agents on which to install an update. This enables you to test the update on a subset of Agents before installing it on all of the remaining Agents. The process for choosing Agents includes defining criteria, such as the type of Agent or a specific Agent Group, that when met determines whether the update is installed. Agents that do not match the specified criteria will not have the update installed.

Controlled update rollout process overview

The following diagram shows the necessary steps for performing a controlled update rollout.



Requirements

To successfully install an update on an Agent when performing a controlled update rollout, the Agent must be a supported bootstrap Agent.

If an Agent does not meet either of these conditions, the new update does not install and the Agent remains at its current version.

Perform a controlled update rollout using the Update Rollout wizard

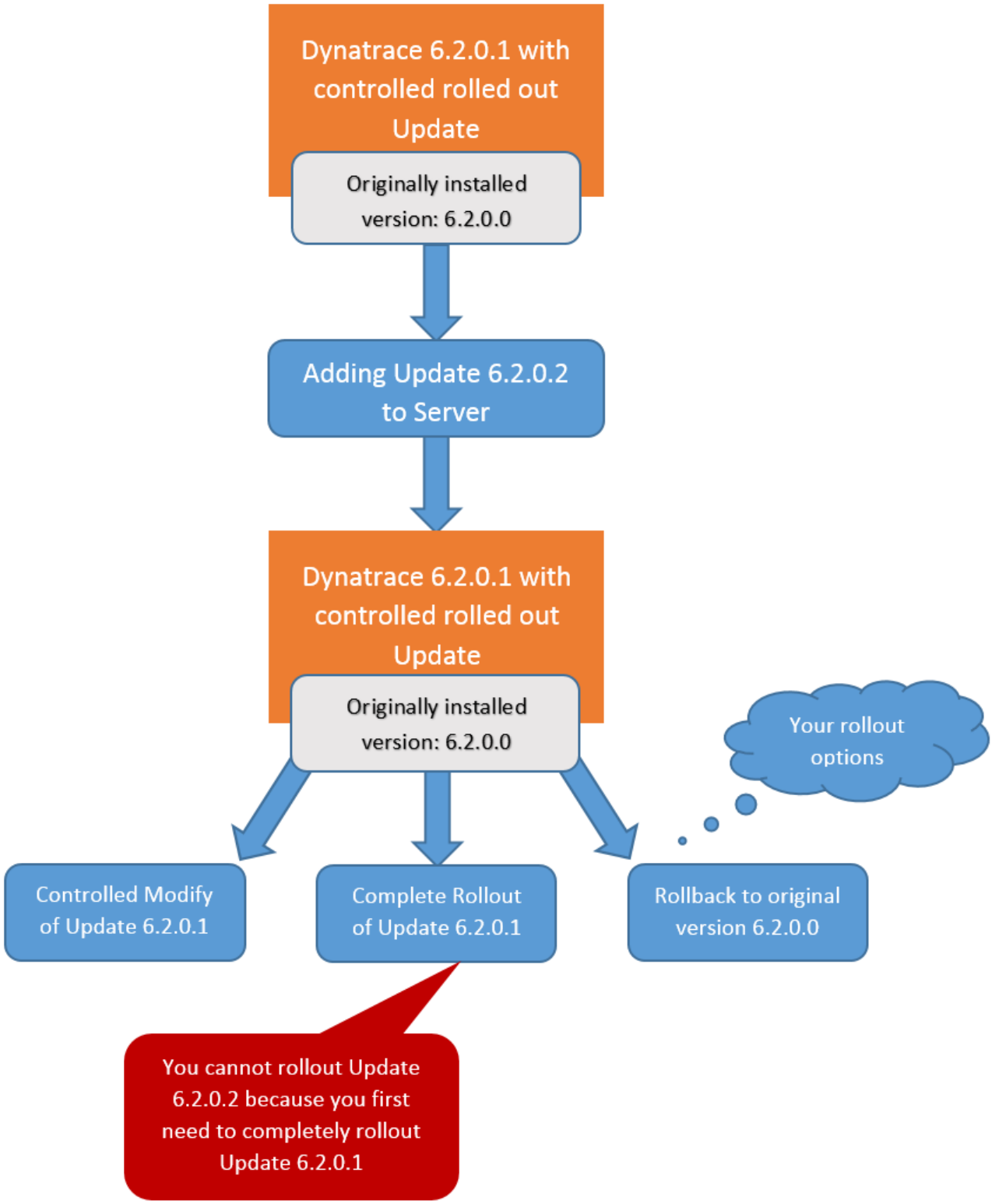
1. Access the **Update Wizard** and upload the update you want to install. See Install an Update.

2. On the **Configure Update Rollout Action** page of the wizard, select **Controlled Modify** and the version of the update you want to install, if not already selected. Click **Next**. The **Configure Agent Rollout** page appears.
3. On the **Configure Agent Rollout** page, choose one of the following options:
   - **Control the Agent rollout by Agent Group and System Profile**: Only bootstrap Agents that match the selected System Profile or Agent Group are updated. If you select an entire System Profile node, all AppMon Agents that match this System Profile are updated.
   - **Control the Agent rollout by Host and Agent name**: Only Agents that are running selected hosts and have one of the selected Agent names are updated. If you select an entire host node, all AppMon Agents that run on the host are updated.
     Additionally, you can refine both rollout options by adding specific technologies to the rollout configuration criteria. For example, if you select a whole System Profile node, but only the technology **Java**, then only AppMon Agents that match the System Profile **and** are Java Agents are updated.
4. Click **Next**. On the **Selected Agent Rollout Overview** page, verify that all Agents that you want to update are shown.
5. Complete the remaining steps of the wizard.

Post-installation considerations for a controlled update rollout

After the successful completion of a controlled update rollout, the Agents in your deployment are at different versions. As a result, installing a newer update is **not** permitted until you perform a complete install and update all of your Agents to the same update version. Alternatively, you can roll back the installed update before installing a newer update to achieve the same Agent version consistency.

The diagram below shows the available options for installing update 6.2.0.2 to replace Update 6.2.0.1 that was installed using the controlled update rollout option.



Perform the following steps to change the configuration of a controlled update rollout.

1. Access the **Update Wizard**.
2. On the **Configure Update Rollout Action** page, select **Controlled Modify** and then click **Next**.
3. On the **Configure Agent Rollout** page, you can change the configuration for selecting Agents on which to install the update.

4. Apply the update changes. The new rollout configuration deploys to the AppMon Collector(s).
5. Restart the affected Agent tiers and complete the steps of the wizard.

> **Note**
>
> Restarting the AppMon Server, Collector or Client is not needed when changing the rollout configuration.

Troubleshooting

Problem: an Agent did not update or the agent update did not have the desired effect.

Resolution Steps

- Check whether a bootstrap Agent is used. The automatic update works only for bootstrap Agents.
- Restart the AppMon Collector to which the Agent is connected after installing the update, then check whether the used Collector runs with the version of the update.
- Restart the Agent tier after installing the update.
- If it is a Web Server Agent, restart the AppMon Web Server Agent (the Master Agent) that runs as a service.
- Restart the AppMon Web Server Agent **before** restarting the monitored Web Server.
- If using a controlled rolled out update, make sure the Agent matches the controlled rollout criteria.

Problem: after uploading an update to the server, it does not appear on the Configure Update Rollout Action wizard page.

Resolution Steps

- Is another update currently in use?
- Is the used update newer than the one you want to install? If so, you can install the older update by a using the rollback feature to its version.
- Was the update that is currently active on your Server installed using the controlled rollout feature? If so, then you either need to perform a complete install of the used update or roll back to another version before you can install the new update.

Problem: an update was successfully installed, but no longer appears in the update overview.

Resolution Steps

- Do you use an instance Server?
- Have you installed the update with the same instance Server that is currently running? If not, you have to install the update also for this instance Server.
- Have you checked the Server log? Make sure to check the fixpack `_*<update_VERSION>*_osgi.log` (for example, `fixpack_7.0.0.2469_osgi.log`) shown below your Server node in the **System Information** dashlet.

Roll back or remove an update

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Roll back an update

Remove an update

Related pages

Updates

Install an update

Plugins and updates (REST)

Roll back an update

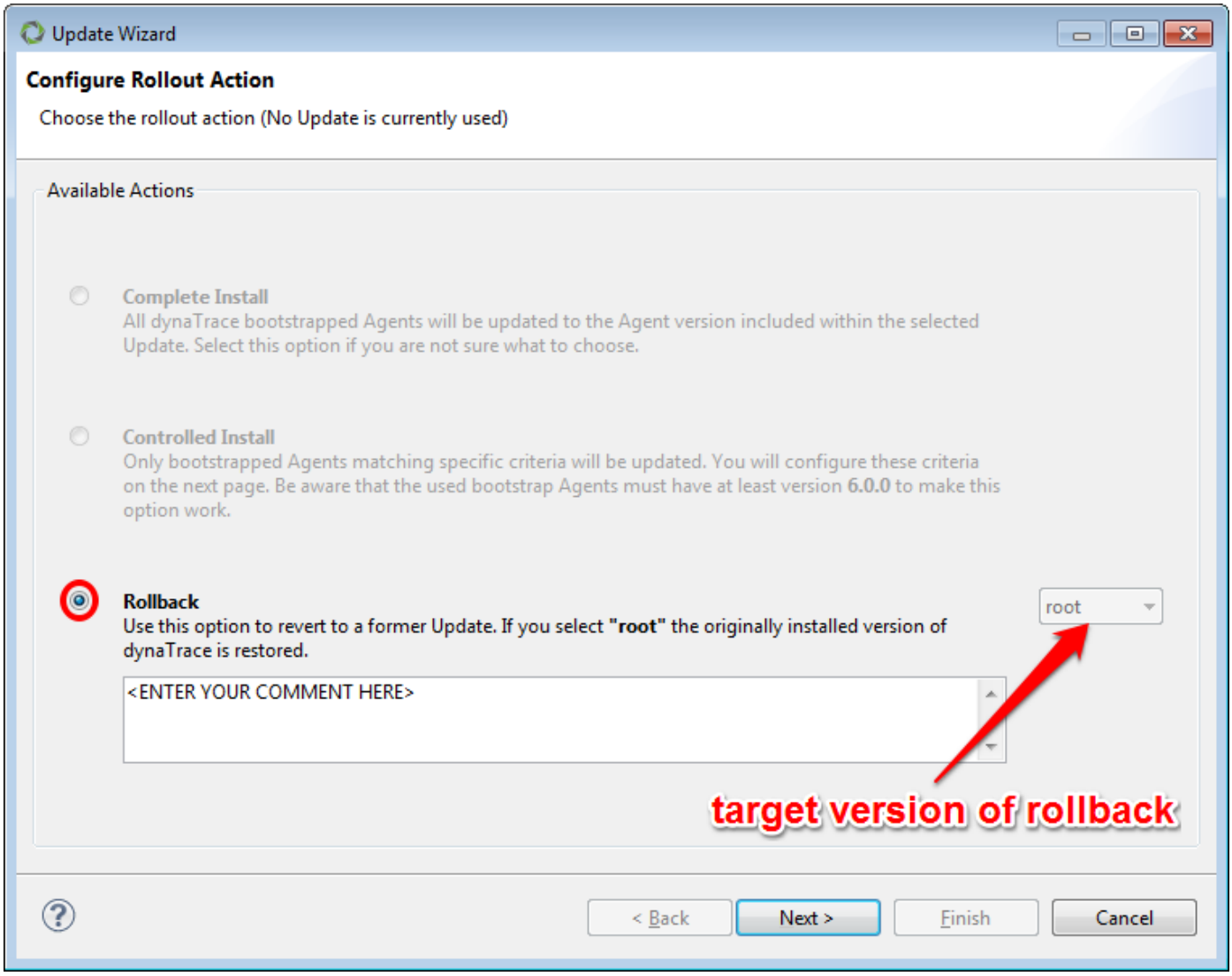
Use the update rollback feature to revert to a previously installed update or restore the originally installed version of AppMon. To successfully complete the rollback process, the following conditions must be met:

- You can immediately restart the AppMon Server and its Collectors.
- You can restart related applications and Agent services.

To perform an update rollback:

1. On the Client menu bar, choose **Tools > Manage Updates**. The **Dynatrace Server Settings** dialog box appears and displays the **Updates** pane.
2. Click the **Manage** button. The **Configure Update Rollout Action** page appears.

3. Select **Rollback** and the version to revert to. If you select **root**, all updates are disabled and the originally installed AppMon version is restored.



4. Click **Next** and follow the instructions in the wizard to complete the rollback process.

After you roll back the update, it appears as inactive on the **Management** tab. In addition, the **History** tab displays a record of when the update was rolled back.

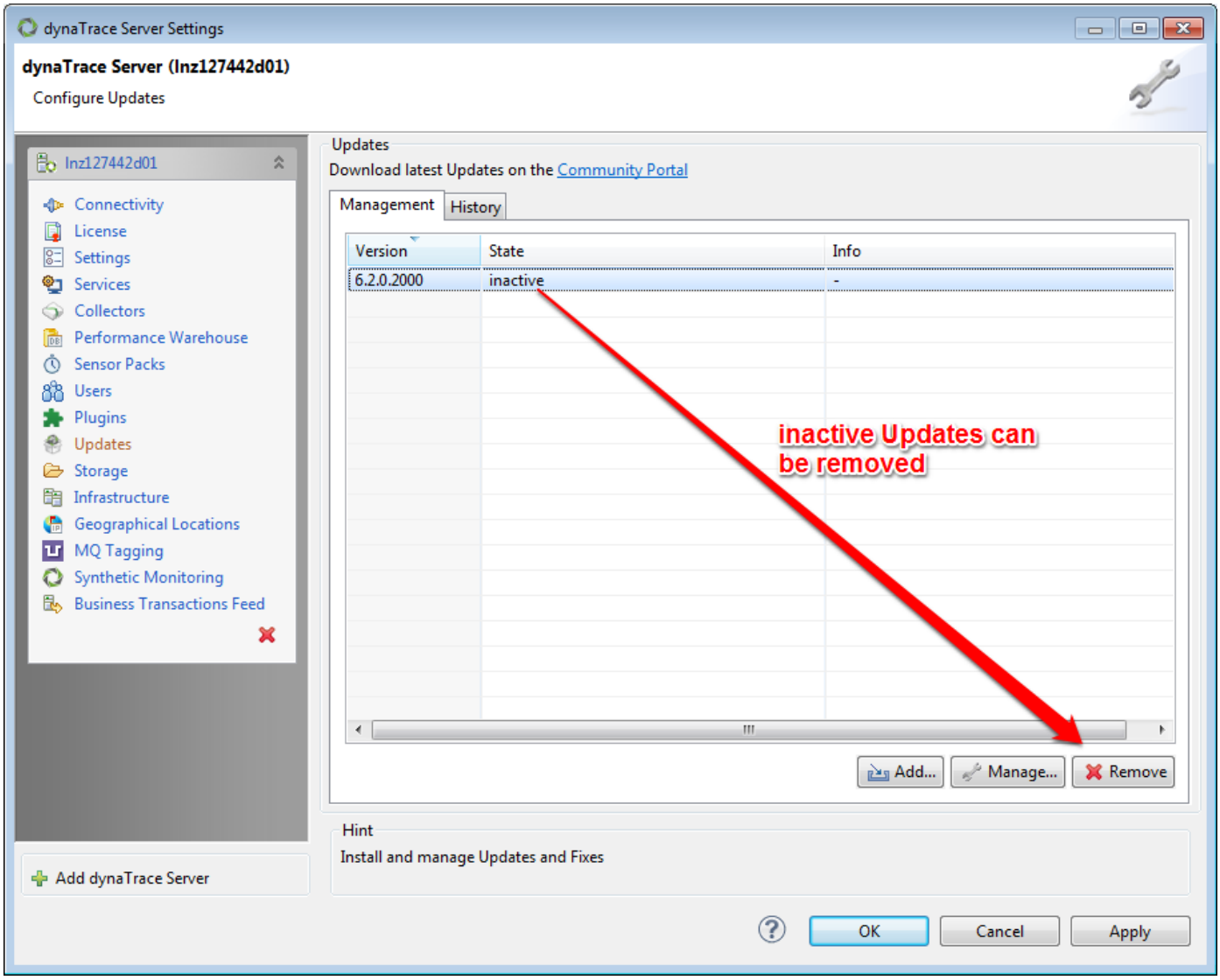
Remove an update

An update must be inactive to remove it from your installation. The result of the removal process is that the update is deleted from the Server machine on which it was originally uploaded.

To remove an update:

1. On the Client menu bar, choose **Tools > Manage Updates**. The **Dynatrace Server Settings** dialog box appears and displays the **Updates** pane.

2. Select the update you want to remove.
3. Click **Remove**, then click **Yes** to initiate the removal process.



After you remove the update, it's deleted from the file system and no longer appears on the **Management** tab. In addition, the **History** tab displays a record of when the update was removed.

> **Note**
>
> Never manually delete an update from the file system. This may result in the loss of any fixes installed by the update, or may corrupt your installation.

User experience management data files

User Agent strings

Provided by Udger

This data file allows user agent information to be determined from a given user agent string.

Apple devices

Maintained internally by AppMon. This provides a cross reference of Apple Device ID to Apple Device name.

Android devices

Provided by Google Play Store.

This provides a cross reference of Android Device ID to Android Device name.

ISP

This is a combination of look up tables that resolve a specific IP address to the Internet Service Provider (ISP) who host that IP address.

GeoLite2 city

Provided by MaxMind.

This database determines the Geo location for a given IP address.

Updated files are as follows:

A number of data files and databases are used by User Experience Management. These data files and databases contain data that can change over time. Performing an update automatically provideS new versions of these files.

Update a standalone Memory Analysis Server

1. Fully install and roll out update with VERSION on the AppMon Server.
2. Shut down the Analysis Server.
3. In the Analysis Server installation directory (referenced here as `AS_INSTALL -` ensure that the directory exists `AS_INSTALL/server/updates`.
4. From the AppMon Server installation directory `<DT_HOME>`, copy the full directory `<DT_HOME>/server/updates/update_VERSION` to the Analysis Server's `AS_INSTALL/server/updates/` so you should then have for instance on the Analysis Server: `/opt/dynatrace-6.3/server/updates/update_6.3.8.1056`, for example with the command `cp -r DT_HOME/server/updates/update_6.3.8.1056/ AS_INSTALL/server/updates/`.
5. Make sure that the permissions of the new directories match those of the root `AS_INSTALL` directory.
6. Start the Analysis Server.
7. Verify that the correct version was started by checking the version header in the `AS_INSTALL/log/analysisserver/analysisserver.0.0.log`, for example:

```
2016-09-12 08:20:34 INFO [OOPAnalyzer] ------------------------------------------
2016-09-12 08:20:34 INFO [OOPAnalyzer]  dynaTrace Memory Analysis Server  Copyright (C
2016-09-12 08:20:34 INFO [OOPAnalyzer]   ------------------------------------------
2016-09-12 08:20:34 INFO [OOPAnalyzer]  Version 6.3.8.1056  built Wed Sep 07    17:15:
```

8. Test connections from the AppMon Serverby selecting **Client -> Server Settings -> Services -> Dynatrace Memory Analysis Server -> Test connection**.

These are only relevant for analysis server installations that are in an installation directory separate from the AppMon Server, as otherwise an update rollout on the AppMon Server also patches the Analysis Server.
Use these steps to manually update an AppMon Memory Analysis Server.

Upgrade and migration guide

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Dynatrace AppMon 2017 May Release Notes

Migration tool details

Deployment Guide

Downloads

dynatrace-migration tool

Product Downloads

Migration steps

Prepare to upgrade

Upgrade server, collectors, and clients

Upgrade agents

Final migration checks

Watch and learn

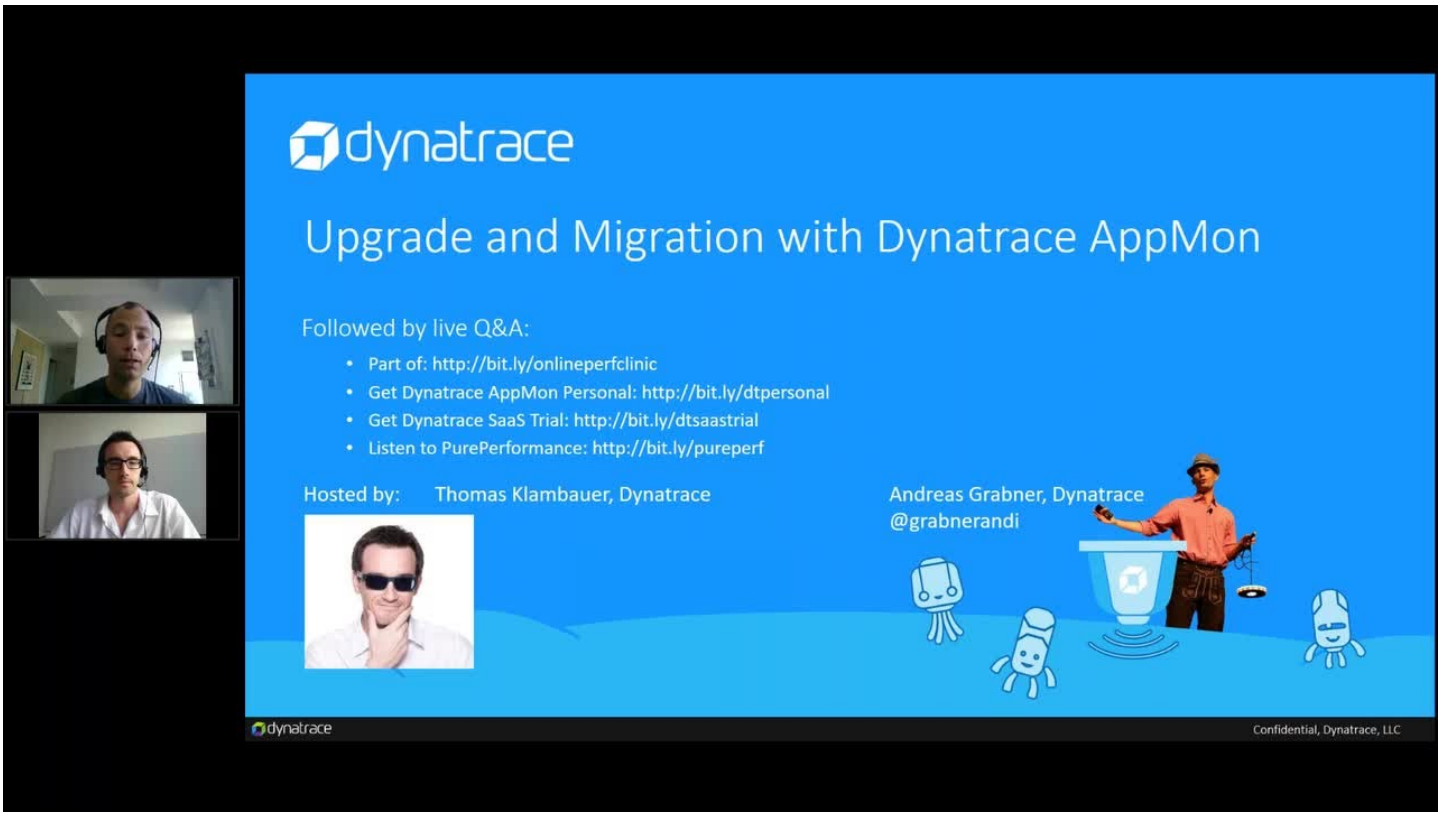
Deploying, Upgrading, and Migrating Dynatrace AppMon

Follow the instructions in this guide to:

- Upgrade your AppMon installation
- Migrate your AppMon installation to another host machine

1. **Prepare for your migration**
   Before you start, review your current installation releases for Servers, Collectors, and Agents to determine if those components can be directly upgraded to the new release. Also, review the System requirements to make sure you are using supported hardware and platforms.
2. **Install the new components and transfer data to it**
   Install the new components and **use the latest Migration Tool** to move your current data and configurations to the new installation. Check the version displayed when you run the tool from the command line. See Migration tool details for manual steps and edits needed to complete migration, and for more information on using the Migration tool.
3. **Upgrade your Agents**
   Most Agents automatically upgrade with the new release. In some cases, you may need to manually upgrade your Agents.
4. **Final checks and verification** Start AppMon and ensure Agents are sending data and your PurePaths and dashboards display properly.

See also the following Performance clinic videos for an upgrade demo:



Prepare to upgrade

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Dynatrace AppMon 2017 May Release Notes

Deployment Guide

Component compatibility

License upgrade

Downloads

dynatrace-migration tool

Product Downloads

Migration steps

Upgrade server, collectors, and clients

Upgrade agents

Final migration checks

Review this checklist to ensure you are ready to migrate.

Is this the right guide?

- This guide only applies if you want to upgrade or migrate to version 7.0. Otherwise see the documentation of that respective version.
- For AppMon 6.2 and later, a direct upgrade is possible.

- For AppMon 6.1 and earlier, **you must migrate to 6.5 first**. See the 6.5 Documentation for instructions.
- For Proof of Concept (POC) and evaluation installations, a clean new installation is highly recommended.
- For Independent Software Vendor editions (PTC Windchill), use the documentation supplied by the vendor.

Get an overview

Create an inventory of your installed Collectors, Agents, and Clients to understand the scope of the upgrade.

Check new requirements

Verify the system requirements:

- Make sure the **Hardware requirements** is fulfilled for your sizing. AppMon 2017 May is more strict in hardware requirements than previous releases.
- AppMon 2017 May and later **does not** include a 32-bit Client anymore. See Set up the Client and Set up the Webstart Client for more details.
- This migration does **not** support a Server-embedded Collector. Install a stand-alone service for your current version before you upgrade.
- An embedded Performance Warehouse is not supported for migration. Switch to an external one first, as described here.
- If LDAP authentication is enabled, users need to be members of at least one AppMon LDAP group to log in. See LDAP for details.

Double-check on critical fixes

If you recently **received an update** with a critical fix or change in response to a support case, check the latest update Release notes to see if the issue id (for example, JLT-160157) is included.

This release together with its latest public update contains all changes of:

- Public update 6.5.18 and earlier (not affected by JLT-192329)
- Public update 6.3.25 and earlier
- All updates released for versions 6.2 and earlier

Later versions will be included in following 2017 May public updates. If you are missing required changes, contact support before scheduling the upgrade to clarify and resolve.

Update the source installation

It is recommended to roll out the most recent public update on the old installation before the upgrade.

Check Compatibility

We recommend to upgrade all Collectors and Agents along with the Server. This is also required to guarantee that certain new features will work.

However, if you don't plan on doing so, check the component compatibility beforehand.

As a best practice, plan your upgrade around scheduled maintenance windows. You must be able to restart Agent Groups / tiers should incompatibilities arise.

Split server instances

If in your current AppMon server installation path `<DT_HOME_OLD>` there exists a `server/instances` folder and you are using these **Server instances:** This feature and the `dtserver`'s and `dtfrontendserver`'s `-instance` parameter are deprecated as of AppMon 6.3. Collector instances are still supported.

Use this guide and check requirements there to migrate the Server instances to separate installations before upgrading.

Review custom-sized installations

Review your Server sizing by contacting support beforehand, if your `dtserver.ini` or `dfrontendserver.ini` file contains `-memory unsupported`.

Take notes

Document whether you are using **continuous transaction storage**. Check **Settings > Dynatrace Server > Storage** in the AppMon Client.

Document which additional plugins were installed by clicking **Settings > Dynatrace Server > Plugins** in the AppMon Client.

Document all Downtimes and incident rules of the Self-Monitoring System Profile.

Document the passwords to the following services: Email, LDAP, Proxy, DC-RUM, Gomez.

Performance Warehouse database

- Be prepared to create backups as part of this guide.
- Check in the System Requirements whether your SQL DB server is supported. If not, create a **backup** and switch to a supported DB before the upgrade.
- Verify that the configured **Performance Warehouse** user has the permissions to create and drop tables for the DBMS.

Session Store

- If the current storage location is within the installation directory (which is default) and you want to keep the stored sessions, move the session storage into a separate directory accessible from the both old and new installations before the migration. It will ease this and future migrations.
- Document the current configuration. In the AppMon Client, click **Settings > Dynatrace Server > Storage**.
- Be prepared to create backups as part of this guide.

License

If you use z/OS Agents, AppMon 6.3 and later no longer requires an additional license. Instead, contact license management to get the needed number of CICS and IMS agents and/or Java Agents added to your Server license.

Download files

Download the software required for the migration:

- **Installers** and ADKs for your application platforms and operating systems.

- The latest version of the **dynatrace-migration tool**
- The **latest Update**, if available, is located at the top of the 7.0 downloads list. See Install an Update.
- If you don't have internet connectivity from the Server, download the **upgraded licenses** from eServices.

Running two versions

You can run parallel versions of AppMon to test your upgrade, but be aware of the requirements for this configuration.

Next: Upgrade Server, Collectors, and Clients

Component compatibility

Looking for the 6.5 version of this page? Open in 6.5 documentation

Migration steps

Prepare to upgrade

Upgrade server, collectors, and clients

Upgrade agents

Final migration checks

Although certain version combinations may be compatible, we **strongly advise** all users to upgrade all components as soon as possible to the same version as the server.

Compatibility and Support

Components are said to be **compatible**, if they can connect and offer any kind of functionality together. For a **specific feature** to work, the support of **all involved components** (Agents, Collectors, Server, Client) is required.

Example 1: Infrastructure: Process Availability

When using AppMon 6.2 Host Agents with a 6.5 Server and Client, you will be able to configure Process Availability patterns, but these will have no effect and not fire incidents as the 6.2 Agent does not support them: This feature was only introduced starting with 6.5.

Agent-Collector Compatibility

Agents are upgraded depending on their type:

- **Bootstrapped Agents** update automatically when the instrumented application is restarted.

- Upgrade **Non-bootstrapped Agents** manually

To determine Agent versions and whether they are bootstrapped, use the AppMon Client's **Cockpits > Agent Overview** Dashboard.

Collector:	6.0	6.1	6.2	6.3	6.5	7.0
Agent						
5.6	OK	OK	OK	OK	OK	OK
6.0	OK	OK	OK	OK	OK	OK
6.1	n/a [1]	OK	OK	OK	OK	OK
6.2	n/a [1]	n/a [1]	OK	OK	OK	OK
6.3	n/a [1]	n/a [1]	n/a [1]	OK	OK	OK
6.5	n/a [1]	n/a [1]	n/a [1]	n/a [1]	OK	OK
7.0	n/a [1]	n/a [1]	n/a [1]	n/a [1]	n/a [1]	OK

[1] Upgrade the Server and Collectors before non-bootstrapped Agents.

Collector-Server Compatibility

Collectors are upgraded manually like Servers.

Collector:	6.0	6.1	6.2	6.3	6.5	7.0
Server						
7.0	OK	OK	OK	OK	OK	OK

Client-Server Compatibility

The major and minor version of the client has to match that of the Server. Lower-version Clients can auto-upgrade to the Server version if the major version matches.

Examples:

- 6.5.0 Client cannot connect to a 2017 May Server - requires a new installation
- 6.5.0 Client can connect to a 6.5.10 Server
- 6.3.0 Client when connecting to a 6.5.0 Server can auto-upgrade to 6.5.0.
- After such an auto-upgrade: a 6.5 Client cannot connect to a 6.3 Server

Running versions in parallel for migration

Multiple collectors on the same host

If you run more than one Collector version or instance on the same machine, verify that in their respective `collector.config.xml` files, on the `collectorconfig` XML element, they still have:

- unique ports (`agentport` attribute) assigned on which they listen for Agents
- unique `name` attributes
- `serveraddress` and `serverport` still indicate the correct Server information.

- Licensing allows for some flexibility allowing running versions in parallel (three-day grace period or three-week trial).
- Be sure to configure different ports for different version AppMon components to run in parallel and start / stop them at the right time.
- **Same major version *non-Webstart Clients*:** If you want to connect an old version Client to an old version Server again, answer **No** when it asks if it should update itself on connection to a new version Server.
- If you allow the update of the non-Webstart Client, it no longer connects to the older Server. To resolve this issue, use the Web Start Client to connect to the old Server or run / install a *same old version* Client.
- You cannot use the same Performance Warehouse database with two different Versions, thus you have to:
  - At some point disconnect the old version (only upgrades are possible) and connect the new version, or
  - Create a separate database with a different name, but then you will not have access to the data in the other database.
- **Do not** connect an older Server to the same Performance Warehouse DB once you connected the new version Server (and thus updated the DB).

Some components and connections in AppMon may result in potential pitfalls. To avoid some of these, consider:
In migration scenarios, having two versions of AppMon running at the same time is **not** supported. The standard migration procedure can be found here. The content of this page is meant as an aid for

experts.

Upgrade server, collectors, and clients

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

1. Install the new Server, Collectors, and Clients
2. Deactivate and Upgrade License
3. Shut down the old version
4. Create backups
5. Services migration (Server, Collector and Memory Analysis Server)
6. Auto-start new services
7. Start the new Server components
8. Client migration
9. Start the new Client
10. Activate licenses for the new Server
11. Select server sizing
12. Apply update to new AppMon
13. Connect the Performance Warehouse
14. Configure session storage
15. Start new Collectors

Related pages

Dynatrace AppMon 2017 May Release Notes
Deployment Guide
Migration tool details

Downloads

dynatrace-migration tool

Product Downloads

Migration steps

Prepare to upgrade

Upgrade agents

Final migration checks

If you are migrating the Server to a new host without upgrading to a new version, you only need to perform the applicable steps.

1. Install the new Server, Collectors, and Clients

Install the new AppMon Server, AppMon Collectors and Clients into a **new directory**, referred to as `<DT_HOME_NEW>` in these instructions.

> **Important - Windows**
>
> At the end of the installation, you will be prompted to start the Server, Collector, and Client. **Do not start these components at this time**.

> **Important - Unix**
>
> Do **not** unzip the `.jar` installation files, but run `java -jar <FILENAME>.jar`.

See Silent Installation for automation.

2. Deactivate and Upgrade License

Deactivate and **upgrade** your AppMon license in eServices to not lose UEM volume.

3. Shut down the old version

1. **Stop** the old Client, Server, Frontend Server, Collector and Memory Analysis Server services.
2. **Disable** automatic startup of the old Server, Frontend Server, Collector and Memory Analysis Server:
   - Windows: Choose **Control Panel > Administrative Tools > Services**. In each service's properties, set the **Startup type** to **Manual**.
   - **Unix-like systems**: Use `chkconfig` or `update-rc.d` to disable all AppMon services.

4. Create backups

Using the configuration information collected during preparation:

1. Back up your **Performance Warehouse DB**.
2. Back up your **Session Store**.
3. **On *NIX**, back up all AppMon scripts in `/etc/init.d`.

Files in `DT_HOME` are backed up during the next step.

5. Services migration (Server, Collector and Memory Analysis Server)

Download and use the current Migration Tool on all Server and Collector machines as follows.

For migration without backup archives and further options see Migration tool details or run `java -jar dynatrace-migration.jar`.

5.1. File collection

Use `dynatrace-migration` to create a migration archive. For example: `java -jar dynatrace-migration.jar -migration -sourceDTHome "<DT_HOME_OLD>" -targetArchiveDir "<ARCHIVE_DIR>"`

Where `<DT_HOME_OLD>` is the old AppMon installation path. A `<MIGRATION_ARCHIVE>` file named `<Server_name>_<creation_dateTime>.dtma` is created in `<ARCHIVE_DIR>`.

Keep the resulting file as permanent backup.

5.2. File migration

On the target host, as Administrator or configured user, migrate the data to `<DT_HOME_NEW>` using the following: `java -jar dynatrace-migration.jar -migration -sourceArchive "<ARCHIVE_DIR>/<MIGRATION_ARCHIVE>" -targetDTHome "<DT_HOME_NEW>"`

Backups of files are changed in `<DT_HOME_NEW>` and a migration log are created in a new directory in `<DT_HOME_NEW>/migration/backup/`.

5.3. Edit configuration files

dynatrace-migration lists files with configuration settings that may need to be migrated manually to the new release. Therefore, the applicable old files are copied with the suffix `.toBeMigrated`. These files are found in `<DT_HOME_NEW>`. **Do not reuse** the old files, because for AppMon 6.3 and later, changed boot bundles cause your Server to not start.

Here is a sample output:

```
There are some files left, which have to be migrated manually:

In <DT_HOME_NEW>:
 \dtanalysisserver.ini -> edit and integrate custom settings from old \dtanalysisserver.i
 \dtcollector.ini -> edit and integrate custom settings from old \dtcollector.ini.toBeMig
 \dtfrontendserver.ini -> edit and integrate custom settings from old \dtfrontendserver.i
 \dtserver.ini -> edit and integrate custom settings from old \dtserver.ini.toBeMigrated
 \server\selfmonitoring\dtselfmon.ini -> edit and integrate custom settings from old \ser

Do NOT re-use the old files, this may cause components to not start!
```

- `dtserver.ini`: If you are upgrading and using continuous transaction storage (see Prepare to Upgrade), do **not** migrate `-Dcom.dynatrace.diagnostics.memory.maxPurePathBufferSize=<X>` if present.
- `dtserver.ini`: If it contains an entry regarding SQLSERVER, follow this KB article.
- Migrate all `-Xmx`, `-Xms`, `-XX:PermSize`, and `-XX:MaxPermSize` entries.

See more at Set up the Server and Set up Collectors.

5.4. Register and Auto-start additional collector instances

Expand for information on instances

If you used Collector instances in the old installation:

- **Windows:** In the directory `<DT_HOME_NEW>`, for each Collector instance, using the same `<CollectorInstanceName>`s as in the old version, execute:

```
dtcollector -service install -instance <CollectorInstanceName>
```

- ***NIX:** In `/etc/init.d`, for each Collector instance, there is an old startup script named `dynaTraceCollector<NN>`, where `<NN>` is a number. For each such file:
  - Take note of the values for the variables `DT_OPTARGS` and `DT_INSTANCE`.
  - Replace it with the new version: `cp <DT_HOME_NEW>/init.d/dynaTraceCollector/etc/init.d/dynaTraceCollect`
  - Edit `/etc/init.d/dynaTraceCollector<NN>` and fill in the noted values for `DT_OPTARGS` and `DT_INSTANCE`.

6. Auto-start new services

***NIX**: With `chkconfig` or `update-rc.d`, make sure that the new Server, Collectors, Memory Analysis Server, and Frontend Server are started automatically with `/etc/init.d` shell scripts and that the permissions are correct.

7. Start the new Server components

If you configured a `DT_HOME` environment variable, update it to `<DT_HOME_NEW>`, and then start the AppMon Server and Frontend Server.

8. Client migration

If it is acceptable for your users to reconfigure their client themselves (proxy settings etc), skip this step.

Otherwise, compare old and new files and carry over your changes:

- After a first client start in each user home the old and new `dtclient.ini`
- Defaults for new users on a machine: `<DT_HOME_OLD>\dtclient.ini.template` and `<DT_HOME_NEW>\dtclient.ini.template`

Where is the Dynatrace user home?

- **Windows**: `C:\Users\<username>\.dynaTrace\Dynatrace 7.0\`
- ***NIX:** `~/.dynaTrace/Dynatrace 7.0/`

9. Start the new Client

In `<DT_HOME_NEW>` start `dtclient` or use the Webstart Client at `https://<servername>:8021`.

Be sure to connect to the right AppMon Server (hostname) in the **Settings > Dynatrace Server > Connectivity** pane.

10. Activate licenses for the new Server

If you require a proxy server to access the Internet, manually configure the Client proxy settings to access the online licensing eServices site.

To import the Server and UEM licenses, click **Settings > Dynatrace Server > License > Import** and follow the instructions.

11. Select server sizing

After the license import, you are presented with different sizing options (see Sizing Settings for more details). Select the Sizing that you identified during the Preparations. It is important to perform this step **before starting the new Collectors**, to avoid Out-of-Memory problems.

12. Apply update to new AppMon

Check the download page if there is an update available for AppMon 7.0, use the Client to install the Update and follow the instructions. This restarts and updates Collectors and Agents.

13. Connect the Performance Warehouse

In **Settings > Dynatrace Server > Performance Warehouse**, verify configuration for the old database, click **Test** and **Connect**.

If you encounter an error, please see Performance warehouse migration troubleshooting.

14. Configure session storage

If you want to migrate your stored sessions, choose **Settings > Dynatrace Server > Storage** and verify that the directory points to your session storage data.

15. Start new Collectors

Start the new Collectors, then choose **Settings > Dynatrace Server > Collectors** and verify that all Collectors are connected to the Server. If you applied an update(to the new Server), you should immediately restart the Collectors from this dialog box for the update to be applied.

The AppMon Server and Collector now accept Agent connections.

Next: Upgrade Agents

Upgrade agents

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Related pages

Component compatibility

Upgrading zOS Agents

Set up agents

Agent Development Kit (ADK)

Migration steps

Prepare to upgrade

Upgrade server, collectors, and clients

Final migration checks

Upgrading the Agents is **recommended** and required to receive updates. Restarting an agent here means usually to restart **the instrumented application**.

Old Agents are compatible and continue to function.

1. Bootstrapped Agents

Includes: Java, .NET, Host, Native ADK, PHP, CICS, and IMS

Make sure the Agents are **connected**, then **restart** to upgrade them automatically. They continue to use the `<DT_HOME_OLD>` directory for retrieving their configuration. You can switch them to `<DT_HOME_NEW>` by upgrading the Bootstrap Agents (see below).

1.1. Bootstrapped Web Server Agents

Includes: Web Server

In the correct order:

1. Restart the Web Server Agent **Service** and wait until it has reconnected.
2. Restart the Web Server Agent.

2. Non-bootstrapped Agents

Includes: Mobile App ADK

Switch your Mobile App to the new ADK and deploy the new version to upgrade. The compatibility statement also applies to Mobile ADK Agents.

3. Bootstrap Agents

For AppMon 6.5 and later, **Bootstrap** Agents with versions 5.6 and earlier are unsupported. Verify their version and if unsupported, upgrade the Agent's bootstrap component.

These features also require minimum versions:

- Restart Webserver Agent Services from the Client (Bootstrap >= 6.2)
- Disable automatic agent updates (Bootstrap >= 6.3)

4. z/OS Agents

z/OS Agents can be bootstrapped or non-bootstrapped. See Upgrading zOS Agents for more information.

Next: Final Migration Checks

Upgrading Bootstrap Agents

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Preparation

Bootstrap agent upgrade process

Uninstall the outdated agents

Related pages

License upgrade

zOS Agent troubleshooting

zDC

Migration steps

Prepare to upgrade

Upgrade server, collectors, and clients

Upgrade agents

Final migration checks

AppMon 6.5 and later no longer supports bootstrap agents earlier than 6.0. This page describes how to upgrade your pre-6.0 bootstrap agents to the newest version.

Preparation

1. Download a new agent installer for your platform from downloads.dynatrace.com.
2. Consider a backup of your current agent configuration directory.
3. Deploy and install it on your target platforms next your agents installation which is currently in use.

Bootstrap agent upgrade process

- Host Agent
  1. Stop the old host agent.
  2. Migrate the content in `/conf/dthostagent.ini` (mainly the name and the server) from your old to your new installation directory.
  3. Start the new host agent.
- Webserver Agent
  1. Stop the old web server agent.
  2. Migrate your webserver agent settings in `conf/dtwsagent.ini` from your old to your new installation directory.
  3. Start the new web server agent.
- Java Agent
  1. Reconfigure the path to the agent in you local application configuration (for example, from `-agentpath:/opt/dynatrace-<loldversion>/agent/lib64/libdtagent.so=name=Application_Monitoring` to `-agentpath:/opt/dynatrace-<latestver>/agent/lib64/libdtagent.so=Application_Monitoring`).

     You can also install to and configure a version-independent agent path (for example, /opt/dynatrace-agent/) and move the old version out of that location and install the new version into the same directory.
  2. Restart your application.
- .NET Agent
  - Restart your application after installing the new agents.
- zOS zLocal / zRemote agent
  - Upgrading the zLocal / zRemote agents will automatically upgrade the bootstrap agents. Please follow the instructions here.

Uninstall the outdated agents

- **Linux**: Delete the installation directory of the outdated agents.
- **Windows**: Uninstall outdated agents using Add / Remove programs.

Upgrading zOS Agents

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Requirements

Upgrade procedure

Related pages

License upgrade

zOS Agent troubleshooting

zDC

Migration steps

Prepare to upgrade

Upgrade server, collectors, and clients

Upgrade agents

Final migration checks

Requirements

Upgrade AppMon Servers, Collectors, Agents and Performance Warehouse

Before starting any migration action for your z/OS Agents, read the general Upgrade to 7.0 guide and create a detailed plan for the upgrade of your AppMon Servers, Collectors, other Agents, and the Performance Warehouse and other components. For less maintenance overhead coordinate these upgrades to coincide with application restarts.

Important

The z/OS Agent can't directly connect to an AppMon Collector without first going through a zRemote Agent. Using the zRemote Agent reduces CPU consumption on z/OS. If your current pre-6.2 z/OS Agent deployment omitted the zRemote Agent, you must install it as a part of your upgrade. See zRemote Agent for more information.

License

You need valid licenses for the AppMon Server. See License Upgrade for more information. Although z/OS Agents are licensed MSU-based, the AppMon Server enforces the number of current connected CICS and IMS regions. See Prepare to Upgrade and License Upgrade for more information.

Upgrade procedure

1. Upgrade the AppMon Server and Collector as described in Upgrade to version 7.0 prior to upgrading the mainframe components. While newer zOS Agents are generally not compatible with older AppMon Collectors or Servers, newer AppMon Collectors or Servers usually work with older zOS Agents.

2. Follow the standard Agent installation process. This consists of running the JCL to download the agent distribution file and running SMP/E to populate the target datasets for the new release. For more information, see Download and SMP/E install of the z/OS Agent Components.

3. Do one of the following:

- Stop your current zDC and update the zDC authorized libraries to the version 7.0. Before you restart the z/DC, you should ensure that the zDC parm `DTMSG_SMOSIZE` has been uncommented and that parm `DTMSG_QSIZE` has been deleted. These parameters control whether messages are queued in the zDC dataspace or in the zDC 64-bit shared memory object (SMO) pending their writing to the AppMon Agent.
  The use of dataspaces has been deprecated. If you customize the ZDCSYSIN sample contained in the 5.6 version, you can skip this step because the sample already incorporates this change. After the zDC has been started, the CICS and IMS agents should register within 5 minutes. **- OR -**
- Create a parallel 7.0 environment for the zDC and zLocal (formerly USS) Agent. Follow the directions as described in Installing the zDC. With a parallel 7.0 environment, you can continue running the old z/DC while you migrate individual CICS regions and IMS control regions over to the new zDC, or you can test with a few regions prior to upgrading the default zDC. If multiple zDCs are running, the one that specifies DEFAULT(YES) will be chosen unless an INITPARM parameter in the CICS or IMS system initialization parameters specifies that it must connect to a zDC with a particular name: INITPARM=(ZDTPLT6x='xxxx').

Notes

- There can be only one zDC per LPAR with the DEFAULT(YES) option.
- The SUBSYSTEM_ID must be unique to each zDC.
- The library used in the zDC STEPLIB must be authorized.

4. The bootstrap Agent automatically upgrades the zLocal Agent when it first connects to an upgraded AppMon Collector / Server. If you use the non-bootstrap version of the z/OS UNIX Agent, you must run COPYAGNT to copy the new version to the location in the z/OS UNIX filesystem where it executes. See the COPYAGNT step in Installing the zDC.

5. Because the AppMon new version zDC is compatible with older CICS and IMS Agents, you don't need to upgrade all CICS regions and IMS control regions at once. However, if you monitor transactions cross CICS regions or IMS control regions, those related regions should be at the same Agent version otherwise incomplete PurePaths will result. Consequently, upgrading CICS regions and IMS control regions should be done with consideration to application groupings. Plan on upgrading all of your CICS regions and IMS control regions as soon as possible to take advantage of the features that come with the new AppMon version.

- To upgrade your CICS regions, replace the DFHRPL concatenation of the old PDS (or its contents) with the new PDS (or its contents). For example: `// DD DISP=SHR,DSN=<hlq>.LZDT700.SZDTLOAD`
- To upgrade your IMS control regions, replace the STEPLIB PDS (or its contents) in the JCL that executes the IMS Agent injection program, ZDTIINST, with the new PDS (or its contents). `// DD DISP=SHR,DSN=<hlq>.LZDT700.SZDTAUTH`

> **Note**
>
> The library used in the STEPLIB must be authorized.

Please refer to Migration Troubleshooting for the AppMon component compatibility table.

Final migration checks

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

1. Manual configuration migration

2. Confirm operation of the new AppMon Server

Related pages

Troubleshooting migration

Migration steps

Prepare to upgrade

Upgrade server, collectors, and clients

Upgrade agents

1. Manual configuration migration

1.1. Reapply modifications to built-in items: sensor packs

Modifications to out-of-the-box sensor packs in the **Server Settings > Sensor Packs > Edit** dialog (only possible in debug mode) are not migrated. Do not export and reimport these old sensor packs, but rather reapply the edits in the UI.

Check user plugins

In the AppMon Client: **Settings > Dynatrace Server**, if plugins installed by the user were not migrated, re-install them.

1.2. Reimport LDAP certificates

If an additional certificate for LDAP (e.g. CA or self-signed certificate) was imported to the AppMon keystore:

- `server/conf/jssecacerts`

don't copy the old keystores, to avoid security issues through potentially compromised public certificates.

Please re-import the previous certificate by logging on with an AppMon local user and click in the AppMon Client: **Settings > Dynatrace Server > Users > LDAP > Test Connection.**

2. Confirm operation of the new AppMon Server

To ensure your new Server operates correctly, verify the following:

- All Agents displayed in **Cockpits > Status Overview > Agent Overview**:
  - Are connected
  - Have data coming in: scroll over horizontally to verify that **Event Count** or **Class Load Count** (or both) change
  - Have the correct Agent version (7.0) and Update version if one was installed.
- PurePaths come in.
- Dashboards are properly displayed.
- Historical data is present if you migrated the Session Storage.
- Verify that the WebUI (default port 9911), Server website (default port 8020) and REST services are reachable and have correct certificates if those were customized.
- Verify that these services and connector work: Email, LDAP, Proxy, DC-RUM, Gomez
- Review the **Start Center > Administration** for errors or warnings

> **Congratulations!**
>
> Migration successful!

Frequently Asked Questions

Forum posts

Open Q&A

Related pages

Troubleshooting migration

Migration steps

Prepare to upgrade

Upgrade server, collectors, and clients

Upgrade agents

Final migration checks

Which agent version runs after the Server/Collector upgrade?

After the Server/Collector upgrade, the Agent bootstrap upgrades the actual Agent, but the old version Agents run until you restart your applications.

Due to a change of the license handling of .NET and IIS agents in AppMon 6.3 and later, it is strongly recommended to restart all .NET and IIS agents on a host after an upgrade from a version earlier than 6.3 to AppMon 6.3 or later.

Please have a restart plan/coordination for your application tiers.

You can find the Agent version (bootstrap and active part) information in the **Agents Overview** dashboard, accessed from the AppMon Client cockpit.

Will old Agents work with the new license version of the Server?

Yes. For example, if you have a 7.0 Server and 7.0 license active on that server, 6.5 Agents consume the license slots as with a 6.5 Server and license. However, be sure to check the general Agent compatibility.

Do I need to install new Agents?

No for most Agents, because they are bootstrapped and upgrade automatically with a restart. But check if your bootstrap version is still supported. See Upgrade Agents for more information.

Do I need to create a Migration archive (.dtma)?

No, but it is highly recommended for backup purposes. See the tool syntax for more information.

Is the Performance Warehouse database schema backward compatible?

No. Please use your database backup to go back.

What version is my Performance Warehouse database schema?

Connecting the new AppMon Server to the old Performance Warehouse database updates the AppMon schema. You can check the version of the DB database.

For example, in SQL Server:

```
SELECT MAX(version) FROM schema_version;
```

If there is no table `schema_version` in your schema (in AppMon 6.3 and earlier), use the following statement:

```
SELECT * FROM dynatrace70.dbo.version;
```

Is the data between Server start and Performance Warehouse connection + end of migration lost (after upgrade)?

Yes, but only the last 10 minutes. Older data is lost.

How much time does the Performance Warehouse migration take during an upgrade?

This depends on the version you are upgrading from and the amount and type of data stored in the database. In general it should be done in a few minutes, but can take up to one hour in worst case scenarios.

Is it possible to migrate old Performance Warehouse data to a new database vendor?

No, if you require this, please submit a request for enhancement.

Is there any effect while the Performance Warehouse migration is running?

No, except only the last 10 minutes of data collected while the migration is running is stored.

In case of a rollback, how do I restore the Performance Warehouse?

Be sure that AppMon is disconnected while restoration. For the restoration, refer to your datyabase vendor documentation on how to restore snapshots.

Will deactivating the license cause an outage?

Not immediately for regular licenses (ecxludes demo and POC). After deactivation, you have a grace period where the license still functions. See license deactivation for more information.

UEM volume deactivates immediately however and does not have a grace period, as the voucher with the remaining visit counts is already returned to eService.

If I need to roll back, how can I get an old license as the original was upgraded?

Keep in mind that roll back should be a last resort scenario, first open a support ticket for your upgrade problem. Then open a license support ticket to get an old license.

How do I handle multiple Collector instances?

Collector instances share one single installation directory and are started and installed with the `-instance` parameter.

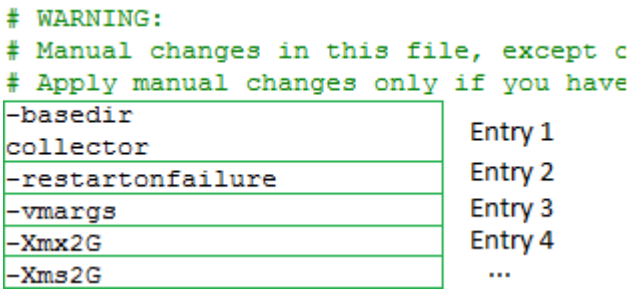
As these instances share certain files such as `dtcollector.ini`, you should shut down all instances when migrating files. The migration tool migrates all files for all collector instances by default, so you don’t need any special parameters. But you must make sure all instances are started from the new location using `init.d` scripts or installing new Windows services (as described in the guide).

How do I handle `.ini` and `.toBeMigrated` files

Using the file `dtcollector.ini` as an example, migration creates two files in your new installation directory:

- `<DT_HOME_NEW>/dtcollector.ini`: This is the new file from the installer for the version 7.0. It is used by the new collector. Any settings from your old installation are included in this file.
- `<DT_HOME_NEW>/dtcollector.ini.toBeMigrated` is a copy of your old `<DT_HOME_OLD>/dtcollector.ini` and stored here by the migration tool for convenience. This file potentially contains settings that you want to keep and integrate into the new file.

You can use a text comparison tool to clearly see the differences. Each entry begin with a "-" and ends right before the next "-", entries can be one-line or two-line. For example:



For each entry that is in `dtcollector.ini.toBeMigrated` and **not** in `dtcollector.ini`, decide if you ignore it or add it to `dtcollector.ini`.

Entries that you should migrate:

- Any entries that you added yourself, either in the file or through the "Debug Options" (Debug flags), mostly they should begin with `-Dcom.dynatrace`.
- Memory configuration lines, which begin with `-Xmx`, `-Xms`, `-XX:PermSize`, `-XX:MaxPermSize`.
- Memory and sizing entries: **-memory VALUE** and **-sizing VALUE**.

Troubleshooting migration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Forum posts

Open Q&A

Related pages

Keystore Migration Troubleshooting

Migration steps

Prepare to upgrade

Upgrade server, collectors, and clients

Upgrade agents

Final migration checks

The migration archive is too large, migration is taking too long.

Case 1: Collectors: Large class cache

The class cache is found in `DT_HOME/collector/cache`. You can use the `-noclasscache` parameter, but keep in mind these remarks

Case 2: Many agentres files

Each update will install one in `DT_HOME/server/lib` and `DT_HOME/collector/lib`. These contain old agent versions that are required if agents of that version connect to the Server or Collectors. Some of those can be removed, but be sure to **carefully** follow these steps to not affect the monitored application:

1. Reduce the number of different Agents **versions** that connect to this installation. In the best case, update to the latest update and roll this out to Server, all Collectors and all Agents.
2. Make a list of all versions that any of your Agents are running on.
3. Shut down all components installed in this `DT_HOME` directory.
4. Do **not** delete `agentres.jar` (without version) or the subdirectory `minified`.
5. Back up and delete any files named `DT_HOME/server/lib/agentres-X.jar` and `DT_HOME/collector/lib/agentres-X.jar` for all versions X that are **not** in use on any of your Agents.
6. Continue with migration guide.

Collector or Server not starting?

See Keystore Migration Troubleshooting

Performance warehouse migration not successful?

See Performance warehouse migration troubleshooting.

PurePaths are missing after upgrade.

If you upgraded without migrating the class cache and are missing PurePaths or have PurePaths that contains `<unknown>` nodes, try to restart the instrumented applications. **Two** restarts of the instrumented applications are required to avoid this problem.

Memory settings in dtserver.ini and dtfrontendserver.ini are ignored and even removed

For the **server** '`-Xmx`' and '`-Xms`' VM arguments are ignored. Please select an appropriate sizing in **Settings > Dynatrace Server > Settings pane > Sizing**. See Sizing Settings for more details.

AppMon support

If you need further assistance, go to our support page.

Collecting support information

For issues with the upgrade or migration process and the migration tool, please provide the following information on support tickets

- A *support archive* with all logfiles if the affected components could be started and did connect - or a zip of the log folder of components that have problems
- The full migration archive (.dtma) if any was created. If no explicit location was specified, it is found it the target DT_HOME/migration/ directory. It can become quite large, if necessary we will provide an FTP location for upload
- All dtmigration.log *files* (source and target) from where the migration tool was executed (working directory)
- The versions of the source and target Dynatrace installation (eg. 6.2 -> 6.5)
- The version of the Migration Guide used (eg 6.3 or 5.6)

Performance warehouse migration troubleshooting

Error Message	DBMS	Affected version	Descriptio
SEVERE [FlywayRepositorySchema] error during performance warehouse migration: com.dynatrace.diagnostics.repository.flyway.FlywayRepositorySchema a:314 org.flywaydb.core.api.FlywayException: Found non-empty schema “DT_APM “ without metadata table! Use baseline() or set baselineOnMigrate to true to initialize the metadata table.	DB2 only	6.3 and later	For DB2, must have the name blanks in to fill up t name car correctly.
SEVERE [AbstractRepositoryAccessor] [PWH SCHEMA UPDATE] Migration V6.3.0.7__createIncidentProperties.ddl failed Error Code : -613 Message : The primary key, unique key, or table partitioning key identified by “INCIDENTPROPERTIES_PKEY” is too long or has too many columns.. SQLCODE=-613, SQLSTATE=54008, DRIVER=4.14.122 Location : ddls/db2/update/6.3.x/V6.3.0.7__createIncidentProperties.ddl (/ddls/db2/update/6.3.x/V6.3.0.7__createIncidentProperties.ddl) Line : 1 Statement : CREATE TABLE incidentproperties (uuid_lsb BIGINT NOT NULL, uuid_msb BIGINT NOT NULL, propkey VARCHAR(100) NOT NULL, value VARCHAR(1000) NOT NULL, CONSTRAINT incidentproperties_pkey PRIMARY KEY (uuid_lsb, uuid_msb, propkey, value))	DB2 only	6.3 and later	The Perfo requires a Performa administra page size since App message unique ke key identi “INCIDEN is too lon columns” this issue
SEVERE [AbstractRepositoryAccessor] cannot create performance warehouse schema: com.dynatrace.diagnostics.sdk.logging.ThrottlingLogger a:150 org.flywaydb.core.api.FlywayException: Found non-empty schema [MS\dtuser] without metadata table! Use baseline() or set baselineOnMigrate to true to initialize the metadata table.	SQL Server only	6.3 and later	Some spe the SQL s usage of the user r name. Th the two co backslash causes pr JDBC driv

The following table shows known PWH migration issues and the solutions. Please have a look at the error messages in the client and in the repository log.

If the migration did not finish successfully or the connection status shows “migration needed”, as a first

step, please check the Performance warehouse administration page for the requirements and needed database permissions. There may be an error message in the client, when you try to connect the Performance Warehouse (depending on your type of issue), which contains good hints for further analysis. Also, the repository log contains very helpful information. See **System Information > Server > Support Information > Log Files**.

Keystore Migration Troubleshooting

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Troubleshooting migration

Certificate management

Advanced features - certificates, private keys and keystore

Symptoms

- Collector or Server is not starting.
- Jetty for DTANG, web services, or WebUI is not starting.

Also, one of the Logfiles in:

- `DT_HOME/log/collector/NAME/Collector.X.Y.log`
- `DT_HOME/log/server/Server.X.Y.log`
- `DT_HOME/log/server/FrontendServer.X.Y.log`

- `DT_CLIENT_HOME/log/client/Client.X.Y.log`
  Has one or several of the following strings:
  - The keystore file 'XXXX' is invalid
  - BrokenKeystoreFileException
  - javax.crypto.BadPaddingException: pad block corrupted
  - The subsequent `java.lang.NullPointerException` and `java.security.UnrecoverableKeyException` may differ a little, depending on the component (server, frontend server, collector, client):
    - [AbstractLifeCycle] FAILED SslContextFactory@47252b0d(/home/dynatrace/dynatrace-7.0.0/server/conf/jetty.jks,/home/dynatrace/dynatrace-7.0.0/server/conf/jetty.jks): java.lang.NullPointerException: org.eclipse.jetty.util.log.JavaUtilLog warn:71 java.lang.NullPointerException
      at org.eclipse.jetty.util.ssl.SslContextFactory.getKeyManagers(SslContextFactory.java:911)
    - [AbstractLifeCycle] FAILED SslContextFactory@1190d87(/home/dynatrace/dynatrace-7.0.0/server/conf/jetty_fe.jks,/home/dynatrace/dynatrace-7.0.0/server/conf/jetty_fe.jks): java.security.UnrecoverableKeyException: Password must not be null: org.eclipse.jetty.util.log.JavaUtilLog warn:71 java.security.UnrecoverableKeyException: Password must not be null

Solution 1

1. Download new version of the migration tool.
2. Recreate migration archive and re-try.

Solution 2

1. Shutdown all components using the affected installation directory.
2. Backup the full `<DT_HOME>/conf` folder.
3. Execute steps that apply.

Troubleshooting dt_pwdstore

This troubleshooting applies to `dt_pwdstore` files only.

In this case 'XXXX' in the exception is one of the following:

- `dt_pwdstore.bks/salt/key`

- `java.lang.NullPointerException at org.eclipse.jetty.util.ssl.SslContextFactory.getKeyManagers(SslContext`
- `java.security.UnrecoverableKeyException: Password must not be null`

1. Delete all `dt_pwdstore.*` and `.dt_pwdstore.*.bkp` files (the default `dt_pwdstore` is extracted at next startup) if you have a normal deployment.
2. Extract the `dt_pwdstore.zip` if you have a customized `dt_pwdstore`.
3. If still failing, delete all keystores, including `*.bks`, `*.salt`, `*.key`, and `*.bkp` keystore files in the folder.

Data lost:

- Nothing.
- All re-initialized stock passwords. Restore these by re-extracting `dt_pwdstore.zip`, or copy all `dt_pwdstore.*` files from a working component.
- *Step 3 only*: See sections of the specific component there data is lost.

Troubleshooting Collector

This troubleshooting applies to Collector only.

In this case 'XXXX' in the exception = `dt_co_keystore.bks/salt`.

The migration tool may copy potentially corrupt files from older deployments and leave needed ones behind.

1. If `.dt_co_keystore.*.bkp` files are present, restore them by removing `'.'` from the prefix and `'.bkp'` from the suffix.
2. Start the component.
3. If still failing, delete `*.bks`, `*.salt`, `*.key`, `*.bkp`.

Data lost:

- Collector proxy password.
- Private key and certificate of the collector.
- Accepted certificates from the AppMon server.

Troubleshooting Client

This troubleshooting applies to the AppMon Client only.

In this case ‘XXXX’ in the exception = `dt_cl_keystore.bks/salt`

Client keystores may be corrupted.

1. Delete `dt_keystore.*`, `.dt_keystore.*.bkp`, `dt_cl_keystore.*`.
2. In case `.dt_cl_keystore.*.bkp` files are present, restore them by removing `.` from the prefix and `.bkp` from the suffix.
3. Start the component. If still failing, delete `*.bks`, `*.salt`, `*.key`, `*.bkp`.

Data lost:

- Collector proxy password.
- Private key and certificate of the Collector.
- Accepted certificates from the AppMon server.

Troubleshooting Frontend Server

This troubleshooting applies to Frontend Server only.

In this case ‘XXXX’ in the exception = `dt_fe_keystore.bks/salt`.

The migration tool may copy potentially corrupt files from older deployments and leave needed ones behind.

1. Delete `dt_fe_keystore.*` if still present.
2. If `.dt_fe_keystore.*.bkp` files are present, restore them by removing `'.'` from the prefix and `'.bkp'` from the suffix. Otherwise, simply start the component.
3. If it’s still failing and `.dt_keystore.*.bkp` files are present, delete `dt_keystore.bks` and `dt_keystore.salt`, restore the `.bkp` files and restart.
4. If still failing, delete `*.bks`, `*.salt`, `*.key`, `*.bkp`.

Data lost:

- Private key and certificate of the frontend server.
- Accepted certificates from the backend server.

Troubleshooting Backend Server

This troubleshooting applies to Backend Server only.

In this case ‘XXXX’ in the exception = `dt_keystore.bks/salt`.

The migration tool may copy potentially corrupt files from older deployments and leave needed ones behind.

1. Delete `dt_keystore.*` if still present.
2. If `.dt_keystore.*.bkp` files are present, restore them by removing `'.'` from the prefix and `'.bkp'` from the suffix. Otherwise, start the component.
3. If still failing, delete `*.bks`, `*.salt`, `*.key`, `*.bkp`.

Data lost:

- Passwords like Gomez integration, PWH integration, SMTP integration, BT Export, Analysis Stream etc.
- Private key and certificate of the backend server.
- Accepted certificates from the memory analysis server.

Troubleshooting Memory Analysis Server

This troubleshooting applies to Memory Analysis Server only!

In this case ‘XXXX’ in the exception = `dt_al_keystore.bks/salt`.

The migration tool may copy potentially corrupt files from older deployments and leave needed ones behind.

1. Delete `dt_al_keystore.*` if still present.
2. If `.dt_al_keystore.*.bkp` files are present, restore them by removing `'.'` from the prefix and `'.bkp'` from the suffix, Otherwise simply start the component.
3. If it’s still failing and `.dt_keystore.*.bkp` files are present, delete `dt_keystore.bks` and `dt_keystore.salt`, restore the `.bkp` files restart.
4. If still failing, delete `*.bks`, `*.salt`, `*.key`, `*.bkp`.

Data lost:

- Private key and certificate of the memory analysis server.

Migration Tool

Related pages

Migrating server instances to separate installations

Migration tool details

Migration tool syntax

These pages provide detailed instructions and examples for using the migration tool when using the Upgrade and Migration guide to perform an AppMon installation migration or upgrade.

Click an entry below or in the Related pages to learn more about using the migration tool.

Migration tool details
Migration tool syntax
Migrating server instances to separate installations

Migration tool details

On this page

Overview

Migration Tool requirements

Use the Migration Tool

Examples

Related pages

Migration tool syntax

Troubleshooting migration

Migrating server instances to separate installations

Running multiple server instances on one system

Server.config.xml changes for server Instance removal

Upgrade server, collectors, and clients

Upgrade and migration guide

Overview

The AppMon Migration Tool (`dynatrace-migration`) is a command line tool that can backup and migrate configuration data from an existing version to a new AppMon version.

You can download the latest version here. The download is an executable, so there is no need to extract the jar file.

See the AppMon 2017 May Upgrade and Migration Guide for full instructions on upgrade or migration. Click here for previous versions of the Upgrade and Migration Guide. These pages also contain details on Performance Warehouse and Session Store handling.

By their very nature the Performance Warehouse DB and the session store in the file system need a different strategy for backup / migration (database / file-based backup).

Migration Tool requirements

- Java 7 or later is recommended, but Java 6x is also supported. AppMon includes a Java version in `<DT_HOME>/jre` that can be used if Java is not otherwise available.
- All source or target installation components such as the Server and Client must be shut down before running the Migration Tool. This is necessary because files like the Collector class cache are constantly changing, and files may also change while the tool runs. This can result in corrupt files.

Use the Migration Tool

Call the Migration Tool without command line parameters for a list and descriptions of the parameters you can use.

Examples

Here are some usage examples beyond the basic one or two-step services migration as described in the first two service migration steps in Upgrade server, collectors, and clients section.

Upgrade and migrate in one command

You can migrate using a single command if you can access the file systems of both old and new installations at the same time. Specify target locations for `sourceDTHome` and `targetDTHome`. This example migrates an AppMon full package installation on Windows:

```
java -jar dynatrace-migration.jar -migration -sourceDTHome "C:\Program Files\dynaTrace\Dynatrace 6.5" -targetDTHome "C:\Program Files\dynaTrace\Dynatrace 7.0"
```

Creating a backup archive

See AppMon Backup for how to use the migration tool to backup and restore critical files.

Upgrading with old Agent versions

If you have a need to keep old agent versions, it is possible during upgrade to keep the same agent version across application restarts. This command requires a target version of AppMon 6.3 or later, as well as bootstrap Agents 6.3 and later. The new bootstrap Agents must be connected to the old version Collectors before starting the Server upgrade.

```
java -jar dynatrace-migration.jar -migration sourceArchive "C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program Files\dynaTrace\Dynatrace 6.3" -upgradeAgents no
```

Agentres modification

Upgrading from AppMon 6.2 and earlier to AppMon 6.3 and later modifies the current version's `agentres.jar` to include the JavaScript Agent. This lets existing Webserver Agents inject the exact matching version. If the exact version match isn't found, the next higher version is injected. This is only relevant if there are Agents that have not been upgraded to AppMon 6.3 or later.

Use the `-noAgentResModification` flag to disable this modification, for instance:

```
java -jar dynatrace-migration.jar -migration -sourceDTHome "C:\Program Files\dynaTrace\dynaTrace 6.2" -noAgentResModification -targetDTHome "C:\Program Files\dynaTrace\Dynatrace 7.0
```

Upgrade and migrate without class cache

Data in the Collector class cache can be more than 10GB, which can take a significant amount of time and can pose a problem when transferring the file to the target location.

You can use the `-noclasscache` parameter to skip the class cache migration. However classes need to be newly discovered, therefore until the next Agent restart:

- HotSensor placement will not work.
- Class Browser will not show classes.
- Unknown PurePath nodes can happen with Agents earlier than 6.0.

Therefore the usage of this flag is *not recommended* and *by default* the Collector class caches are migrated.

This command migrates the configuration data from the archive, but not the class cache. `java -jar dynatrace-migration.jar -migration -sourceArchive "C:\migration\mymachine_<date>_<time>.dtma" -noclasscache -targetDTHome "C:\Program Files\dynaTrace\Dynatrace 7.0`

Select default and additional server and collector instances to migrate and upgrade migrate from archive

Upgrade and migrate all

This command example migrates and upgrades all default and additional AppMon Server and Collector instances from a specified archive. The command demonstrates the default behavior with for all parameters.

```
java -jar dynatrace-migration.jar -migration -sourceArchive "C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program Files\dynaTrace\Dynatrace 7.0"
```

Upgrade and migrate default server and collector instances only

```
java -jar dynatrace-migration.jar -migration -sourceArchive "C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program Files\dynaTrace\Dynatrace 7.0" -migrateInstances default
```

Upgrade and migrate default server instances

```
java -jar dynatrace-migration.jar -migration -sourceArchive "C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program Files\dynaTrace\Dynatrace 7.0" -migrateInstances defaultServer
```

Upgrade and migrate default collector instance

```
java -jar dynatrace-migration.jar -migration -sourceArchive "C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program Files\dynaTrace\Dynatrace 7.0" -migrateInstances defaultCollector
```

Upgrade and migrate additional instances only (not default server and collector)

```
java -jar dynatrace-migration.jar -migration -sourceArchive
"C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program
Files\dynaTrace\Dynatrace 7.0" -migrateInstances instances
```

Upgrade and migrate additional server instances (not default server)

```
java -jar dynatrace-migration.jar -migration -sourceArchive
"C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program
Files\dynaTrace\Dynatrace 7.0" -migrateInstances ServerInstances
```

Upgrade and migrate additional collector instances (not default collector)

```
java -jar dynatrace-migration.jar -migration -sourceArchive
"C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program
Files\dynaTrace\Dynatrace 7.0" -migrateInstances CollectorInstances
```

Upgrade and migrate, and specify alternative migration backup directory

```
java -jar dynatrace-migration.jar -migration -sourceArchive
"C:\migration\mymachine_<date>_<time>.dtma" -targetDTHome "C:\Program
Files\dynaTrace\Dynatrace 7.0" -backupDir "D:\migrationBackup"
```

Undo last migration step

```
java -jar dynatrace-migration.jar -undomigration -targetDTHome
"C:\Program Files\dynaTrace\Dynatrace 7.0"
```

Migration tool syntax

Related pages

Migration tool details

Troubleshooting migration

Upgrade and migration guide

```
java -jar dynatrace-migration.jar
--------------------------------------------------------------------------------
 Dynatrace Migration Tool       Copyright (C) 2004-2017 Dynatrace
--------------------------------------------------------------------------------
 Version 10.0.0.1258 built 2017-01-25T08:18:53.172-0500
  Startup 2017-03-30 14:40:19-0400

 -migration [{<option>}]
   Migrate, upgrade or backup an installation. Select an <option>:
     -sourceDTHome <directory> [-noClassCache]
            Dynatrace installation directory to migrate from.
            [-noClassCache]
            Instructs to not include the class cache (can require gigabytes).
     -sourceArchive <archivepath>
            Migration archive (.dtma) to migrate from.

    Select exactly one target option:
     [-targetArchiveDir <directory>]
            Target directory for the migration archive.
     [-targetArchive <filename>.dtma]
            Target path and filename for the migration archive.
     [-targetDTHome <directory> [-migrateInstances <instanceoption>]]
            Dynatrace installation to migrate to.

            -migrateInstances <instanceoption>
            Without -migrateInstances:
             All default and additional Server and Collector
             instances will be migrated.
            With <instanceoption>     will be migrated:
```

```
            with <instanceoption> ... will be migrated.
             default: only the default Server and Collector instance
             defaultServer: only the default Server instance
             defaultCollector: only the default Collector instance
             instances: only Server and Collector instances
             ServerInstances: only Server instances
             CollectorInstances: only Collector instances

            -makeServerInstanceDefault <instancename>
              EXPERIMENTAL Convert server instance to default instance
            -makeCollectorInstanceDefault <instancename>
              EXPERIMENTAL Convert collector instance to default instance
            -upgradeAgents [yes/no] Agents keep current version (target >= 6.3)
            -noAgentResModification Don't modify active agentres file

     [-backupDir <directory>]
            Optional: backup directory for files replaced during migration and
              to log migration steps and status
            If not present the backup directory is
                targetDTHome/migration/backup.

     -silent                 execute command without prompt.
     -skipDTHomeCheck        allow source or target directories that aren't Dyna
trace installations
     -skipRunningCheck       ignore running components
     -config <customConfig>  Use <customConfig> instead of default.
                             Mutually exclusive with [-migrateInstances <instanc
eoption>]]!
     -ignoreSpace            ignores required free space calculations
     -dropInvalidFilenames   drops files with invalid filesnames
     -ignoreNoReadAccess     ignore missing read access on sourceDTHome

 -undomigration [{<option>}]
   Where <option> includes:
     -targetDTHome <directory>
                             Dynatrace home directory for which you want to
                             revert the last migration step
     -silent                 execute command without prompt.

 -releaseNotes
 -version
```

Migrating server instances to separate installations

Related pages

Running multiple server instances on one system

Server.config.xml changes for server Instance removal

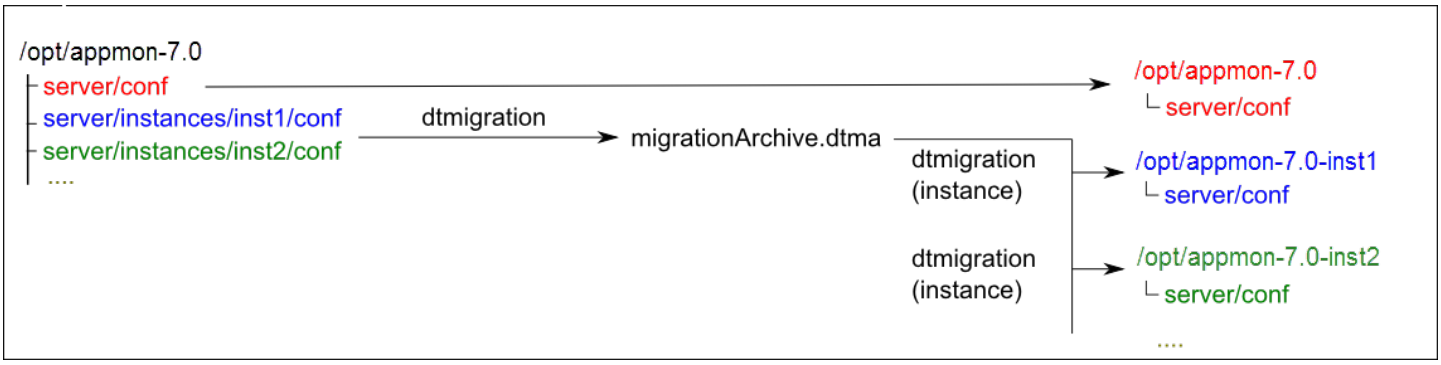
Upgrade server, collectors, and clients

Upgrade and migration guide

Overview

If you use AppMon Server instances that you created using the `dtserver -instance` command, you likely have two or more servers configured. The following steps let you to separate them into two separate installations, or distinct installation folders.

You should not perform an upgrade and separate the instances in a single step. First, split the instances into two distinct installations, for example `inst1` and `inst2`. Once you have split the installation instances, then upgrade each one separately. The following shows the workflow.



Windows requires a separate OS installation for each of the server installations, matching its sizing requirements. This can imply new hardware, virtual machines or Windows licenses. The `.msi` installer prohibits multiple installations on the same OS.

With separate server, differentiate between the following:

- The default server instance with most configuration files in `<DT_HOME>/server/conf`.

- The (zero to many) server instances with the most configuration files in `<DT_HOME>/server/instances/<INST_NAME>/conf`.

Perform the following before making any changes:

1. Download the latest Migration Tool.
2. Stop all components in all instances running from `<DT_HOME>`.
3. Back up all system before starting, especially the existing full installation directory using the following command:
   `java -jar dynatrace-migration.jar -migration -sourceDTHome "<DT_HOME>" -targetArchiveDir "<ARCHIVE_DIR>"`

Separation

Migrate the default server instance using the additional parameter `-migrateInstances default` when migrating from the migration archive to `<DT_HOME_NEW>` which is the target migration directory. See the AppMon 2017 May Upgrade and Migration Guide for more information.

For the remaining server instances, where `<DT_HOME_OLD>` is the original AppMon installation directory, for each AppMon Server instance, where `<INST_NAME>` denotes the instance name, perform:

1. Install a new AppMon Server using the installer on the target host.
2. Migrate all files, including files such as `.ini` files and agent resources shared between the default server instance and the server instance by executing the following command.
   `java -jar dynatrace-migration.jar -migration -sourceDTHome "<DT_HOME_OLD>" -targetDTHome "<DT_HOME_NEW>"` where `<DT_HOME_NEW>` is the selected installation folder, for example `/opt/dynatrace-7.0-instance1`.
3. Execute the following command to migrate the server instance files to the default server instance.
   `java -jar dynatrace-migration.jar -migration -sourceDTHome <DT_HOME_OLD>" -targetDTHome "<DT_HOME_NEW>" -migrateInstances serverinstances -makeServerInstanceDefault "<INST_NAME>"`.
4. Delete the `<DT_HOME_NEW>/server/instances` folder for cleanup and configuration verification.

Port offsets for the previous instance server remain in place.

Manual steps

1. All system profiles of the default instance and the migrated instance are migrated. From `<DT_HOME_NEW>/server/conf/profiles`, remove all `*.profile` files that are only in `<DT_HOME>/server/conf/profiles` and not also in `<DT_HOME>/server/instances/<INST_NAME>/conf/profiles`.
2. Dashboard files have server instance references included. Either open each dashboard in the AppMon Client and set correct data source, or replace the server instance name in XML files before starting the server.
3. For Linux only, update all `init.d` files. To do this, remove the variable values for `<DT_INSTANCE>` and `<DT_FE_INSTANCE>`. Keep the `-portoffset` parameter in the `<DT_OPTARGS>` and `<DT_FE_OPTARGS>` for new services (DTANG Adapter).

Apply general rules for multiple installations on the same system, for instance:

- different startup and shutdown scripts referencing the correct installation (init.d)
- separate TCP / UDP ports

Continue with the Upgrade and Migration Guide after the file migration for each instance.

Running multiple server instances on one system

Related pages

Server.config.xml changes for server Instance removal

Upgrade server, collectors, and clients

Upgrade and migration guide

Note

While running multiple server versions on one machine is possible, it is not recommended. Each server should run on a separate machine to provide optimum server performance and prevent port and other issues.

When different versions of AppMon are installed to the same machine, they have fully separate installation directories, for example `C:\Program Files\dynaTrace\Dynatrace 6.5` and `C:\Program Files\dynaTrace\Dynatrace 7.0`. Each installation instance must have different port assignments. However, because AppMon assigns ports during installation, the same default ports are potentially assigned for each of two separate installations on the same machine. One installation must use non-default port settings. AppMon writes the port settings during initial Server or Frontend Server startup. One of the installation instances must have different ports assigned.

To best resolve this potential issue, you must change the one of the installation instances to use different ports.

To do this, do the following to the second instance installed.

1. During installation, clear the **Start Server** option in the installer.
2. Do one of the following, depending on the version of AppMon being installed:

Then as a user with the necessary permissions, continue depending on the version in that installation directory, with an example offset of 100:

AppMon 6.3 and later* In the proper AppMon installation directory run the following:
`dtserver -portoffset 100`
`dtfrontendserver -portoffset 100`

AppMon versions prior to 6.3 The DTang port is best calculated (to have the same offset for all ports) for the server as `8040+offset` and for collector as `8042+offset`. So in `dtserver.ini` add the entries like this:

```… (other entries) …

-server
-Dcom.dynatrace.diagnostics.dtangport=8140 -Dcom.dynatrace.diagnostics.PortOffset=100

… (other entries) …
```

```
In `dtfrontendserver.ini` (with no DTang entry)

```... (other entries) ...

-server
-Dcom.dynatrace.diagnostics.PortOffset=100

... (other entries) ...
```
```

Server.config.xml changes for server Instance removal

Related pages

Running multiple server instances on one system

Upgrade server, collectors, and clients

Upgrade and migration guide

As part of Migrating Server Instances to Separate Installations, when moving the `<DT_HOME>/server/instances/INSTANCE_NAME/conf/server.config.xml` file to `<DT_HOME_NEW>/server/conf/server.config.xml`, the current migration tool makes the following changes. The changes here are only for reference and manual execution if necessary.

In `server.config.xml` : On the following XML-**elements**:

- `/dynatrace/serverconfig/collectorconfig/loggingconfig` the `path` attribute changes from `../../../log/collector/<INST_NAME>` to `../log/collector/dynaTrace Collector`.
- `/dynatrace/serverconfig/loggingconfig` the `path` attribute changes from `../../../log/server/<INST_NAME>` to `../log/server`
- `/dynatrace/serverconfig/settings/server` the attribute `instancename` is removed.
- `/dynatrace/serverconfig/settings/oopanalyzer` the `logpath` attribute changes from `../../../log/analysisserver` to `../log/analysisserver`.
- If the embedded repository is used: `dynatrace/repositoryconfig/repository/connection` the `embededdatapath` (sic!) attribute changes from `"instances/<INST_NAME>/repository"` to `"repository"`.

Administration

- How to start and stop AppMon services or daemons

- Application environment administration
  - Sensor Configuration
    - Class Browser
    - Configuration change immediacy matrix
    - Hot Sensor Placement
  - System Profile
    - Guided creation of a System Profile
    - Manual creation of a System Profile
    - Enable or disable a System Profile
    - Edit a System Profile
      - System Profile — General
      - System Profile — Sensors
      - System Profile — APIs
      - System Profile — Measures
      - System Profile — Business Transactions
      - System Profile — Schedules
      - System Profile — Tasks
      - System Profile — Monitors
      - System Profile — Incidents
      - System Profile — Error Detection
      - System Profile — Test Automation
      - System Profile — Applications
      - System Profile — User Experience
        - CDN content analysis
        - Resource summary configuration
        - Action groups advanced configuration
      - System Profile — Data Privacy
      - System Profile — Integration
      - System Profile — Observed Tiers
      - Agent Group — General
      - Agent Group — Agent Mapping
      - Agent Group — Sensor Placement
      - Agent Group — Sensor Configuration
        - Sensor Properties
    - Export a System Profile
    - Import a System Profile
    - Rename a System Profile
    - System Profile - Business Hours
    - System Profile templates
- Client administration

- Command line tool
- Deployment dashboards
- Environment incidents
- AppMon Web administration
- JMX management
- Performance Warehouse administration
- Security and compliance whitepaper
- Server administration
  - Connectivity
  - License
  - Sizing settings
    - Confidential strings
  - Services
    - Email
  - Collectors
  - Performance Warehouse
  - Sensor packs
    - Sensors for JavaScript frameworks
  - Users
  - Plugins
    - Language packs
  - Storage
  - Infrastructure
  - Geographical locations - mapping of IP addresses
    - Defining custom locations in the world map
  - Synthetic Monitoring configuration
  - server.config.xml
  - Backup of AppMon settings and data
- User permissions and authentication
  - Permissions mapping

This section explains how to configure and administer the following AppMon components and features:

Deployment dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Working with Deployment dashboards

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Overview

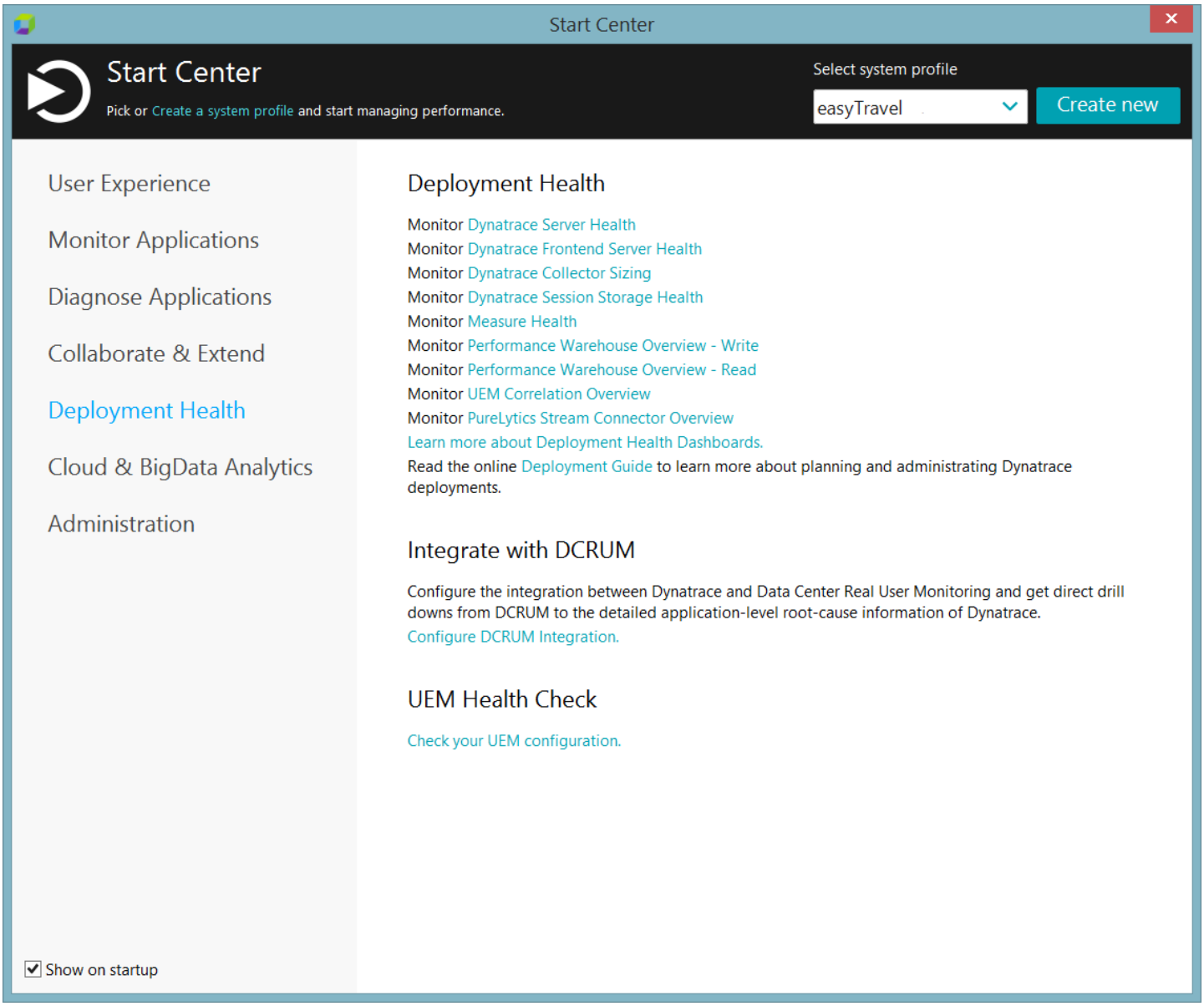
The Deployment dashboards are essential to tune, run, and optimize AppMon deployments. These dashboards help you answer the following questions:

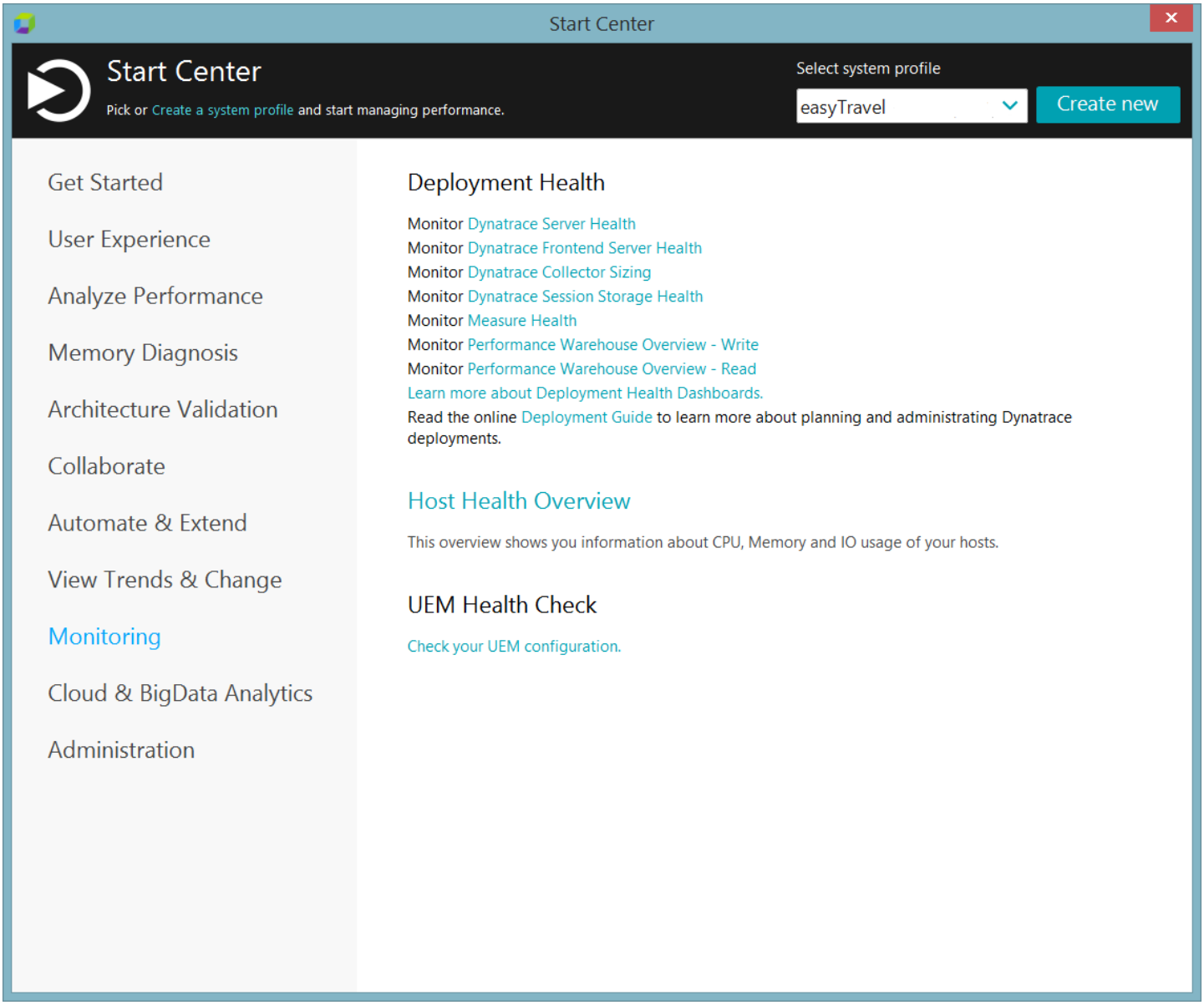
- Does my AppMon deployment perform within the defined parameters?
- Is memory consumption as expected or should I increase it?
- Is the load distribution optimal or should I add an additional Collector?
- Is the load growing or should I add another Collector or Server?

- How can I avoid skipped events, which result in the loss of transactional information on the Agent, Collector, or Server?
- Are total and system profile-based storage requirements sufficient?

Access these dashboards from the **Deployment Health** section of the **Start Center** for the Production Edition and the **Monitoring** section of the **Start Center** for the Pre-Production / Test Center Edition.

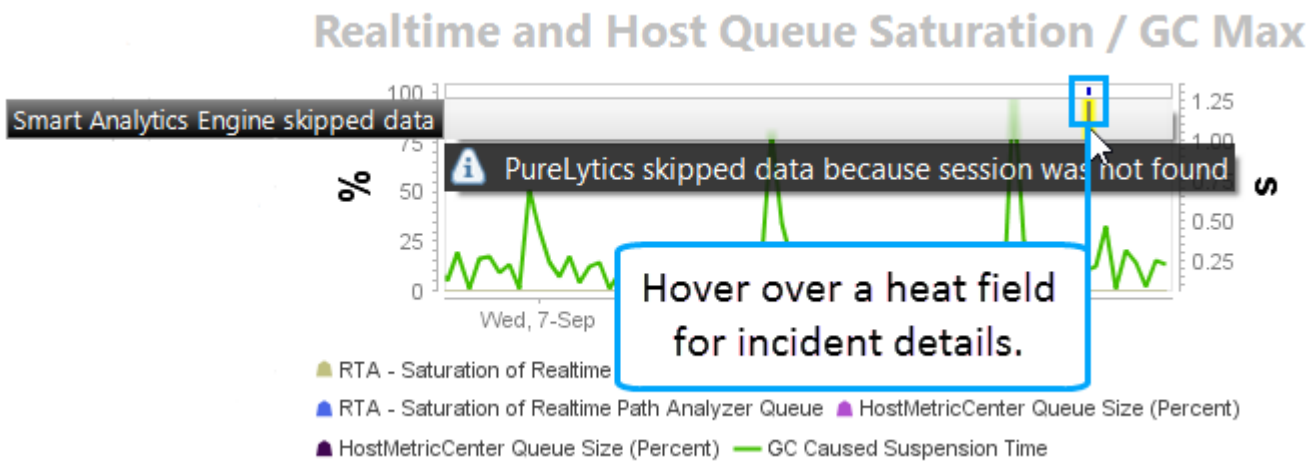
Access the Deployment dashboards from the Start Center





Working with Deployment dashboards

Where applicable, tiles on Deployment dashboards include heat fields, which indicate when incidents related to the chart data have occurred. Hover over a heat field to get details about the incident.
See Heat Field and Incidents Overview Dashlet for more information.



You can arrange any Deployment dashboard's tiles by clicking the tile title and dragging to the desired place on the dashboard, and customize time frames for individual times on any Deployment dashboard. These features let you tweak Deployment dashboards to convey the most pertinent and precise picture of your AppMon deployment. Share any Deployment dashboard to ensure all members of your deployment team share a common understanding of deployment state.

You can see the measurements and metrics used for any deployment dashboard tile, and even edit the measures to suit your needs. Simply click the **Table Visibility** button in the tile toolbar in either the standard dashboard view or the detail tile view and select **Table** or **Chart and Table** to see a combination view. Double-click any tile title to toggle between a detail view and the standard dashboard view.

The following pre-defined dashboards display health-related information about your AppMon deployment:

- AppMon Server Health dashboard
- AppMon Frontend Server Health dashboard
- AppMon Collector sizing
- AppMon session storage health
- Measure Health dashboard
- Performance Warehouse Overview - Write dashboard
- Performance Warehouse Overview - Read dashboard
- UEM Correlation Overview dashboard (Production Edition only)
- PureLytics Stream Connector Overview dashboard (Production Edition only)

AppMon Server Health dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Frontend Server Health dashboard

AppMon Collector Sizing dashboard

AppMon Session Storage Health dashboard

AppMon Measure Health dashboard

AppMon Performance Warehouse Overview - Write dashboard

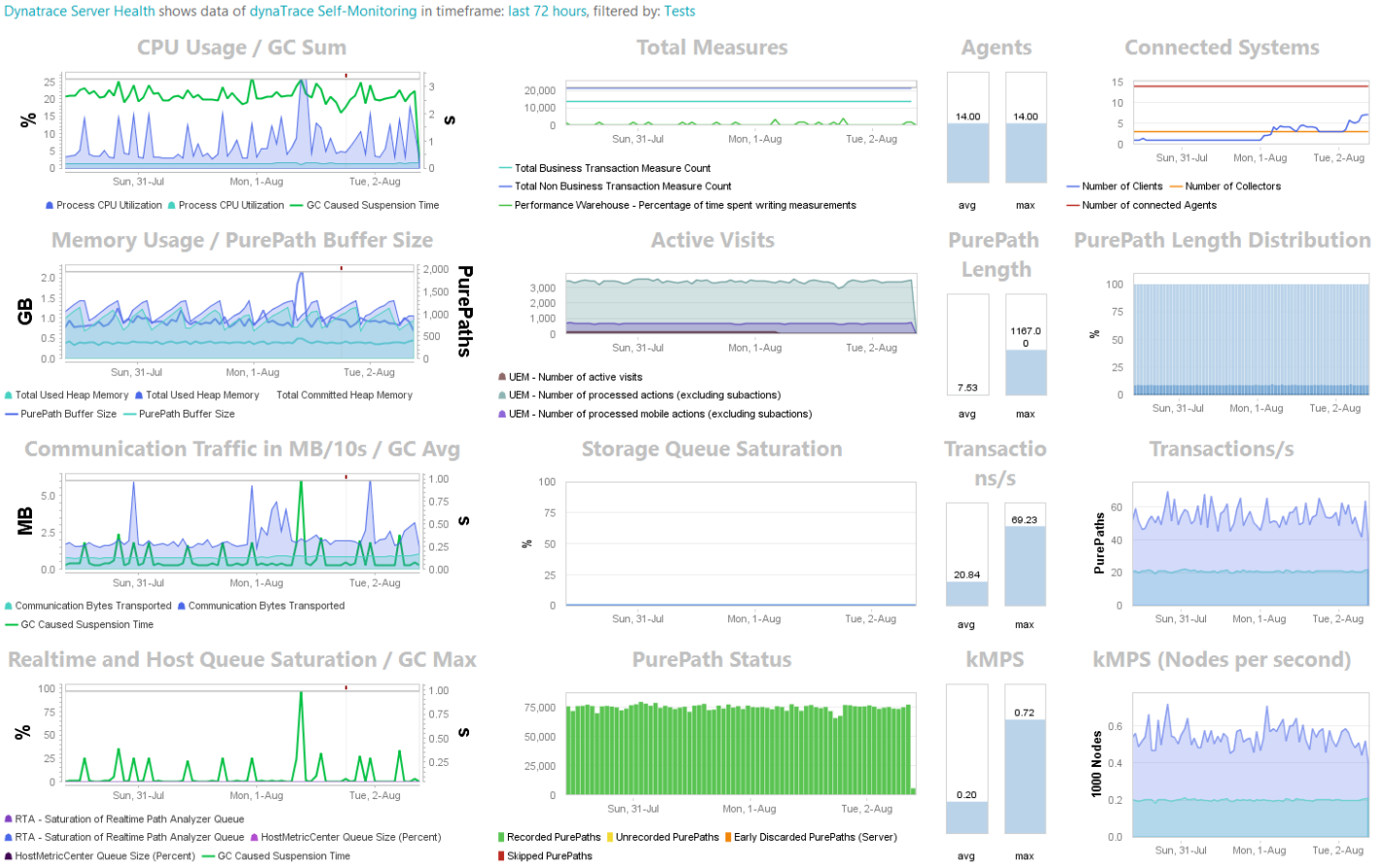
AppMon Performance Warehouse Overview - Read dashboard

UEM Correlation Overview dashboard

PureLytics Stream Connector Overview dashboard

The **AppMon Server Health** dashboard displays the overall health status of your AppMon deployment. It shows the most important performance indicators which help you run a AppMon deployment successfully. This information is valuable to testing teams as they monitor and tune a system, and to production operations teams for continual deployment optimization. Most charts show average and

maximum values, which you can use to differentiate normal loads and spikes.



This dashboard includes the following:

- **CPU Usage / GC Sum**: The left axis shows the percentage of CPU usage for processes (maximum and average aggregation). The right axis shows the amount of suspension time caused by garbage collection, and the maximum CPU consumption on the Server.
- **Total Measures**: Shows the total number of defined Business and non-Business measures, and the percentage of time spent writing measurements to the Performance Warehouse. For real time sessions only, this tile also shows the size of the internal real time analysis and cyclic event queues, and the number of measurements handled by the time series center per second.
- **Agents**: The average and maximum number of connected agents.
- **Connected Systems**: The average number of connected collectors, agents, and clients.
- **Memory Usage / PurePath Buffer Size**: The average and maximum memory that the Server consumes. If you exceed the free memory allocation, Garbage Collection suspensions occur. You can see these in the Suspension Time and Suspension Count graphs. also shows the average and maximum number of PurePaths kept in memory.
- **Active Visits**: Summarizes UEM visit activity, showing the total number of active visits in all VisitStores, and the number of processed actions and number of mobile actions.

- **PurePath Length**: The average and maximum number of nodes in a PurePath. Based on your environment, the average PurePath length should not exceed 300 for production, 500 for test center, and 1000 for development. The impact of long PurePaths include higher CPU load on the monitored application or on the Server with higher MPS numbers, and shorter live session due to an implicated higher MPS.
- **PurePath Length Distribution**: A percentage chart showing the percentage of PurePaths on the server within set PurePath length (instrumentation depth) ranges. The ranges are:
  - Less than 10
  - 10 to 99
  - 100 to 999
  - 1000 or greater
- **Communication Traffic in MB/10s / GC avg**: The average and maximum traffic volume on the Server. A network overload trend indicates that you should reduce the amount of data that your AppMon configuration captures.
- **Storage Queue Saturation**: The maximum and average storage utilization as a percentage of the total storage queue. A warning alert occurs at 80% utilization and a severe alert at 90% utilization.
- **Transactions/s**: The average and maximum number of transactions per second for all agents.
- **Transactions/s (PurePath)**: The average and maximum number of PurePath transactions per second for all agents.
- **Realtime and Host Queue Saturation / GC Max**: The left axis shows percentage of total utilzation (average and maximum) for the real-time analyzer. The right axis shows the duration of suspensions caused by garbage collections. in efficiently configured deployments, the AppMon Collector does not have major garbage collections.
- **PurePath Status**: Shows the number of PurePaths per interval for all agents. Includes:
  - **Recorded PurePaths**: The number of PurePaths written to session storage from the in-memory live session.
  - **Unrecorded PurePaths**: The number of PurePaths that could not be written to the session storage from the in-memory live session.
  - **RTA - Analyzed PurePaths**: (realtime sessions only) The number of complete PurePaths that the real-time analyzer processes. Under certain circumstances (high load, low PurePath buffer size), completed PurePaths may fall out of the queue before analysis.
  - **Skipped PurePaths**: The number of PurePaths that do not get analyzed due to missing events such as not arriving in time.
  - **Early Discarded PurePaths**: The number of Purepaths discarded early in the processing chain, prior to completion. This could be due to a number of factors, including full queues or processing buffers, invalid start/end dates, age expiration, and low server memory.
- **kMPS**: The average and maximum number (x 1000) of measure nodes per second.

- **kMPS (nodes per second)**: A visualization of the current MPS (measures per second) trend. It shows how the load develops over time, which allows you to see a general growth trend early. In such a case, you should do the following tasks:
  - Consider configuration changes to decrease growth
  - Upsize your Server installation and increase the threshold afterwards

AppMon Frontend Server Health dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Server Health dashboard

AppMon Collector Sizing dashboard

AppMon Session Storage Health dashboard

AppMon Measure Health dashboard

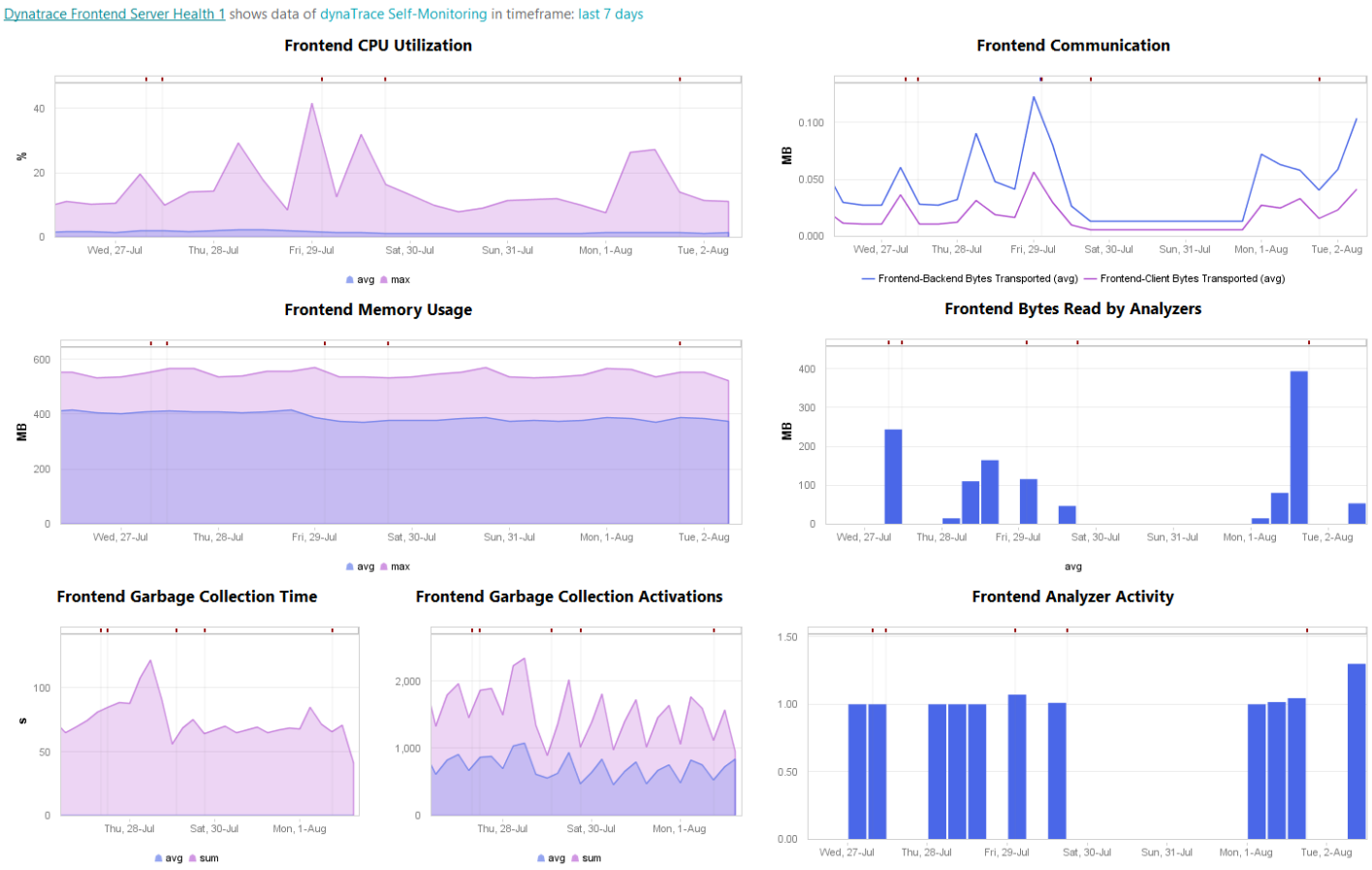
AppMon Performance Warehouse Overview - Write dashboard

AppMon Performance Warehouse Overview - Read dashboard

UEM Correlation Overview dashboard

PureLytics Stream Connector Overview dashboard

The **AppMon Frontend Server Health** dashboard displays important performance and frontend Server activity indicators, such as CPU and memory usage, garbage collection activity, and frontend analyzer activity to the backend and client. This dashboard lets you quickly determine whether frontend capacity is sufficient.



This dashboard includes the following:

- **Frontend CPU Utilization**: The amount of CPU usage as a percentage that the Frontend Server uses.
- **Frontend Communication**: A summary of the frontend to backend traffic and the frontend to Client traffic in MB.
- **Frontend Memory Usage**: The average and maximum memory in MB that the Frontend Server consumes.
- **Frontend Bytes Read by Analyzers**: MB that the analyzers read from the Frontend Server.
- **Frontend Garbage Collection Time**: The average and sum total of Frontend Server garbage collection time in milliseconds.
- **Frontend Garbage Collection Activations**: The average and sum total of Frontend Server garbage collection activations.
- **Frontend Analyzer Activity**: A summary of the Frontend Server activity.

AppMon Collector Sizing dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Server Health dashboard

AppMon Frontend Server Health dashboard

AppMon Session Storage Health dashboard

AppMon Measure Health dashboard

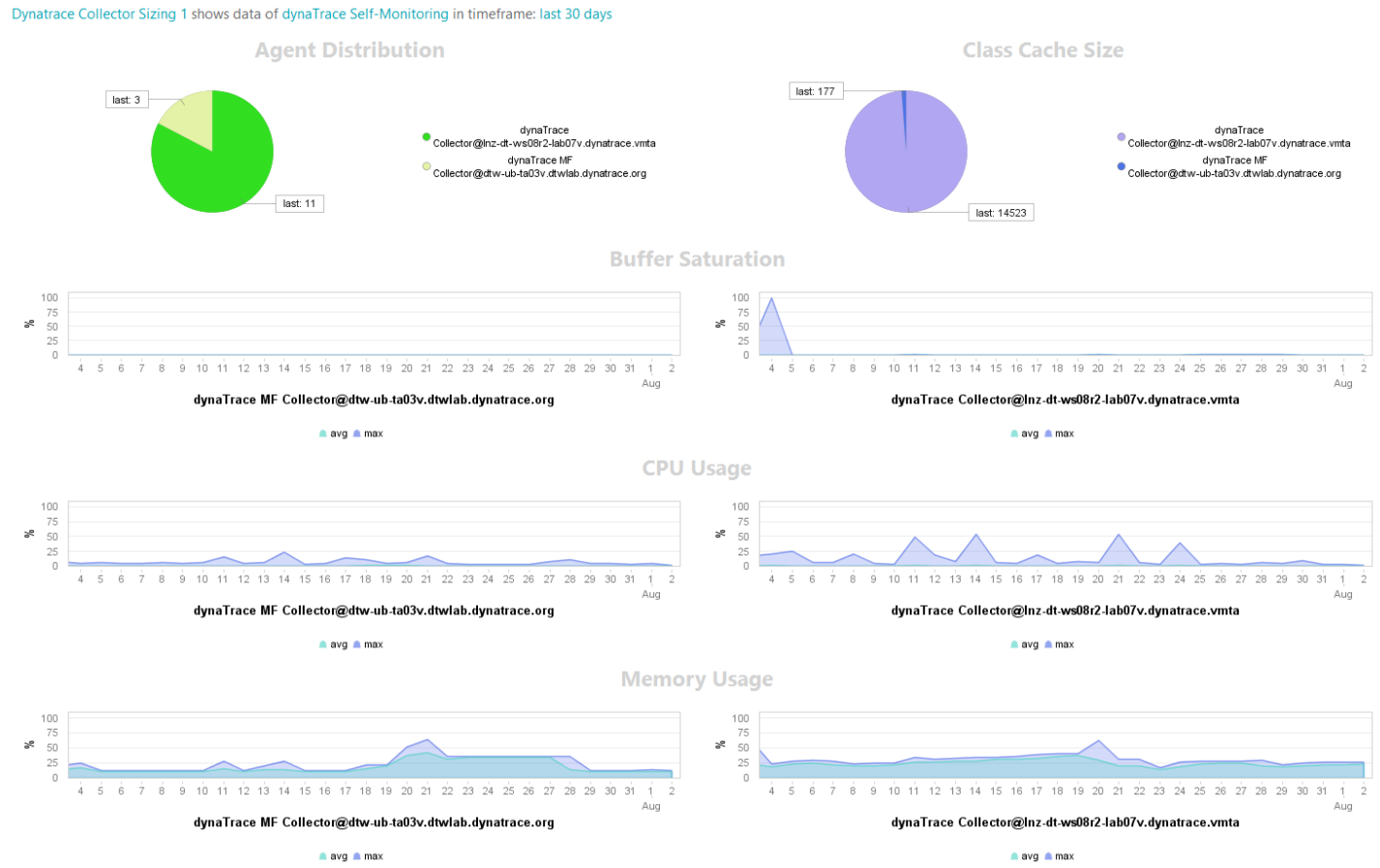
AppMon Performance Warehouse Overview - Write dashboard

AppMon Performance Warehouse Overview - Read dashboard

UEM Correlation Overview dashboard

PureLytics Stream Connector Overview dashboard

The **Dynatrace Collector Sizing** dashboard displays sizing-related configuration issues in your AppMon Collector deployment. Proper Collector sizing ensures a highly scalable deployment that performs well. Use this dashboard to size and tune your system optimally.



Dynatrace Collector Sizing dashboard

This dashboard includes the following:

- **Agent Distribution**: The number of Agents that are connected to each Collector, based on the last available measurement.
- **Class Cache Size**: The number of loaded classes per Collector, based on the last available measurement.
- **Buffer Saturation**: The buffer utilization as a percentage per each connected Collector instance.
- **CPU Usage**: The CPU usage per Collector.
- **Memory Usage**: The memory usage per Collector.

Observe the rules for how to distribute your Agents over Collectors for different applications and subnets. You should balance the load. For example, if there are two Collectors and one receives two thirds of the load (or Agents), redistribute the Agents to balance it.

AppMon Session Storage Health dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Server Health dashboard

AppMon Frontend Server Health dashboard

AppMon Collector Sizing dashboard

AppMon Measure Health dashboard

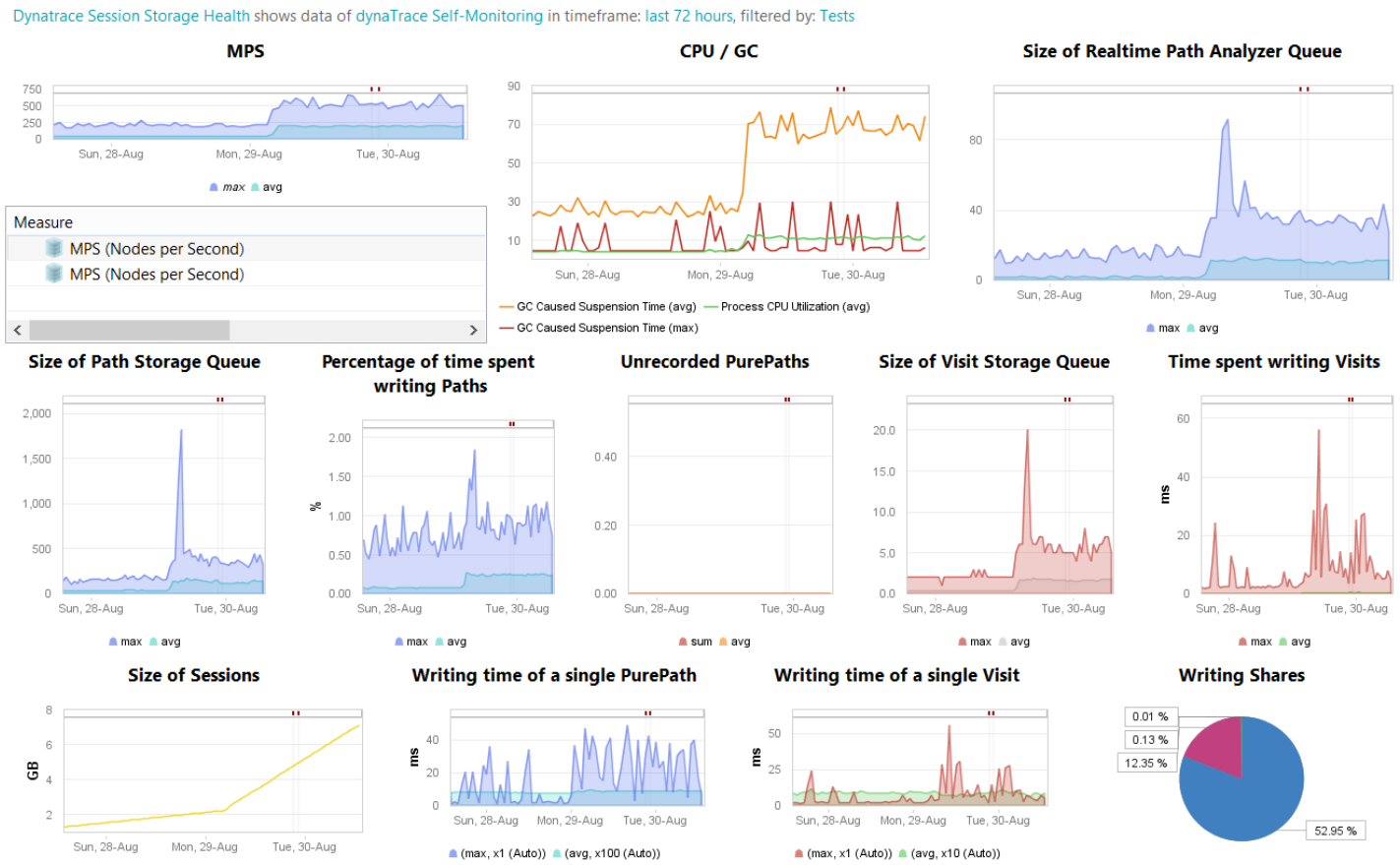
AppMon Performance Warehouse Overview - Write dashboard

AppMon Performance Warehouse Overview - Read dashboard

UEM Correlation Overview dashboard

PureLytics Stream Connector Overview dashboard

Storage requirements change periodically as your AppMon deployment gains or loses users and as user's AppMon activity changes. Use the Session Storage Health Dashboard to help assess whether session sizes are sufficiently configured, whether queue sizes require adjustments, and if any RTA or write issues with storage require investigation.



This dashboard includes the following:

- **MPS**: The average and maximum number of measure nodes per second.
- **CPU / GC**: Shows the percentage of CPU usage for processes in storage, and the average and maximum amount of suspension time caused by garbage collection.
- **Size of Realtime Path Analyzer Queue**: The average and maximum size of the internal real time analysis queue.
- **Size of Path Storage Queue**: The average and maximum size of the internal session storage queue.
- **Percentage of time spent writing Paths**: The time writing sessions to disk as a percentage of total write time.
- **Unrecorded PurePaths**: Total and average number of PurePaths per Interval over all agents that could not be written to the session storage from the in-memory live session.
- **Size of Visit Storage Queue**: The average and maximum size of the internal session storage visit queue.
- **Time Spent Writing Visits**: The average and maximum time amount of time (in milliseconds) spent writing visits to disk.
- **Size of Sessions**: The size per interval of all sessions stored in the designated session directory.
- **Writing Shares**: The percentage of total session write time between writing PurePaths, updates, time series (measurements), and incidents.

AppMon Measure Health dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Server Health dashboard

AppMon Frontend Server Health dashboard

AppMon Collector Sizing dashboard

AppMon Session Storage Health dashboard

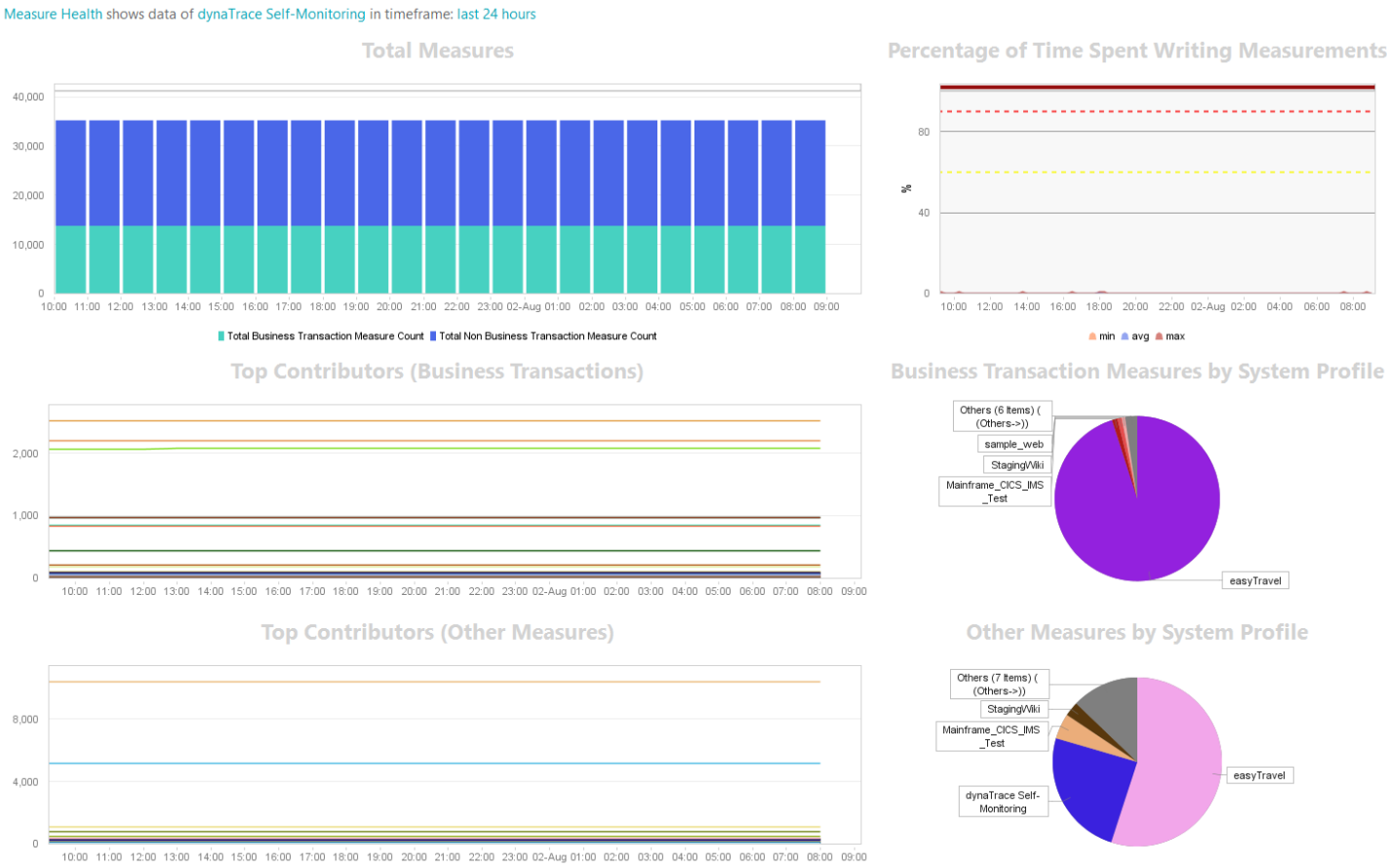
AppMon Performance Warehouse Overview - Write dashboard

AppMon Performance Warehouse Overview - Read dashboard

UEM Correlation Overview dashboard

PureLytics Stream Connector Overview dashboard

The Measure Health dashboard gives you a detailed look into the quantity and efficiency of measure use. Key to this are displayed indicators show you distribution of both business transaction and non-business transaction measures among system profiles, business and non-business transactions measure counts, and the efficiency of measure writing.



Dynatrace Measure Health dashboard

This dashboard includes the following:

- **Total Measures**: The overall number of Measures, with the number of dynamic business transaction Measures in the Performance Warehouse, and the total number of dynamic Measures in the Performance Warehouse.
- **Percentage of Time Spent Writing Measures**: Shows the percentage of a minute that is spent writing Measures over the last 72 hours. A high percentage in this graph indicates slow writing performance.
- **Top Contributors (Business Transactions)**: The business transactions with the largest number of dynamic Measures, split by business transaction name and system profile.
- **Business Transaction Measures by System Profile**: A pie chart that shows how Measures for business transactions are distributed among System Profiles.
- **Top Contributors (Other Measures)**: The non-business transaction Measures with the largest number of dynamic Measures, which are split by Measure type and System Profile.
- **Other Measures by System Profile**: A pie chart that shows how other Measures are distributed among System Profiles.

AppMon Performance Warehouse Overview - Write dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Server Health dashboard

AppMon Frontend Server Health dashboard

AppMon Collector Sizing dashboard

AppMon Session Storage Health dashboard

AppMon Measure Health dashboard

AppMon Performance Warehouse Overview - Read dashboard

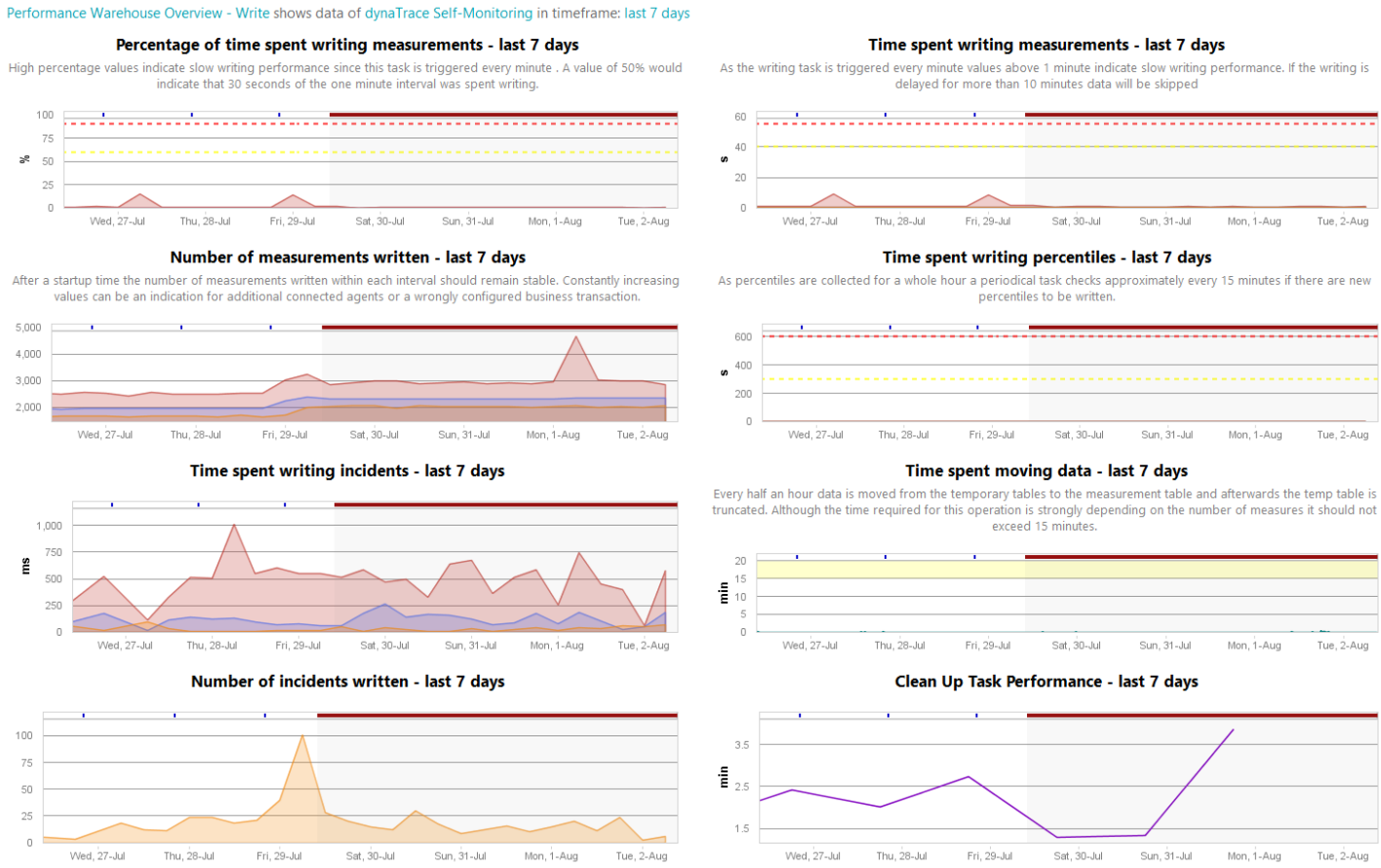
UEM Correlation Overview dashboard

PureLytics Stream Connector Overview dashboard

This dashboard how effectively data is written to the Performance Warehouse. It shows writing times for measures, incidents, and percentiles, time spent moving data from temporary tables to the

measurement table and truncating the temporary table. Performance Warehouse clean-up times also display on this dashboard.

Monitor this dashboard to assess whether high write times require tweaks to Performance Warehouse configuration settings, or if instabilities in measure write times could be caused by improperly configured business transactions.



Dynatrace Performance Warehouse - Write Health dashboard

This dashboard includes the following:

- **Percentage of time spent writing measures**: The amount of time as a percentage spent to write Measures to the Performance Warehouseover the set time frame. A high percentage in this graph indicates a slow writing performance.
- **Time spent writing measurements**: The absolute amount of time used to write Measures. Values that exceed one minute indicate slow writing performance. If writing is delayed for more than 10 minutes, data is skipped.
- **Number of measurements written**: The number of Measures written over the set time frame. After an initial startup phase, this number should remain stable within the time frame. Increasing values can indicate additional connected Agents, or a misconfigured business transaction.

- **Time spent writing percentiles**: The absolute amount of time required to write percentiles which it collects for a whole hour. It writes new percentiles in 15 minute intervals.
- **Time spent writing incidents**: The time spent to write incidents over the set time frame. Spikes in this chart that do not correlate with respective spikes in the number of incidents written may indicate writing performance issues.
- **Time spent moving data**: The amount of time it takes to move data from the temporary measurements table. Values in this chart should not exceed 15 minutes.
- **Number of incidents written**: The number of incidents written to the Performance Warehouse, which is required to interpret the **Time spent writing incidents** chart.
- **Clean Up Task Performance**: The clean up task aggregates data in the Performance Warehouse to the different resolutions and deletes data after the specified duration. Only perform this task when Server load is expected to be low. You may have to run the clean up task more often for high data volume and the resulting long run times. For more information, see Performance Warehouse Administration Storage Management.

AppMon Performance Warehouse Overview - Read dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Server Health dashboard

AppMon Frontend Server Health dashboard

AppMon Collector Sizing dashboard

AppMon Session Storage Health dashboard

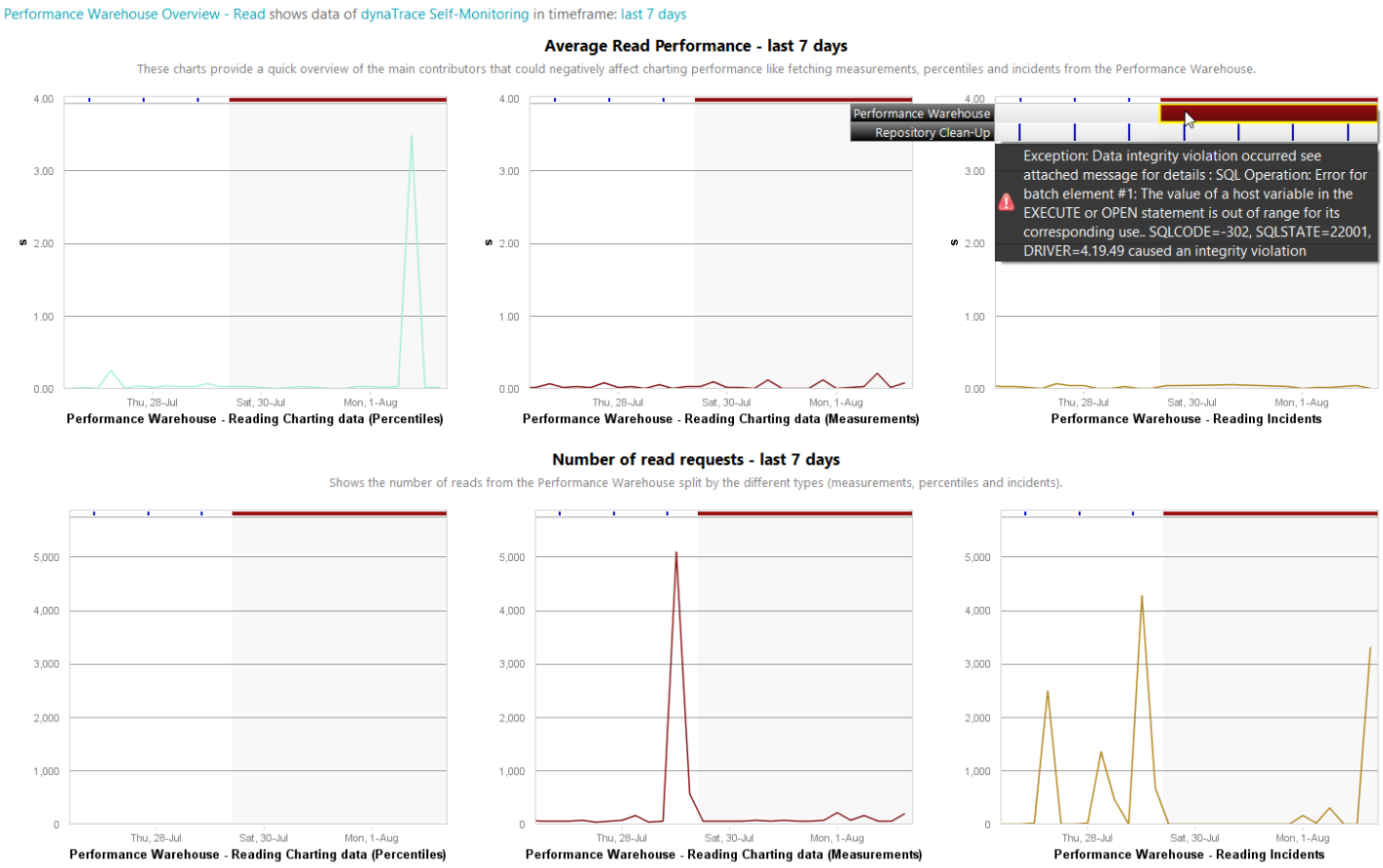
AppMon Measure Health dashboard

AppMon Performance Warehouse Overview - Write dashboard

UEM Correlation Overview dashboard

PureLytics Stream Connector Overview dashboard

Use the Performance Warehouse - Read Health dashboard to determine if factors such as fetching measurements, percentiles, or incidents negatively affect charting performance.



Dynatrace Performance Warehouse - Read Health dashboard

This dashboard includes the following:

- **Average Read Performance**: An overview of main contributors of charting performance over the set time frame.
  - **Performance Warehouse - Reading Chart data (Measurements)**: Shows the average values of the time elapsed for fetching measurements from the Performance Warehouse.
  - **Performance Warehouse - Reading Charting data (Percentiles)**: Shows the average values of the time elapsed for fetching percentiles.
  - **Performance Warehouse - Reading Incidents**: Shows the average values for the time spent reading incidents.

- **Number of read requests**: The number of reads from the Performance Warehouse over the set time frame.
  - **Performance Warehouse - Reading Chart data (Measurements)**: Shows the number of measurements fetched from the Performance Warehouse.
  - **Performance Warehouse - Reading Charting data (Percentiles)**: Shows the number of percentiles fetched from the Performance Warehouse.
  - **Performance Warehouse - Reading Incidents**: Shows the number of incidents fetched from the Performance Warehouse.

UEM Correlation Overview dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Server Health dashboard

AppMon Frontend Server Health dashboard

AppMon Collector Sizing dashboard

AppMon Session Storage Health dashboard

AppMon Measure Health dashboard

AppMon Performance Warehouse Overview - Write dashboard

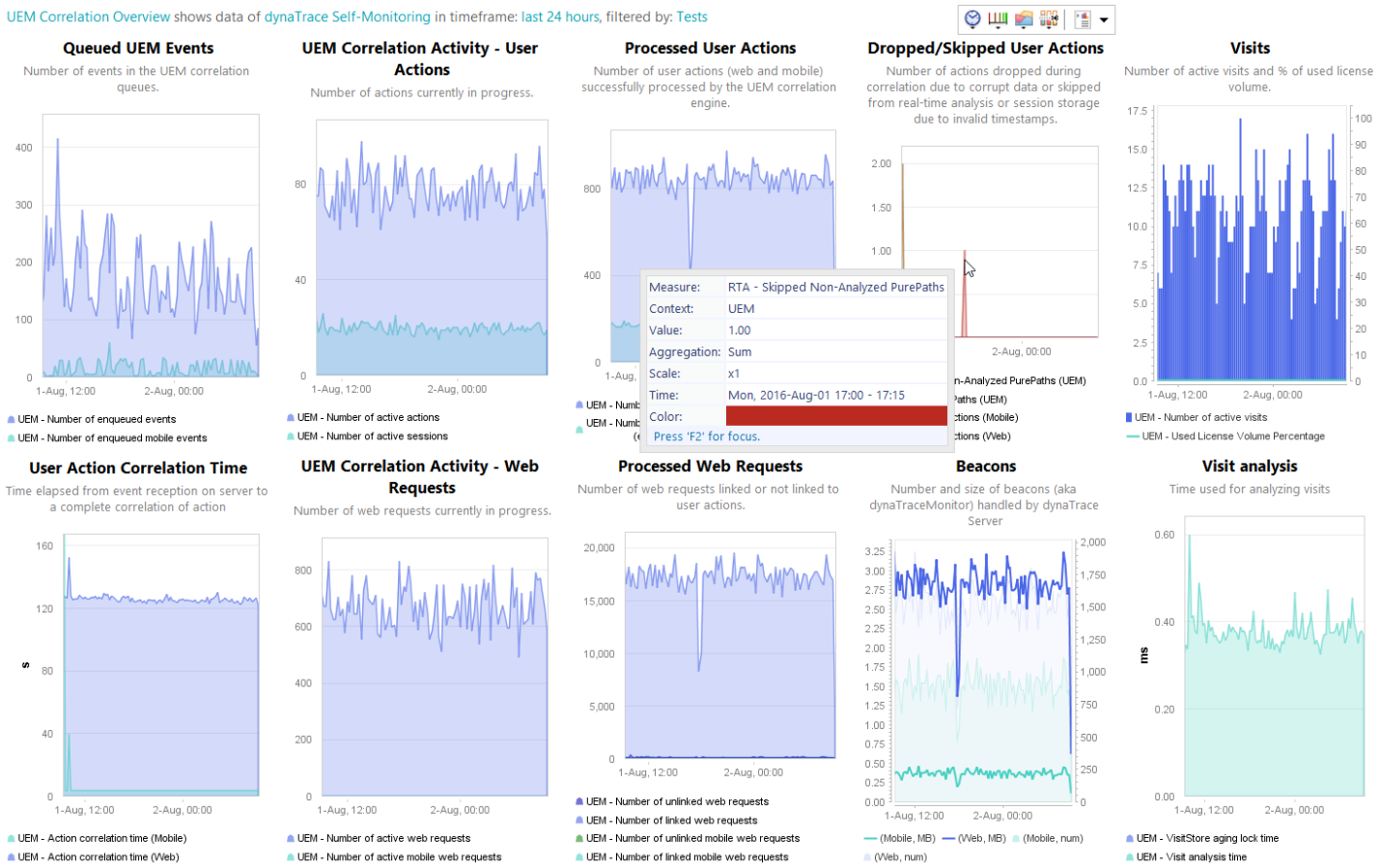
AppMon Performance Warehouse Overview - Read dashboard

PureLytics Stream Connector Overview dashboard

Note

This dashboard is for Production edition only.

For users capturing UEM data, the UEM Correlation Overview dashboard shows a variety of UEM capacity data to give you an overall picture of UEM deployment. User visit throughput efficiency can be assessed from the various correlation tiles, visit handling can be assessed from the Visits, Visit analysis, and other tiles.



This dashboard includes the following:

- **Queued UEM Events**: The number of web and mobile events in the UEM correlation queues.
- **UEM Correlation Activity - User Actions**: The average number of actions in progress per interval.
- **Processed User Actions**: The number of user actions (web and mobile) that the UEM correlation engine successfully processes, and the average number of active sessions per interval.

- **Dropped/Skipped User Actions**: Gives an overview of unsuccessfully processed user actions and PurePaths, including:
  - *RTA - Skipped Non-Analyzed PurePaths*: The number of PurePaths skipped from realtime analysis due to invalid timestamps.
  - *Unrecorded PurePaths*: The number of UEM PurePaths from all agents that could not be written to the session storage from the in-memory live session.
  - *UEM - Dropped Actions*: The number of web actions and mobile actions dropped during UEM correlation due to parsing errors or invalid beacon data.
- **Visits**: The right axis shows the number of active visits. The left axis shows the percentage of used UEM license volumes.
- **User Action Correlation Time**: The amount of time from the reception of an event on the server to a completed correlation of the action and RTA handoff, for both web and mobile-generated actions.
- **UEM Correlation Activity - Web Requests**: The total number of active web requests per interval, and the average number of active mobile web requests per interval.
- **UEM Correlation Activity - CPU**: The CPU utilization by UEM correlation engine.
- **Processed Web Requests**: The number of web and mobile web requests linked or not linked to user actions.
- **Beacons**: The left axis shows the size in megabytes of UEM web and mobile beacons handled by the Server. The right axis shows the number of UEM web and mobile beacons handled by the Server.
- **Visit Analysis**: The average amount of time per interval used to analyze visits.

PureLytics Stream Connector Overview dashboard

Related pages

Host health monitoring

Collector best practices

Storage

System Profile - Measures

Add and edit measures

Server and Performance Warehouse best practices

Performance Warehouse (REST)

UEM Health Check

PureLytics Stream

Other Deployment dashboards

AppMon Server Health dashboard

AppMon Frontend Server Health dashboard

AppMon Collector Sizing dashboard

AppMon Session Storage Health dashboard

AppMon Measure Health dashboard

AppMon Performance Warehouse Overview - Write dashboard

AppMon Performance Warehouse Overview - Read dashboard

UEM Correlation Overview dashboard

Note

This dashboard is for Production edition only.

If you stream real time UEM data an external data source for big data analysis outside of AppMon, the PureLytics Stream Connector Overview dashboard can provide a detailed look into PureLytics streaming activity. You can monitor request, queue and error numbers from a single dashboard to help ensure connection reliability and determine if queue size, dispatch intervals, or other PureLytics configurations need adjusting.



This dashboard includes the following:

- **Number of Requests**: The total number of requests sent to the configured target for the Server Frontend and Backend.
- **Request Size**: The maximum size (in KB) of the requests sent to the configured target for the Server Frontend and Backend.
- **Request Duration**: The maximum duration (in milliseconds) of the requests sent to the configured target for the Server Frontend and Backend.

- **Queue Size**: The size of the queue on the Server Frontend and Backend receives incoming JSON documents.
- **Skipped Documents**: The total number of incoming JSON documents that are skipped and not processed through the PureLytics connector.
- **Number of Errors**: The total number of connection errors for the Server Frontend and Backend

Server administration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Access the AppMon Server settings dialog box

Add or remove a Server

Server configuration

Related pages

Backup of AppMon settings and data

Collectors

Connectivity

Geographical locations - mapping of IP addresses

Infrastructure

License

Performance Warehouse

Plugins

Sensor packs

server.config.xml

Services

Sizing settings

Storage

Synthetic Monitoring configuration

Users

User permissions and authentication

The AppMon Server backend and frontend processes are administered using the Webstart Client or the standalone Client, installed with either the full package, or Client-only installer, or unzipped from the portable Client archive.

From either Client, you can access the **AppMon Server Settings** dialog box if you have an administrator role assigned to your user account. For more information about roles, see User Permissions and Authentication.

Access the AppMon Server settings dialog box

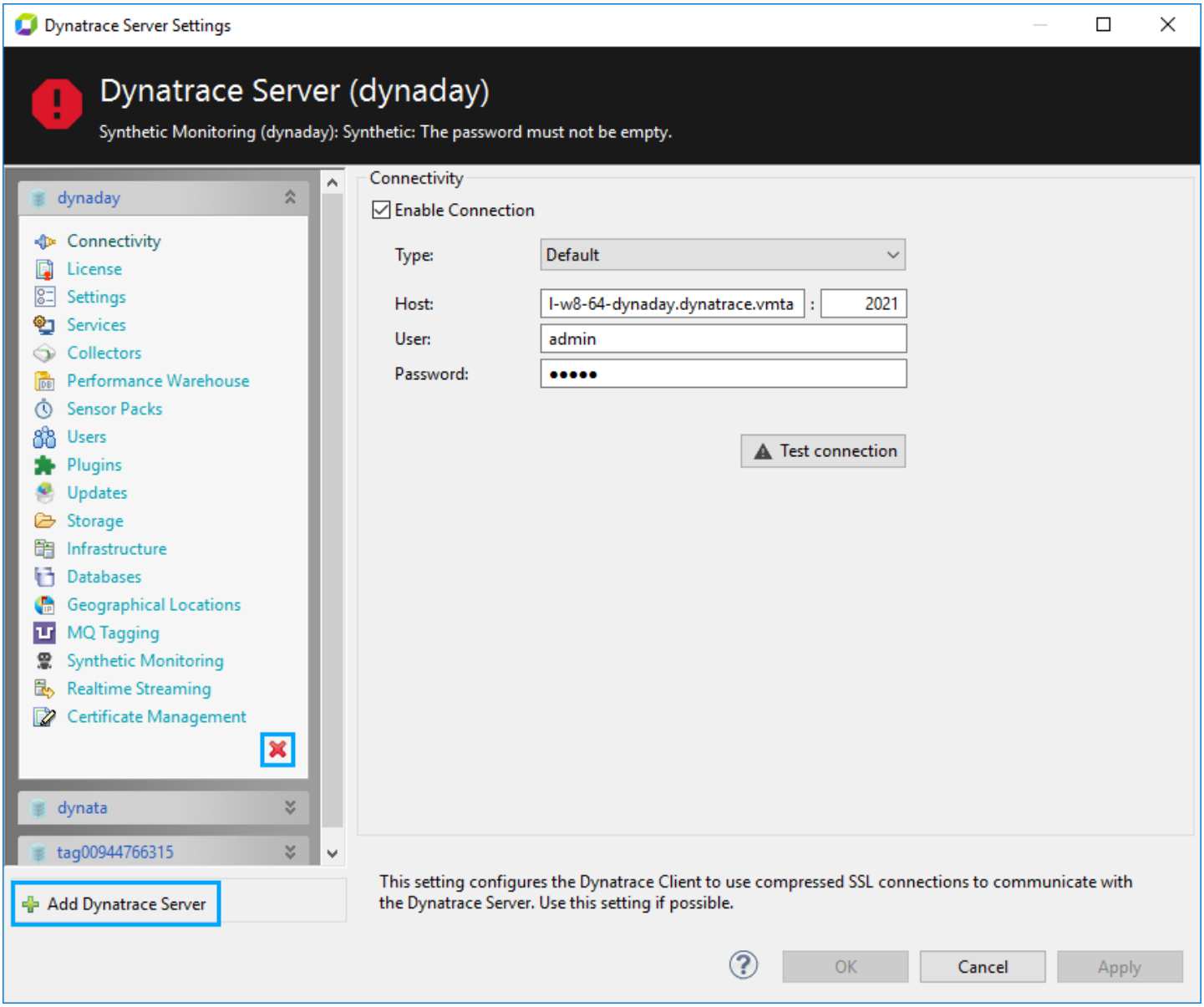
In the Client, select **Settings > AppMon Server**. Alternatively, in the **Cockpit** sidebar, right-click **System Profiles**, and select **Manage Server**.

Add or remove a Server

To add a new Server configuration, select **Add Dynatrace Server** in the lower left corner of the **AppMon Server Settings** dialog box.

To remove a Server connection, press the

button of the Server entry you want to delete.



Add and Delete Server Options

Server configuration

The **Dynatrace Server Settings** dialog box contains the following configuration panes:

- **Connectivity**: Client/Server connection settings, such as user name and password.
- **License**: View and import licenses.
- **Settings**: View Server information and configure Server memory settings.
- **Services**: General service configuration, such as Collector connection ports, REST or web service ports, and email server configuration.
- **Collectors**: Collector overview and management.

- **Performance Warehouse**: Performance Warehouse configuration.
- **Sensor Packs**: View and import, rename, and reorder and prioritize Sensor Packs.
- **Users**: User management for users, user groups, roles, and LDAP configuration.
- **Plugins**: Plugin management for import, configuration, and development instructions.
- **Updates**: Update management for installation, activation, deactivation, and removal instructions.
- **Storage**: Server storage directory configuration and session storage quota definition.
- **Infrastructure**: Infrastructure management for sites, hostgroups, and hosts.
- **Databases**: Manage and configure the database agents to monitor your database instances.
- **Geographical Locations**: Map IP address ranges to locations for **User Experience Management**.
- **MQ Tagging** — WebSphere MQ queue manager configuration.
- **Synthetic Monitoring Integration**: Configure and schedule synthetic tests for AppMon-instrumented web applications.
- **Business Transactions Feed**: Send Business Transaction data in real time to external systems for further storage and analysis.

Connectivity

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

AppMon Client to Server connectivity default values

Related pages

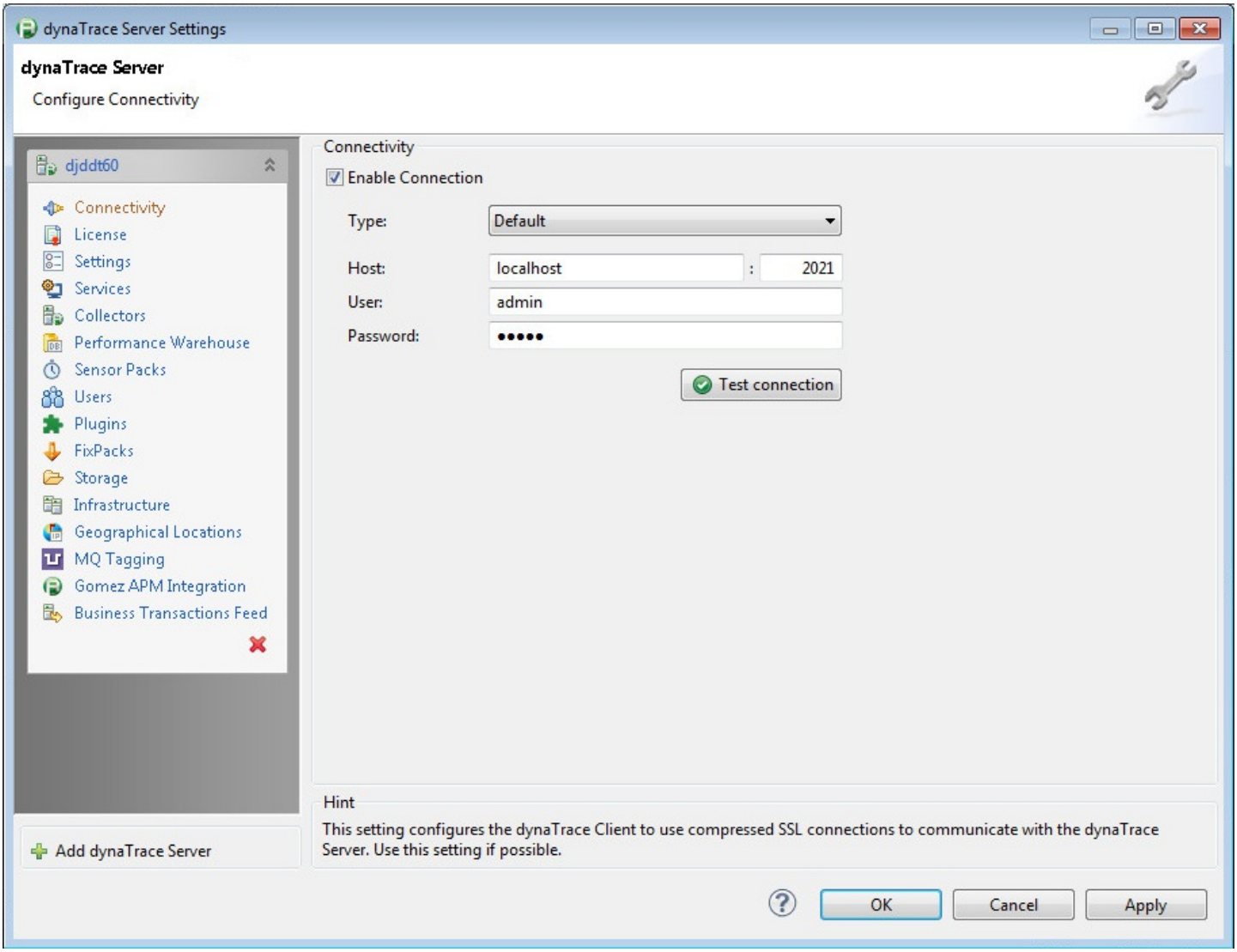
Server administration

Set up communication connections

Choose **Settings > Dynatrace Server > Connectivity** to view and manage communication settings between Client and Server, such as the communication type, AppMon host name, port and login credentials. The default communication type is uncompressed with SSL; HTTP tunneling and / or proxy.

AppMon Client to Server connectivity default values

AppMon assumes a local AppMon Server on port 2021 (localhost:2021) by default. The initial and installation settings for user name and password are **admin** / **admin**. Change these for your connection.



Server Settings Connectivity pane

Learn more about connectivity between Client and Server in Setup Communication > Client Setup.

License

See Licensing for more information.

Sizing settings

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Find the right sizing for your installation

Available deployment sizes

Why is the available physical memory less than expected?

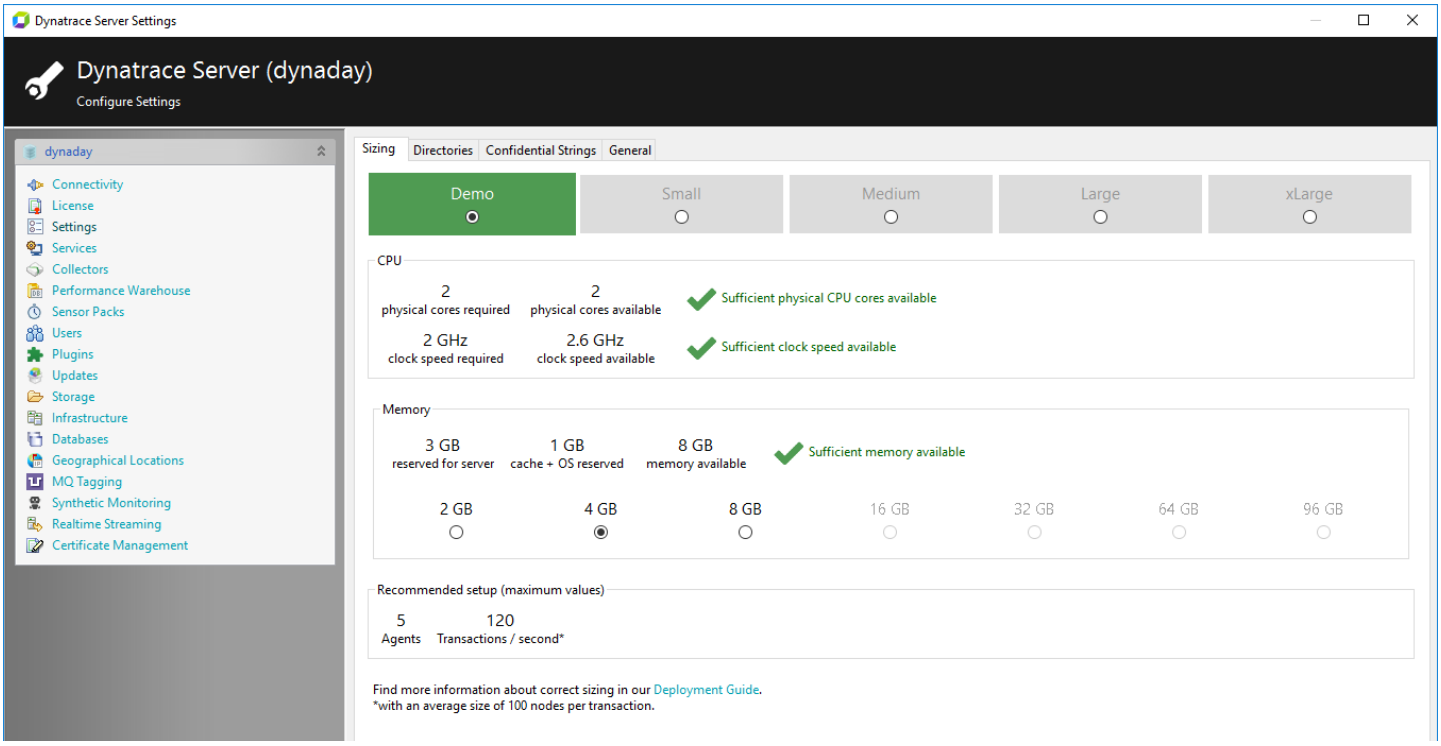
Warnings and errors

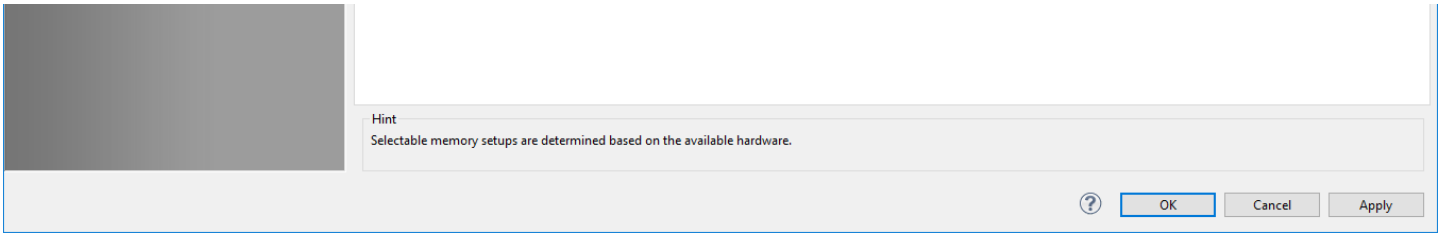
Related pages

Deployment Guide

Handling confidential strings

Collect sizing data





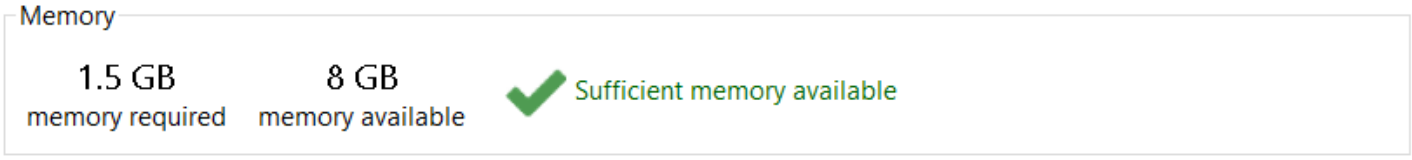
The **Sizing** tab of the **Settings** item from the **Dynatrace Server Settings** dialog box lets you view and manage the memory used by the AppMon Server. See the Deployment Guide for more information about accurately sizing your environment.

You can also define directories for logs, reports, and temporary files at the **Directory** tab, specify which info should be confidential at the **Confidential Strings** tab, and turn on/off self monitoring at the **General** tab.

Find the right sizing for your installation

Depending on the available hardware, up to five different deployment sizes are available. Sizes which require more physical cores than available cannot be set, but you can still select them to get the information about hardware requirements and possible setup for the available memory.

Memory: physical vs. overall used and paging



Compared is the total available physical memory with the memory reserved for the AppMon Server:

- **Available physical memory**: The maximal amount of physical memory which can theoretically be accessed by the AppMon Server.
  To ensure optimal performance, avoid using a deployment size where the overall used memory exceeds the available physical memory. Overall used memory is memory reserved for the AppMon Server plus the memory used by other processes. Although virtual memory would allow you to use more memory, paging can have a negative impact on performance and reliability, especially in high-load scenarios.
- **Reserved for AppMon Server**. The total amount of memory which is constantly reserved for the AppMon Server and the AppMon Frontend Server.

Recommended setup



Recommended setup presents key metrics which help you to find the right deployment size for your setup. The values are standard maximum values:

- **Agents**: Number of concurrently connected AppMon Agent instances.
- **Transactions / second***: Number of traced transactions respective to the number of PurePaths per second. The displayed value is based on PurePaths with an average size of 100 nodes.

After you have found the right sizing for your installation click **OK** or **Apply** to apply the current selection.

> **Note**
>
> Applying a new deployment requires an AppMon Server restart. If you decide against an immediate restart please do not forget to restart the AppMon Server before you connect a higher number collectors and / or agents.

Available deployment sizes

The AppMon Server (backend process / Windows service / *NIX daemon) is only supported on operating systems that support Java 8. Only the included Java Runtime Edition is supported.

- **Windows x64**: 10/8.1/8/7/Vista or Server 2008/2012/2016 with latest service pack; installer (.msi) does not need a pre-installed JRE; package contains a private Java Runtime Edition v8.
- **Linux x64**: With a recent kernel version 2.6 or higher.
  Installation requires a pre-installed Java Runtime Edition v8.
- **LDAP**: For AppMon user authentication.
  - Active Directory
  - openLDAP

Hardware requirements

See the Deployment Guide for detailed hardware recommendations specific to your load.

Deployment size[1]	Physical cores required[2]	Recommended physical memory[3]	Also available memory options	Recommended disk space	A
Demo 5 agents | 150 transactions/sec	2	2GB	4GB, 8GB	1.5GB + 100GB Continuous session storage/week	N
Small 10 agents | 300 transactions/sec	4	4GB	8GB, 16GB	3GB + 200GB Continuous session storage/week	Y
Medium 250 agents | 1,500 transactions/sec	8	8GB	16GB, 32GB	5GB + 1TB Continuous session storage/week	Y
Large 750 agents | 3,500 transactions/sec	16	16GB	32GB, 64GB	10GB + 2TB Continuous session storage/week	Y
XLarge 32GB | 1600 agents | 5,000 transactions/sec	24	32GB	-	20GB + >2TB Continuous session storage/week	N
64GB | 1800 agents | 5,800 transactions/sec	24	64GB	-	20GB + >2TB Continuous session storage/week	N
96GB | 2000 agents | 6,500 transactions/sec	24	96GB	-	20GB + >2TB Continuous session storage/week	N

[1] The maximum agent numbers consider a typical combination of different agents types (Java, .NET, host, WebServer, PHP, etc.).
In special circumstances, significant more agents are supported (e. g. mainly host agents). Please contact support for guidance and confirmation.

[2] No virtual cores (Hyper-Threading), for Medium and above a clock speed of 2.6 GHz or higher is required.

[3] Required memory for the AppMon Server and operating system. If there are other processes running, then additional heap is required.

Note

To improve performance, AppMon always uses the maximum available CPU and memory resources.
If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

VM support

All deployment sizes listed above may be deployed on a virtual machine. However, you must consider the following:

- No over-commitment of resources (CPU, memory, network, disk) and no virtual-only resources that are not backed by dedicated physical resources.
- Large and XLarge only. VM is equivalent to a physical box in all performance metrics. For more details see Guidelines for running AppMon in VMWare.
- XLarge only. Database partitioning is mandatory.

Why is the available physical memory less than expected?

32-bit Windows®: Due to 32-bit Windows' memory model characteristics, maximum usable physical memory is 2 GB.

Shared graphics memory: The graphics chip on your host may not have its own dedicated memory, and instead share main system RAM with the CPU and other components. Default values like 256 MB are not unusual. The default size can usually be changed in the BIOS settings.

Warnings and errors



For the currently selected setup is no applicable memory sizing available

This message displays if there is insufficient physical memory available for the selected sizing. If planning to upgrade hardware, remember that the operating system and other processes also need additional memory.



There are only *X* MB left for the operating system and other processes

To ensure optimal performance it is important to ensure that the AppMon Server is only using **physical** memory. This warning displays if there's insufficient physical memory available for the operating system and other processes for the selected sizing is possible. Although virtual memory would allow using more memory, paging can have a negative impact on performance and reliability, especially in high-load scenarios. Please select a different sizing or upgrade your hardware.



This setup needs at least *X* physical CPU cores. The current system has only *Y* cores

This warning displays if the total number of CPUs (including virtual cores through Hyperthreading) is smaller than the minimal number of required physical cores for the selected sizing. You cannot set this setting, but you can still select them to get the information about hardware requirements and possible setup for the available memory.



This setup needs at least a CPU clock speed of *X* MHz. The system runs currently only with *Y* MHz

This warning displays if the CPU clock speed is less than the minimal required CPU clock speed of the selected sizing. If you select a sizing, although you get this warning the AppMon Server can't handle the maximum number of transactions per second as displayed in the "Recommended setup" section. Select a sizing with this warning only if you do not expect to reach the displayed transactions per second. Otherwise, select a different sizing or upgrade your hardware.

Handling confidential strings

The analyzer receives `**********` as the input and uses it. The original argument is not available. All users see `**********`, regardless of the permissions assigned.

- Confidential argument: `test argument`

The analyzer receives `SELECT **********` instead of SQL string 1 and 2 as input. The two strings are the same, so it aggregates them to one row with a call count of two as a result.
All users see one `SELECT **********` string with aggregated call count, regardless of the permissions assigned.

- SQL string 1: `SELECT * FROM abc`
- SQL string 2: `SELECT * FROM bcd`

In the following example, with:

The original strings are not available to the analyzer, as the confidential data is removed prior to calling the analyzer. If you analyze a stored session without confidential data, the result may be different because some results, which were different prior to anonymization, are the same now and therefore aggregated.

In a session file where confidential strings are removed, all users see asterisks (*) instead of confidential strings. In SQL strings, the first word is not replaced by the asterisk.

To remove confidential strings from stored session, select the **Remove confidential strings from Session** check box in the **Export Session** dialog box.

You can safely use any session file that has all confidential strings removed. Confidential strings are not part of such session file and you cannot restore them.

Remove the confidential strings

The analyzer has the original data and uses it. The user with insufficient permissions sees result as `**********`.
The user with the *ReadConfidentialStrings* permission sees the original data.

- Confidential argument: `test argument`

The AppMon Server database analyzer analyzes the original data. The result contains two rows with a call count of one for each.
The user with insufficient permissions sees two `SELECT **********` strings with separate call counts.
The user with the *ReadConfidentialStrings* permission sees the original data.

- Confidential SQL string 1: `SELECT * FROM abc`
- Confidential SQL string 2: `SELECT * FROM bcd`

In the following example, with:

In this case the confidential strings are captured and used in session analysis. However, only users with the *ReadConfidentialStrings* permission can view it. Users without the *ReadConfidentialStrings* permission see asterisks (*) instead of confidential strings. In SQL strings, the first word is not replaced by the asterisk.

Keep the confidential strings

You can use two approaches to confidential strings:

To define confidential string types when you export a client session, select **Settings > Dynatrace Server > Settings**, and click the **Confidential Strings** tab. In the **Active** column, check strings which should be kept confidential.

For example, the JDBC sensor captures argument values passed to methods or SQL statements.

- String arguments passed to methods.
- String return values.
- Exception messages.
- Message content transported over middleware, such as JMS.
- URI, query string, headers, parameters, request attributes, and Servlet call session attributes of.
- URI, query string, headers, parameters, cookies, session attributes, and ASP.NET call server variables.
- SQL and SQL database call bind parameters.
- Incident messages and descriptions triggered by the AppMon Server.

The list of defined confidential data includes:
AppMon handles confidential string data captured by AppMon Agents or Sensors to restrict view of the sensitive data. To read confidential data, your user account must have the *ReadConfidentialStrings*

permission. See Permissions Mapping and User Permissions and Authentication > Roles for more information about permissions.

Services

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

General tab
Management tab
Webstart tab
Dynatrace Web tab
SSL certificates tab
Email tab
Dynatrace Memory Analysis Server tab
Symbolication service tab

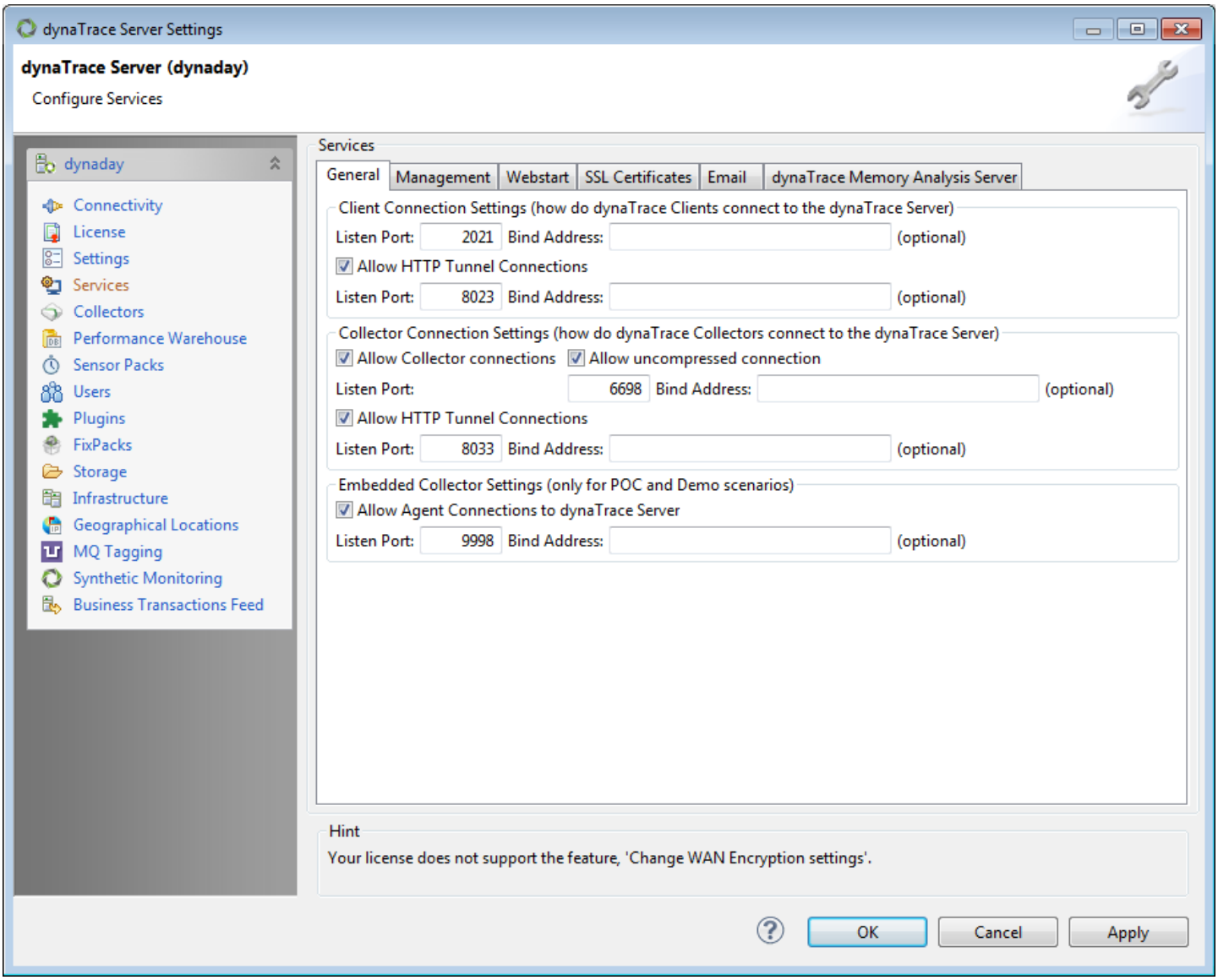
Related pages

Set up communication connections
Set up the Webstart Client
Set up SSL communication
Set up a Memory Analysis Server
Email

The Services tab in **Settings > Dynatrace Server** has several horizontal tabs to configure various service communication settings.

General tab

Configure ports that the AppMon Server listens on for Clients and Collectors, optional bind addresses, if SSL or non-SSL or HTTP tunneling (see Configure Proxies) is allowed.



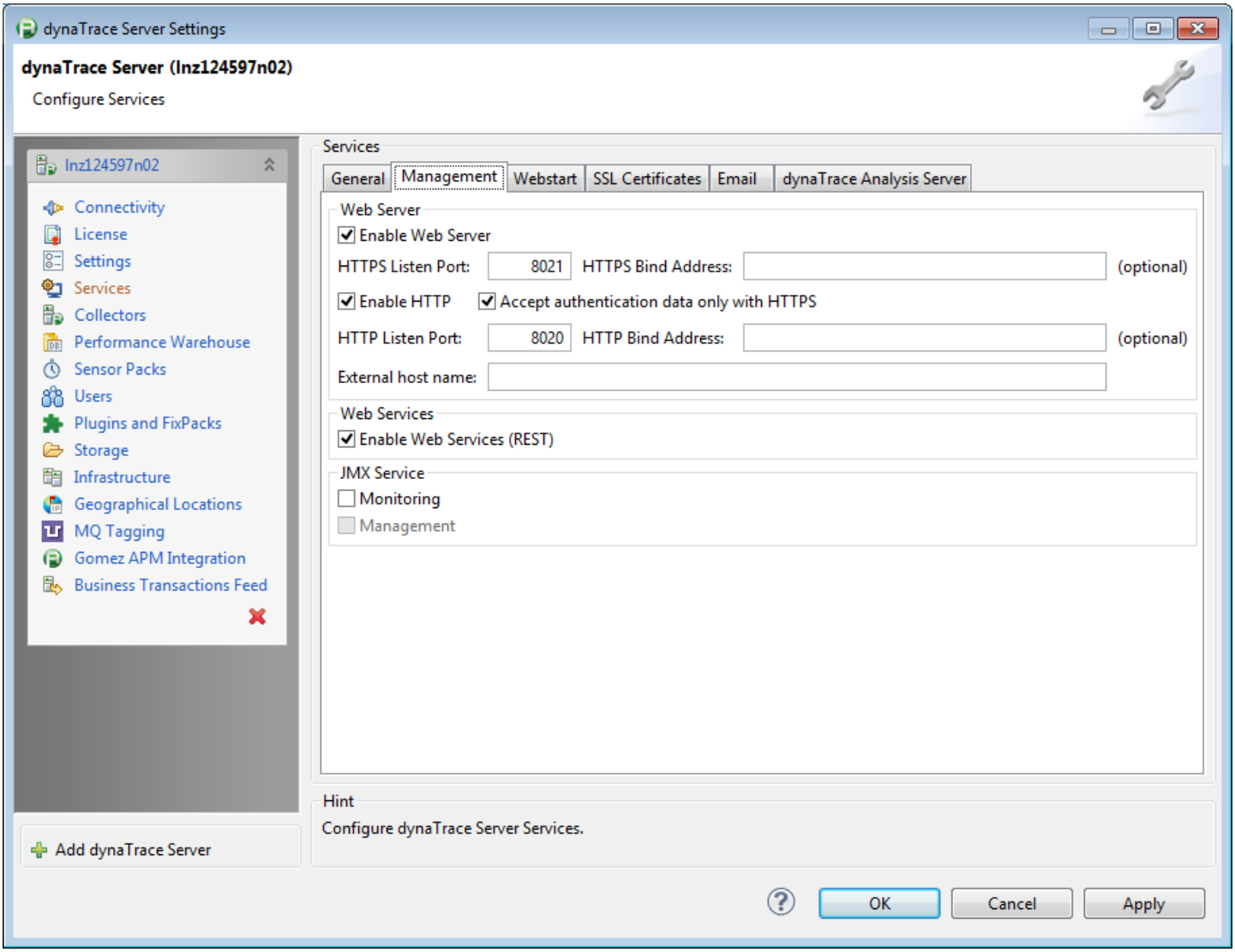
Note

The AppMon Server comes with a Server-embedded Collector used only for test purposes. Your license type enforces Server-embedded Collector use. Select the **Allow Agent Connections to dynaTrace Server** check box to use the Server-embedded Collector. This requires an AppMon Server restart.

Use an AppMon standalone Collector service / daemon for normal load.

See Set up Communication Connections for information on service configuration.

Management tab

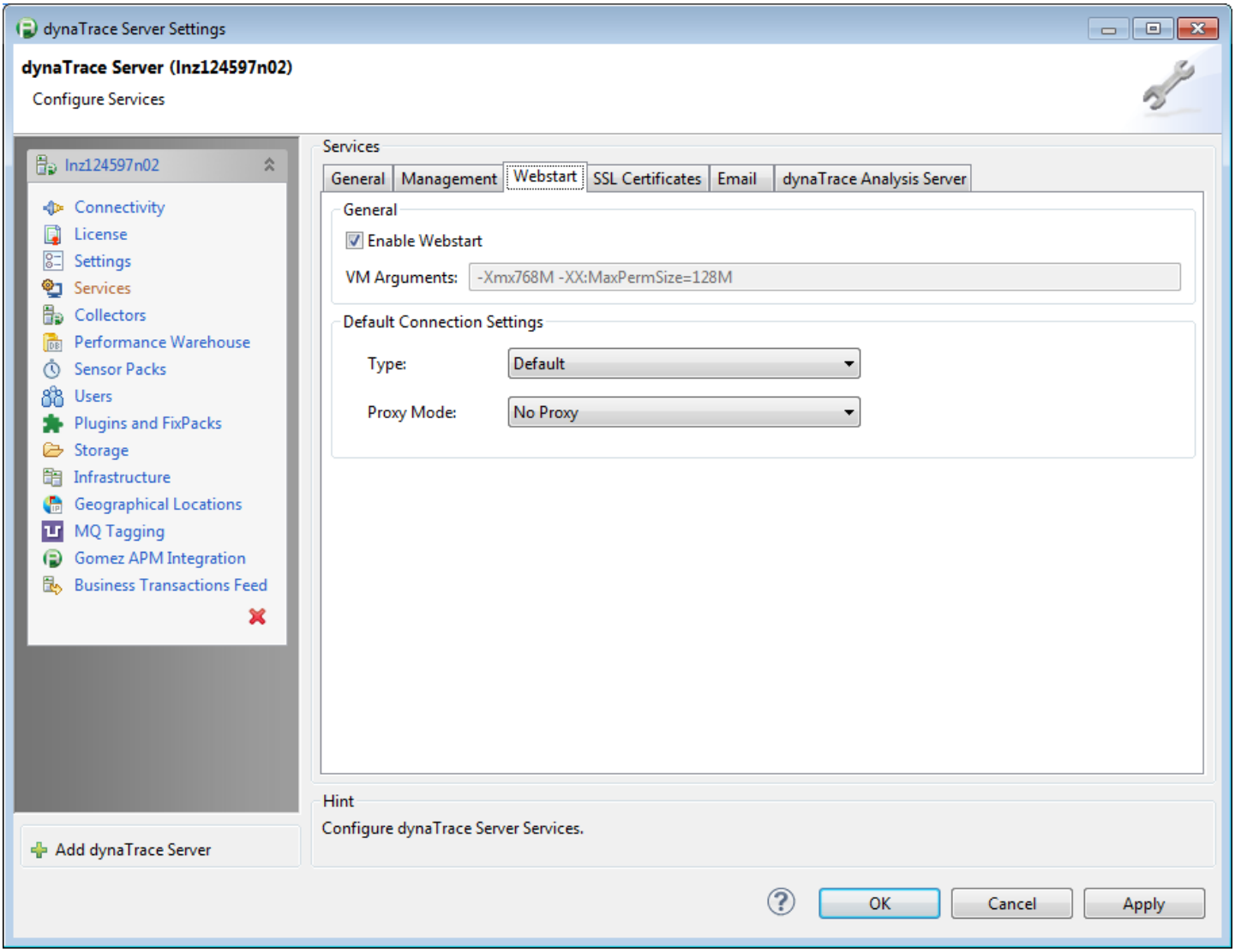


The **Management** tab contains the following configuration options for built-in web server and JMX services.

- **Enable Web Server**: Enable or disable the built-in web server.
- **HTTPS Listen Port**: Specifies the HTTPS port on which the web server listens.
- **HTTPS Bind Address**: Binds the HTTPS service to a specific address in case of multiple network interfaces.
- **Enable HTTP**: Allow or disallow HTTP services.
- **Accept authentication data only via HTTPS**: Doesn't accept authentication data if sent via HTTP. Forwards user to HTTPS pages or rejects REST requests with authentication data sent over HTTP.

- **HTTP Listen Port**: Specifies the HTTP port to which the web server listens.
- **HTTP Bind Address**: Binds the HTTP service to a specific address for multiple network interfaces.
- **External host name**: Configures the server hostname so it can be addressed from external networks.
- **Enable Web Services (REST)**: Enables or disables the server's REST web services.
- **Enable Server Side Reporting**: Enables or disables the server's REST reporting services. See Dashboards and reporting (REST) for more information.
- **JMX Monitoring**: Enables or disables JMX Monitoring.
- **JMX Management**: Enables or disables JMX Management.

Webstart tab



Use the **Webstart** tab to enable or disable the Webstart support for this AppMon Server by selecting the **Enable Webstart** check box. The tab displays arguments for the AppMon Webstart Client Java VM and you can switch the connection type to HTTP tunnel. You can also configure a proxy for the **Webstart** Client. Configure a proxy for the **standalone** Client in the **Settings** pane of the **Dynatrace Client Preferences** dialog box.

Dynatrace Web tab

This tab includes the following options to control AppMon Web:

- **Enable Dynatrace Web**: Enable or disable use of the **AppMon Web** and the ability to drill-down from the Web UI to the AppMon Client for detailed data analysis.
- **HTTPS listen port**: Specifies the secure connection port to which** AppMon Web ** listens. This is set to **9911** by default.
- **Allow non-ssl connections**: Select this check box to enable unsecured Web UI connections.
- **HTTP listen port**: If selecting the **Allow non-ssl connections** check box, specify the communication port for unsecured connections. This is set to **9910** by default.
- **Allow custom HTML**: If embedding custom content in a Web dashboard, select this check box to allow unsecured custom HTML code to create the custom content.

See Set up SSL Communication for more information about secured and unsecured connections.

SSL certificates tab

For AppMon 6.3 and later, the SSL Certificates horizontal tab is removed from Services and has its own Certificate Management tab in the Server Settings dialog box. See the section on customer-provided keystores in Certificate Management for details.

Email tab

See Email to learn how to configure an administrative email address and mail server connectivity.

Dynatrace Memory Analysis Server tab

See Set up a Memory Analysis Server to learn how to configure connection to and from the AppMon Memory Analysis Server.

Symbolication service tab

Use this tab to enable symbolication (for iOS) or de-obfuscation (Android) of mobile application crash reports or handled exceptions. Enabling this feature lets you view stack trace data in plain text. See Mobile Symbolication Service to learn how.

Email

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Configuring the system for sending email

Importing the Email Server Certificate

Related pages

Services

Overview

The AppMon Server can generate an email notification when the following occurs:

- You send a scheduled report to specified email recipients. See Automated Reporting for more information.
- A specified incident is triggered. See Incidents and Alerting for more information.

The AppMon Server uses SMTP protocol for sending email and relies on an external SMTP server for the forwarding of email to recipients. To send email using AppMon, you need the connection details to an SMTP server, which includes:

- SMTP server hostname
- SMTP server port (default is 25)
- Is SSL encryption required?
- Is authentication required?
  - SMTP username
  - SMTP password

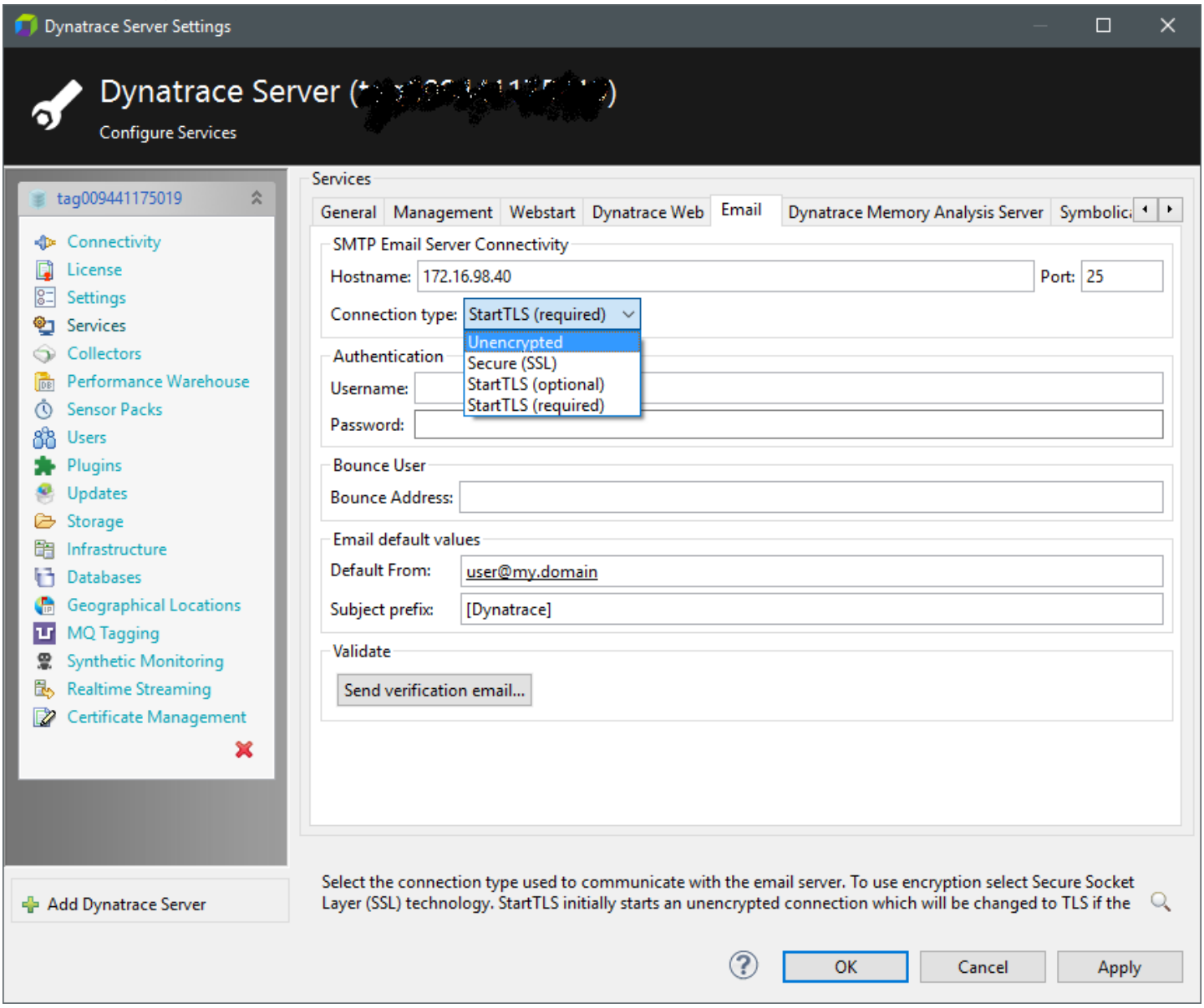
You can use users and groups defined in AppMon as recipients as long as they have an email address assigned. If you use LDAP for user and group information, then the email address and group assignments from the LDAP repository are synchronized to the AppMon user repository and are used automatically.

Configuring the system for sending email

For the system to send email, you or an administrator must set up the system email configuration in the **Email** tab of the **Server Settings Services** pane.

To access this tab, choose **Settings > Dynatrace Server > Services** and select the **Email** tab. You can also access the **Server Settings** dialog box from the **Administration** section of the **Start Center**.

Server Settings Services Pane - Email Tab



The following are the required values:

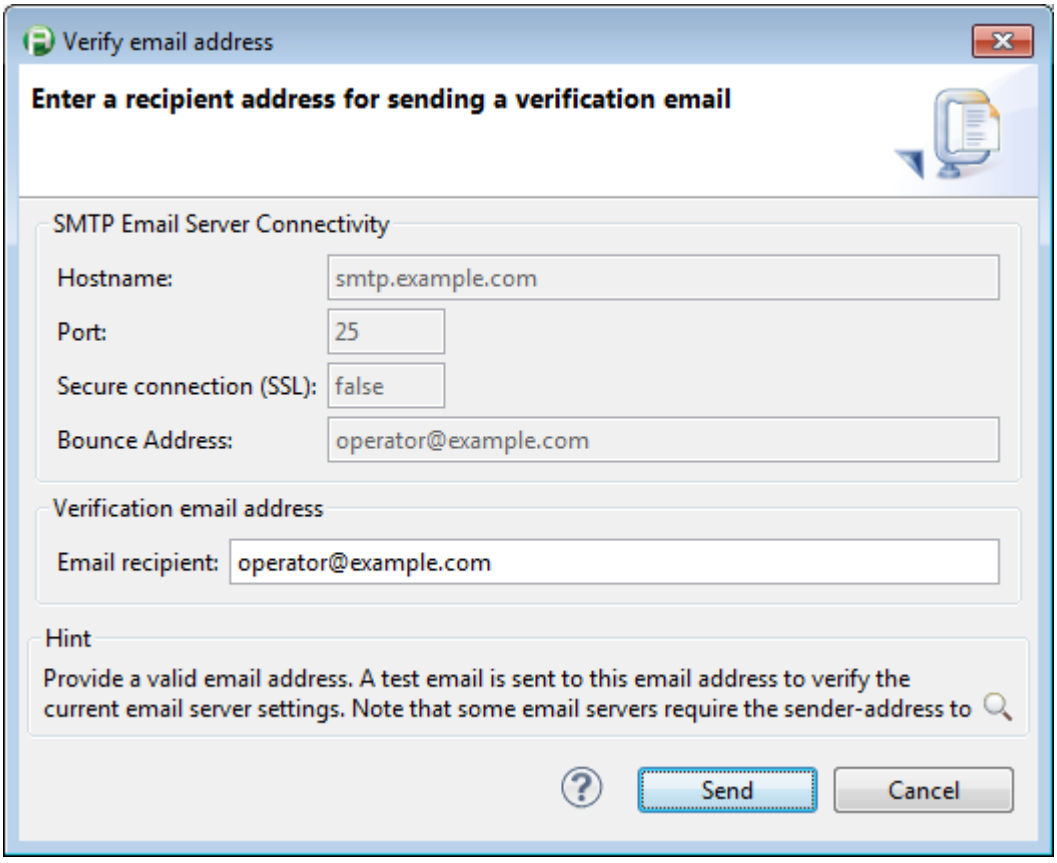
- **Hostname**: Specifies the machine which provides the SMTP server instance.
- **Port**: The port on which the SMTP server instance is listening, default value is 25 for unencrypted (plain) communication and 465 for using secure (encrypted) transport (SSL).
- **Connection type**: Choose one of the following connection types when contacting the email server:
  - Unencrypted: Plain communication, no protection.
  - Secure (SSL): Use SSL protocol to secure the communication with the email server (if supported by the server).
  - StartTLS(optional): Use StartTLS protocol to secure the communication. If the email server supports TLS communication then the mail will be sent encrypted. If the server does not support TLS then the mail will be sent in plain text.
  - StartTLS(required): Use StartTLS protocol to secure the communication. If the email server supports TLS communication then the mail will be sent encrypted. If the server does not support TLS then the mail will not be sent.

You can also specify the following values:

- **Username/Password**: Provides authentication details, only necessary if your SMTP server requires these.
- **Bounce Address**: Specifies an administrative email address that is informed whenever an email can not be delivered by the SMTP server, foe example if it is mistyped or the recipient email address is no longer valid.
- **Default From**: Specifies the email address that is used as sender of emails.
- **Subject Prefix**: Specifies some text that is prepended to the subject-field of all emails that are sent. This is useful to quickly distinguish emails and to easily filter emails in email client software.

In the **Email recipient** box, enter an email address to verify that your provided SMTP server settings work. Press **Send verification email** to test.

Verify Email Address Dialog Box



Verify Email Address dialog box

Importing the Email Server Certificate

If you want to use a secure connection to your email server by specifying either SSL or StartTLS as connection type you might get an error message when sending a mail which looks like this one:

> javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target

This message indicates that the email server certificate could not be found. In that case you have to get/retrieve the email server certificate first and then you have to import it into the Java Runtime certificate store.

This certificate store is named “cacerts” and it is located in the lib/security directory of your JRE installation. You can use the Java “keytool” command line utility to get the certificate imported:

```
keytool -import -alias <alias> -keystore <path-to-
jre>\lib\security\cacerts -file <path-to-certificate-file>
```

The “alias” parameter names the certificate. An example command could look like this:

```
keytool -import -alias MyEmailServer -keystore
%JAVA_HOME%\lib\security\cacerts -file emailserver.der
```

You are prompted for the keystore/certificate store password. If you did not change it, the password is set to **changeit** by default.

Note

If the email server certificate isn’t a self-signed one you have to retrieve the root (CA) certificate for the email server certificate. Using an SSL/TLS connection implies that the certificate chain for the email server certificate must be validated on the client side (the email sender). This chain only validates if its root (CA) certificate is trusted and contained in the cacerts certificate store. Thus one has to import the root (CA) certificate into the store instead of the email server certificate itself.

Collectors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Collector groups for connection balancing and fail over

Collector load balancing

Clearing Collector caches

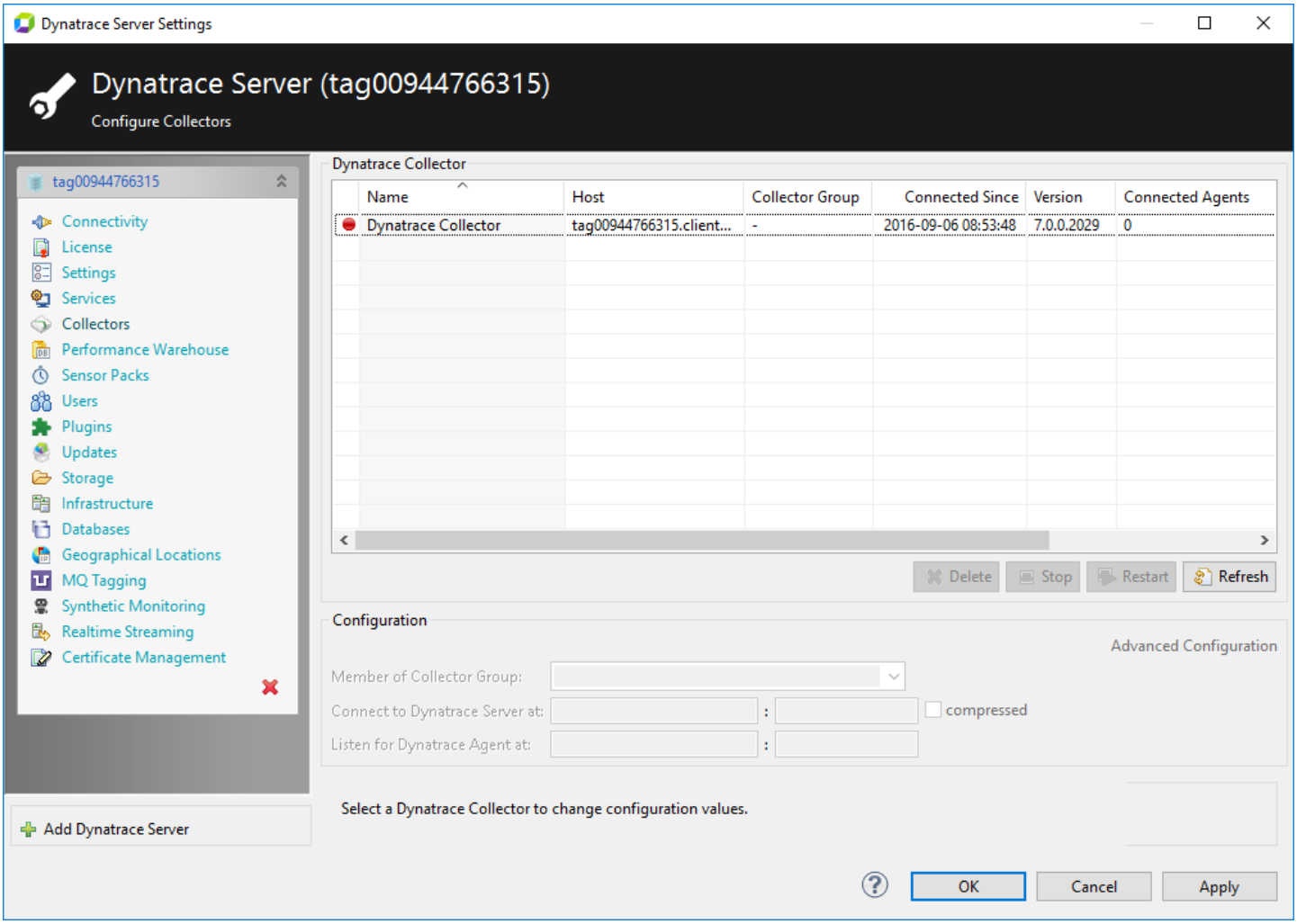
Related pages

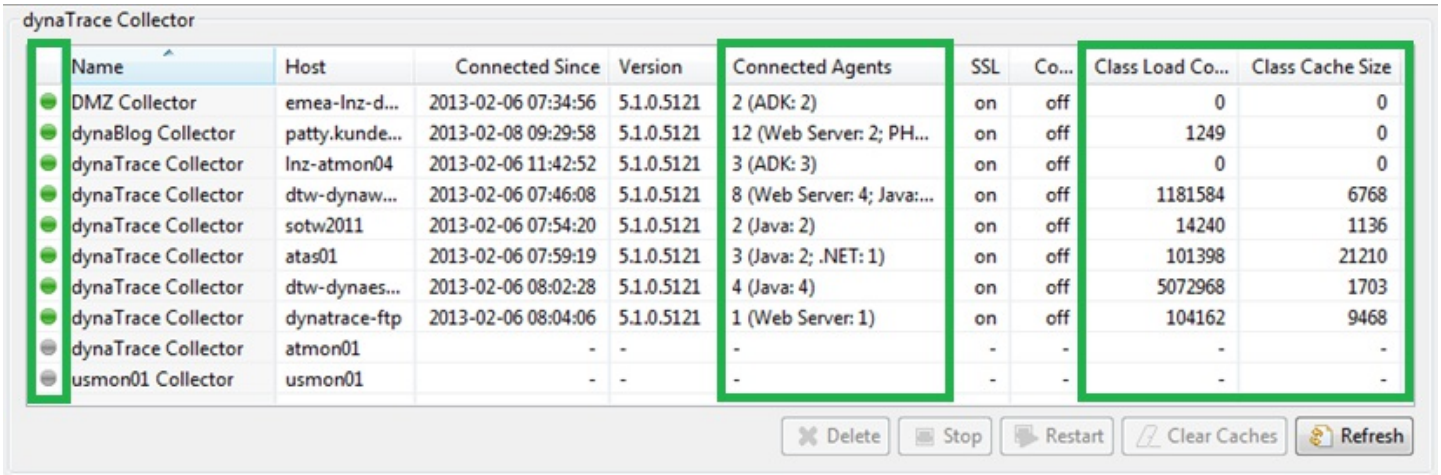
Server administration

Set up the AppMon Server

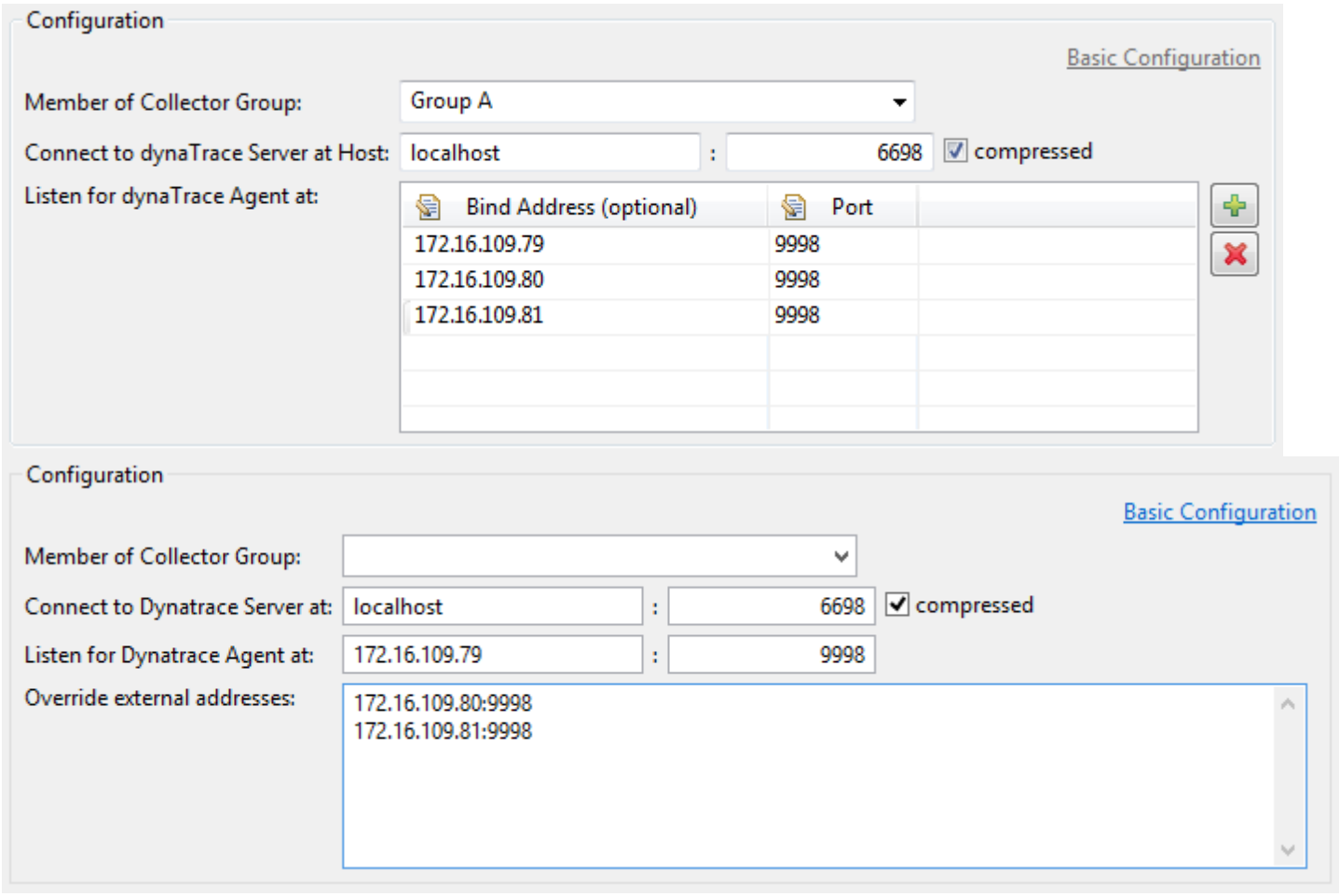
Set up collectors

In the AppMon Client, choose **Settings > Dynatrace Server > Collectors** to view information about Collectors known to the AppMon Server and to configure options to manage connected Collectors.





Configure the port and host for the AppMon Server connection, and the port for the Agent connection to an AppMon Collector, which then requires a restart. The bind address for the Agent listen port is not configurable if set by system properties in the `dtcollecter.ini` file, or as a command-line parameter on AppMon Collector startup. Use Advanced Configuration to configure multiple bind addresses and/or listening ports for Agent connections to the Collector.



Status:

- : Connected - Collector is connected to server
- : Not Connected - Connection from Server to Collector currently lost
- : Offline - Collector shut down properly
- : Restart Pending - Collector currently restarting
- : Agent Listening Error - A error with the connected Agents on the Collector

Collector groups for connection balancing and fail over

Use Collector groups (**Settings > Dynatrace Server > Collectors**) to allow Agent connection balancing / fail-over (but no automatic fail-back) across Collectors in the same group.

Agents configured to connect to one of the Collectors in a group are completely randomly balanced (roughly equally distributed by chance) among available Collectors at Agent start time.

The Agent connects to the Collector that it is pointed to by the connection string. It then gets a list of available Collectors in the group and reconnects randomly. With large, diverse networks, plan which Collector groups and respective Agents should be “close” to the Collectors.

Despite automatic balancing, you should not point all the Agents to one Collector for initial connection, but distribute as you would manually.

Obey the Agents-per-Collector-limits that apply for different technologies, calculate the sum for the group and have sufficient reserve for fail-over.

If a Collector goes down, connected Agents try to reconnect for 20 seconds. if they can’t reconnect during that time span, the Agents fail over to a different Collector in that group.

Agents do not fail back if they are connected when their original Collector becomes available again, but they will fail back / redistribute when the applications or Agents restart. If a Collector does not come up again within half an hour, it no longer displays in the Server Settings > Collectors list.

To set up a Collector group, choose a suitable group name. Select each Collector you want to include in this group and enter the group name. Click **Apply**.

The Collector name stored in the Agent’s `CollectorList` file is the Collector’s detected. If an IP address (multiple NICs in the machine) or fully qualified domain name is required for the Agent to connect to the Collector, you can choose to configure a Collector Bind Address. This bind address is sent to the Agent and populated in its `CollectorList` file. To configure this, switch from the Basic to the Advanced Configuration in the dialog box.

Collectors without a group name are “standalone” Collectors. No Agent connection balancing is performed across them.

Collector load balancing

For AppMon 6.3 and later, load balancing makes sure that no collector is overloaded.

If the agent balancing task realizes that a collector is overloaded, the balancing task tries to move agents to other collectors within the same collector group. By default, collectors with a load of more than 50% of the group average are considered overloaded.

Clearing Collector caches

Automatic Collector cache clearance

The Collector performs daily cleanup to keep the class cache from growing too large. The cache purges Classes not used for a week.

Manual Collector cache clearance

When should I manually clear the class caches? Clear a cache manually only when your class cache is corrupted and you must clear it immediately. To do so, select the Collector, and click the **Clear Caches** button below the Collector. Contact support if this occurs.

Performance Warehouse

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Connection details

Changing the AppMon Performance Warehouse database

Manage System Profiles

Manage storage

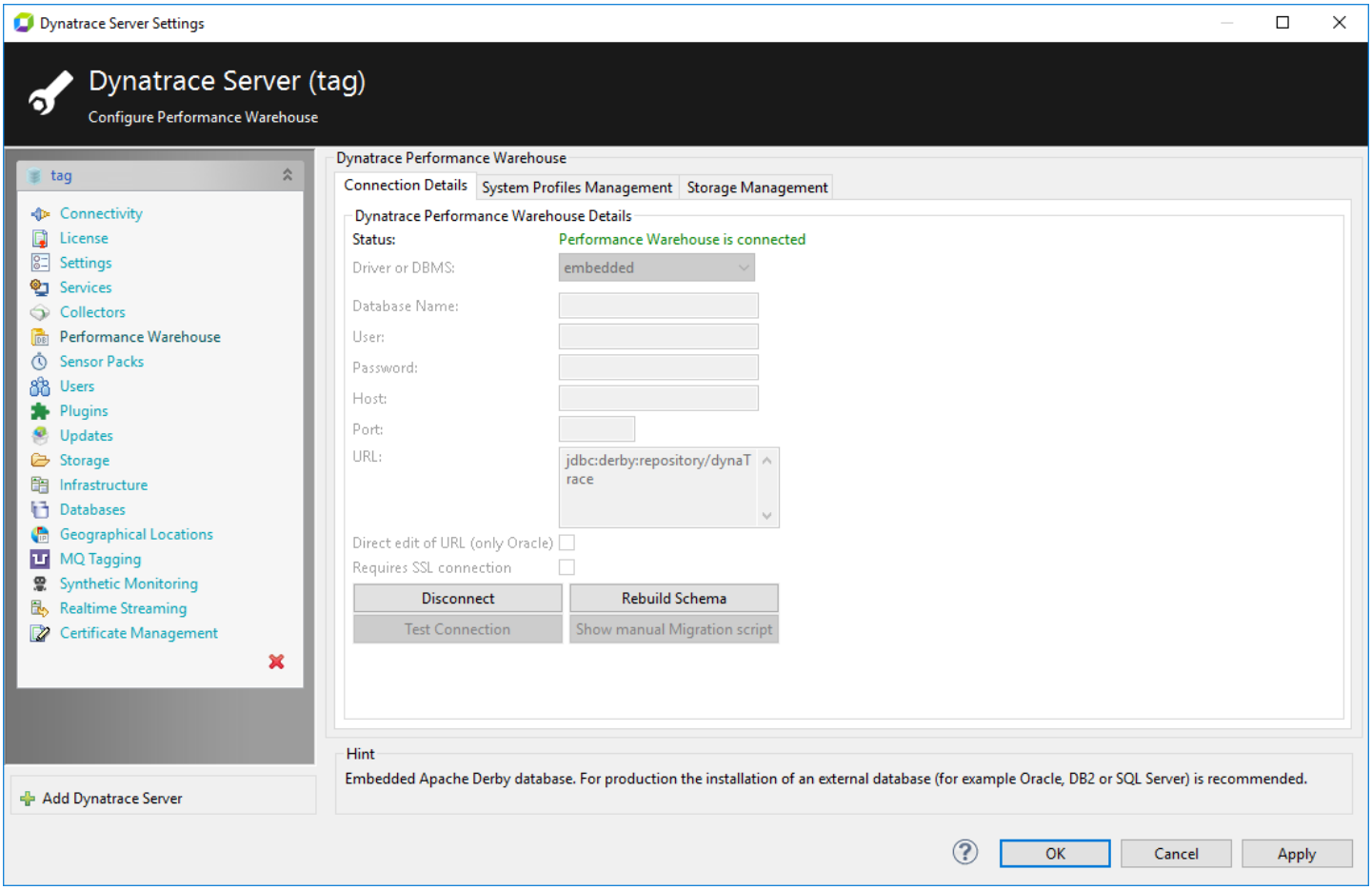
Performance Warehouse database versus session store files

Related pages

Performance Warehouse administration

Embedded Performance Warehouse Usage Restrictions

Use the **Server Settings Performance Warehouse** pane to set up connection details to use an external database. Use the external database to store Performance Warehouse data, manage System Profiles, and set the maximum duration for how data is kept. See Performance Warehouse Administration for more information.



Server Settings Performance Warehouse pane

Connection details

Cyclic measurement data persists in the embedded Derby database or in a SQL DBMS (DB2, Oracle, PostgreSQL and Microsoft SQL Server / SQL Azure). The embedded database is for testing only and its usage is limited to certain licenses. See Embedded Performance Warehouse Usage Restrictions for more information. See System Requirements for SQL DBMS minimum supported versions.

Changing the AppMon Performance Warehouse database

You can only change the database when a database is not connected. Follow these steps to change the Performance Warehouse database.

1. Choose **Settings > Dynatrace Server, Performance Warehouse > Connection Details** and click **Disconnect** to disconnect the current database.

2. Provide a database type, host, port, user, and password to specify the new database. The Java Cryptographic Extension encrypts the password.
3. Click **Test Connection** to verify that the Performance Warehouse database is available.
4. Click **Connect** to connect to the database and create the tables.

The AppMon Performance Warehouse is now set up to the configured database. Click the **Rebuild Schema** to rebuild the whole database schema.

> **Important**
>
> When you rebuild the schema, all Performance Warehouse data is lost.

See Performance Warehouse Administration for more database connection information.

Manage System Profiles

The System Profiles Management tab of the Performance Warehouse pane shows the System Profiles available in the Performance Warehouse. To remove unnecessary measurements or incidents from a System Profile's Performance Warehouse storage, select a System Profile and click **Delete Measurements/Incidents**. Click **Remove** to delete a System Profile and all its data from the Performance Warehouse.

Manage storage

Use the **Storage Management** tab to set the length of time data are kept in high, medium, and low resolutions and when they are consolidated into the next lower level. The data point interval is ten seconds for data in memory and one minute, hour, or day for high, medium, or low resolutions in the database respectively. See Resolution for more information.

Performance Warehouse database versus session store files

The **Performance Warehouse** database continuously stores data from both system monitoring such as CPU and memory usage, and PurePath timings. Data is stored at 10-seconds granularity at the highest resolution. Granularity is diminished at the medium (hour) and low (day) resolutions.

To explicitly store parts of such live sessions, click **Cockpit > Session Storage > Start / Stop Recording or > Store Session** in the AppMon Client and right-click a System Profile. Click **Cockpit > Session Storage** to see them. You access them in the AppMon Server. The **Export Session** option zips up the stored session files (`.dts`). The **Import Session** option from the file system makes it available in **Cockpit > Offline > Local Sessions**.

Stored session files can contain a memory dump, thread dump, sampling, or full PurePath information. Stored PurePath Sessions include time series data also.

Sensor packs

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

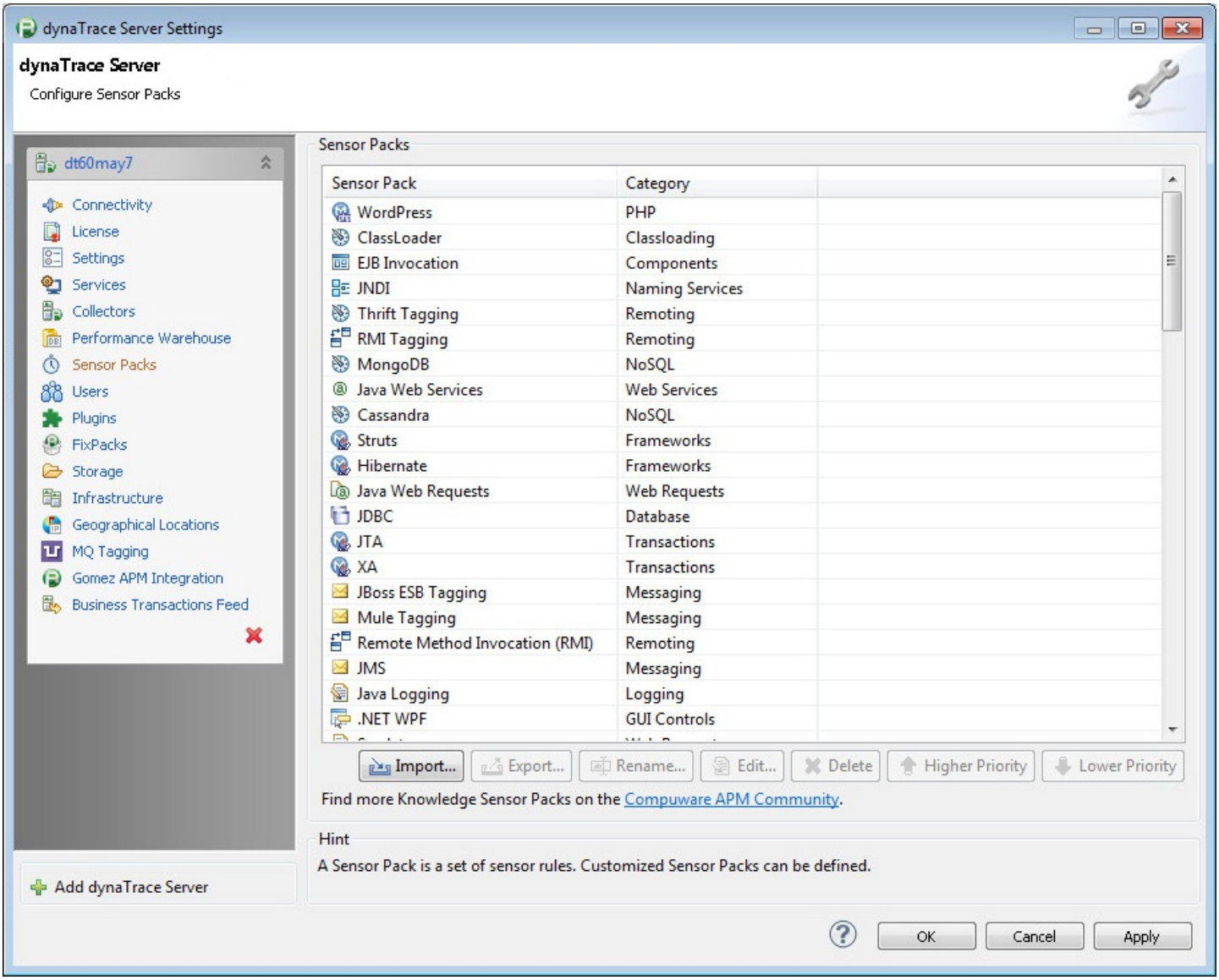
Sensors for JavaScript frameworks

The **Server Settings Sensor** pane displays the available Sensor Packs on the corresponding AppMon Server.

There are two types of Sensor Packs:

- **Sensor Packs**
  - Provided by AppMon
  - Viewable but not editable
  - Not deletable
  - Not exportable
- **Knowledge Sensor Packs**
  - Built by end users
  - Editable (**Edit** button)
  - Deletable (**Delete** button)
  - Exportable (**Export** button)

You can rename both types of Sensor Packs. Set priority in the list by moving an entry up for higher priority, or down for lower priority. Press **Restore Order** to restore the default order. In addition, you can import Sensor Packs and Knowledge Sensor Packs from this pane.



Server Settings Sensor Pack pane

Sensors for JavaScript frameworks

Looking for the 6.5 version of this page? Open in 6.5 documentation

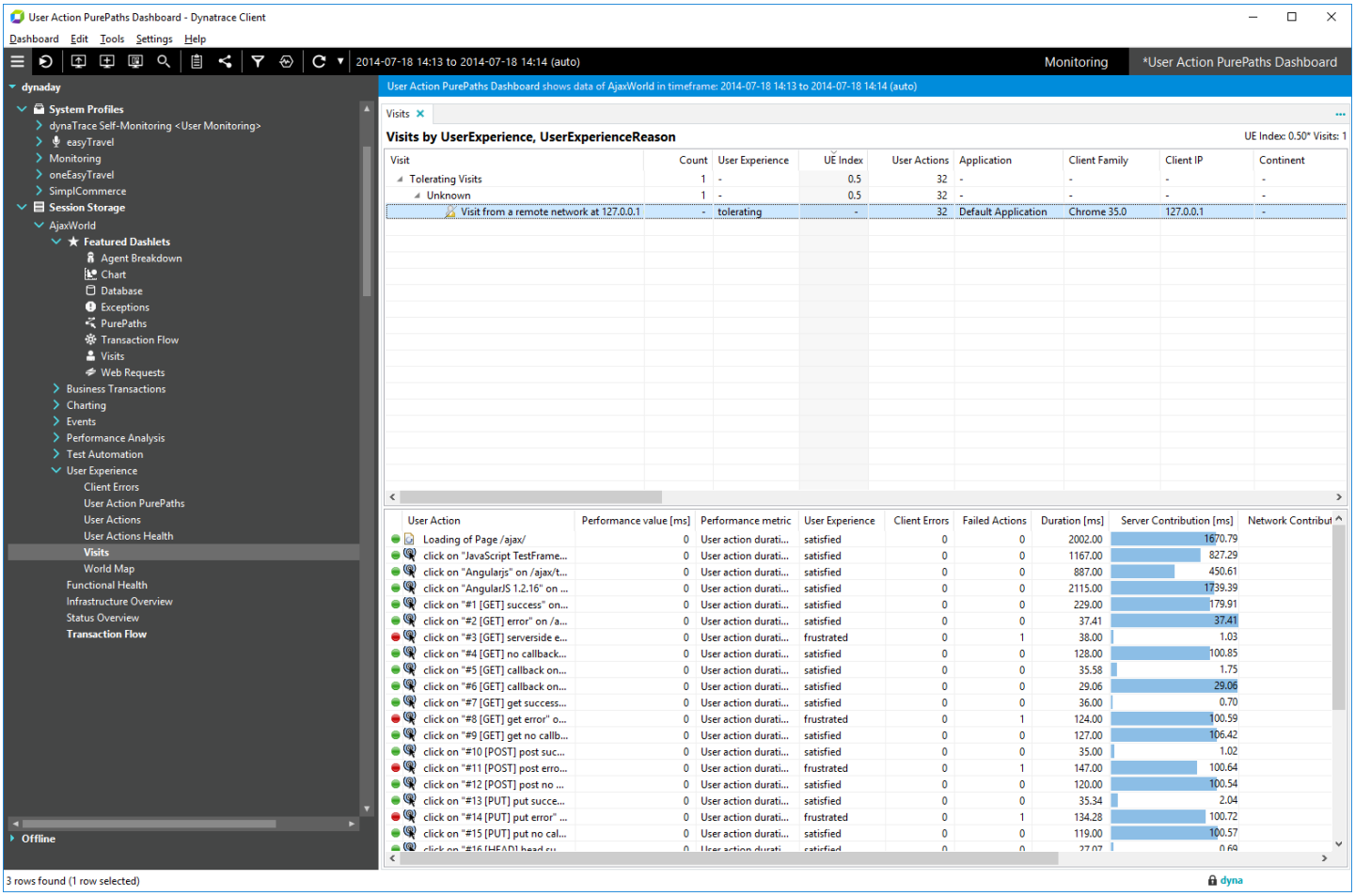
Related pages

Sensor packs

UEM is compatible with all JavaScript frameworks, however automatic click detection requires special sensor packs for several frameworks. Additional information on supported frameworks can be found in the Release notes.

AngularJs

The following screenshot shows a UEM visit capturing user actions. This particular example is using AngularJs XHR support. You can download this sample visit here: Sample_AngularJs_Support_Visit.dts



Backbone.js

Backbone.js usually uses jQuery to send web requests. In most cases it is sufficient to enable the jQuery sensor if you are using Backbone.js. If you are using the default `Backbone.sync` and `Backbone.ajax` calls, you can use jQuery sensor.

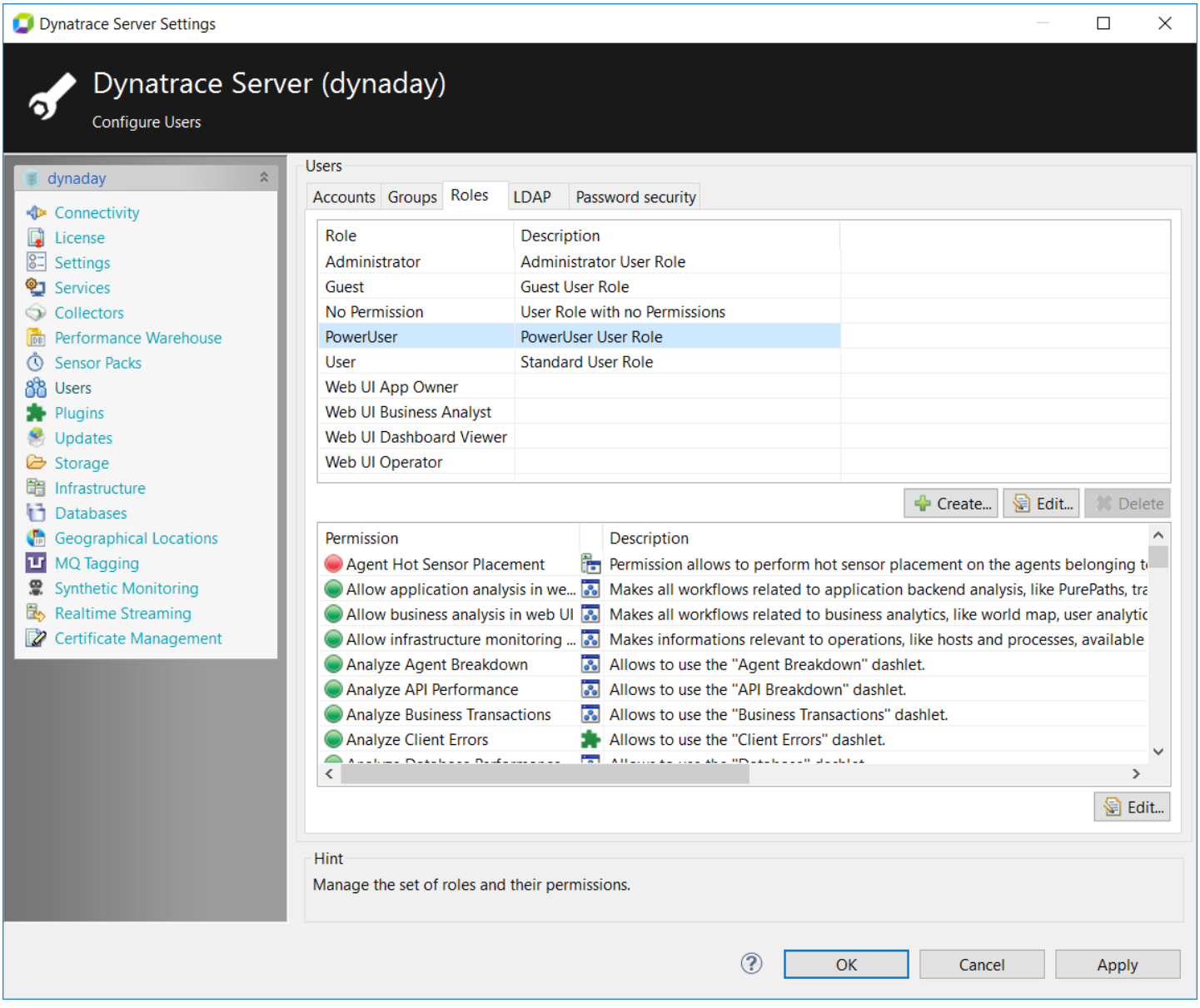
- jQuery.Ajax is the default way for Backbone RESTful Ajax request. But the users are able to override it by using native XMLHttpRequest or other libraries. Then the basic XHR sensor or other framework sensors may needed.
- Backbone.js also provides the ability to override the default RESTful Ajax request through a custom `Backbone.sync` method. The custom method could use local storage, web sockets or XML Transport. From most online examples:
  - XML Transport method eventually goes with jQuery ajax as well.
  - Local storage method is all about locally persistence, not to the server side.
  - Web sockets method is using web socket rather than ajax. In this case, examples use socket.io and Node.js a lot. This is currently not supported.

Users

Learn more about User Permissions and Authentication.

Important

For security purposes, change the password of the administrative user account immediately after installation.



Manage Use Accounts in the AppMon Server Settings Users Pane

A default administrative user account with full access rights is available after you install AppMon Server. The user name and password of this user account is **admin**. You cannot delete this user account or create another user account with the user name *admin*.

Plugins

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installed plugins tab

Plugin development tab

Related pages

Updates

Language packs

Manage and develop user plugins

Plugins are supplemental product libraries that extend AppMon with features and functions beyond the functionality in the installed product. To access the **Plugins** pane of the **Server Settings** dialog box, choose **Settings > Dynatrace Server > Plugins** or **Tools > Manage Plugins** if your Client is connected to at least one AppMon Server. See Plugin Architecture for more information about plugins.

What's the difference between a plugin, FastPack, update, and product upgrade?

FastPacks are downloadable packages that can contain resources such as preconfigured plugins, sensor packs, dashboards, and System Profiles. A FastPack bundles these different resources into one installation file that you can install and import into AppMon like a plugin. See Building FastPacks for more information.

Updates are software updates that may contain defect fixes, performance improvements, or usability and feature enhancements. To install an update, you must upload the update to a running AppMon Server. Manage the update installation with the **Update Wizard**, which you can access from the Client UI. See Updates for more information.

A **product upgrade** requires a completely different installation process from an update, plugin, or FastPack. You must install a product upgrade to a separate directory structure. To persist your data, you must migrate settings and data from the old to the new version. See the Upgrade and Migration Guide to learn how.

Installed plugins tab

On this tab, you can view all of the installed plugins. There are several different types of plugins that are editable, as shown in the following table:

Plugin Type	activatable/deactivate	Editable in Plugin Development Environment
Action	yes	yes
Library	no	no
Metric Group	no	yes
Monitor	yes	yes
System	no	no
Task	yes	yes
Webstart	no	no

To install the default plugins, such as System plugins or downloaded Monitors, click the **Install Plugin** button. For plugins available in a separate folder instead of an installation file, click the **Import Folder** button, which is available for actions such as developing a new Task or Monitor.

To install a system plugin:

1. Download a plugin installation file from the Dynatrace Community.

2. Install the file using either of the following methods:
   - In the **Plugins** pane of the **Server Settings** dialog box.
   - Double-click the installation file (`.dtp` files on Windows only) to start the installation.
3. Follow the instructions. You must restart Server or Collector. You can see pending restarts in the 
   Start Center.

You can install and upgrade only if the plugin is a **newer** version.

Plugin development tab

On this tab, you can manage custom plugins that you develop, test, and deploy on your own AppMon system. See Manage and Develop User Plugins for more information.

Language packs

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installing a language pack

Selecting a language

Related pages

Plugins

Downloads

Dynatrace Product Downloads

A *Language Pack* is an AppMon system *Plugin* that provides a localized user interface for a given language. While installing a language pack does not immediately change the user interface of your AppMon installation, it does enable your AppMon users to select the installed language.

Like other plugins, a language pack is downloaded and installed on your AppMon Server. You can install as many languages packs as you like and switch between them in the AppMon Client by selecting **Settings > Dynatrace Server > Plugins**. Unlike other plugins, which are available for download from the FastPacks and Plugins page, language packs are available from the AppMon Product Downloads page.

Installing a language pack

Required permissions

To install a language pack, your user account must have the **Manage Plugin Bundles** permission enabled. See Permissions Mapping for more information.

1. Log in to the Dynatrace Community and browse to the AppMon Product Downloads page.
2. Scroll down to the **Language Packs** section and download the desired language pack to a convenient location on your system.
3. Start the AppMon Client.
4. Access the **Plugins** pane of the **Server Settings** dialog box. Choose either **Settings > Dynatrace Server > Plugins or Tools > Manage Plugins**.
5. Click **Install Plugin** and select the language pack file (`.dtp` extension) that you downloaded in step 2. AppMon installs the language pack, which is now available for users that log into this AppMon Server.
6. If you have multiple AppMon Servers, perform steps 4 and 5 for the other servers.
7. Click **OK** to close the server settings.

Note

Language packs are system plugins and become a native part of your AppMon installation. Once a language pack is installed, it cannot be deleted.

Selecting a language

1. Start the AppMon Client.
2. Choose **Settings > Dynatrace Client > Settings**.
3. Click the **Language** list and select the desired language.
4. Click **OK**.

Storage

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Storage quota for sessions

Stored sessions directory

On the **Settings menu > Server Settings > Storage** pane, you can configure storage location, *total* session storage, and reserve storage *per System Profile*.

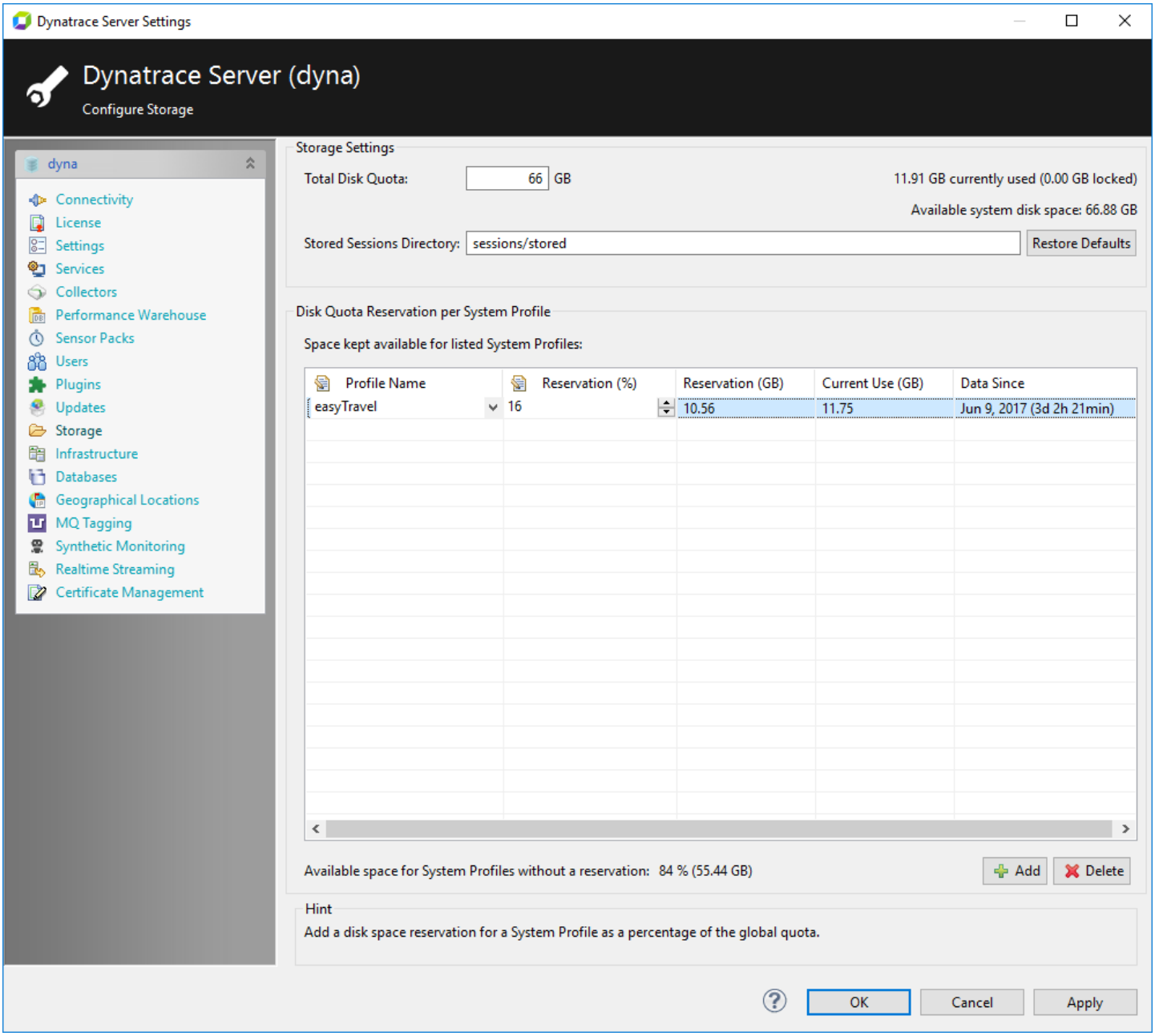
Storage quota for sessions

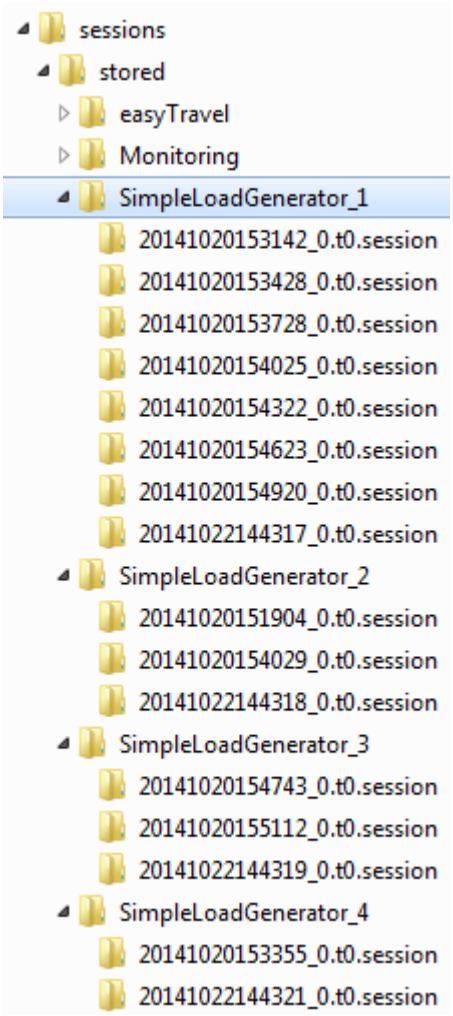
Total quota vs. reservations per System Profile

Use **Total Disk Quota** to limit the disk space the AppMon Server can use. When the quota limit is met, the oldest session that is not locked is deleted regardless of the System Profile to which it belongs. If space cannot be freed up, session recording is paused until there is space available.

Use **Disk Quota Reservation per System Profile** to reserve storage per System Profile and specify System Profile priority in terms of choosing a System Profile from which to delete a session. By making reservations for System Profiles, the oldest from the specific session data is deleted to meet the quota again. System Profiles without a reservation compete for the remaining resources. This means reservations for System Profiles are not simply a sub-quota of the **Total Disk Quota**.

Server Settings > Storage


Dynatrace Server Settings
Dynatrace Server (dyna)
Configure Storage
dyna
Connectivity
License
Settings
Services
Collectors
Performance Warehouse
Sensor Packs
Users
Plugins
Updates
Storage
Infrastructure
Databases
Geographical Locations
MQ Tagging
Synthetic Monitoring
Realtime Streaming
Certificate Management
Storage Settings
Total Disk Quota: 66 GB
11.91 GB currently used (0.00 GB locked)
Available system disk space: 66.88 GB
Stored Sessions Directory: sessions/stored
Restore Defaults
Disk Quota Reservation per System Profile
Space kept available for listed System Profiles:
Profile Name
Reservation (%)
Reservation (GB)
Current Use (GB)
Data Since
easyTravel
16
10.56
11.75
Jun 9, 2017 (3d 2h 21min)
Available space for System Profiles without a reservation: 84 % (55.44 GB)
Add
Delete
Hint
Add a disk space reservation for a System Profile as a percentage of the global quota.
OK
Cancel
Apply



Quota and reservation limits and dependencies

Total quota lower limit without reservations

The smallest number you can enter for total storage is 20 GB, but you must ensure that other processes don't consume system disk space, which could reduce the amount of total storage to less than the specified amount.
If you consider that the AppMon Server always keeps 5 GB free (or 1 GB in demo/trial settings) and sessions can be as large as 5 GB (depending on the AppMon sizing), it would not make much sense to allow a lower value.

Total quota lower limit with reservations

This can be no lower than 50 GB, for reasons stated above.

Reservations per System Profile lower limits

This is limited to 20%, which amounts to a 10 GB minimum, considering the total quota is a 50 GB minimum to enable reservations per system profile. Less than 10 GB would result in the deletion of the session that was just written to make room for a new one.

Reservations per System Profile overall limits

It is necessary to preserve a reasonable amount of remaining disk space for the System Profiles that do not have reservations. It is not possible to predict exactly how much is required, and it does not make sense to enforce some limit because the requirements depend on the storage requirements of the individual System Profiles without reservation. The user interface can only issue a general warning when space may be low.

Total storage quota

Select this option to schedule a task that monitors disk space at an interval of 10 seconds and checks whether the following occurs:

- Disk quota is exceeded.
- Minimum free disk space is reached.

> **Note**
>
> For stability reasons, the AppMon Server always keeps free disk space (1 GB for the demo settings and free trial; 5 GB otherwise) that is independent of the total storage quota settings.

Storage quota Per System Profile

The **Storage Quota Assignment** section of the dialog box allows to define disk space quota per System Profile (percentage of the total setting or part in GB) and shows the current use in GB and the date of the oldest continuous recording session.

The Server takes the following actions if no quota for System Profile are defined:

- If the size of the stored sessions exceeds the quota, the AppMon Server deletes the oldest unlocked sessions first, until conditions are met. As much disk space as required is freed to fall below the quota limit. Only whole sessions can be deleted. Deleting a 50GB session may result in falling significantly below the limit or just a bit below the limit. So never exceeding the limit is not guaranteed, but action is taken as soon as possible.
- If disk space cannot be freed up, for example if all sessions are locked, all ongoing recording sessions including session writers and dump creation stop, and an incident is raised.
- If you reduce quota, unlocked sessions are deleted without warning until quota limits are met.

The Server takes the following actions, if quota for System Profiles are defined:

- If the size of the stored sessions exceeds the disk quota, the AppMon Server deletes the oldest unlocked sessions in the following order:
  - From System Profiles without quota.
  - From System Profiles with quota until the limit is reached. Because only whole sessions can be deleted, this mat result in falling a little bit below the limit.
  - If the disk quota is still exceeded, sessions from System Profile with quota below the limit are deleted.

The following are possible user actions for resolution:

- Increase the disk space quota.
- Unlock sessions to allow the task to delete them.
- If the previous actions are unsuccessful, manually unlock and delete sessions using the AppMon Client, which requires a special procedure.

To the right of the Storage Quota option, you can view the **used storage quota**, the amount of **locked storage**, and the currently **available disk space**. If the specified storage quota is higher than that available, AppMon does not use the entire quota and displays a warning message.

Stored sessions directory

Specify a valid path (Windows, Linux, UNC) for stored sessions.

Note

Make sure the AppMon Server service (the accounts it runs under, for example Local Service) has the necessary rights to the directory. Check the **Properties > Security** settings tab for the sessions/stored directory in Windows Explorer.

When you map a path to a drive on a Windows computer and the sessions directory, the AppMon Server must be allowed to query the available drive's disk space. Consequently, the right permissions must be set for the entire drive at the root, and not only the sessions folder.

Infrastructure

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Hostgroups tab

Labels tab

Sites tab

Alerts tab

Exclusions tab

Troubleshooting

Related pages

Infrastructure monitoring dashboards

Infrastructure monitoring

Host health web view

Overview

In the AppMon Client, choose **Settings > Dynatrace Server > Infrastructure** to configure and manage sites, hostgroups, hosts, and labels applicable to hosts and alert configurations and their exceptions.

Hosts tab

In the AppMon Client, choose **Settings > Dynatrace Server > Infrastructure > Hosts** to see all connected Agents, Collectors and AppMon Server hosts. Each entry represents a single host on the network. A host belongs to exactly one detected or selected site, and exactly one hostgroup (address matches the defined pattern). See Troubleshooting > Hostgroup Priority for details on guaranteed pattern uniqueness.

If no Agent is connected to the backend server process, the host is determined as unavailable. A host can have any number of labels for flexible categorization.

Click **Create** to manually create a new host using the **Create / Edit Host** dialog box. Double-clock a host in the lists, or click the host listing then click Edit to changes settings for an existing host.

Create / Edit Host dialog box

Auto-detection typically creates all needed hosts. Create a host entry manually only in very special cases, such as:

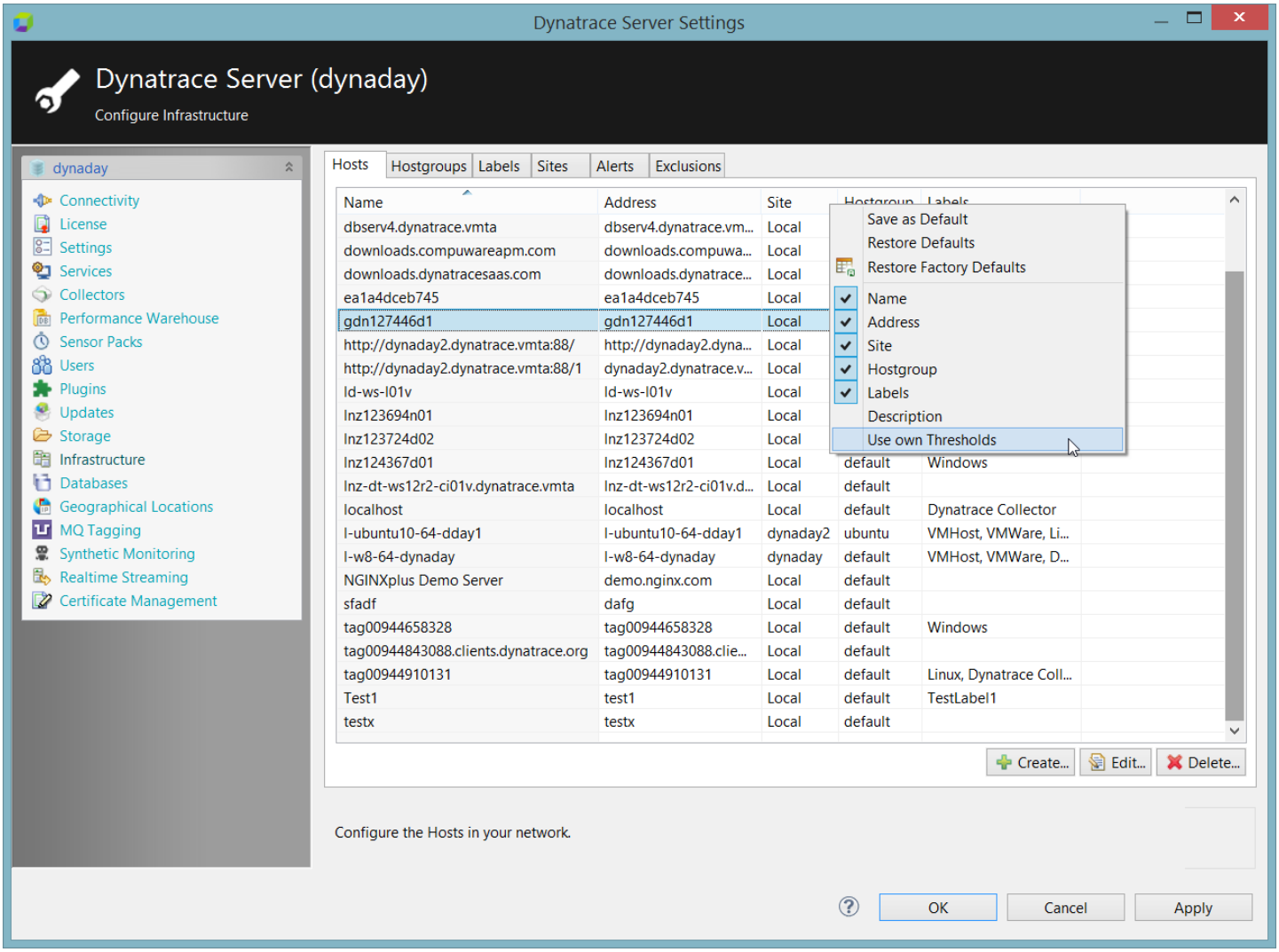
- When there is no Agent active on the host.
- You have the ping plug-in installed and you want to define an host incident rule to get an alert when the host can't be pinged.

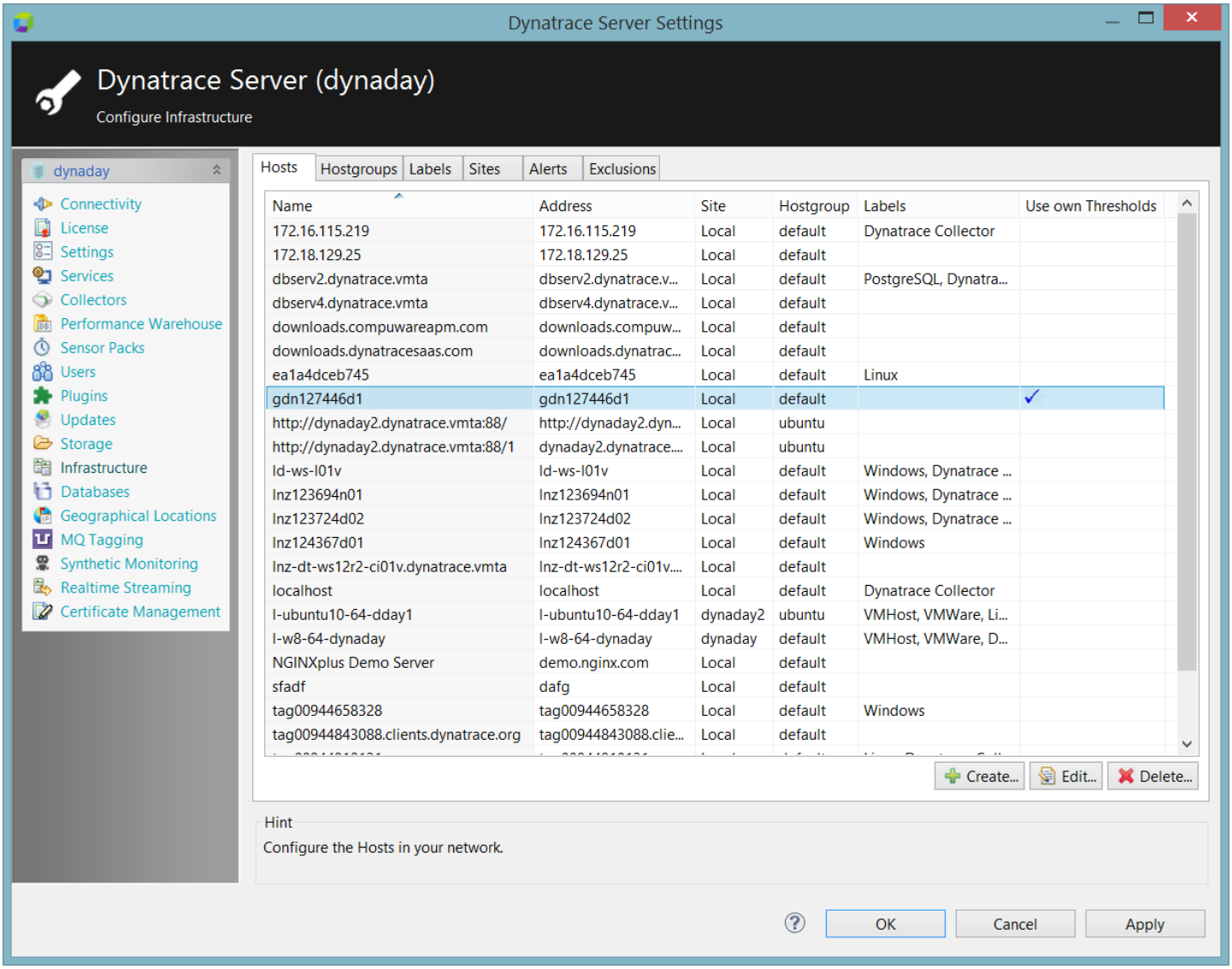
Edit a host entry to:

- Update the name that was automatically derived from the address.
- Change the host's site.
- Add a label.
- Configure specific thresholds availability for this host.
- Manually create a host when needed.

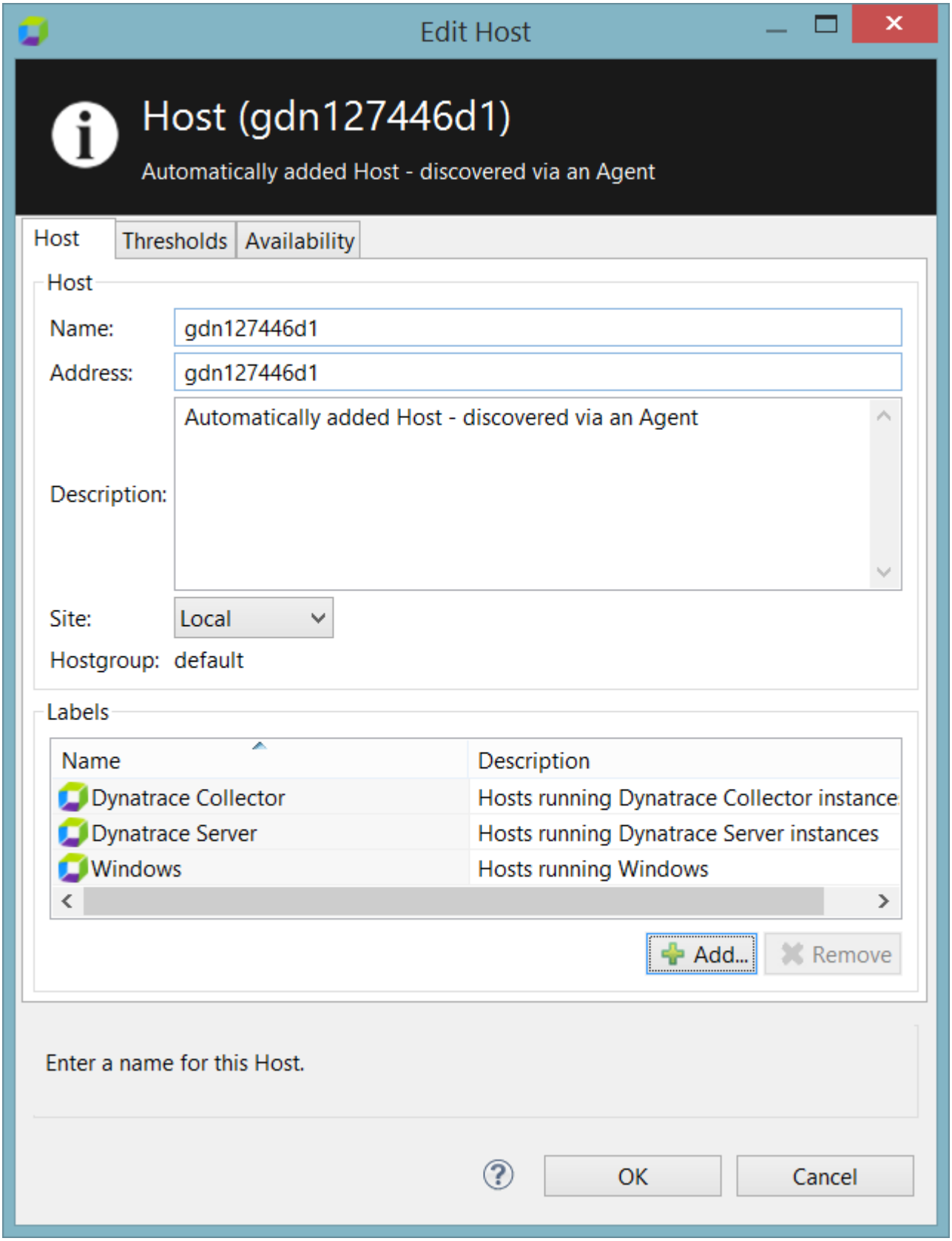
The host uses the assigned hostgroup thresholds by default.
Clear the **Use Thresholds of the Selected Hostgroup** check box in the Thresholds Tab to use host-specific thresholds.

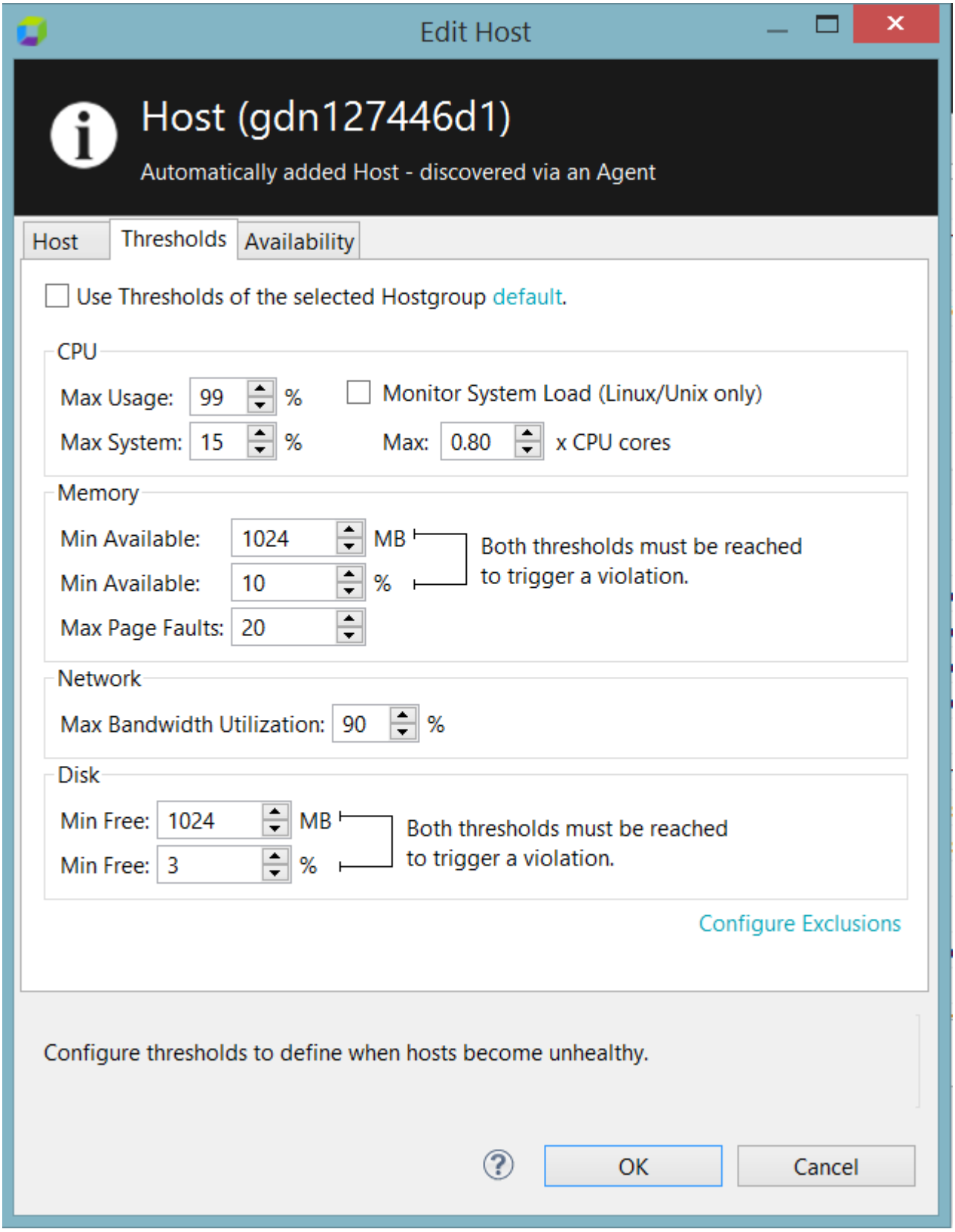
> **Tip**
>
> If using host-specific thresholds, consider displaying the **Use Own Thresholds** column on the Hosts tab. To do this, right-click on the column headers and select it from the list.


Dynatrace Server Settings
Dynatrace Server (dynaday)
Configure Infrastructure
dynaday
Connectivity
License
Settings
Services
Collectors
Performance Warehouse
Sensor Packs
Users
Plugins
Updates
Storage
Infrastructure
Databases
Geographical Locations
MQ Tagging
Synthetic Monitoring
Realtime Streaming
Certificate Management
Hosts
Hostgroups
Labels
Sites
Alerts
Exclusions
Name
Address
Site
Hostgroup
Labels
dbserv4.dynatrace.vmta
dbserv4.dynatrace.vm...
Local
downloads.compuwareapm.com
downloads.compuwa...
Local
downloads.dynatracesaas.com
downloads.dynatrace...
Local
ea1a4dceb745
ea1a4dceb745
Local
gdn127446d1
gdn127446d1
Local
http://dynaday2.dynatrace.vmta:88/
http://dynaday2.dyna...
Local
http://dynaday2.dynatrace.vmta:88/1
dynaday2.dynatrace.v...
Local
ld-ws-l01v
ld-ws-l01v
Local
lnz123694n01
lnz123694n01
Local
lnz123724d02
lnz123724d02
Local
lnz124367d01
lnz124367d01
Local
default
Windows
lnz-dt-ws12r2-ci01v.dynatrace.vmta
lnz-dt-ws12r2-ci01v.d...
Local
default
localhost
localhost
Local
default
Dynatrace Collector
l-ubuntu10-64-dday1
l-ubuntu10-64-dday1
dynaday2
ubuntu
VMHost, VMWare, Li...
l-w8-64-dynaday
l-w8-64-dynaday
dynaday
default
VMHost, VMWare, D...
NGINXplus Demo Server
demo.nginx.com
Local
default
sfadf
dafg
Local
default
tag00944658328
tag00944658328
Local
default
Windows
tag00944843088.clients.dynatrace.org
tag00944843088.clie...
Local
default
tag00944910131
tag00944910131
Local
default
Linux, Dynatrace Coll...
Test1
test1
Local
default
TestLabel1
testx
testx
Local
default
Save as Default
Restore Defaults
Restore Factory Defaults
Name
Address
Site
Hostgroup
Labels
Description
Use own Thresholds
Create...
Edit...
Delete...
Configure the Hosts in your network.
OK
Cancel
Apply



See Hostgroup Dialog Box - Thresholds Tab and Hostgroup Dialog Box - Availability Tab for more information on the **Thresholds** and **Availability** tabs.


Edit Host
Host (gdn127446d1)
Automatically added Host - discovered via an Agent
Host
Thresholds
Availability
Host
Name:
gdn127446d1
Address:
gdn127446d1
Description:
Automatically added Host - discovered via an Agent
Site:
Local
Hostgroup:
default
Labels
Name
Description
Dynatrace Collector
Hosts running Dynatrace Collector instance
Dynatrace Server
Hosts running Dynatrace Server instances
Windows
Hosts running Windows
Add...
Remove
Enter a name for this Host.
OK
Cancel



Hostgroups tab

Choose **Settings > Dynatrace Server > Infrastructure > Hostgroups** to see the custom hostgroups and the patterns that define their members. Use this tab to access the **Add / Edit Hostgroup** dialog box to create, edit, and configure hostgroups.

Add/Edit Hostgroup dialog box

The **Add/Edit Hostgroup** dialog box includes tabs that let you:

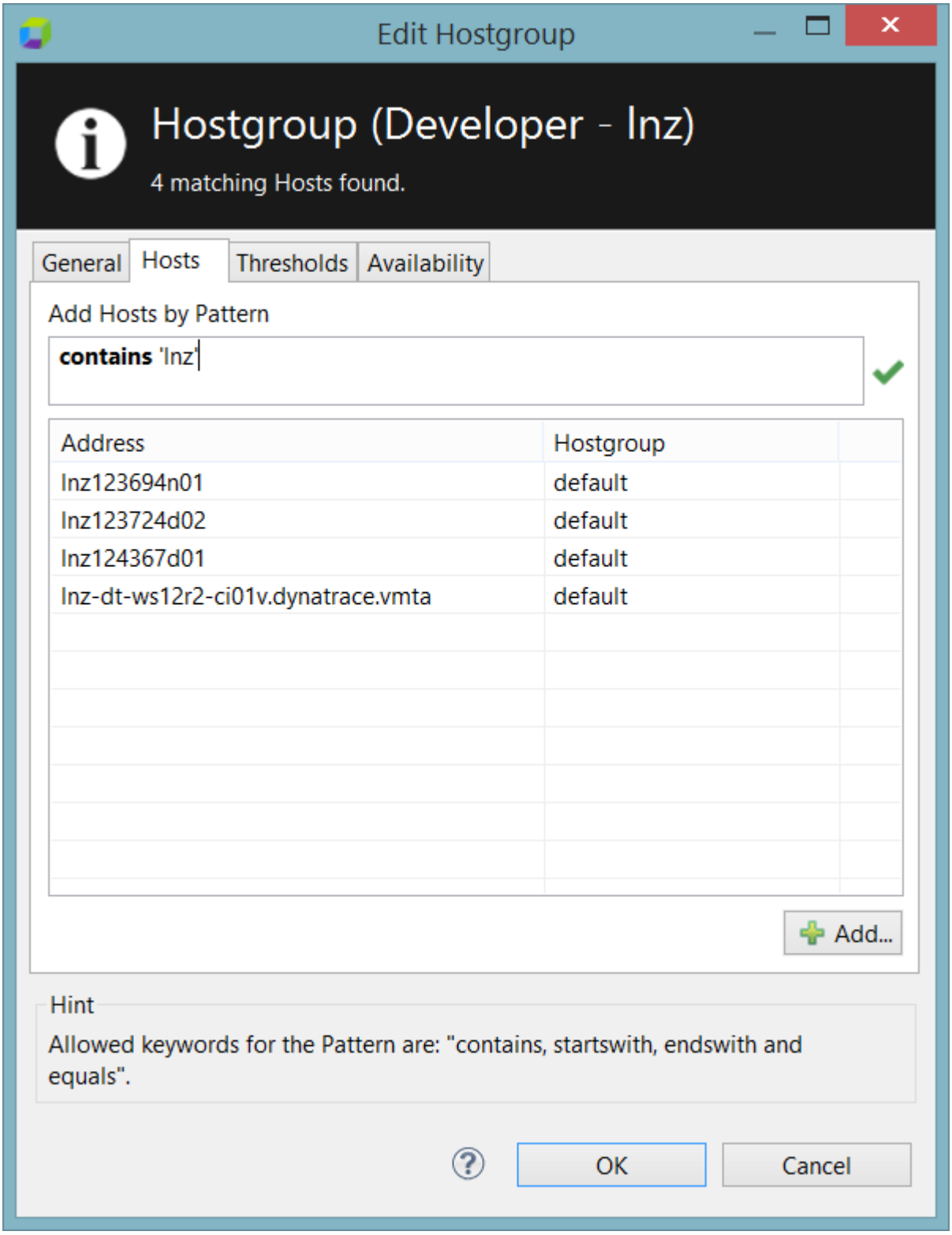
- Set the pattern to match hosts into a group by comparing their address to the pattern.
- Configure group-specific thresholds.
- Set number of hours from the time the host is offline until removal from the infrastructure overview.
- Define host availability and processes monitored by the host agent.

> **Tip**
>
> Click the **Add** button to add hosts manually to the hostgroup. The pattern text is updated with `equals 'hostaddress'`.

Hostgroup Dialog Box - Hosts Tab - Pattern Definition

To add a host by pattern, type one of the string comparison operators in the **Add Hosts by Pattern** text box. Matching operators appear in the box as you type, so you can select from the list of matches. Patterns are simple logical expressions that can include the following:

- One of the string comparison operators: **equals**, **startswith**, **contains** and **endswith**.
- A string in single quotes. Escape any strings containing a single quote with a backslash character, for example `'isn\'t'`.
- The logical operators: **not**, **and**, **or (in order of precedence)** can be grouped together with parentheses to change order of precedence. For example: `(contains 'at' or contains 'de') and endswith 'corp'`.



All functions are case-insensitive and require one string literal as parameter. For example:

- equals ‘34325.clients.emea.dynatrace.corp’
- startswith ‘lnz’
- contains ‘emea’
- endswith ‘.com’
- not contains ‘internal’

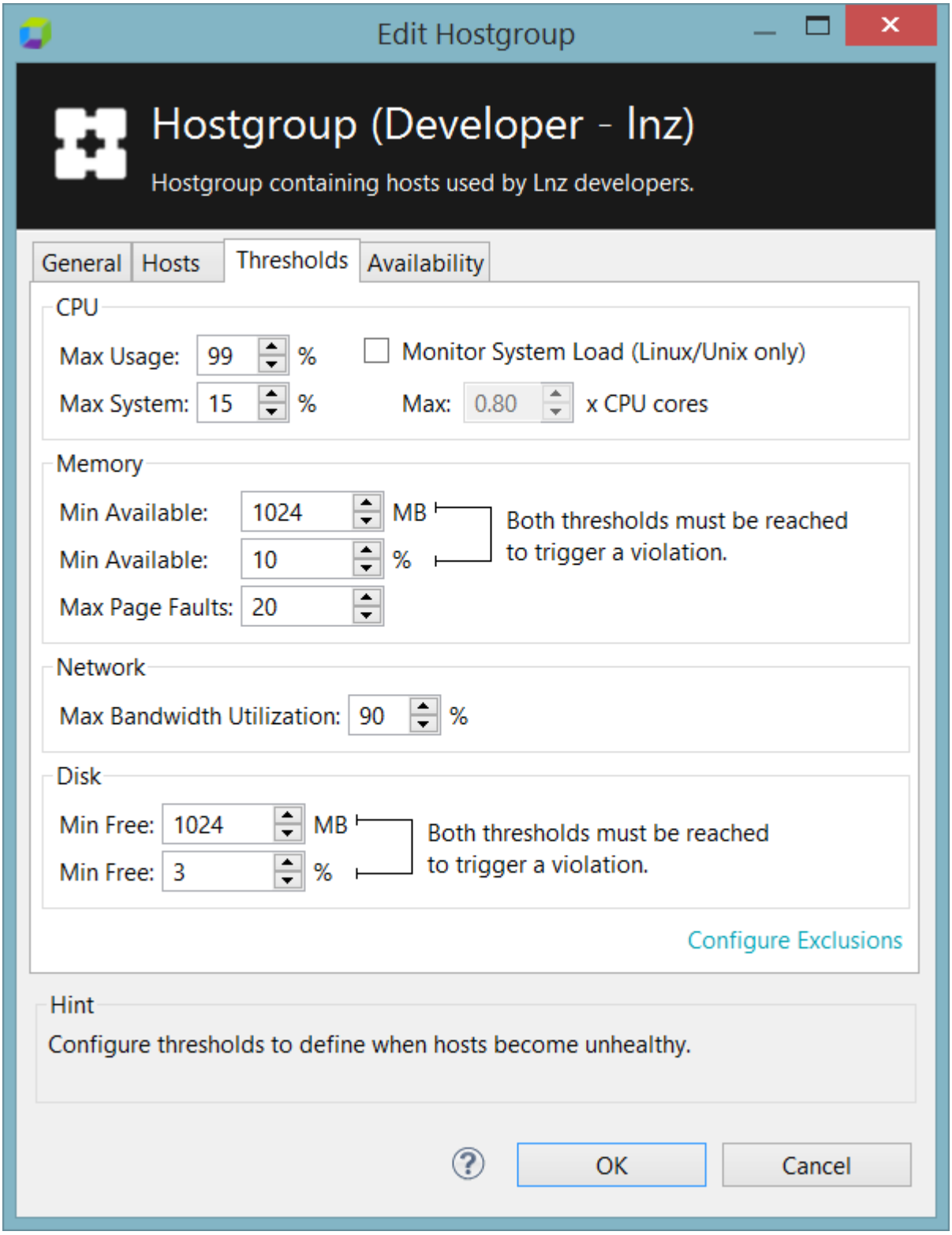
- contains ‘at’ or contains ‘de’, and endswith ‘corp’

Hostgroup Dialog Box - Thresholds Tab

Set thresholds on this tab to determine exactly when the hosts in the selected hostgroup show as unhealthy in the Infrastructure Overview dashboard or in the Hosts tile of the Operations dashboard in AppMon Web.

Generally, if any of the four groups (CPU **or** memory **or** network **or** disk) is unhealthy, the whole host is considered unhealthy. More specifically:

- The server evaluates the host health every minute. The CPU, memory and network measurements of the last 15 minutes are split into one minute chunks, by average aggregation.
  If 13 chunks violate the relevant criteria, the host is considered as unhealthy. The health is deemed good again when at least three of the chunks are healthy. The healthy chunks don’t have to be consequential.
- Disks health status changes immediately without a watch period when a threshold is crossed, so the host health displays changes accordingly.



Threshold groups and subgroups

- **CPU group**: Any of the three values (usage % or system % or load) spoils CPU health after the above mentioned formula.

- **Memory group**: Minimum available subgroup **or** maximum page faults per second spoil memory health after the above mentioned formula.
  - Both minimum available MB **and** minimum available % of the Minimum Available subgroup must exceed to contribute.
  - In all: (minimum available MB **and** minimum available %) **or** minimum page faults per second.
  - To ignore one of two **anded** values, just set one threshold so it is always reached. For example, you can set a very high size or percentage. `Memory available < 100%` returns true, so a violation occurs when `min. available MB < threshold` becomes true.
- **Disk group and subgroup**: Both minimum free MB **and** minimum free % must exceed to trigger, but they trigger immediately. To ignore one of two **anded** values, just set one threshold so it is always reached.

Click **Configure Exclusions** to exclude specific disks, mount points, or NICs from monitoring. Explicitly include what would otherwise be excluded. See Exclusions Tab for more information.

For the **Add/Edit Host** dialog box, select the **Use Thresholds of the Selected Hostgroup** check box in the Thresholds Tab to use host-specific thresholds.

> **Tip**
>
> The host uses the thresholds of the assigned hostgroup by default. You can override hostgroup-wide thresholds and set host-specific ones in the **Thresholds** tab.

Compare measures between operating systems

Measures are relatively comparable between different operating systems, with the following limitations:

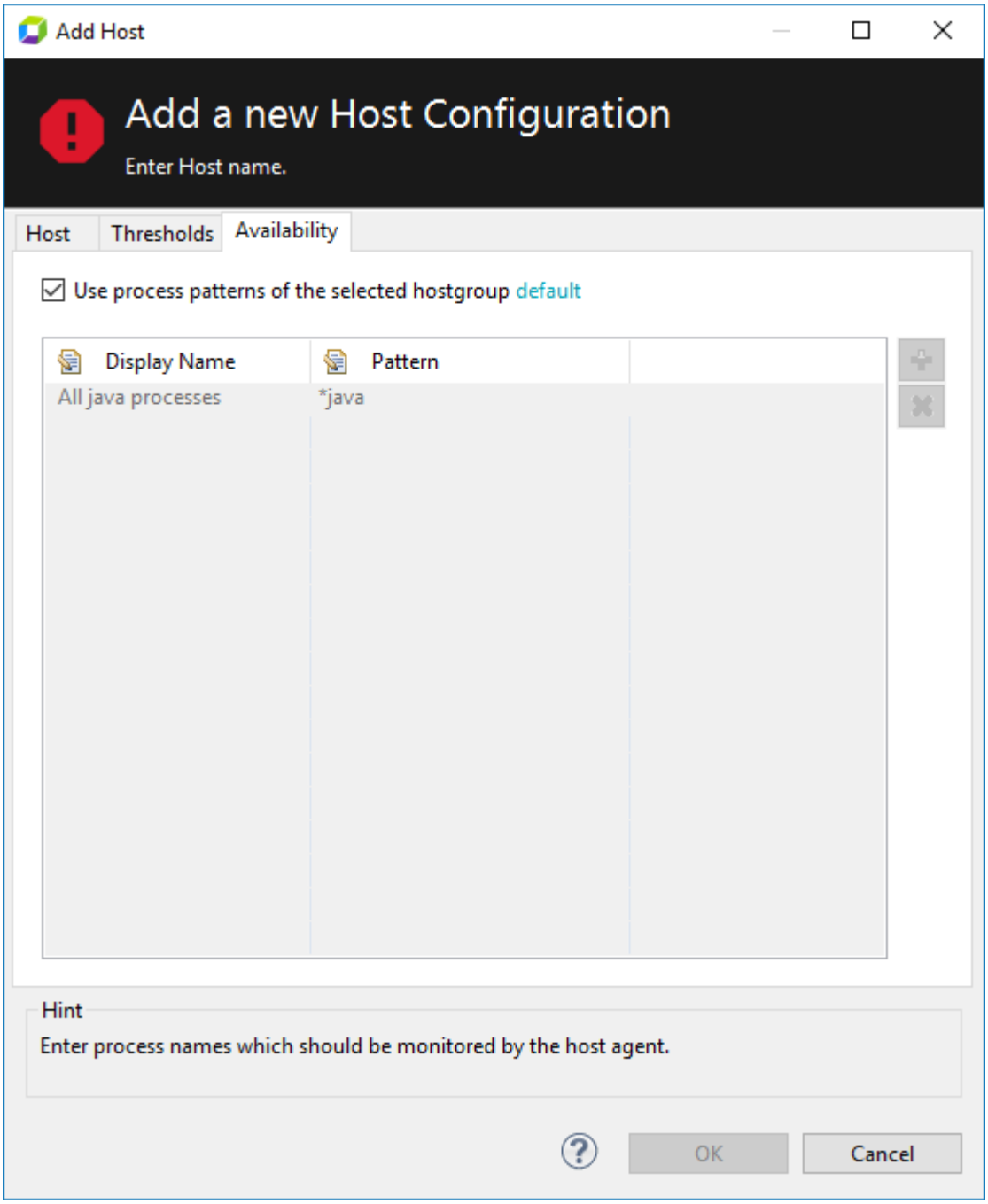
- Windows does not deliver a CPU load like *NIX systems. CPU load is omitted in health calculations on Windows hosts.
- Only hard page faults are considered as page faults. Windows systems have hard page faults even with free memory.
- Page faults on AIX versions earlier than 5.2 report soft and hard page faults.
- Disk space available is determined from the point of view of the Agent. If the Agent has restrictions such as disk space quota, only the disk space available to the Agent is reported as free space.
- Page faults require disk access, which includes access to memory mapped files. Applications such as backup software use memory mapped files a lot and might cause temporarily high page fault rates.

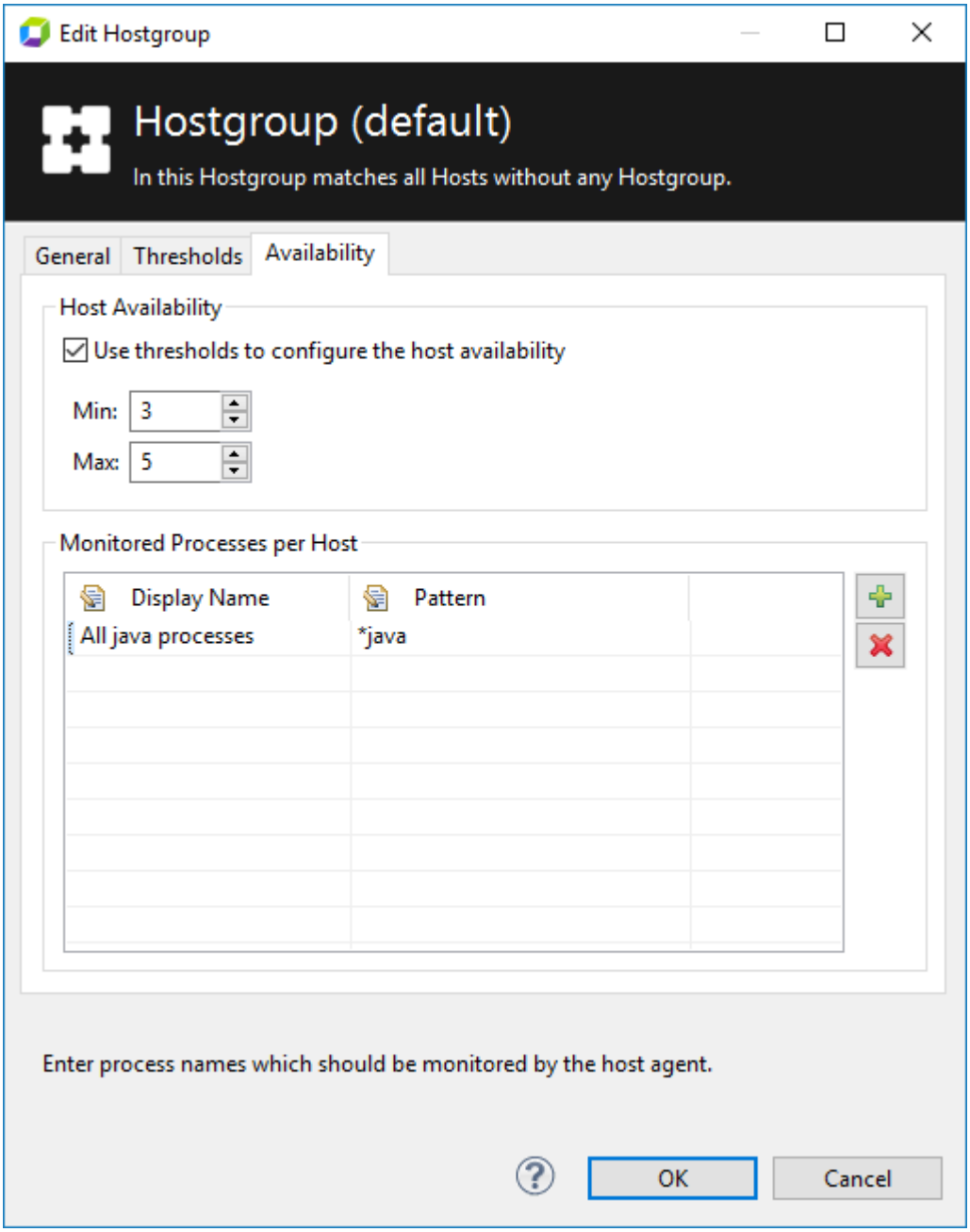
See Host Health Monitoring > What is Monitored on What Platform? for a detailed comparison of operating systems.

Hostgroup Dialog Box - Availability Tab

Host Availablity: Define Minimum and Maximum thresholds of the number of required hosts in the hostgroup. **Process Availability**: Specify processes for monitoring by the Host Monitoring Agent. Running and Unavailable processes display in the infrastructure overview. Once a process is not available, an incident is created. For the **Add/Edit Host** dialog box, you can select the check box to use the process patterns of the selected hostgroup, then specify the processes using the **Add / Edit Hostgroup** dialog box.

You can monitor processes by using pattern matching. Click the button to the right of the process name list, then click the top available line in the Pattern column to enter a pattern match string. The process name to match with the pattern includes the process path. The pattern matching supports a * wildcard at the beginning and end. For example, *w3wp.exe -ap ".NET 4.5"*. This specifies that the host monitors all processes matching the pattern. Click in the Display Name column to add a descriptive name for the process pattern match.


Add Host
Add a new Host Configuration
Enter Host name.
Host
Thresholds
Availability
Use process patterns of the selected hostgroup default
Display Name
Pattern
All java processes
*java
Hint
Enter process names which should be monitored by the host agent.
OK
Cancel

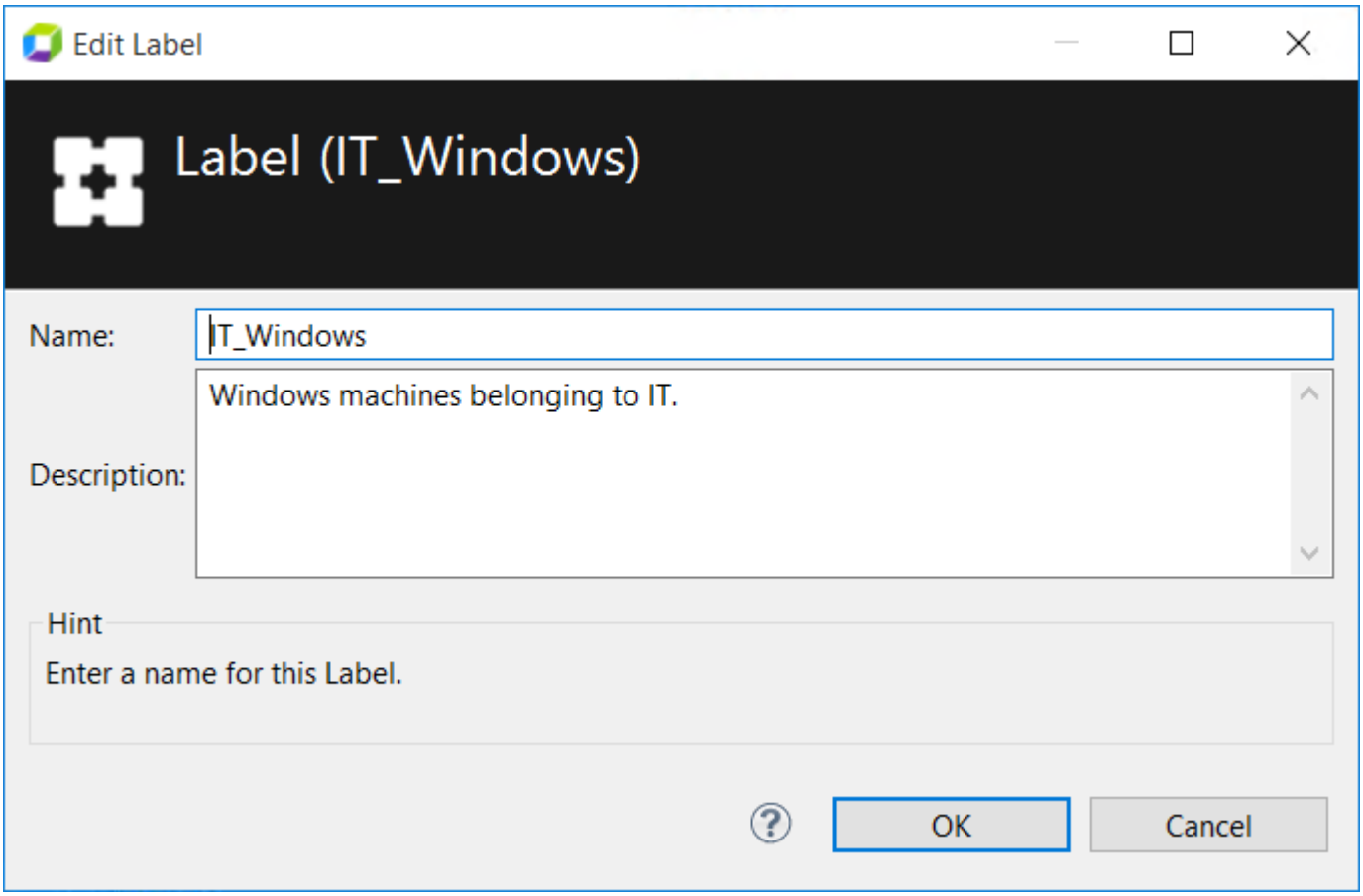


Labels tab

The Labels tab lists default and custom labels to logically group hosts when you placing the same host into more than one group. This is diametrically opposed to patterns which, should uniquely group hosts into hostgroups and are guaranteed to do so with the help of the Priority property.

> **Tip**
>
> Default labels are indicated with the Dynatrace logo.

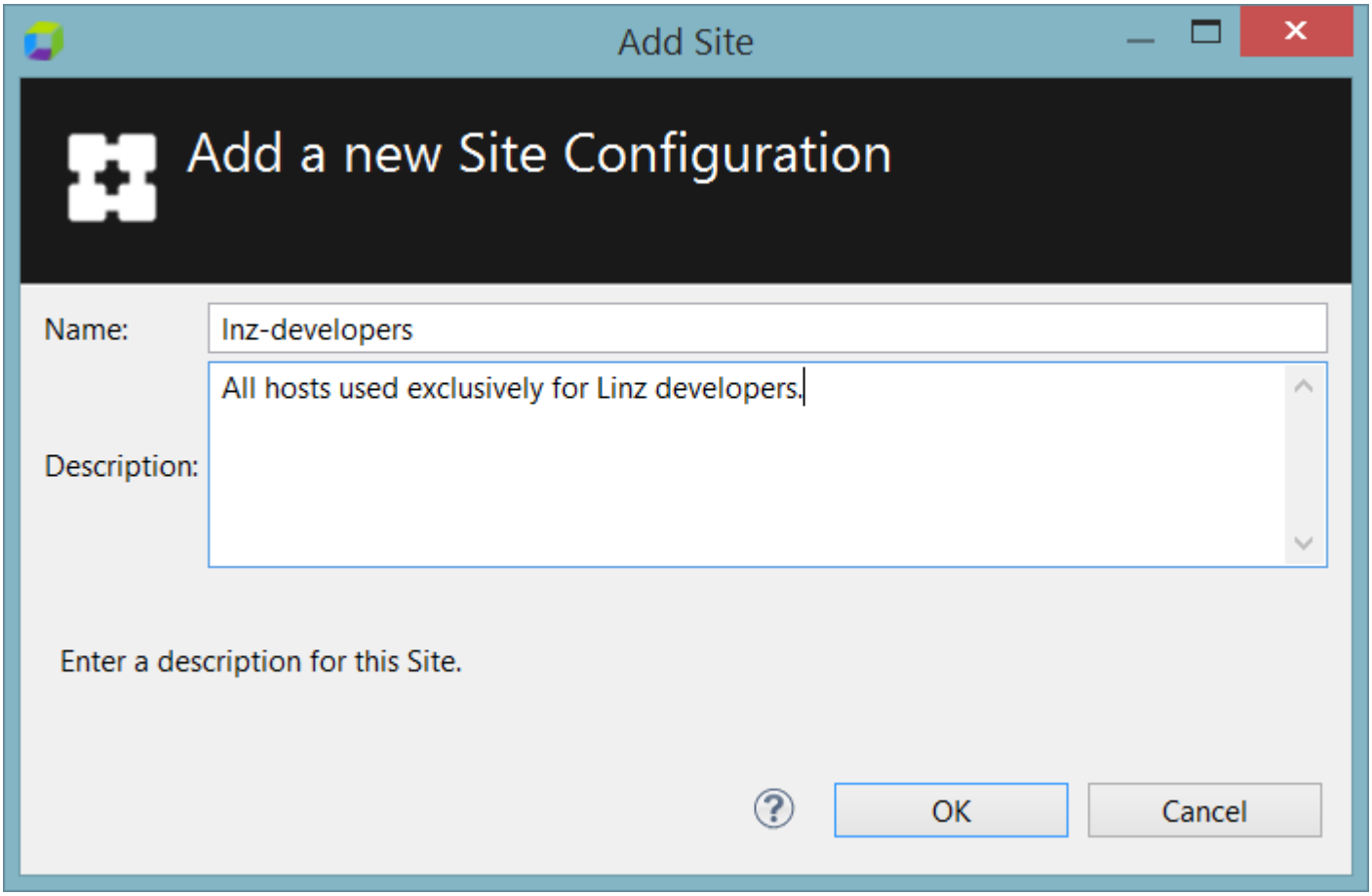
Click **Create** or **Edit** to open the **Add/Edit Label** dialog box to create and edit any type of labels that fit your purpose. For example, create labels for specific operating systems, or separate labels for development, testing or production. If you create a Monitor for a label, AppMon automatically executes it for equally labeled hosts. You are not able to edit the default labels.



Sites tab

The Sites tab displays default and custom site names used to assign hosts to a specific location. A site consists of a name and description.

Click **Create** or **Edit** to open the **Add/Edit Site** dialog box, where you can specify a site for a host. For example, you can configure sites for different departments or cities. Each host has exactly one site. As with labels, you are not able to edit the default site.

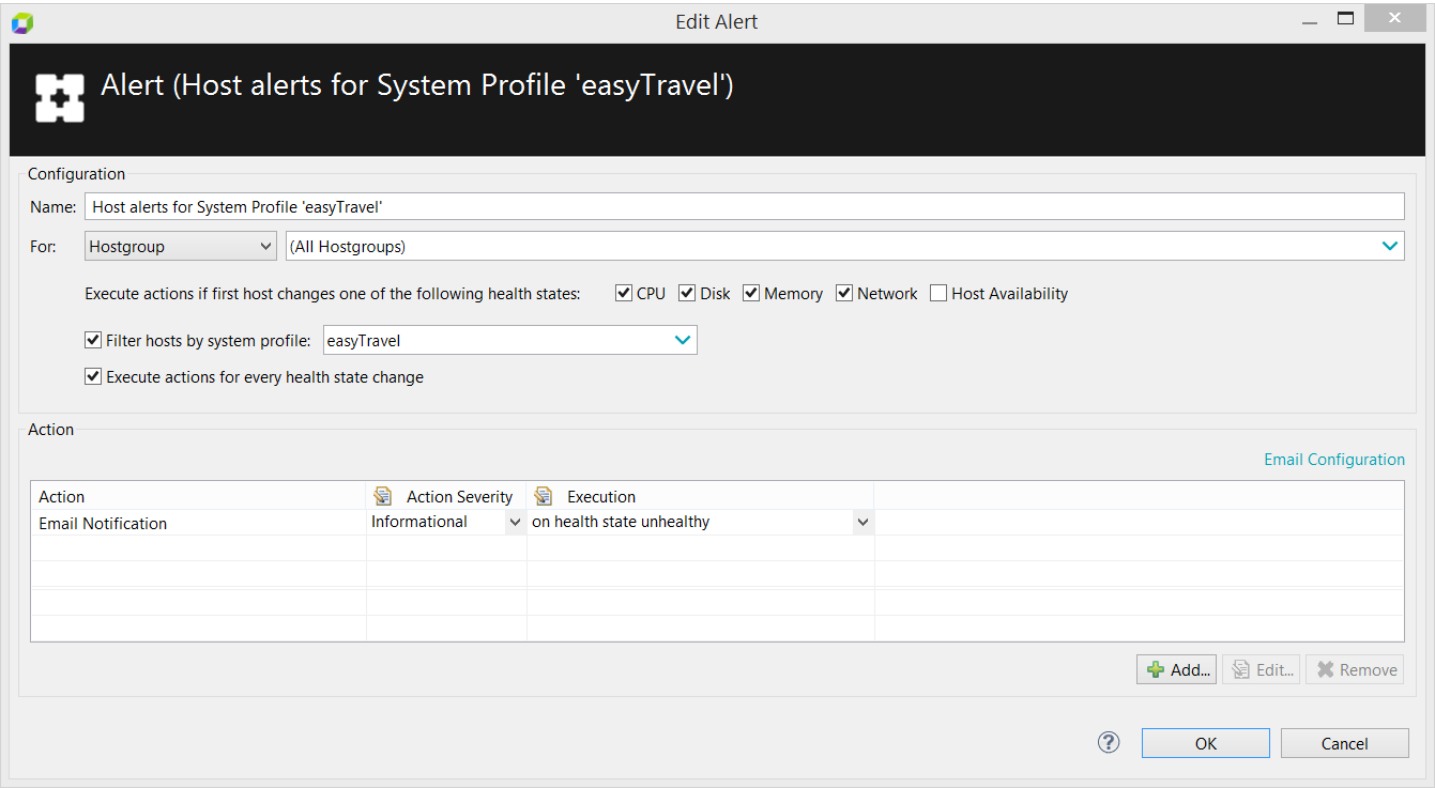


Alerts tab

This tab lists the default and custom alerts created to send notifications when alerts occur. After configuring thresholds for the hosts, create alerts to define whom to notify when the thresholds are exceeded and how to act on violations.

Click **Create** or **Edit** to open the **Add/Edit Alert** dialog box. To create a custom alert, provide the following.

- **Name**: Give the alert a unique name that describes its purpose.
- **For** list box: Select the category (**Host, Hostgroup, Label, Site)** by which the actions should trigger.
- **System Profile** check box (optional): Filter the hosts further by the selected System Profile.
- **CPU, Disk, Memory, Network**: Check the threshold groups that should trigger.
- **Execute action if first host changes one of the following health states**: Every time one threshold of any affected host is exceeded, the actions are executed. Select check boxes to indicate in which areas the actions are executed when the first host becomes unhealthy.
- **Action**: Configuration of the actions for this alert. See Incidents and Alerting for more information about actions.



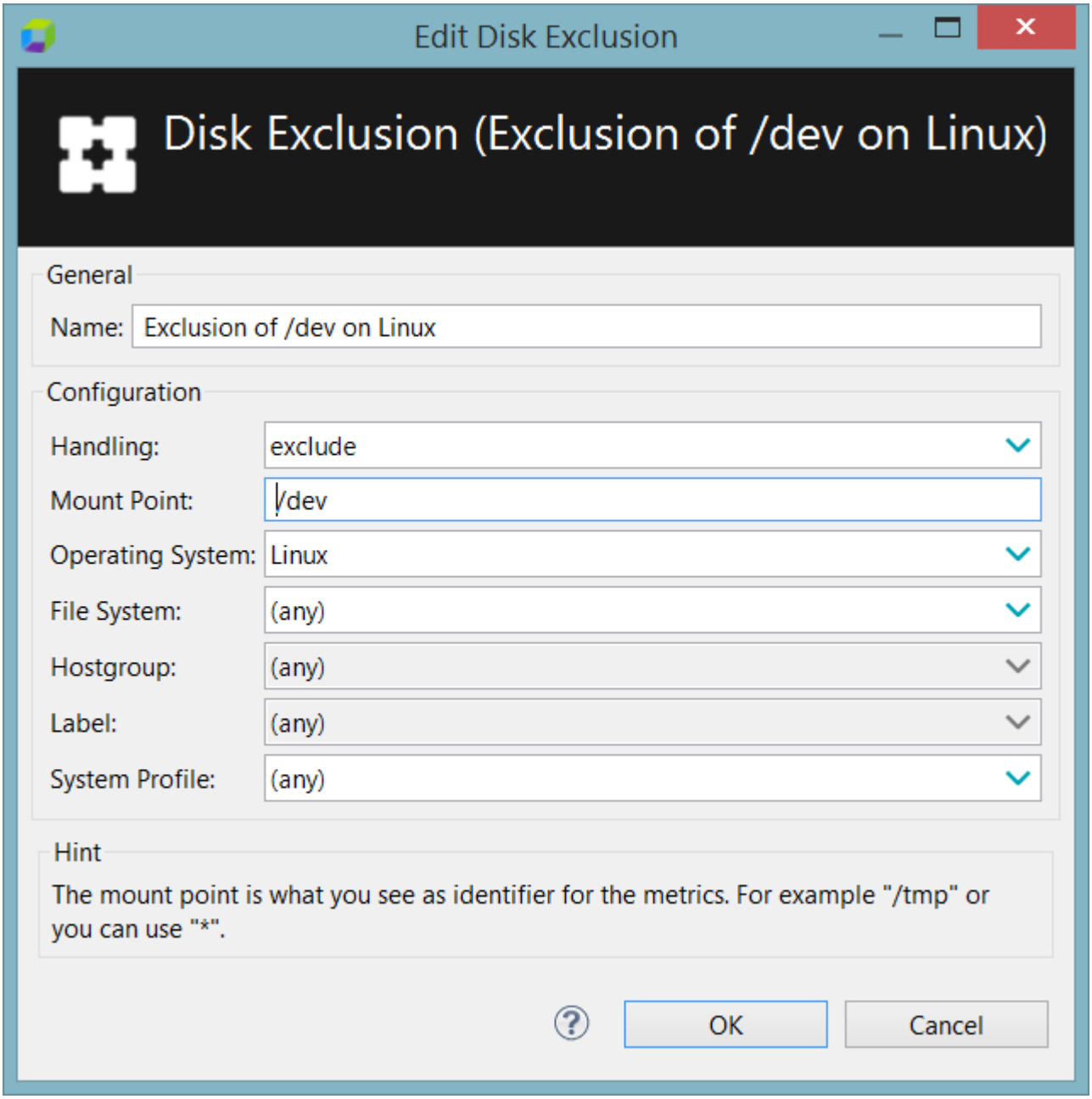
Exclusions tab

The Exclusions tab lists specific disks and mount points than can be excluded or explicitly included for display on the Infrastructure Overview dashboard. Click **Create** or **Edit** to open the **Add/Edit Alert** dialog box and exclude specific disks and mount points or NICs from monitoring. Explicitly include what would otherwise be excluded. See the dialog box or the screenshot below, for predefined, standard *NIX exclusions or inclusions that may fit your needs.

A disk, mount point or NIC is first matched against inclusions. If it matches, it is not excluded. If it does not match, it is matched against exclusions. If it matches, it is excluded. All outstanding non-matches are included. Specify the following to configure disk exclusions.

- **Name**: Give the inclusionor exclusion, or ignore a unique descriptive name.
- **Handling** types: The disk, mount point or NIC. Select one of the following:
  - **Exclude**: Disk is not visible in the Infrastructure Overview.
  - **Include**: Is not excluded.
  - **Ignore**: Disk is visible, but does not trigger any action at the defined threshold.

- **Mount Point**: The metrics identifier, for example `/tmp` on *NIX or `c:\` on Windows.
  Show *NIX mount points with `cat /proc/mount`.
  The **asterisk wildcard** is allowed, but only at the end. For example:
  - `/h*` means all mount points starting with `/h`.
  - `/home/*` works, but only if you have mount points in the `home` directory.
  - `/home*` to catch the whole `/home` mount point.
  - `/tmp*` should exclude `/tmp/foo`.
  - `/tmp/*` does not exclude `/tmp`.
- **Operating System**: The host operating system.
- **File System**: The host file system. You can add a new file system to the drop down list or group them by using the asterisk wildcard at the end. For example, `ext*` indicates all EXT versions.
- **Hostgroup, Label**: These two can only be selected if the **Handling** setting is **ignore**.
- **System Profile**: Filter the hosts further by the selected System Profile.



Disks

Name	Mount Point	Handling	Operating System	File System
Exclusion of debugfs on Linux	*	exclude	Linux	debugfs
Exclusion of iso9660 on Linux	*	exclude	Linux	iso9660
Exclusion of tmpfs on Linux	*	exclude	Linux	tmpfs
Exclusion of tmpfs on Linux	*	exclude	Linux	tmpfs
Exclusion of udf on Linux	*	exclude	Linux	udf
Inclusion of /run on Linux	/run	include	Linux	(any)
Inclusion of /run on Linux	/run	include	Linux	(any)
Inclusion of /tmp on Linux	/tmp	include	Linux	(any)
Inclusion of /tmp on Linux	/tmp	include	Linux	(any)

Find: Linux Match case 20 items found

Create... Edit... Delete...

Configuring NIC exclusions

NIC exclusion settings are generally the same as disk settings. Specific is only the name of the network interface.

Troubleshooting

Hostgroup priority

A host has to belong to one and only one hostgroup If defined patterns let a host accidentally match into more than one hostgroup there, use the hostgroup `Priority` property to guarantee a unique match. When you create a hostgroup, a hidden `Priority` property gets assigned. See Hostgroup Dialog - Hosts Tab - Pattern Definition) for more information.

You can uncover the priority property OR column by right-clicking the column header on the **Hostgroups** tab and selecting **Priority**. Change a hostgroup's priority by right-clicking on its row and selecting Higher or Lower Priority.

Geographical locations - mapping of IP addresses

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Geographical locations database update

Map IP address ranges to locations

HTTP request headers

Locations in the world map

Related pages

Defining custom locations in the world map

Defining custom locations in the world map

AppMon uses the GeoLite2 City database by MaxMind to resolve IP addresses to geographical locations. For typical applications, no further setup is required. See Configure Geographic Location to learn how AppMon obtains the Client IP address.

Geographical locations database update

The database containing IP addresses to geographical locations mapping updates as part of a product update. See User Experience Management Data Files for more information.

Map IP address ranges to locations

If monitoring intranet applications, visitors may have private IP addresses. It is impossible to determine the location of these IP addresses. As a result, features such as grouping visits by location are available in this scenario. To exploit the advantages of User Experience Management, map the private IP address ranges to the locations where these ranges are used.

Consider the following example:

A company operates an ALM application that is accessed from a web browser. To ensure availability and application performance and to detect bugs easily, AppMon uses UEM to monitor the ALM application. There are three offices which access this application. They are in New York, Boston, and San Francisco. Internally, the offices use private IP address ranges that the DHCP servers provide. The San Francisco office also uses public IPv6 addresses. The table below shows the IP address setup:

Location	Net ID	Subnet Mask
New York	192.168.10.0	255.255.255.0
Boston	192.168.20.0	255.255.255.0
San Francisco	192.168.30.0	255.255.255.0
San Francisco	2001:1234:C0A8:1E00::	/64

To map these private IP ranges, access the **Geographical Locations** pane of the **Server Settings** dialog box. You can edit the mapping from IP ranges to geographical locations manually.

The image details the configuration for the example. Use the **Create** and **Edit** buttons to set up and change the mappings for all the relevant IP address ranges.

> **Note**
>
> To map a single IP addresses, use a range of 32 for IPv4 and 128 for IPv6 when you enter the IP address.

Click **OK** to apply the settings on the AppMon Server immediately.

The `subnetmask` attribute value is the number of non-zero bits as used in the CIDR notation. The region information is optional but it provides more details within AppMon. In the corresponding XML file, the region is encoded using the ISO 3166-2 standard for the US and Canada and the FIPS 10-4 standard for the rest of the world. The city name is not validated and may be a random string. The country is encoded using the ISO 3166-1 Alpha-2 country codes.

Important

The IP address ranges specified must be disjoint subnets. Foe example, 192.168.10.0/24 and 192.168.20.0/24 are disjoint. However, 192.168.10.0/16 and 192.168.20.0/24 are not.

Note

For optimal performance, IP mappings are limited to 10000 entries. To modify this limit, edit `ipmappings.xml` with any text or XML editor. The attribute named `limit` inside the xml-node `ipmappingconfig` must be added or modified to contain the desired number of maximum allowed IP mappings.

Import IP address mappings from a CSV file

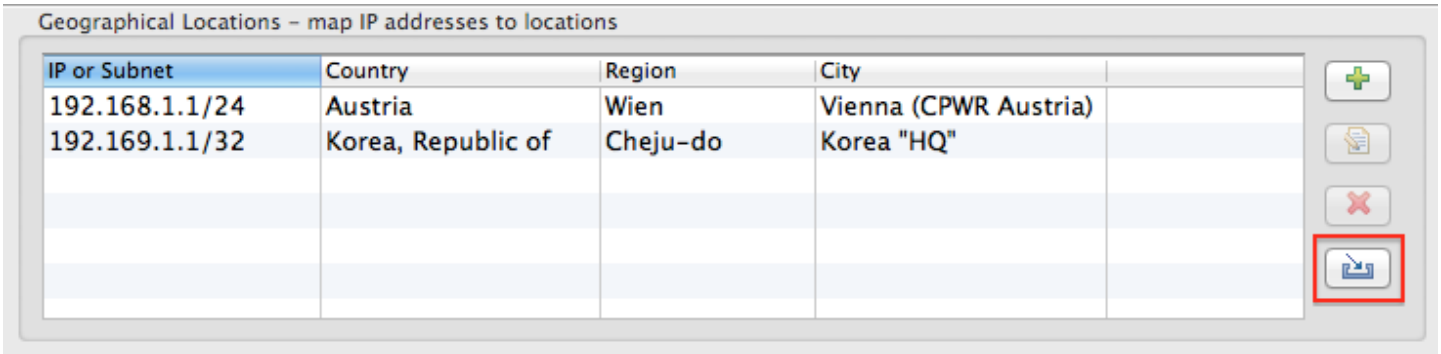
You can also import IP Address mappings from a CSV file. The **Import** button next to the table opens a dialog box in which you can choose the CSV file that contains your mappings.
AppMon skips conflicting address ranges or invalid lines and logs them to the Client log. The imported file data overwrites existing mappings with identical address ranges.

The following are requirements for the CSV file:

- Must be UTF-8 encoded.
- Allowed delimiters are comma, semicolon and tab.
- Use double quotes if a value contains a delimiter.
- Escape double quotes inside a value with a preceding double quote.
- Valid values for country and region are the same as shown in the drop-down menus when manually adding a mapping.

The following is an example:

```
192.168.1.1,24,Austria,Wien,Vienna (Dynatrace Austria)
192.169.1.1,32,"Korea, Republic of",Cheju-do,Korea ""HQ""
```



HTTP request headers

Use HTTP request headers to determine the Client IP address or Server host name.

Locations in the world map

The Geographic locations database contains a list of geographic locations for specific IP addresses throughout the world. However this is never a definitive list. There could be IP addresses that don't appear in this list. You may also want to associate unique custom IP addresses directly to a city's location. To do this, define a geo location for a specific city in the server settings. See Defining Custom Locations in the World Map for more information.

Defining custom locations in the world map

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Edit the Business Transaction corresponding to the app's System Profile

Specify the IP address for a custom city

Viewing visits and the world map

Finding out the geographic location for a given point

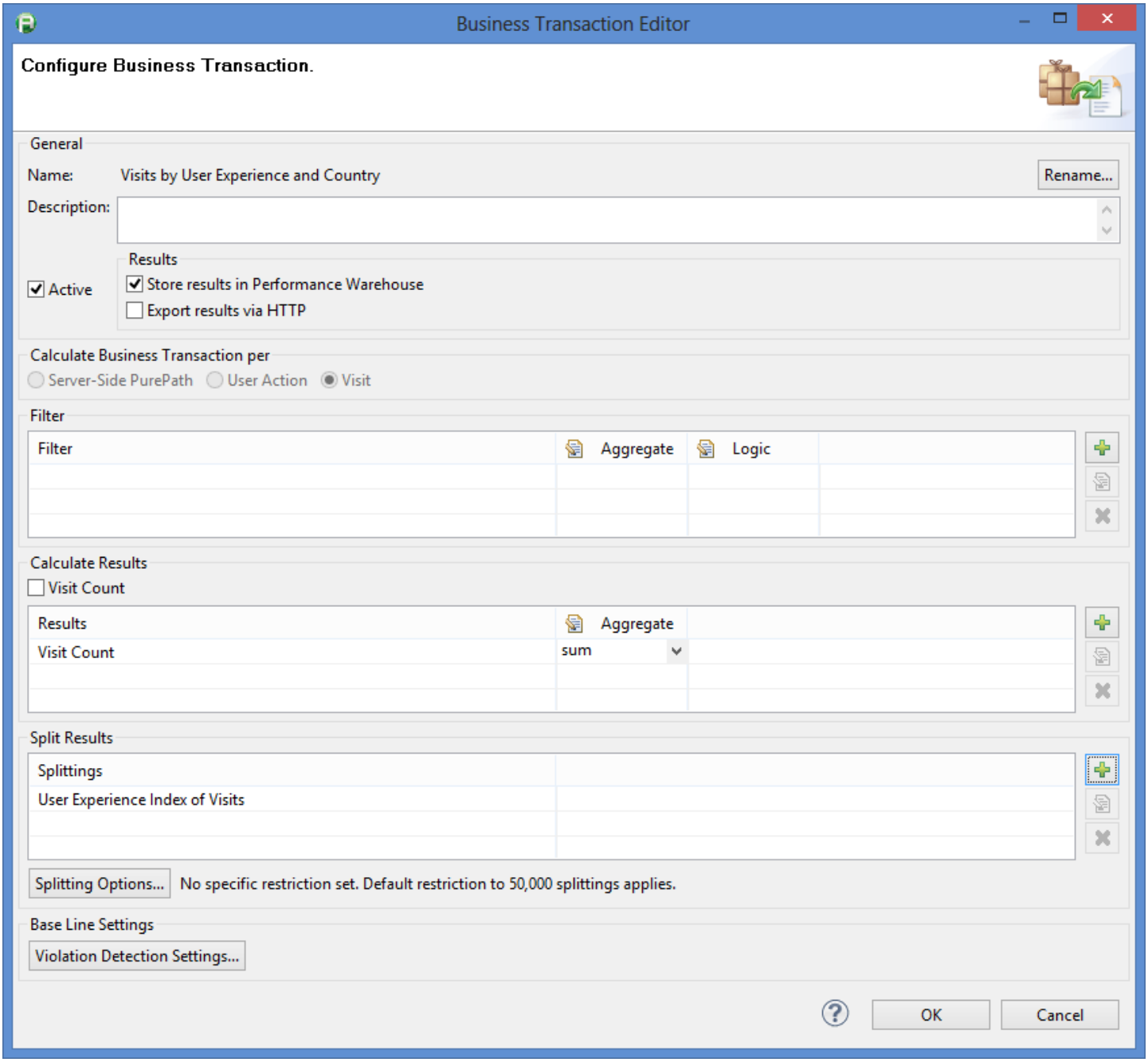
The geographic locations database maps countries and regions to IP addresses. You may want to adapt it due to a number of reasons.

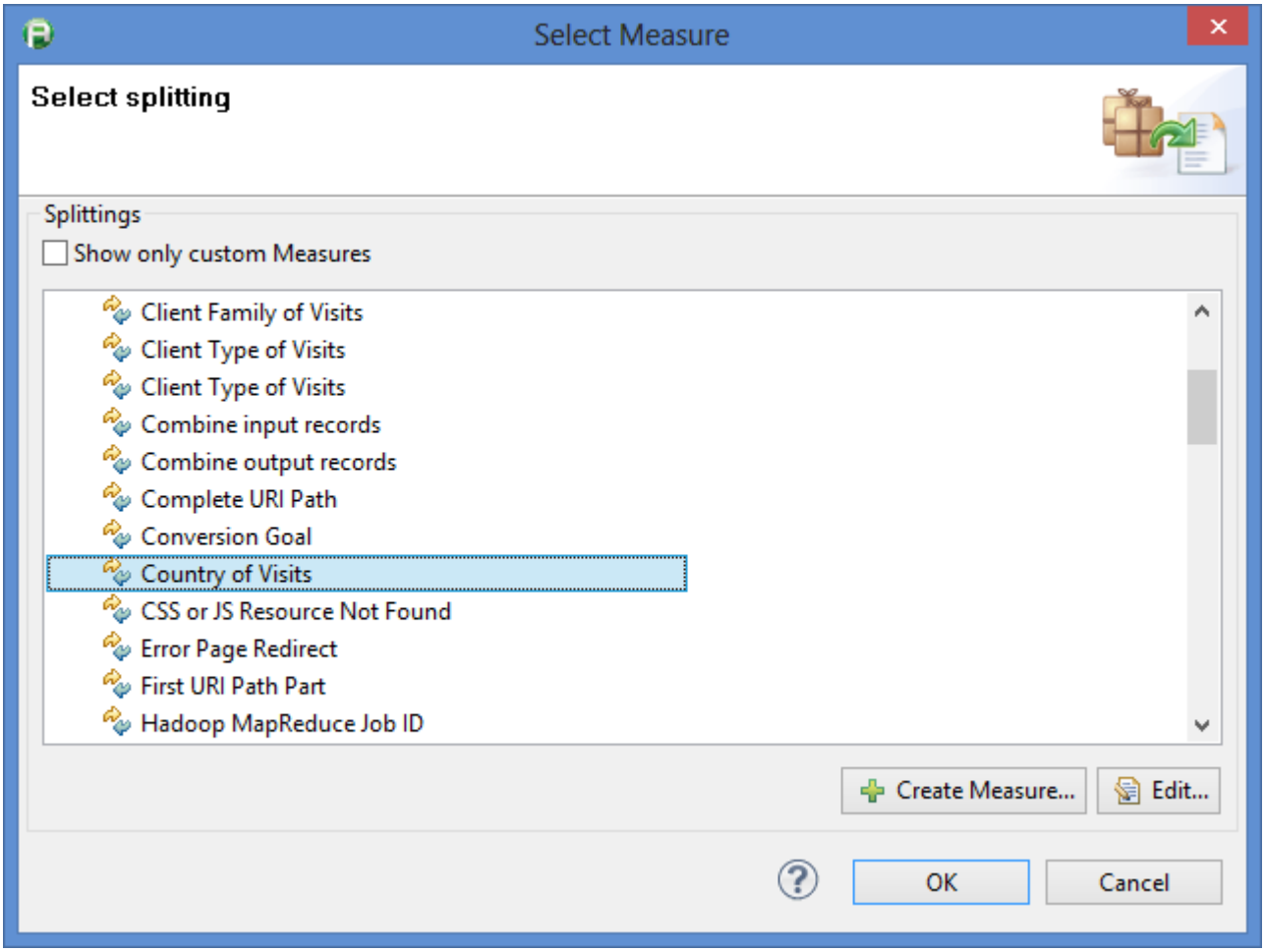
- IP addresses (that don't appear in the geographic locations database) directly with a city's location (as opposed to a country or region).
- Office-internal IP addresses with specific office locations, such as 10.0.0.0/8, 172.16.0.0/12 or 192.168.0.0/16.

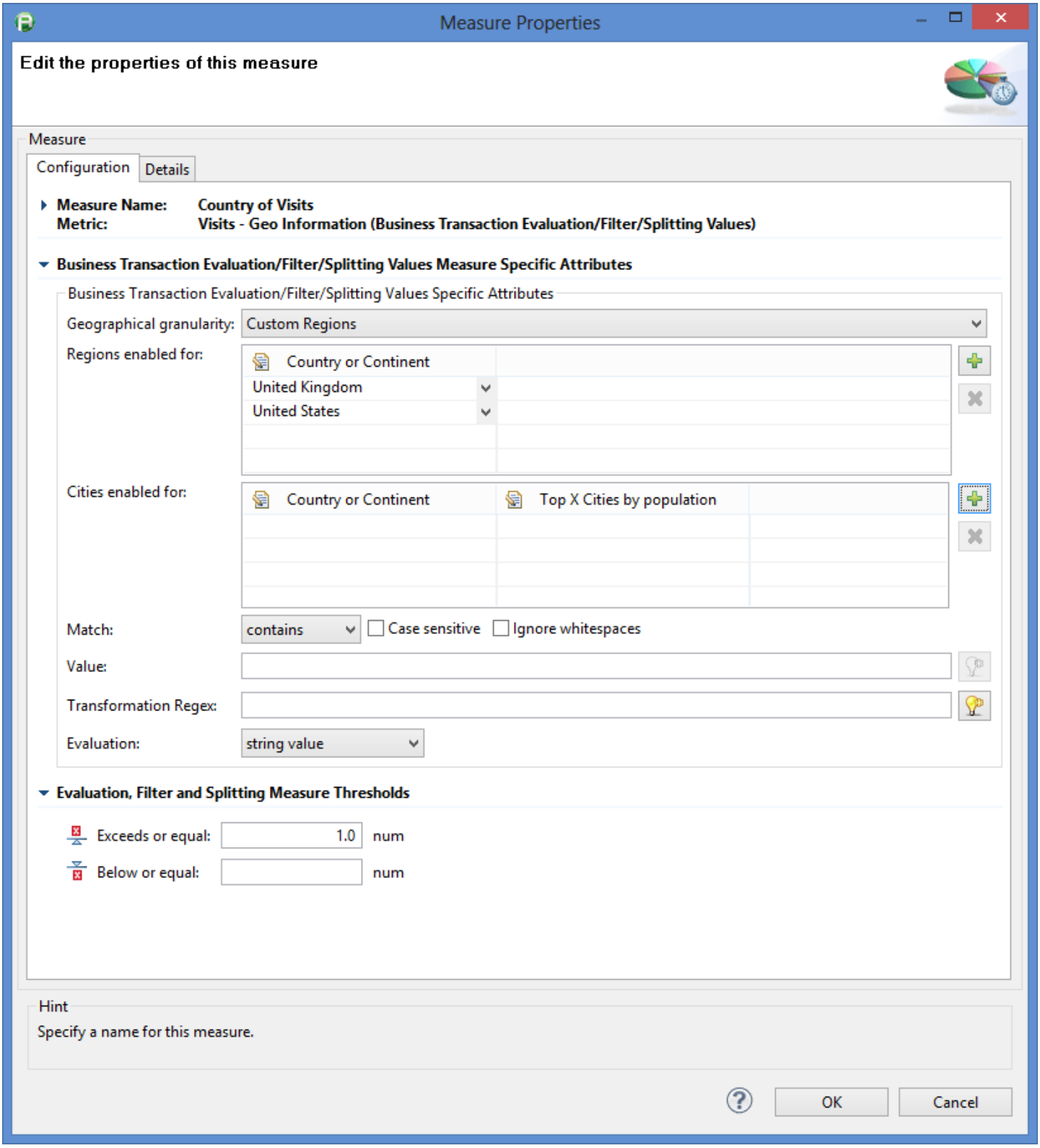
Do the following to define custom locations in the World Map.:

Edit the Business Transaction corresponding to the app's System Profile

Edit the System Profile for your given application by double-clicking it in the Cockpit. Select **Business Transactions** vertical tab and expand **Built-in transactions**. Double-click the option **Visits by User Experience and Count**. Dismiss "Do you really want to edit the Business Transaction?" with Yes.


Business Transaction Editor
Configure Business Transaction.
General
Name: Visits by User Experience and Country
Rename...
Description:
Active
Results
Store results in Performance Warehouse
Export results via HTTP
Calculate Business Transaction per
Server-Side PurePath
User Action
Visit
Filter
Filter
Aggregate
Logic
Calculate Results
Visit Count
Results
Aggregate
Visit Count
sum
Split Results
Splittings
User Experience Index of Visits
Splitting Options...
No specific restriction set. Default restriction to 50,000 splittings applies.
Base Line Settings
Violation Detection Settings...
OK
Cancel


Select Measure
Select splitting
Splittings
Show only custom Measures
Client Family of Visits
Client Type of Visits
Client Type of Visits
Combine input records
Combine output records
Complete URI Path
Conversion Goal
Country of Visits
CSS or JS Resource Not Found
Error Page Redirect
First URI Path Part
Hadoop MapReduce Job ID
Create Measure...
Edit...
OK
Cancel

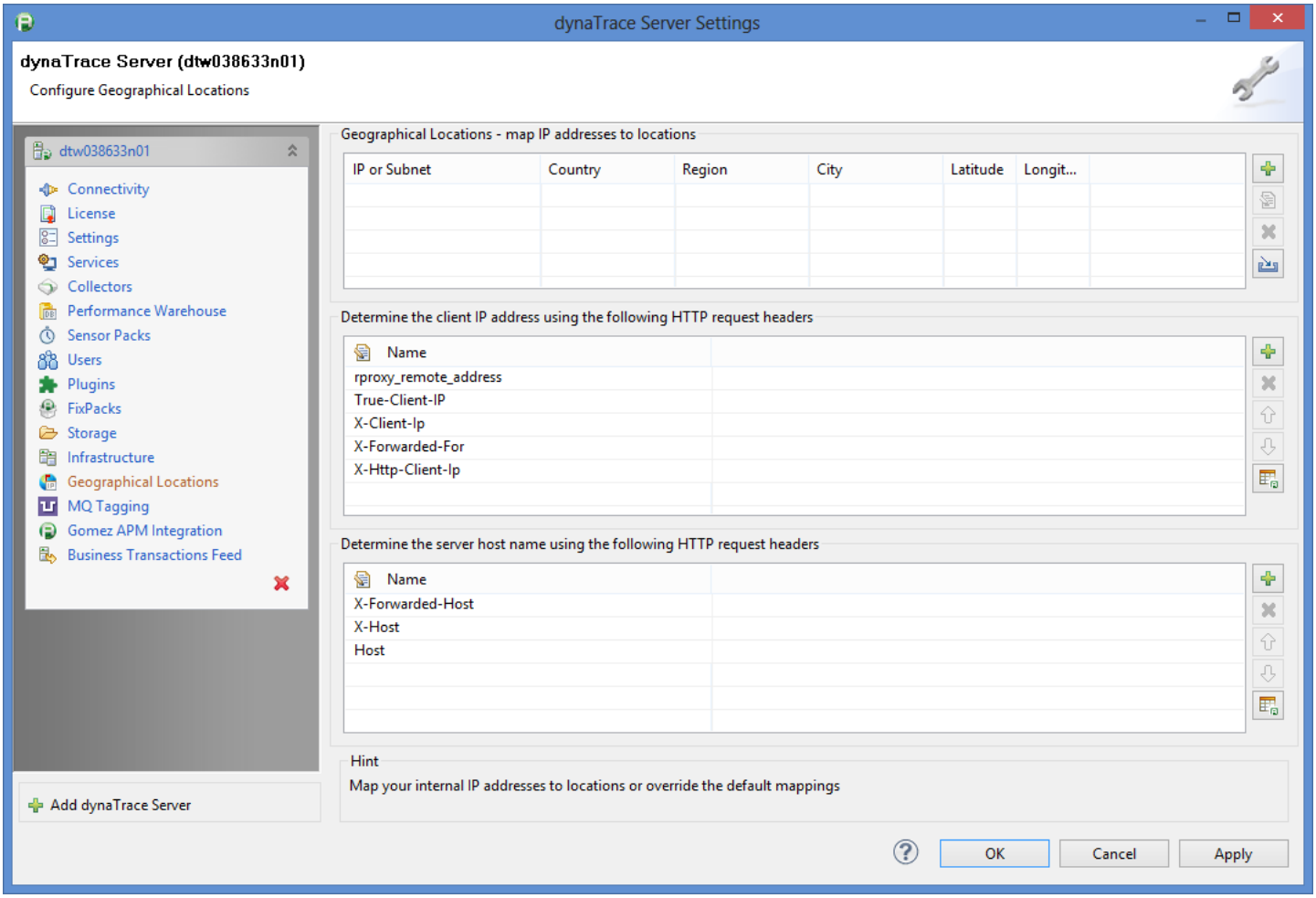


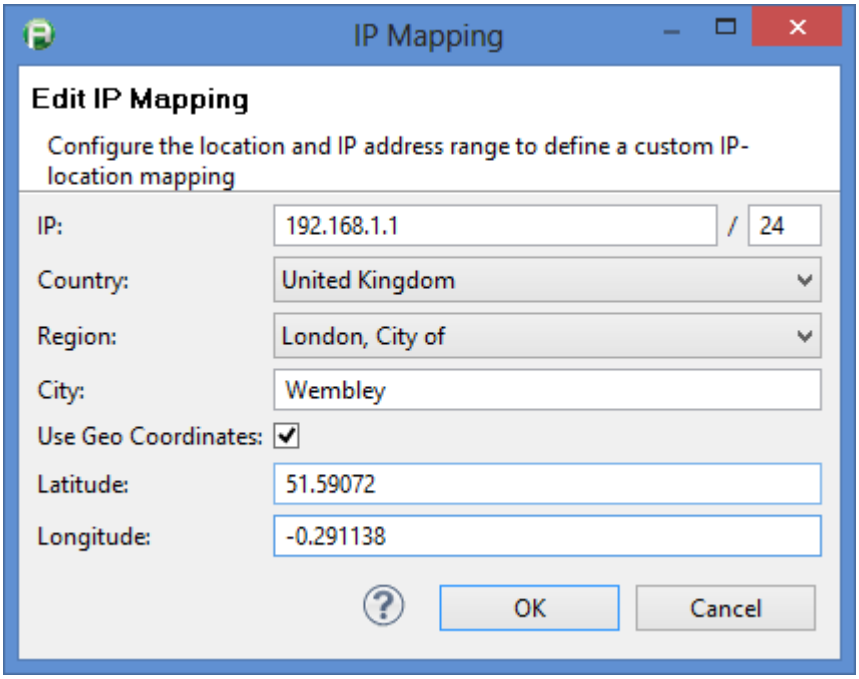


AppMon is now set up to resolve the selected countries to the city level.

Specify the IP address for a custom city

The next step is to define an IP address for a given city. Select **Settings > Dynatrace Server > Geographical Locations**.

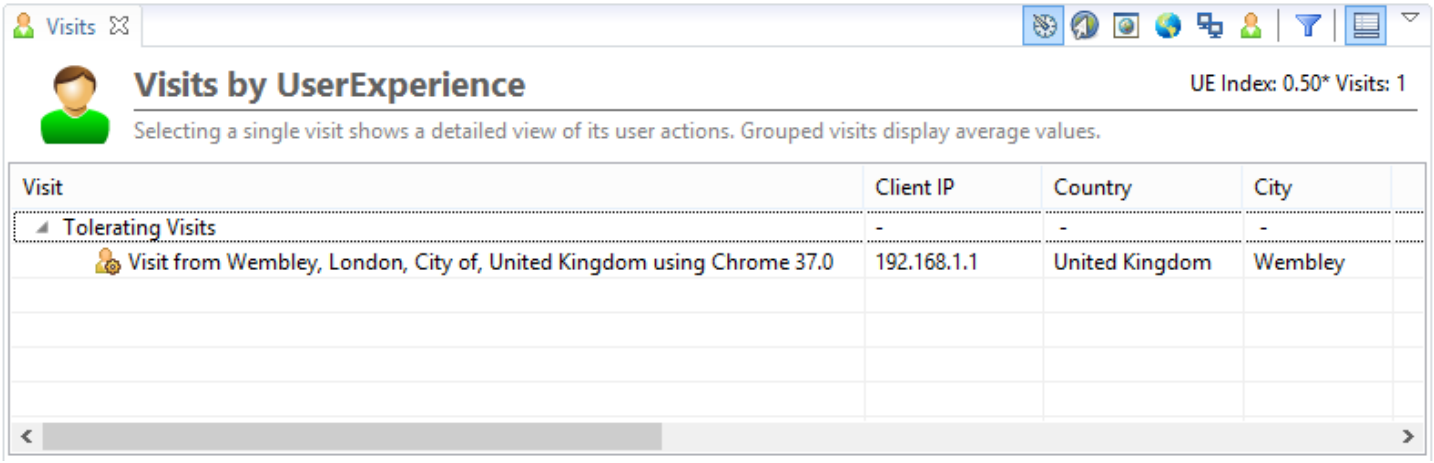




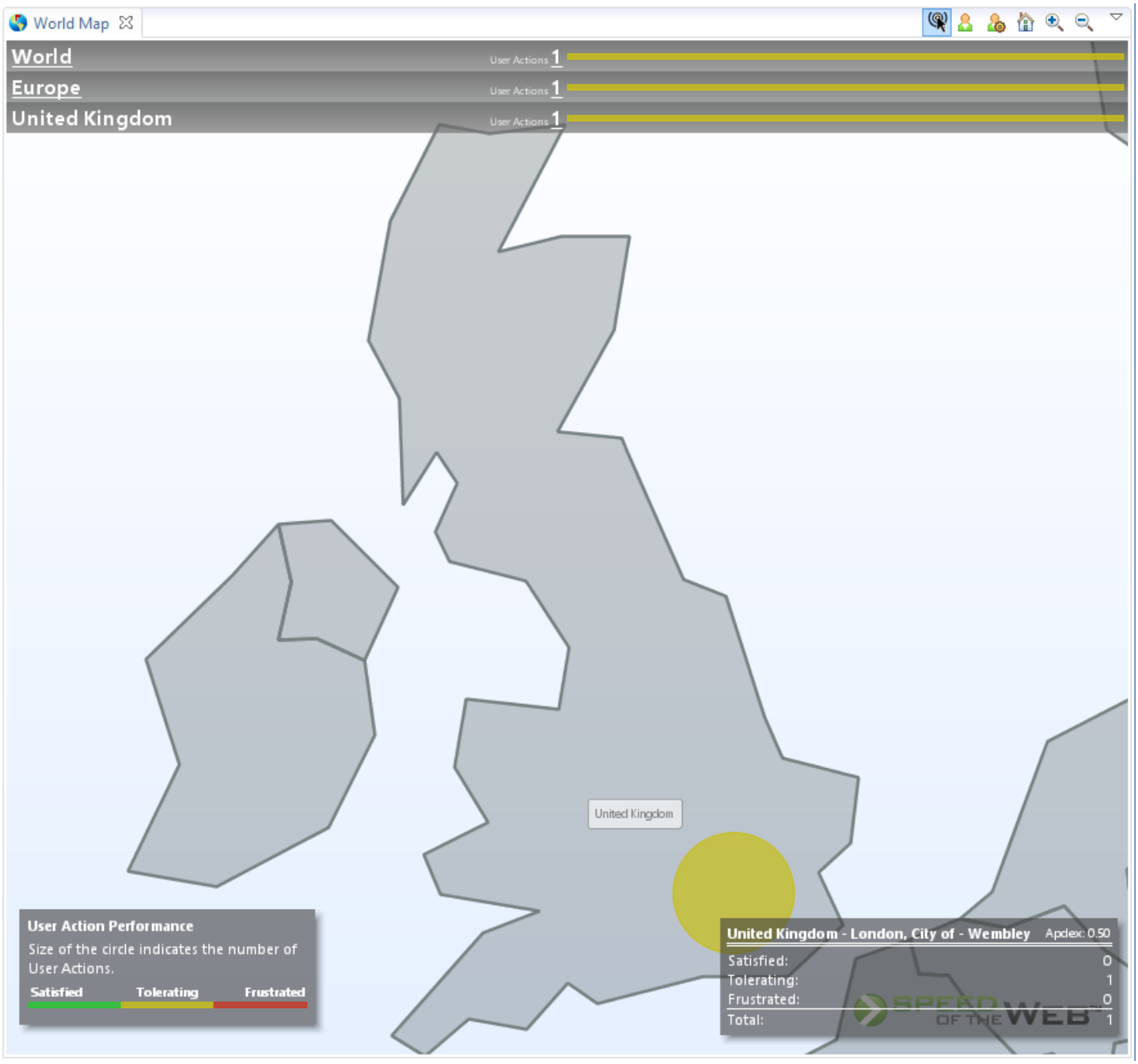
The **Geographical Locations - Map IP addresses to locations** table now summarizes the custom cities.

Viewing visits and the world map

When a visit occurs from the given IP address, the **Visits Dashlet** displays the custom definition.



<System Profile> > Diagnose User Experience > World Map



The tooltip shows the visit has come from **Wembley** and the visit data point is at the geographic location.

Finding out the geographic location for a given point

You can use http://maps.google.com to go to the defined point (latitude and longitude) and right-click and select **What's here**.

Synthetic Monitoring configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

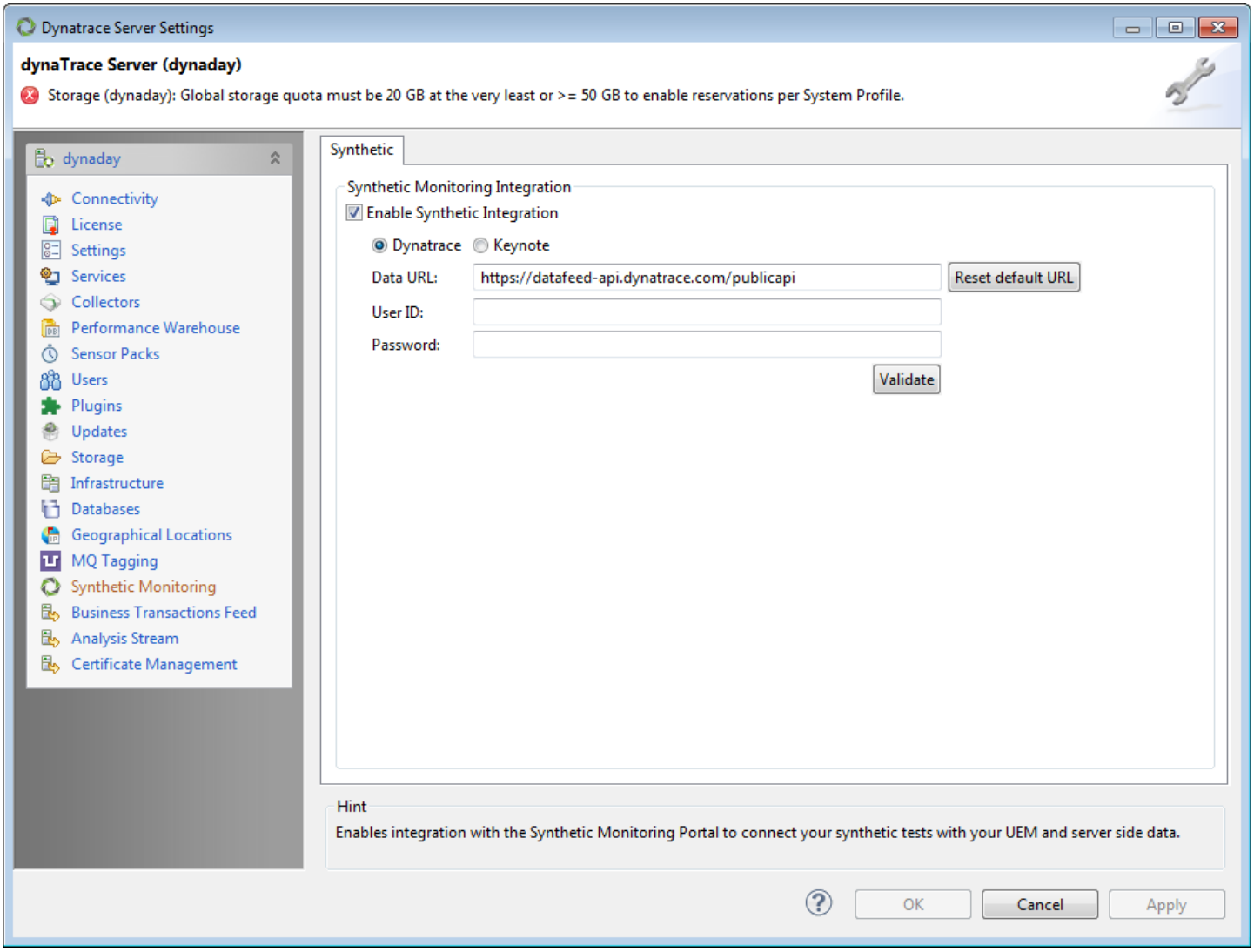
Related pages

Synthetic Monitoring integration

In the AppMon Client, choose **Settings > Dynatrace Server > Synthetic Monitoring** to configure the built-in integration with Synthetic Monitoring and Keynote Synthetic Monitoring.

Enabling Synthetic Test Integration for Application Monitoring

Use the **Synthetic** tab to specify the Dynatrace Portal or MyKeynote Portal connection details required to view synthetic tests in AppMon Web.



Server Settings Synthetic Monitoring pane

To enable Synthetic Test Integration for Application Performance Monitoring:

1. In the AppMon Client, select **Settings > Dynatrace Server**.
2. Select **Synthetic Monitoring**.
3. Select **Enable Synthetic Integration** and select the appropriate button depending if you are setting Dynatrace Synthetic or Keynote Synthetic Monitoring credentials. The default data URL for the applicable Synthetic Monitoring portal is displayed. You can change the URL if needed for a different server.
4. Enter your Dynatrace Portal or MyKeynote Portal login credentials.
5. Click the **Test connection** button to validate the Synthetic Monitoring authentication settings.

server.config.xml

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Server Settings table

Repository configuration

Generating the default configuration file

This page describes the commonly adjusted values in the AppMon Server configuration file (`server.config.xml`).

> **Important**
>
> Do not edit the Server configuration file while the Server is running.

To explain the context of the **XML paths** in the Server Settings table below, the following XML code snippet shows the XML path `serverconfig/settings/server` with its attribute **name**. All XML paths are relative to the containing root element **dynatrace**.

```
<dynatrace version="<version>">
  <serverconfig memento.version="<version>">
    ...
    <settings>
      <server ...
       name="hostNameLowerCased" ... />
    ...
    </settings>
  ...
```

Server Settings table

The following lists the commonly adjusted values in `server.config.xml`:

XML Path	Attribute Name	Default Value	Descript
serverconfig/collectorconfig	agentport	9998	The Co specifie Agent c Restart making
serverconfig/directories/storedessions	path	sessions/stored	The ses the spe is relati installat
serverconfig/directories/temp	path	temp	Tempor in the s which is AppMo folder. F after ma
serverconfig/directories/reports	path	reports	Tempor automa reports interfac are sto folder, v the App folder. F after ma
serverconfig/settings/collector	allowcollectorconnections	true	Allow C connec
serverconfig/settings/collector	requiressl	false	Require
serverconfig/settings/collector	collectorport	6698	The Se specifie Collecto connec Server change
serverconfig/settings/collector	collectorsslport	6699	The Se specifie Collecto connec Server change
serverconfig/settings/client	clientsslport	2021	The Se specifie Client c the Ser any cha

serverconfig/settings/client		clientallowsavecredentials	true	Enable “Remer option a Disablir client fr passwo
serverconfig/settings/server		name	<hostnameLowercased>	Server with low hostnar change make su does no charact Restart making
serverconfig/settings/server		continuoussessionrecording	false	Enable continu recordi
serverconfig/settings/server		jmxexportmanagement	true	Activate additior this Ser some S JMX.
serverconfig/settings/server		jmxexportmonitoring	true	Activate JMX inte Server. interfac Server monitor standar interfac
serverconfig/settings/server		jmxport	1099	The JM
serverconfig/settings/server		tunnelenabled	true	Allow H connec
serverconfig/settings/server		tunneladdress		The Se specifie incomin connec Server change
serverconfig/settings/server		tunnelport	8023	The Se specifie Client c the Ser any cha
serverconfig/settings/server		selfmonitoringenabled	true	Self-mo Server Perform and lets Server

serverconfig/settings/server	agentbasedselfmonitoringenabled	true for memory > demo	Agent-b monitor AppMo AppMo the usu AppMo enablin diagnos
serverconfig/settings/oopanalyzer	address		Defines AppMo Server Memory Process
serverconfig/settings/oopanalyzer	port	7788	Defines the App Analysi
serverconfig/settings/oopanalyzer	logpath	../log/analysisserver	Log pat process Server.
serverconfig/loggingconfig	append	true	Whethe log line
serverconfig/loggingconfig	console	INFO	Log lev output.
serverconfig/loggingconfig	level	INFO	Log lev
serverconfig/loggingconfig	maxfiles	5	Maximu and au
serverconfig/loggingconfig	path	../log/server	The log specifie relative installat
serverconfig/loggingconfig	maxbytes	10485760 (1MB)	The ma log/aud
serverconfig/http	externalhostname		Lets yo server- the web remote used fo dashbo other pl the Ser This is It is onl Server with the under C

serverconfig/http		httpport	8020	Enter th service listened request request ignored number
serverconfig/http		httpsport	8021	Enter th service listened request request ignored number
serverconfig/http		requiressl	false	Require
serverconfig/http		webservicesenabled	true	Enables services informa system informa some S
serverconfig/http		webserverenabled	true	Enable
serverconfig/http		externalhttpsport		Defines port to a service machin This se Server system configu differen than `ht` forward web ser links us configu
serverconfig/http		externalhttpport		This op differen access remote network notifies operati is confi differen than `ht` forward web ser links us configu

serverconfig/http	webserviceusertimeout	-1 (means disabled)	A secur a timeo how lon server v be idle session which th authent is enab REST c this ser header `UserP` `false` Otherwi clients a Unauth even if data ar request
serverconfig/http	uniquewebservicerealm	false	A secur prevent to reus authent protecte server v set to `t` generat authent every ti an auth from the
serverconfig/webstart	webstartenabled	true	Enables Java W which y a Web Client d Server.
serverconfig/webstart	tunnelenabled	false	The Cli HTTP t commu setting configu HTTP t to conn proxy.
serverconfig/webstart	tunneladdress	localhost	Enter th address tunnel e
serverconfig/webstart	tunnelport	8023	Enter th with tun

serverconfig/webstart	proxyenabled	false	The Cli CONNE create connec using a this set connec proxy.
serverconfig/webstart	proxyaddress		Enter th address
serverconfig/webstart	proxyport	-1	Enter th proxy.
serverconfig/webstart	proxyauthenabled	false	Whethe (true) o anonym
serverconfig/webstart	proxyauthusername		Enter th authent
serverconfig/webstart	proxyauthpassword[1]		Enter th authent
serverconfig/webstart	anonymouswebstartenabled	true	Control web inte authent or can l Websta authent
serverconfig/oopanalyzerconfig	address	localhost	Defines Memory address Snapsh Process
serverconfig/oopanalyzerconfig	port	7788	Defines Memory listening
serverconfig/mailserver	address		The hos address server t You mu email se emails u protoco
serverconfig/mailserver	port	25	Specify the ema configu email. T 25, but differen environ

serverconfig/mailserver	ssl	false	An SMT support encrypt using S SMTP s requires commu
serverconfig/mailserver	userid		Specify is used on the empty i does n authent
serverconfig/mailserver	password[1]		Specify is used on the empty i does n authent
serverconfig/mailserver	subject	[dynaTrace]	Specify to the s email th
serverconfig/mailserver	bounce		Specify to which email sl (option the ema delivere recipier email a correctl is sent informa email c delivere copy of Usually either r adminis adjustir of the e the per AppMo adjustir addres recipier email d addres recipier delivery Failures report a situatio the Ser
serverconfig/mailserver/from	email		Email a appear of the a the Ser

serverconfig/mailserver/from	type	Unresolved	If the **e** configu change
serverconfig/webui	webuihttpsport	9911	The po listens HTML a REST r
serverconfig/webui	webuienabled	true	Enables AppMo
repositoryconfig	querytimeout	7200 (2 hours)	The ov queries
repositoryconfig/repository/database	dbms	embedded	See Pe Wareh
repositoryconfig/repository/database	name	dT4 for DB2, dynaTrace4 for all others	Specify name. exist an
repositoryconfig/repository/connection	host		Specify the dat
repositoryconfig/repository/connection	port	Oracle: 1521, DB2: 50000, SQLServer: 1433, PostgreSQL: 5432	Specify
repositoryconfig/repository/credential	user		Specify
repositoryconfig/repository/credential	password[1]		Specify passwo
repositoryconfig/repository/connection	useurl	false	Direct e URL (O
repositoryconfig/repository/connection	url		Connec only).
repositoryconfig/repository/connection	connectonstartup	true	Connec

[1] For migration, starting the Server transfers the passwords to the keystore and the field (including the identifier)is removed from the configuration file.

Repository configuration

The following is a list of the possible values for attribute **dbms** under XML path `repositoryconfig/repository/database`:

DBMS	Description
embedded	Embedded Apache Derby database. For production, install an external database such as Oracle, DB2, or SQL Server.
Oracle	For integration, an Oracle database must be created and configured (user, password) by the database administrator.
DB2	For integration, a DB2 database must be created and configured (user, password) by the database administrator.
SQLServer	For integration, a SQLServer database must be created and configured (user, password) by the database administrator.
PostgreSQL	For integration, a PostgreSQL database must be created and configured (user, password) by the database administrator.

Generating the default configuration file

On the first startup, the Server detects various settings and generates the initial configuration file. Typically you do not have to modify this file. To generate a configuration file without starting the Server, you can define a command line property, as shown below:

```
# ./dtserver -Dcom.dynatrace.diagnostics.server.init_config_and_exit=true
```

Backup of AppMon settings and data

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

AppMon Server and Collector settings and data backup

Optional session store and Performance Warehouse backup

Related pages

Migration tool details

Migration tool syntax

Performance warehouse migration troubleshooting

This page describes AppMon backup and disaster recovery. Backing up AppMon settings and data is simple compared to most enterprise software data backups.

AppMon Server and Collector settings and data backup

Backup

1. Ensure that all components in the directory to be backed up are shut down, including the Server, Frontend Server and Collector.
2. Use the dynatrace-migration tool to backup essential Server and Collector settings such as `.ini` and configuration files, and data such as Collector caches, dashboards, and custom Sensor packs.
3. Create an archive using the first step in Services Migration as a guideline.

```
java -jar <path>/dynatrace-migration.jar -migration -sourceDTHome "<DT_HOME>" -targetArchiveDir "<archiveDir>"
```

A file named `<Server_name>_<creation_dateTime>.dtma` (actually a `.zip` file with all the backup items) is created in `<archiveDir>`.

If you have a distributed architecture, please repeat this with `<Collector_home>`.

Restore

If you need to restore a backup, use the second step in Services Migration as a guideline. Install the Server and / or Collector component and restore settings and data with the following command.

```
java -jar <path>/dynatrace-migration.jar -migration -sourceArchive "
<archiveDir> /<archiveFile.dtma>" -targetDTHome "<DT_HOME>"
```

Optional session store and Performance Warehouse backup

Session Store file backup is normally not needed. The data is very dynamic, of limited historical value, and the files are potentially large.

The situation is similar for the Performance Warehouse database, though older data in the Performance Warehouse is aggregated for better insights in the developments over time, which may warrant backup.

An AppMon upgrade may be the situation with the best argument for a backup if you want to keep your data, or be able to roll back in case something goes wrong. Otherwise the Session Store starts from scratch and a new Performance Warehouse database must be created.

Performance Warehouse administration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What's new and upgrade recommendations

Data stored in the Performance Warehouse

Supported databases

Embedded Performance Warehouse usage restrictions

Performance Warehouse resolution

Performance Warehouse configuration

Backups

Usage of PostgreSQL

FAQs / issues

Related pages

Upgrade and migration guide

Performance Warehouse partitioning for Dynatrace

Getting support

Watch and learn

Configuring the Performance Warehouse

External links

PostgreSQL website

What's new and upgrade recommendations

An upgrade automatically occurs if needed after you connect to the Performance Warehouse. You must have appropriate permissions for the upgrade, since new columns and tables are introduced. See the Upgrade and Migration Guide or contact support for more information on upgrades.

If migrations are pending, AppMon does not automatically connect to or upgrade the AppMon Performance Warehouse when starting or restarting the AppMon Server. Pending migrations occur when there is a major or minor version upgrade to the AppMon Server.

> **Note**
>
> You must re-enter your DB-connection password after migrating to a newer AppMon release.

Data stored in the Performance Warehouse

Use the Performance Warehouse for long term storage of measurements, incidents, thresholds, and tests. The Performance Warehouse does not store PurePaths, or memory and thread dumps. See Test Automation Explained) and their configuration data for more information.

Supported databases

The AppMon Performance Warehouse uses a relational database to store long-term historical data. By default, AppMon installs and uses an embedded database for demo/testing purposes. Any production installation of AppMon **must use one of the following** database management systems **with the specified version** to host the Performance Warehouse database:

- Oracle 10g/11g/12c
- Microsoft SQL Server 2008 / 2012 / 2014 / 2016
- IBM DB2 Version 9.7 / 9.8 / 10.1 / 10.5 / 11.1
- PostgreSQL 9.2 / 9.3 / 9.4 / 9.5 / 9.6

AppMon supports partitioned tables for high-load scenarios for all supported databases. For all

AppMon supports partitioned tables for high-load scenarios for all supported databases. For all AppMon installations **greater than Large**, a partitioned database **is mandatory**. You can find more details in Performance Warehouse Partitioning for Dynatrace.

To determine disk-space requirements for the AppMon repository upfront, factor 15GB per 1,000 persistent measures (of all system profiles within the AppMon repository). You can find more details in the Deployment Guide.

See below for information on using the embedded Apache Derby database.

Embedded Performance Warehouse usage restrictions

You can use the embedded database on developer machines only, because this database runs in the server process and directly affects server performance.

The following table lists the enforced restrictions.

Editions (horizontal) versus Types (vertical)	Pre-Production / Test Center	Production
POC	✓	✓
Eval	✓	✓
Perpetual	✗	✗
Term	✗	✗

Recommended database settings

Embedded database

The embedded database is fully preconfigured. You don't have to change the settings.

SQL Server, DB2, Oracle, PostgreSQL, SQL Azure

Following database settings must be configured:

Following database settings must be configured:

Read/write permission: You must be able to read and write data.

Truncate tables: Table truncation must be allowed.

Create/drop tables, indexes: Schema modification rights like create table are required during installation or migration from an earlier version. During operation it is optional.

Create and execute stored procedures: Creation and execution of stored procedures is required during installation or migration from an earlier version. During operation it is optional.

UTF 16 compatible character set / char set / code page / codepage: A non UTF16 compatible charset ("Test Connections" checks compatibility) may cause problems Measure names which use the full UTF16 char set spectrum. See Database Not Fully utf16 Capable and Default Char Set utf8 below. Don't use special characters and accent marks. Use standard English alphabet characters, digits, and underscores for names.

Page Size 16k: Necessary on DB2. Create the database with the following command:

```
CREATE DB database_name PAGESIZE 16384
```

User Temporary Tablespace: Necessary on DB2. Create this with the following command:

```
CREATE USER TEMPORARY TABLESPACE user_temporary1 MANAGED BY AUTOMATIC STORAGE;
GRANT USE OF TABLESPACE user_temporary1 TO USER DYNATRACEUSER;
```

Collation: Create the database with insensitive collation.

> **Tip**
>
> For a database admin who wants to fine tune the database, it is important to know the typical behavior of the Performance Warehouse.

Measurement writing supports high load environments. Data is inserted once per minute in one of the

Measurement writing supports high load environments. Data is inserted once per minute in one of the measurement_temp tables. AppMon inserts in a temp table for 30 minutes, then switches and moves the data for the now inactive temp table to the measurement_high table.

Keep the statistics (and indexes) up to date for the following tables (for databases other than embedded):

- measurement_temp1
- measurement_temp2
- measurement_high
- measurement_mid
- measurement_low
- percentiles_high (hourly inserts)
- percentiles_low (daily inserts)
- incidentrecords
- dynamic_measure
- test_expectation (for usage of Test Automation Explained only).

Performance Warehouse resolution

Due to performance reasons the measurement data is held in the following different resolutions:

- High resolution: All data received within a minute is aggregated to a single data point.
- Mid resolution: All data received within an hour is aggregated to a single data point.
- Low resolution: All data received within a day is aggregated to a single data point.

> **Note**
>
> AppMon calculates and stores minimum, maximum, and average values for the aggregation interval in a single data point.

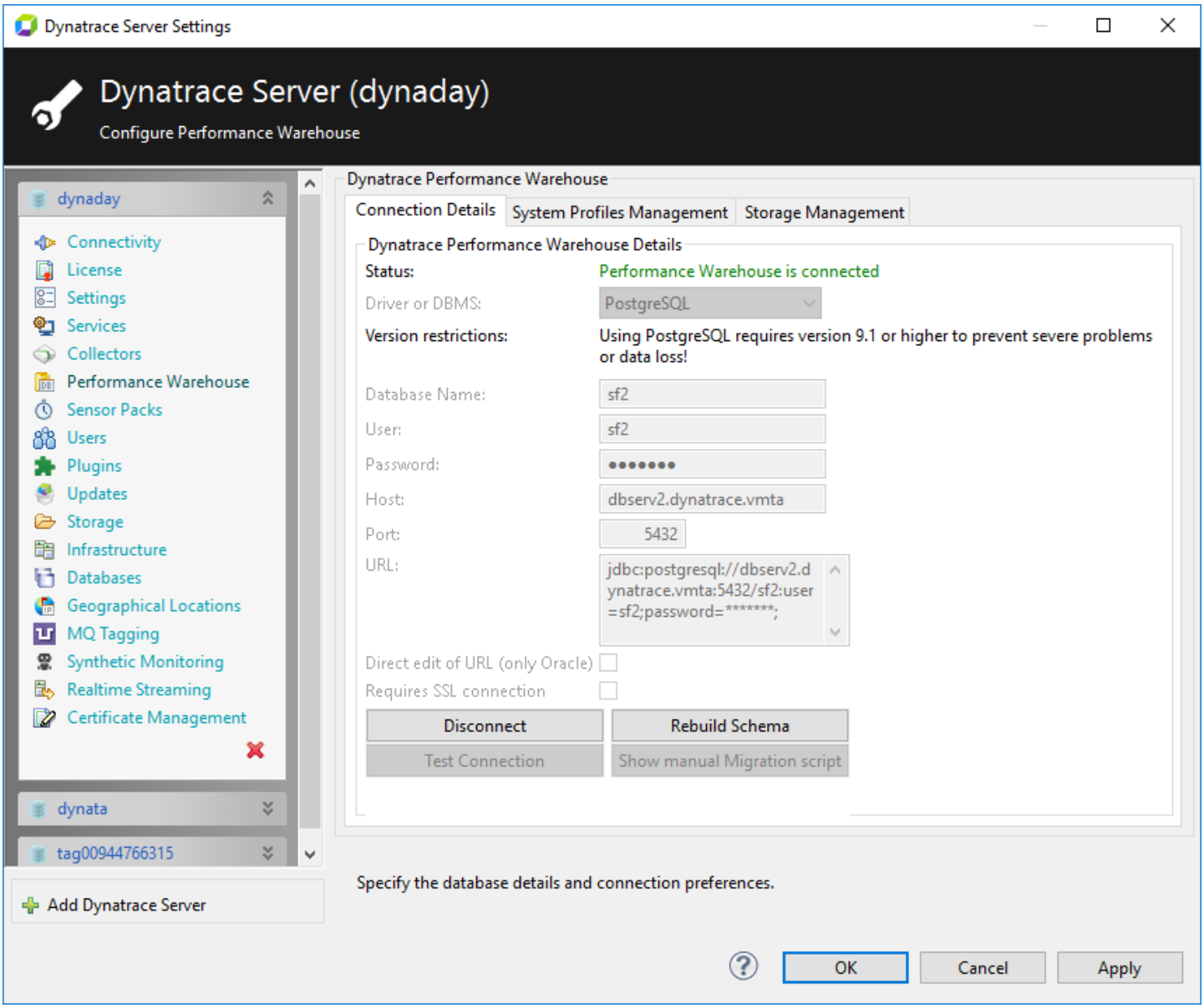
Due to the different nature of percentiles, the data is held only in the two following different resolutions:

- High resolution: All data received within an hour is used for calculating the percentile.
- Low resolution: All data received within a day is used for calculating the percentile.

Performance Warehouse configuration

Performance Warehouse configuration

Connection details



PostgreSQL is used as DB (server).

Use this dialog box to connect to a database and create a schema.

The default activated database is the embedded database. Use an external database server for storage of more than four days of recorded data. See Embedded Performance Warehouse Usage Restrictions for more information.

Using an external DB server

You must create the DB manually before you connect to the DB server or an error will occur. The

You must create the DB manually before you connect to the DB server or an error will occur. The **Database Name** field in the dialog box defaults to `dynaTrace4` (`dt4` in DB2), but you can use any valid DB name. If a schema is not available, the tables are created automatically at first connect. If the table/schema creation process fails, an error message appears.

How to create a schema

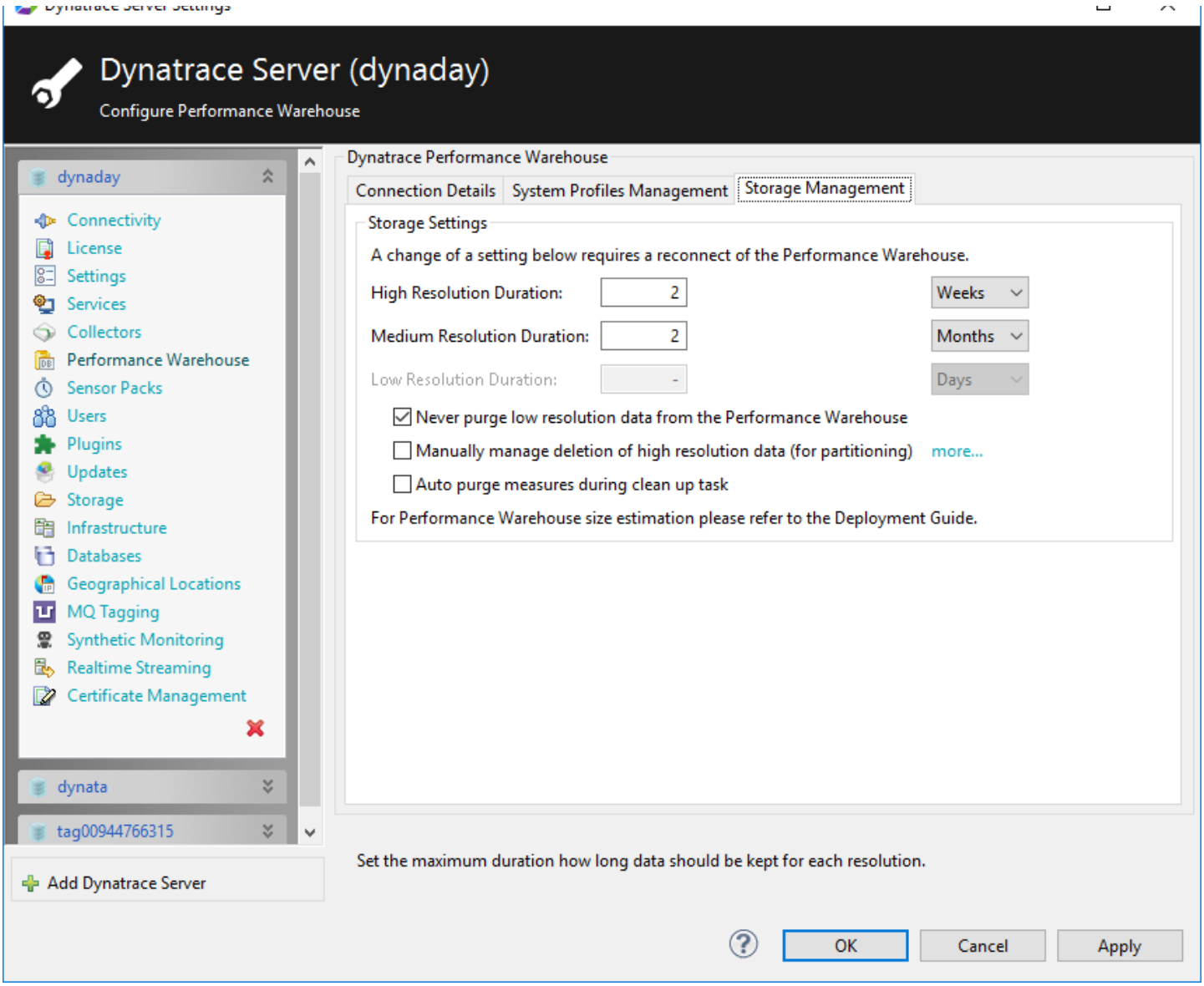
Ensure that the user has sufficient rights to create tables. If the schema is not created automatically at first connect, for example due to insufficient rights, you must manually create the schema process. Use the **Create Schema** button to do this.

Rebuilding a schema

This button only appears if the AppMon Server is connected to a Performance Warehouse with a valid schema. The schema is dropped and recreated which leads to a loss of all data in the Performance Warehouse.

Storage management





Resolutions or data aging

The duration is the time that the data is kept in the corresponding resolutions.

For example, a duration of one week for high resolution means that data with a resolution of one minute is dropped if it is older than seven days. A duration of two months for mid resolution denotes that data with a resolution of one hour is deleted if it is older than 60 days. A duration of one year for low resolution means that data with a resolution of one day is deleted after 365 days.

Partitioning

Enable partitioning to speed up read and write performance, and for high load scenarios. See Performance Warehouse Partitioning for dynaTrace

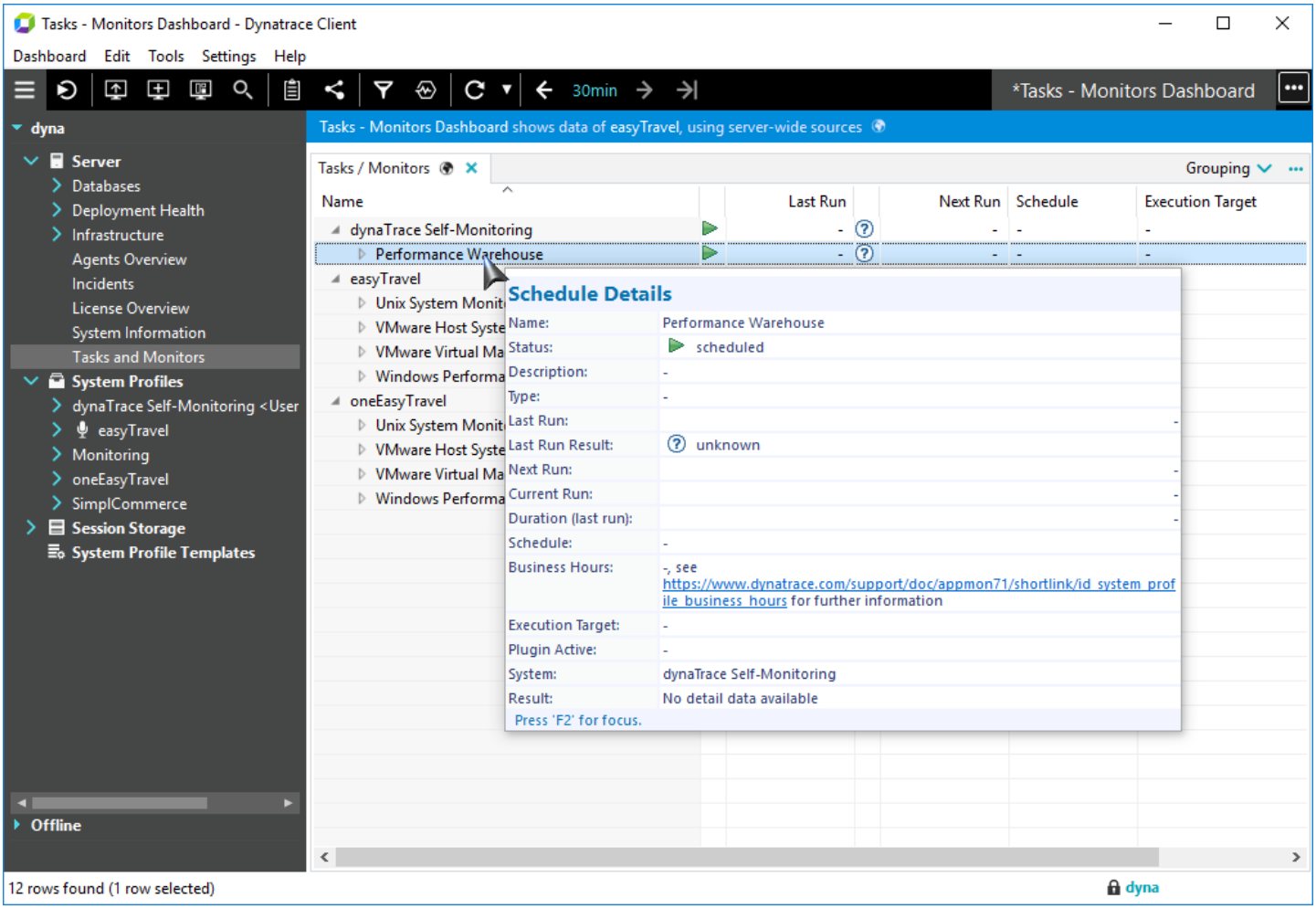
Auto purge

Auto purge

Cleanup deletes old dynamic measures from the Performance Warehouse. In other words, each dynamic measure for which no new measurements have been received as long as the duration of the mid resolution are purged from the Performance Warehouse. Auto Purge helps to clean up unused measures in case of measure explosions or misconfigurations. If you think you need to tweak AutoPurge further, contact support.

Performance Warehouse clean up task

Use the Clean Up Task to aggregate data for the different resolutions and delete data after the specified duration. You can also track and schedule data. To use this click **Cockpits > Status Overview > Tasks and Monitors**.



Schedule these tasks during times when the load on AppMon Server and database is expected to be low.

If the amount of data in the Performance Warehouse is expected to be very high, try to run the clean up

If the amount of data in the Performance Warehouse is expected to be very high, try to run the clean up task at least once a day so that the DB server works on smaller amounts of data, which results in a lower load.

If you do not execute the task, aging is not performed and data is kept in high resolution only and is never purged.

See the Deployment Guide for information on disk space requirements.

Recommend settings for the different durations

When you set durations you may have to make a tradeoff between performance and granularity. The longer the higher resolution is kept in the Performance Warehouse, the more the performance is negatively affected. The formulas below provide insight into the expected amount of storage in the different tables and their impact on performance:

An active Measure is one for which **measurements are actually taken**. The expected number of active Measures is only estimated. It depends on the amount of subscribed Measures and how many of them are configured to create dynamic Measures.

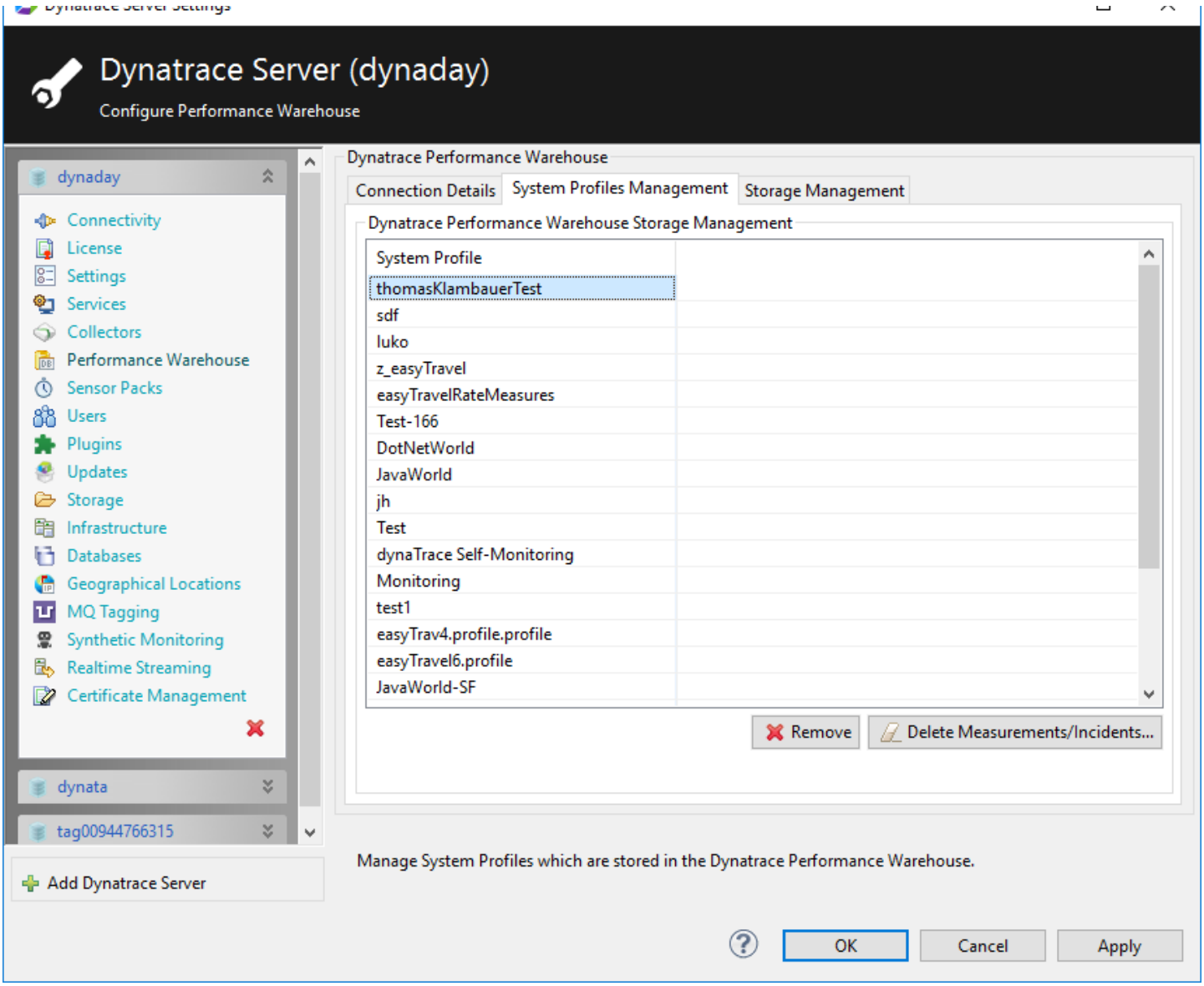
1000 active measures example:

Resolution	= 1000 DB Entries per	Duration of Storage	= DB Entries per Day	= DB Entries per Resolution
High	minute	1 week	1,440,000	10,080,000
Medium	hour	3 months	24,000	2,160,000
Low	day	1 year	1,000	365,000

The more high resolution data kept in the Performance Warehouse, the bigger the impact on performance.

System Profiles management





Use this dialog box to remove measurements and incidents for System Profiles manually or remove System Profiles altogether. If a System Profile is deleted, the Performance Warehouse access should also be disabled. If not, AppMon synchronizes and writes the System Profile in the Performance Warehouse the next time a System Profile is saved.

> **Note**
>
> Do not delete a specific time span or System Profiled, or perform a calculation about the usage percentage of each System Profile in a production environment. These operations can lock the database.

Backups

Backups

The AppMon Server does not back up the database. You have to schedule backups manually on the database server itself.

Usage of PostgreSQL

Download

Download PostgreSQL from http://www.postgresql.org/.

Installation

On Windows, execute the setup file. For performance and stability reasons the data directory should be on its own disk. Use the default locale. You do not have to install Pg/PLSQL for the template database.

Schema/user creation for Performance Warehouse

If the PostgreSQL database instance is only used for the Performance Warehouse and other applications do not perform write operations on this instance, generate a user with read/write access for the public schema, or use the postgres default user. If you don’t do this operation, you should delete the public schema and create one for each user / role.

PostgreSQL clustered indexes

As of Fall 2015, PostgreSQL does not support the automatic reorganization of clustered indices. To ensure the performance, you must schedule the cluster command manually.

> **Note**
>
> The cluster command locks the affected tables. Do this during maintenance intervals when the Performance Warehouse is not connected.

As a workaround, you can use an application runnable by the pgAgent job scheduler for Postgres, which

As a workaround, you can use an application runnable by the pgAgent job scheduler for Postgres, which performs this reorganization task online, but uses almost double the storage. See Script pg_reorg for more information.

Setting statement timeout

Set the statement timeout on the Postgres database itself to prevent blocking database operations.

FAQs / issues

AppMon Server to PWH DBMS to DB relationship

Be careful with semantics. (**DBMSes** / Database Management Systems / database servers using SQL as query language like Oracle, MS SQL Server, IBM DB2, PostgreSQL,… each supporting many *databases* in one instance).

You **may** connect each AppMon Server to a separate / differently named database, where all databases may be located on the same DB server instance, as long as your DB server can take the load. Though you may want to consolidate data or reports from AppMon Servers covering different geographical regions, you **must not** connect more than one AppMon Server to the same (named) database on the same DB server instance. This causes unintended and possibly grave results with your data. Current consolidation options are:

- Connect the stand-alone AppMon Client to your Servers, report and put the reports side-by-side.
- Consolidate the data from different databases using your DBMS.
- Export your data to a central analysis server.
- Stream the data to ElasticSearch and then chart.

Consult your Dynatrace sales representative for further details.

Note

If making any changes to `dtserver.ini` or `server.config.xml`, chances are that you need to apply them to `dtfrontendserver.ini` or `frontendServer.config.xml` as well, because the Frontend Server has its own (read-only) connection to the PWH. For example, Windows authentication for SQL Server needs changes in both `.ini` files, as described below.

Database not fully utf16 capable

The test connection message reports that the database is not fully utf16 capable. In most cases, this doesn't matter, but if you're planning to use measure names, session names, or other data that uses the full spectrum of utf16 char set, the measure names may not store properly in the Performance Warehouse.

Default char set utf8

Test Connection reports that the database is fully utf16 capable, but the default char set of the database is set to utf8, or a different single-byte character set.

The utf16 capability test is very simple. AppMon creates a temporary table. It writes strings from the "upper" sections of the Java character set into this table and reads them. If the strings are equal, the Performance Warehouse stores every string and does not lose information.

User rights

You must be able to select, insert, delete, update, and truncate on all tables that AppMon creates. Schema modification rights (like create table) are only necessary during installation or migration from an earlier version.

Database connection down

The Performance Warehouse tries to reconnect, and measurements are buffered for a maximum of 11 minutes. Therefore, short database outages do not affect daily business.

Timeout occurred - some statements did not complete

As a default, the Performance Warehouse implements a timeout for the SQL Server, Oracle, and DB2

As a default, the Performance Warehouse implements a timeout for the SQL Server, Oracle, and DB2 databases, after which, a statement is automatically canceled. This can happen if some database management operation is running on the tables used by the Performance Warehouse, and the AppMon Server tries to do a cleanup. The default value is two hours, but you can change it in `server.config.xml`, in the `<Repository Config>` section.

As of Fall 2015, the PostgreSQL JDBC driver does not implement the query timeout, so the timeout must be set explicitly in the database with the following code:

```
ALTER DATABASE <databasename> SET statement_timeout=<timeout in seconds>;
```

Does the Performance Warehouse create a new table space?

It only creates the required tables in the DB.

SQL database default ports.

The default ports are SQL Server 1433, Oracle 1521, DB2 50000, and PostgreSQL 5432.

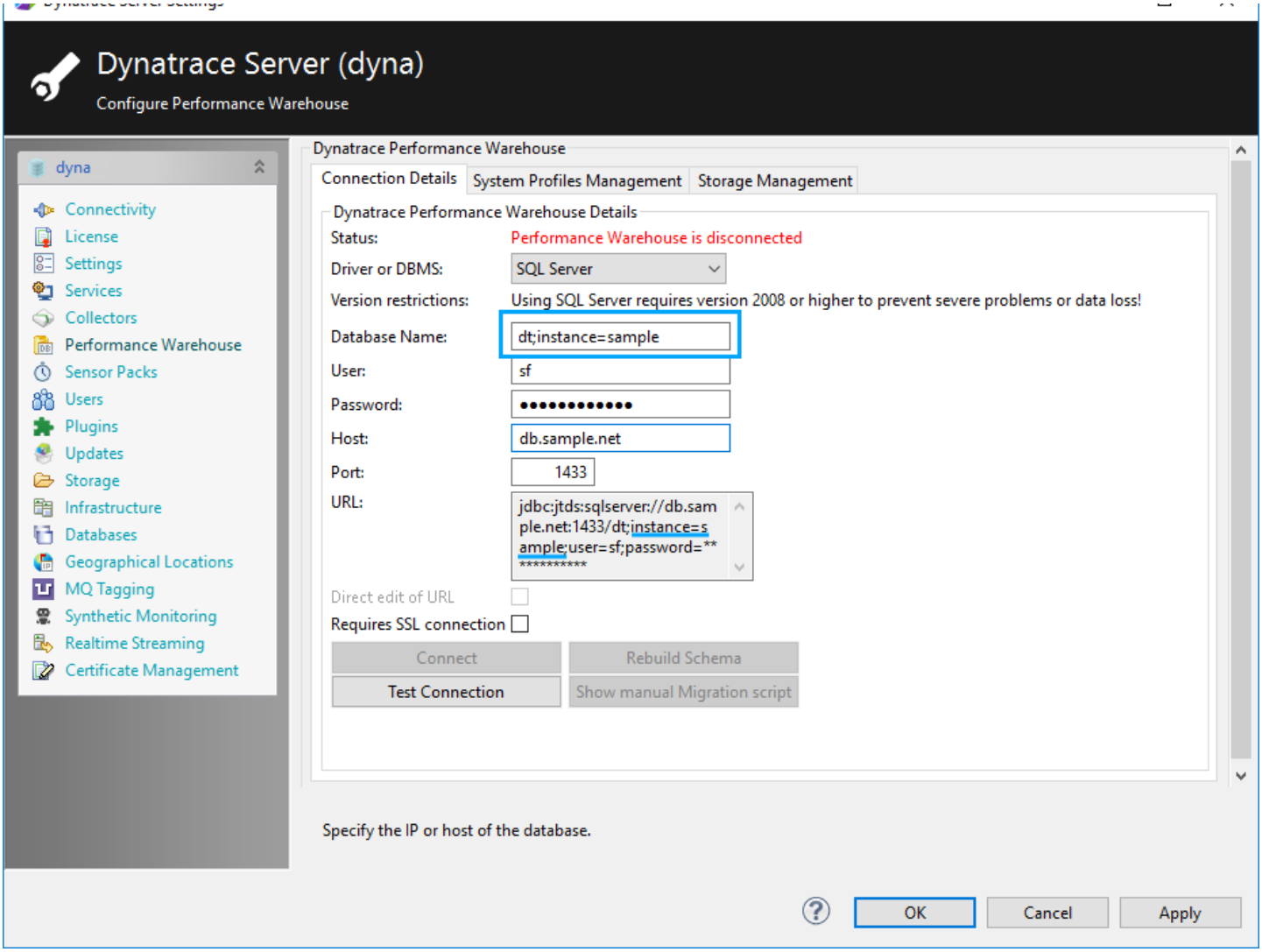
How to connect to a SQL Server instance?

SQL Server instance with static port

To connect to a specific instance of Microsoft SQL, add `;instance=<instancename>` to the Database Name in the Client and choose **Settings > Dynatrace Server > Performance Warehouse > Connection Details**. The valid string for Database Name and the database instance `SQLEXPRESS` would look like `<databasename>;instance=SQLEXPRESS`. The named parameter `instance` is added to the connection string.

Leave out the host portion in the instance name. In the example below, the host's name is `db.sample.net`. A host/instance combo would be `db.sample.net\sample`.





Instance name concatenated to DB name; (static) port 1433 SQL Server normally connects on

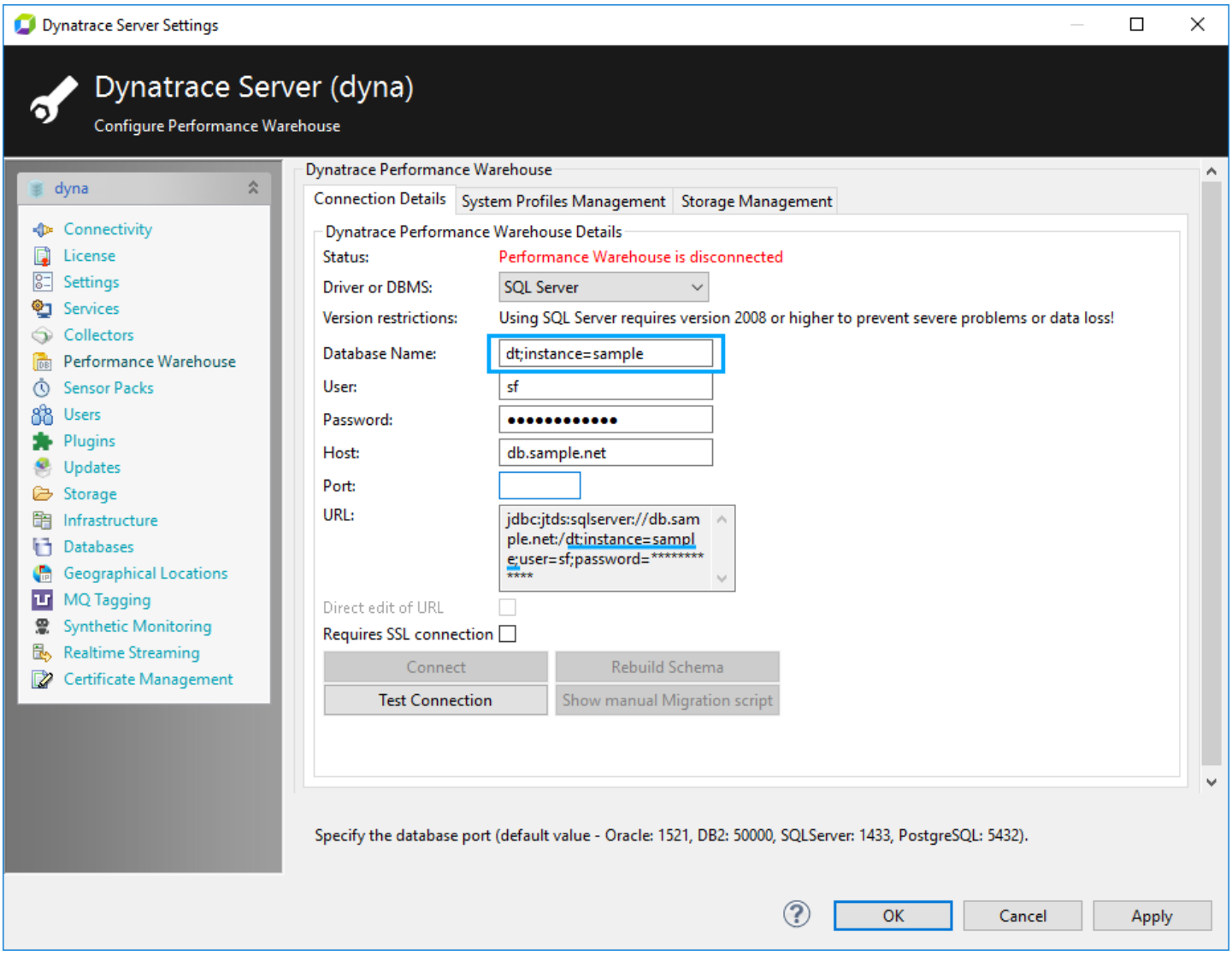
SQL Server instance with dynamic port

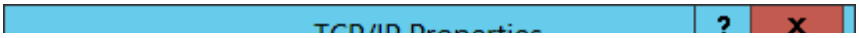
To connect to an instance with a dynamic port setting, click **Settings > Dynatrace Server > Performance Warehouse tab > Connection Details tab** in the AppMon Client and leave the **Port** field blank.

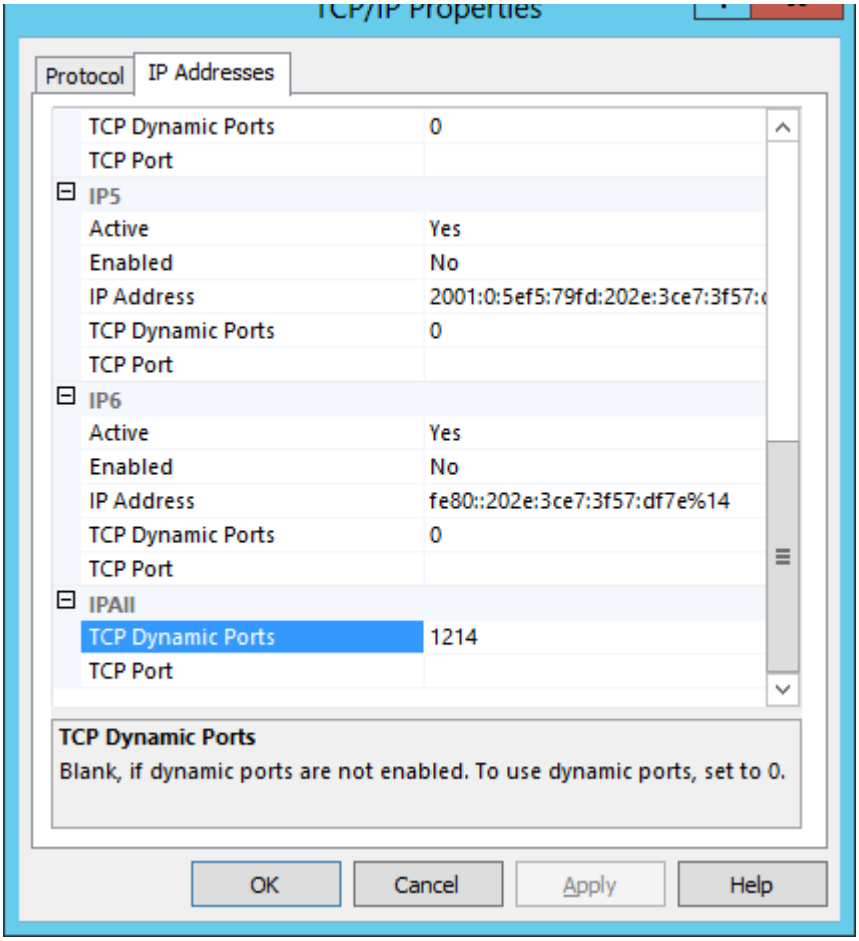
Background:
To run more than one or run the default SQL Server instance on a machine, make sure instances communicate via distinct ports with their respective clients. To do this, set different ports statically or by hand, or let the SS instances dynamically change ports if a conflict occurs.

Ensure that the SQL Server Browser service is started, so it can broker the dynamic port for the named

Ensure that the SQL Server Browser service is started, so it can broker the dynamic port for the named SQL Server instance via UDP port 1434. Also ensure that the firewall is configured accordingly if you access SQL Server remotely. The UDP port is 1434 for SQL Server and Browser and TCP dynamic port range for SQL Server.







The following is the resulting code at the bottom of `<DT_HOME>/server/conf/server.config.xml`:

```
<repositoryconfig memento.version="4.2.0.3154" lowduration="-1" highduration="1209600000"
<repository>
  <database name="dynaTrace4;instance=DYNASQLSERVER" dbms="SQLServer" />
  <credential password="xxxxxxxxxxxx" user="gy" />
  <connection port="" usessl="false" host="GCW8" ignorewarnnonproduction="false" useurl="
</repository>
</repositoryconfig>
```

Note that `connection port=""` is empty and there is no port specified after `//<hostname>:`. The latter is not relevant, as `useurl="false"`.

How to connect to SQL Server with Windows authentication via Active Directory?

Microsoft recommends that you connect to SQL Server using Windows Authentication. This is necessary if you want to use Active Directory services for this purpose.

To enable Windows Authentication for a SQL Server Performance Warehouse DB, you must add two

To enable Windows Authentication for a SQL Server Performance Warehouse DB, you must add two lines to `<DT_HOME>/dtserver.ini`, where `SQLSERVERDOMAIN` is the machine's domain or its name / the host name, if you don't use a domain.

You must add the two lines to `dtfrontendserver.ini` as well, because the Frontend Server can now directly connect to the Performance Warehouse. Add the options before the `-Deof=eof` entry.

```
-Dcom.dynatrace.diagnostics.SQLSERVERUSENTLMV2=true
-Dcom.dynatrace.diagnostics.SQLSERVERDOMAIN=<domain>
```

See How to connect the Performance Warehouse to a MS SQL Server via Windows Authentication for more information. You can also find a detailed description for Single Sign On to SQL Server and AppMon Server, which needs the same two above lines in `dtserver.ini`.

How to specify a schema for a user in DB2?

You can specify a schema other than the users default schema. To do this, add `:currentSchema=` to the database name. A valid string for the database name `dynaTrace4`, and the custom schema `dynaschema`, would look like dynaTrace4:currentSchema=dynaschema.

How do I connect to an Oracle Real Application Cluster?

Instead of a database name in the connection UI, you can apply an Oracle SID to access an RAC.

How many connections are opened by the Performance Warehouse?

The default value for the maximum number of connections is 100. You can change this value via a command line parameter ("com.dynatrace.diagnostics.repository.maxconnections").

Error: DB2 page size too small

`com.ibm.db2.jcc.b.nm: DB2 SQL Error: SQLCODE=-286, SQLSTATE=42727, SQLERRMC=8192; DB2ADMIN, DRIVER=3.50.152 on DB2`

Drop and recreate the database using the following code:

```
CREATE DB database_name PAGESIZE 16384
```

DB2 user temporary table space missing

DB2 user temporary table space missing

```
com.ibm.db2.jcc.am.mo: A default table space could not be found with a
page size of at least "4096" that authorization ID "<dynatrace_user>"
is authorized to use.. SQLCODE=-286, SQLSTATE=42727, DRIVER=4.7.85
```

To create this, use the following code:

```
CREATE USER TEMPORARY TABLESPACE <user_temporary_tablespace_name> MANAGED BY AUTOMATIC ST
GRANT USE OF TABLESPACE <user_temporary_tablespace_name> TO USER <dynatrace_user>;
```

Connection problems

If the database connection does not work, use the following steps to diagnose problems:

- Verify that the database machine is reachable from the AppMon Server machine by pinging the machine.
- Verify that a vendor specific database client like SQLPlus for Oracle can reach the database.
- Verify that it is possible to connect to the database with a JDBC client like Squirrel SQL Client).

If every step is successful, but the AppMon Server is still unable to connect to the database, contact AppMon support.

Performance Warehouse events summary incident

AppMon triggered this incident after the clean up job was executed, when one of the following events, and or exceptions, occurred:

Event/Exception	Description
Not written measurements	Skipping Data occurred where it was not possible to write data in the Performance Warehouse
Exceeded time for writing measurements	It was not possible to write data within one minute (see the Performance Warehouse Write Dashboards). Frequent messages can lead to skipped data
Written duplicated percentiles	If this message occurs frequently contact support.
Written duplicated measurements	If this message occurs frequently contact support. Try to drop the content of the measurement_temp1 or measurement_temp2 tables.
Errors while getting	

Errors while getting baseline Settings	Check server log and contact support if necessary.
A problem occurred during cleanup job	Check server log and contact support if necessary.
Exception text and Count	The text of the exception, and how often the exception occurred, is contained in the message. Please consult the server log and contact support, if necessary.

Application environment administration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Sensor Configuration

System Profile

Agent Groups

Application Environment Configuration

In AppMon, an application environment is typically represented by a System Profile. It serves as a container for configuration and is the workspace for analysis and diagnostics. The first thing to do after or during the installation is to create a System Profile and assign deployed agents to Agent Groups. Without further configuration, the system provides detailed information from its default sensors, enriched with extended information that the Auto Sensors provide.

For a more detailed analysis and diagnostics, you must create a Sensor Configuration suitable for your application and the desired purpose. Sensor Configuration and Sensor Placement have an impact on the application itself, so this task often takes multiple iterations and should be well tested before deploying into production environments.

Detailed documentation can be found here:

- System Profile Configuration describes the general operations on System Profiles.
- Sensor Configuration describes how sensors work, which different types of sensors can be placed and how to use the Class Browser to create your own sensor rules.

System Profile

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Operations on System Profiles

System Profile storage

Related pages

System Profile - Business Hours

Edit a System Profile

Enable or disable a System Profile

Export a System Profile

Guided creation of a System Profile

Import a System Profile

Manual creation of a System Profile

Rename a System Profile

System Profile templates

Manual creation of a System Profile

Edit a System Profile

Guided creation of a System Profile

Hot Sensor Placement

Watch and learn

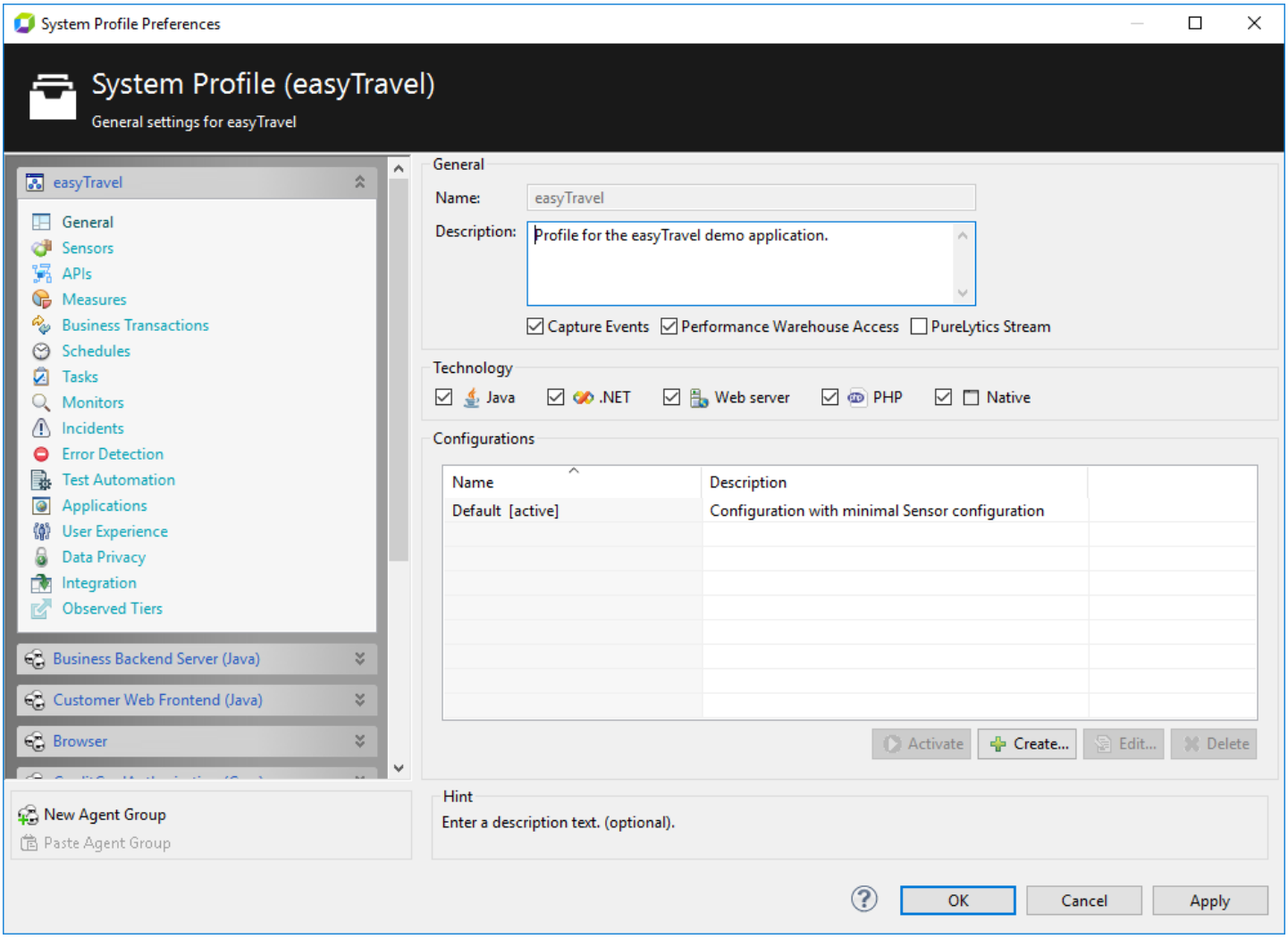
Exploring the System Profile

This page describes how to set up a System Profile, which is the “workspace” where you administer a certain application environment.

A System Profile contains all configuration settings that belong to a certain set of AppMon Agents, which represent the system under diagnoses (SUD). When a new AppMon Agent connects to the AppMon Server, it is matched to exactly one System Profile. As a result, AppMon applies all configuration settings within the System Profile to the Agent.

System Profiles are listed in the AppMon Client cockpit and serve as one possible starting point for all interactive actions applied to the data. The data is provided by the AppMon Agents associated with the System Profile.

You can manage any number of System Profiles on one AppMon Server. To access the **System Profile Preferences** dialog box, right click the required system profile, and select **Edit System Profiles** from the context menu.



System Profile Preferences dialog box

Operations on System Profiles

Create System Profile	Manually create a new System Profile. You can also use a Wizard to create a System Profile. Only visible in context menu of the **System Profiles** node.
Edit System Profile	Manually edit the System Profile.
Search PurePath	Find a single PurePath by HTTP response header. Only visible in context menu of the **System Profiles** node.
Open Session Overview	A new dashboard opens that contains the Transaction Flow Dashlet, PurePaths Dashlet, and Status Overview Dashlet.
Connect Application	Guided creation of a new Agent Group within the System Profile.
Session Storage	Manual session storage functions. See Session for more information.
Hot Sensor Placement	Apply configuration changes to all connected AppMon Agents (if supported).
Synchronize Discovery Data On Collectors	Reuse data gathered during the discovery run of this System Profile and deploy the information to other AppMon Collectors. All connected collectors immediately deploy. Collectors connecting in the future also use this information for initialization. This allows AppMon to rely on the data already collected by other AppMon Collectors and omit the Discovery run for new AppMon Collectors.
Enable disable a System Profile	Activate or deactivate a System Profile. You can't use disabled System Profiles for Agent Mapping.
Import System Profile	Import a System Profile from a local XML file. Only visible in context menu of the **System Profiles** node.
Export System Profile	Export the System Profile to a local XML file.
Rename System Profile	Rename the System Profile.
Active Configuration	Switch the currently active Configuration.

System Profile storage

System Profiles are stored as XML files on the AppMon Server in the folder `<DT_HOME>\server\conf\profiles\<application_name>.profile.xml`. To edit a System Profile, you must have a connection to the AppMon Server and the appropriate permissions You don’t have to run a SUD to define and edit System Profiles. See Server Administration for more information.

> **Note**
>
> Administer System Profiles within a version control system.

Manual creation of a System Profile

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

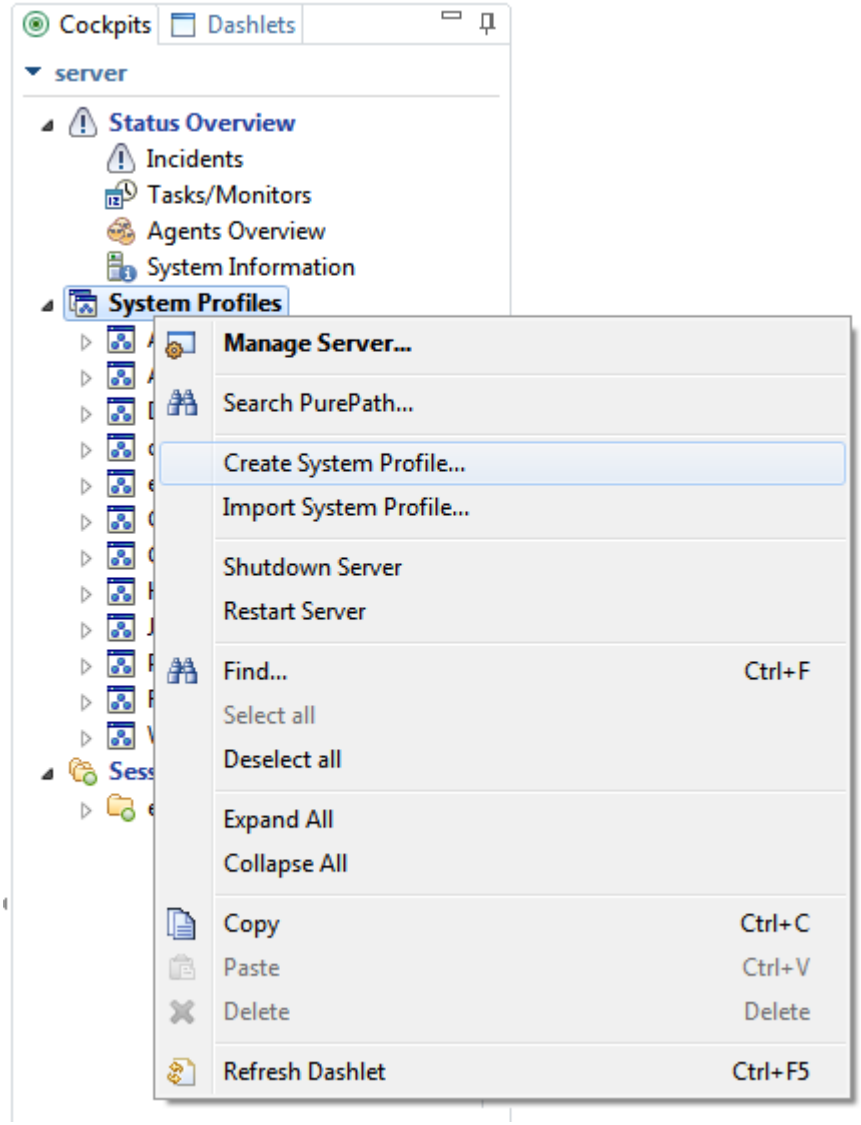
Guided creation of a System Profile

Edit a System Profile

To manually create a System Profile, select **Create System Profile** from the context menu of the System Profiles node in the **Cockpits** tab. A **System Profile Preferences** dialog box appears with an empty System Profile. You must specify a system-wide unique name to create a valid (empty) System Profile. For more information on System Profile setup, see Edit a System Profile.

Note

Not applying a license or reaching the enabled System Profile limit disables new System Profiles after creation.



Manually create a System Profile.

Guided creation of a System Profile

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

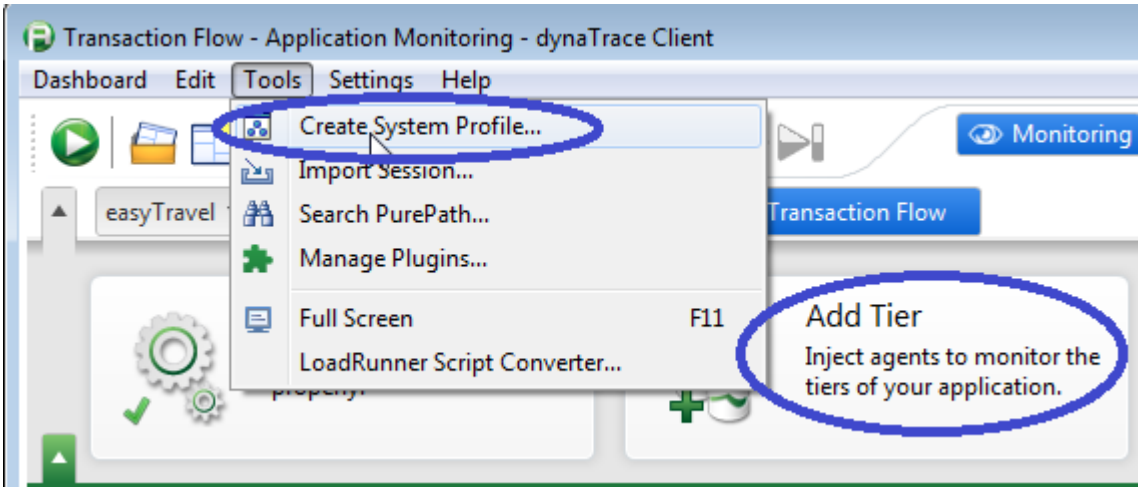
Manage System Profile with the (Add Tier) wizard

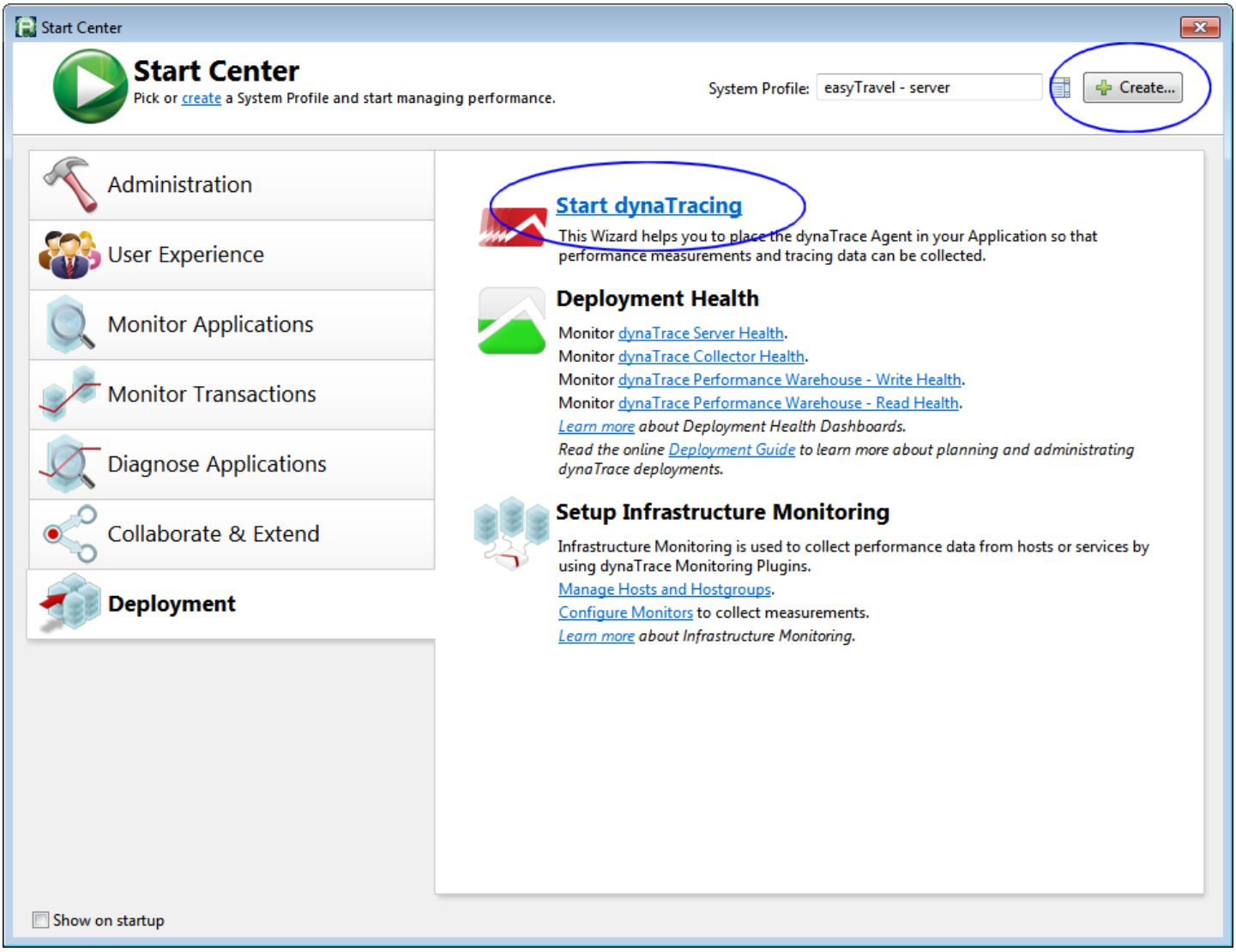
Related pages

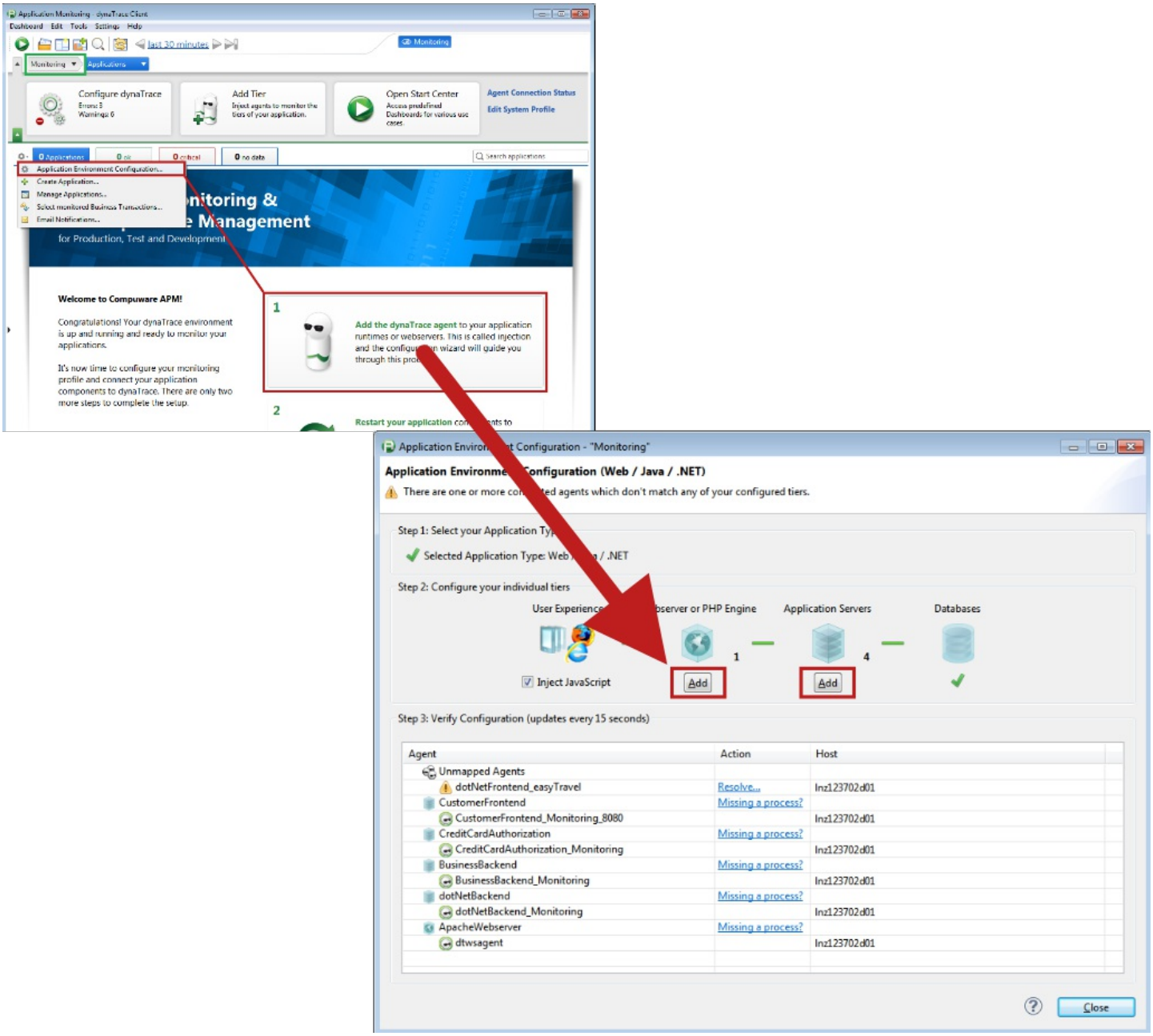
Manual creation of a System Profile

Edit a System Profile

Enable or disable a System Profile




Start Center
Start Center
Pick or create a System Profile and start managing performance.
System Profile: easyTravel - server
Create...
Administration
User Experience
Monitor Applications
Monitor Transactions
Diagnose Applications
Collaborate & Extend
Deployment
Start dynaTracing
This Wizard helps you to place the dynaTrace Agent in your Application so that performance measurements and tracing data can be collected.
Deployment Health
Monitor dynaTrace Server Health.
Monitor dynaTrace Collector Health.
Monitor dynaTrace Performance Warehouse - Write Health.
Monitor dynaTrace Performance Warehouse - Read Health.
Learn more about Deployment Health Dashboards.
Read the online Deployment Guide to learn more about planning and administrating dynaTrace deployments.
Setup Infrastructure Monitoring
Infrastructure Monitoring is used to collect performance data from hosts or services by using dynaTrace Monitoring Plugins.
Manage Hosts and Hostgroups.
Configure Monitors to collect measurements.
Learn more about Infrastructure Monitoring.
Show on startup



Manage System Profile with the (Add Tier) wizard

Depending on your entry point, you have slightly different ways to manage or create a System Profile:

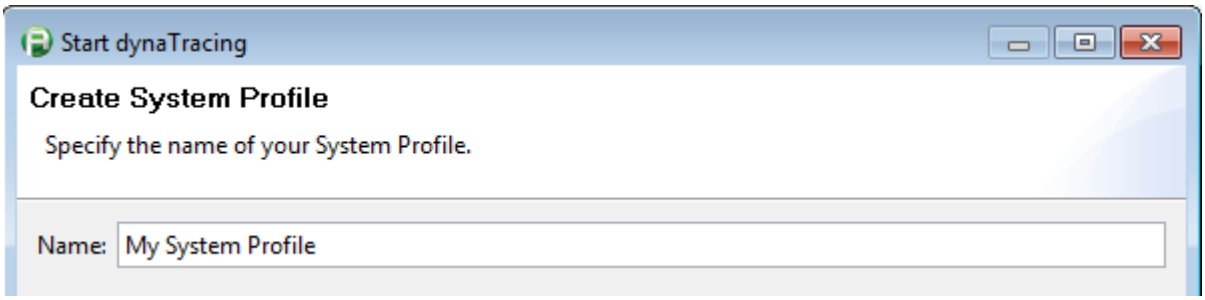
- In the AppMon Client´s **Tools > Create System Profile** menu, you enter the wizard and name a System Profile in step 1.
- From the AppMon Start Center, you begin at Step 2 since you already created or selected a System Profile there. At the right top of the window, click the **Create** button.

- Through the **Add Tier** button in the **Configuration** panel, add to the **active** System Profile (shown in the breadcrumbs - **here: 'easyTravel'** in the first screenshot) or - in a new installation when there is only an AppMon Self-Monitoring profile - a new System Profile is created, implicitly named Monitoring.
- From the **Add** buttons in the Application Environment Configuration Dialog, add a new tier configuration to the active System Profile (shown in the breadcrumbs - **here: 'Monitoring'** in the last screenshot).

The whole process has up to seven basic steps.

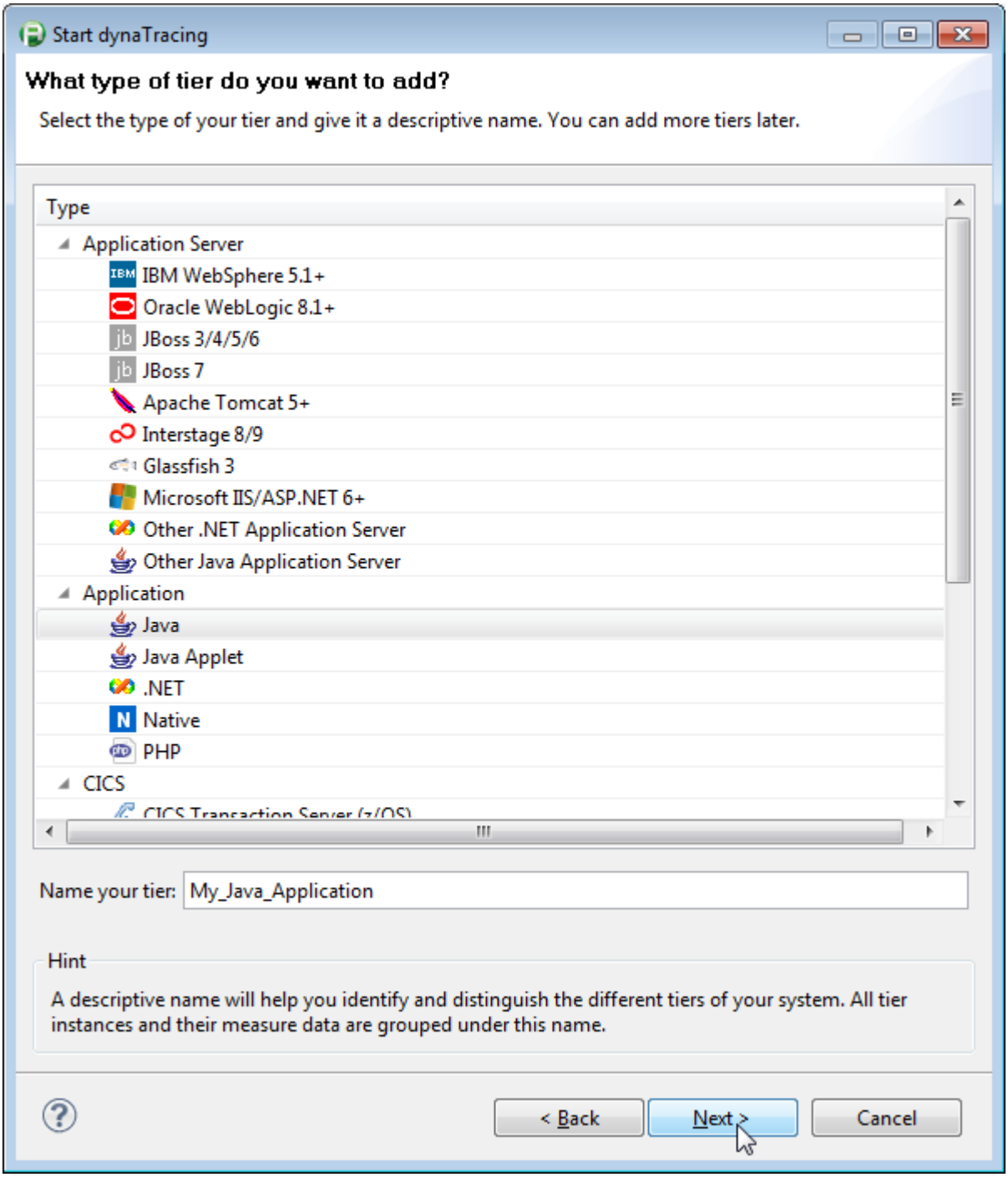
Step 1: Select a System Profile, or Create a new System Profile

Specify the name of the new System Profile, or edit an existing one when you enter the wizard from **Tools > Create System Profile**.



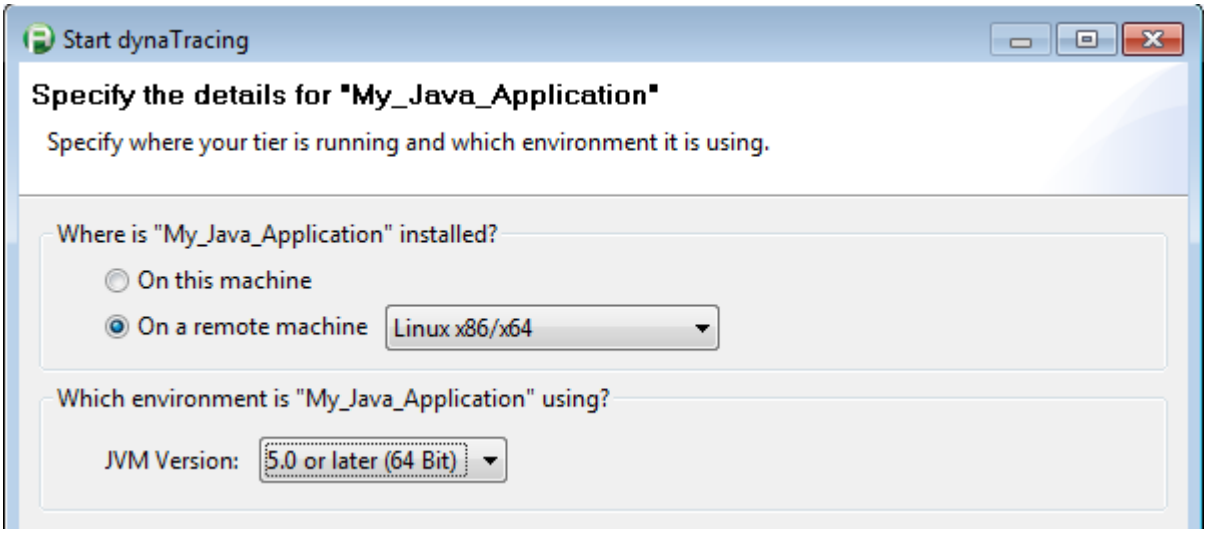
Step 2: Add a new tier. Select the technology of new tier

A new Agent Group is created for each tier. See the Agent Groups later by double-clicking the newly created System Profile in the **Cockpits** sidebar.



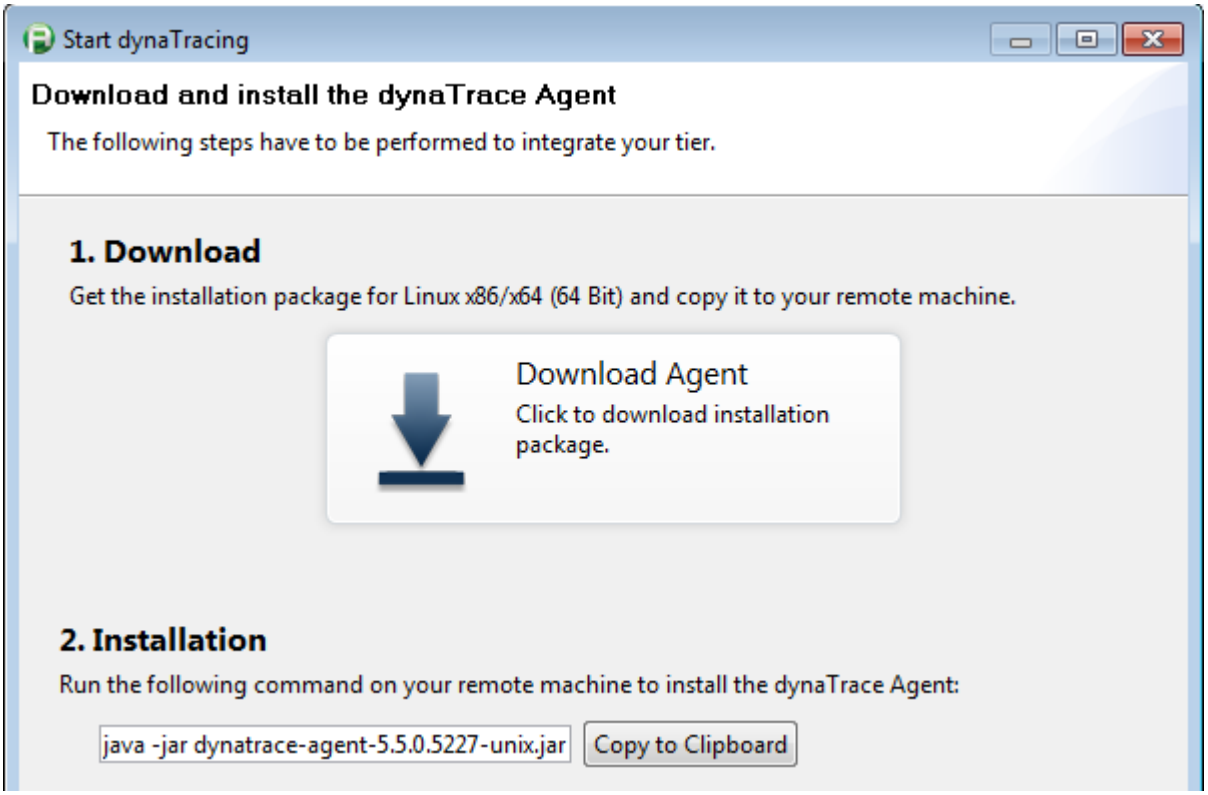
Step 3: Define technology details of new tier

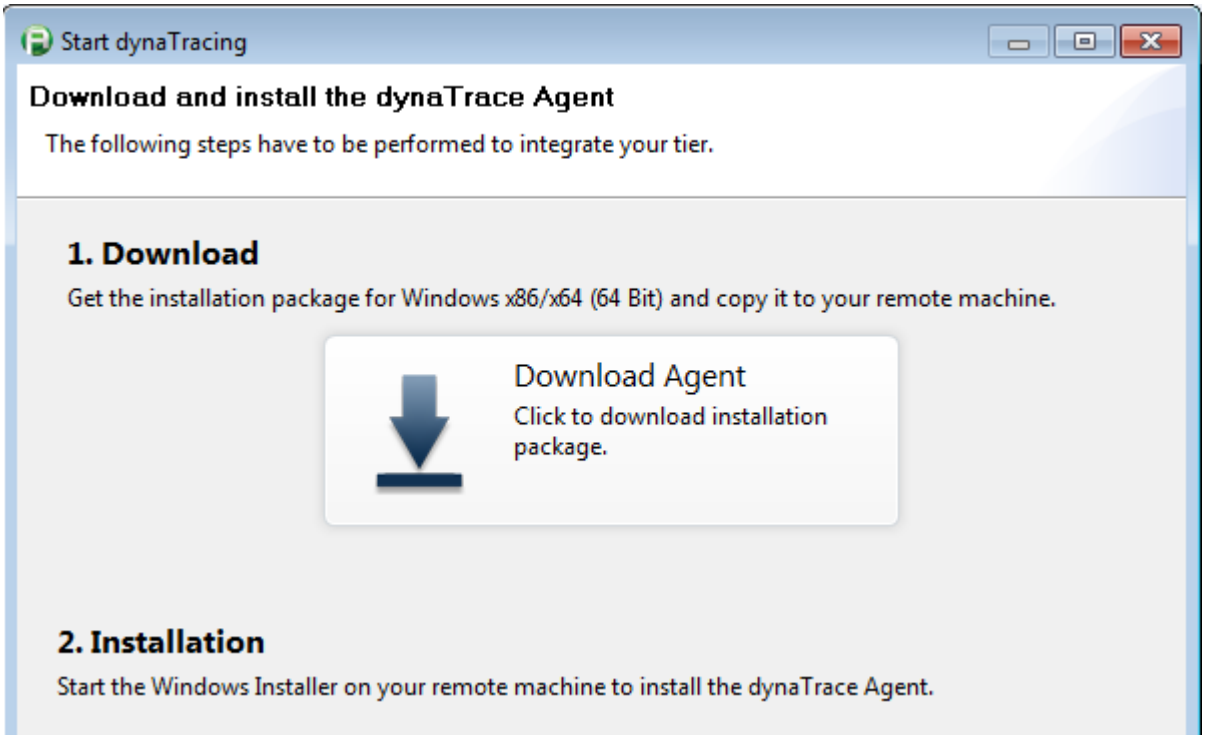
Depending on the technology of the tier, specify the location and the technology details of the tier. The definition steps vary depending on the selected technology (Java/.NET/Native). For Java this includes the specification of the used JDK/JRE version.



Step 4 (Remote machines only): Install the Agent

Download the appropriate Agent version from the Dynatrace Community. The installation steps vary depending on the selected technology (Java/.NET/Native).



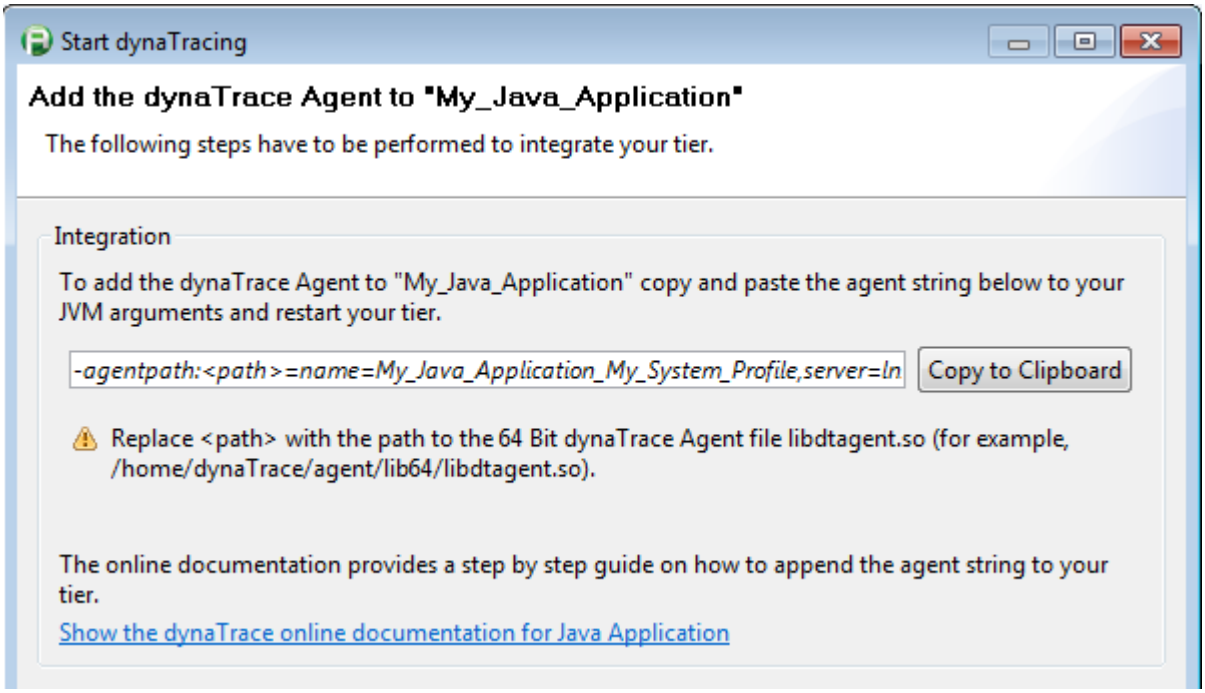


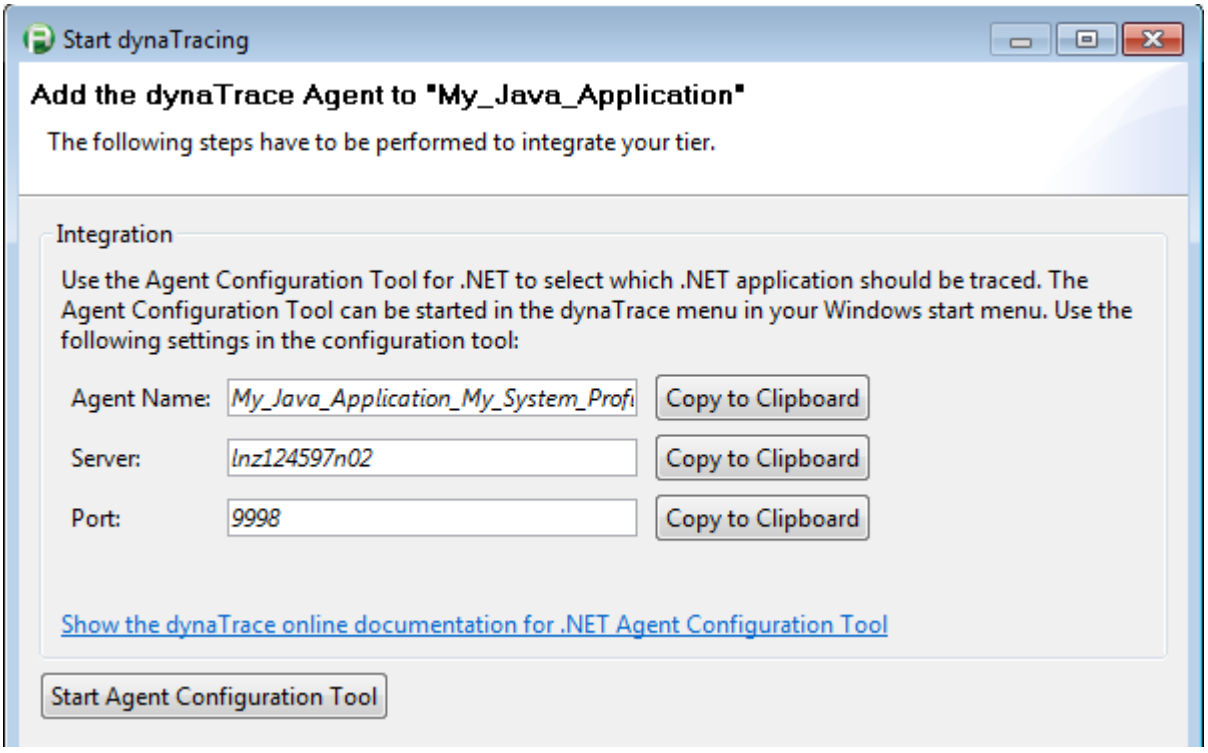
Step 5: Integrate the Agent into the SUD

Configure the SUD to have an Agent integrated. The integration steps vary depending on the selected technology (Java/.NET/Native).

For example, in Java this includes placing specific command line parameters into the startup scripts of the SUD, while in .NET this is done using the Agent Configuration Tool.

> **Note**
>
> The provided agent string for Java-based SUDs is optimized to be used in startup scripts or in a command line. To use the agent string in Eclipse-based `.ini files`, the double quotes (“) must be removed.





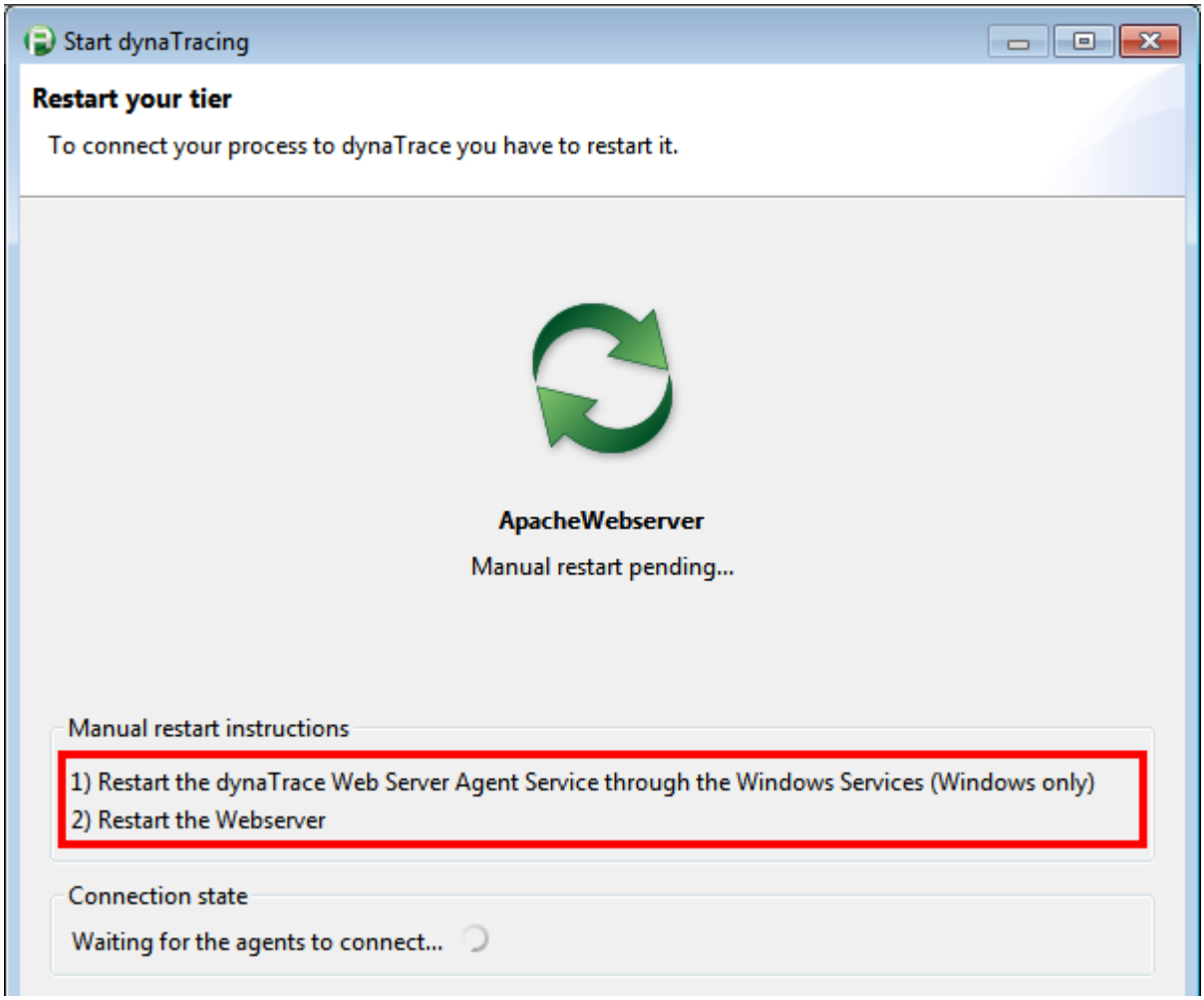
Step 6: Manually restart components

The manual restart page indicates which components to restart before the agents can send data.

Usually this is just the process for the just-created configuration. However, Windows systems for web servers and IIS require two or even three restart steps.

Web server on Windows systems:

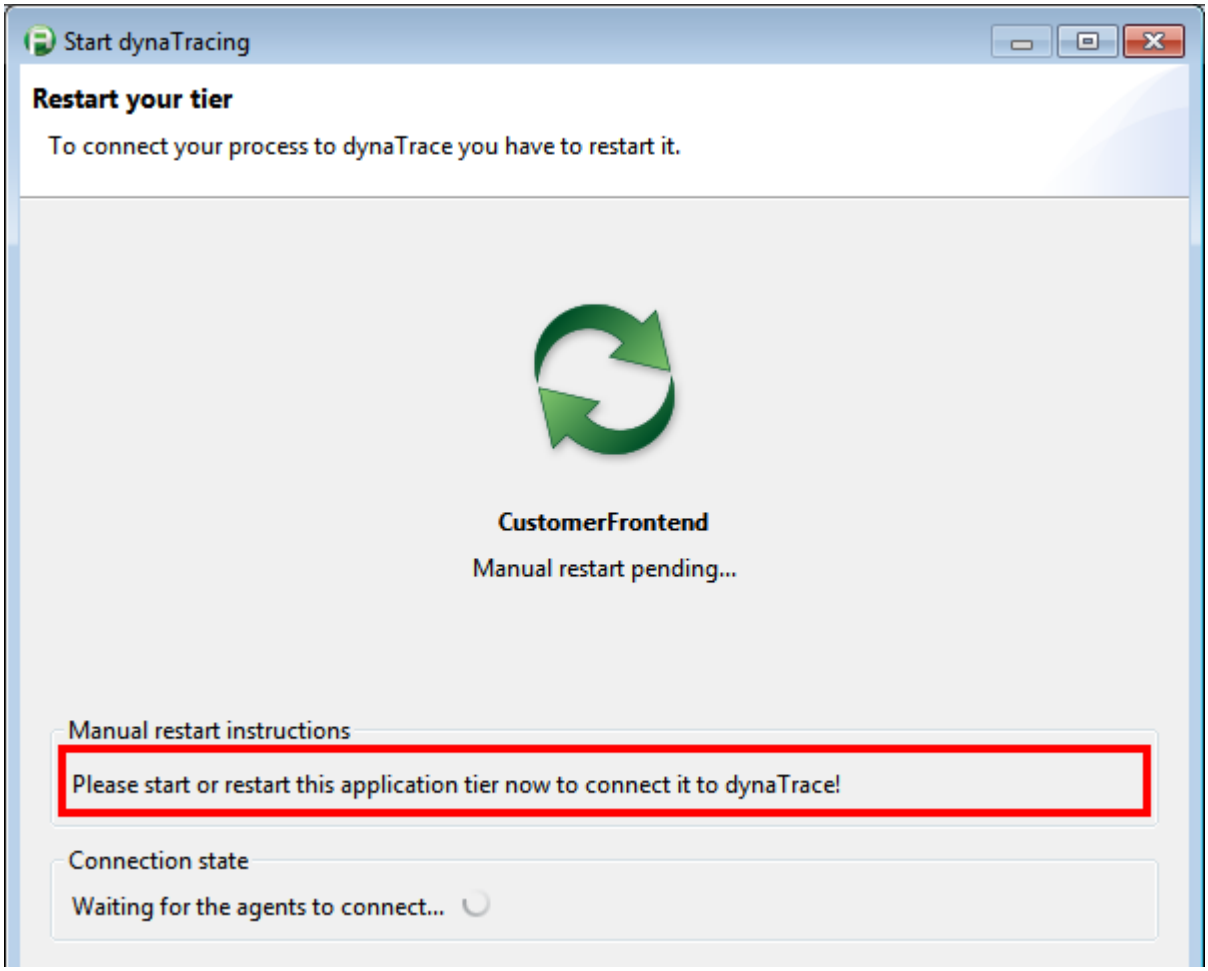
1. Restart the AppMon Web Server Agent Service through the Windows Services.
2. Restart the web server.



IIS:

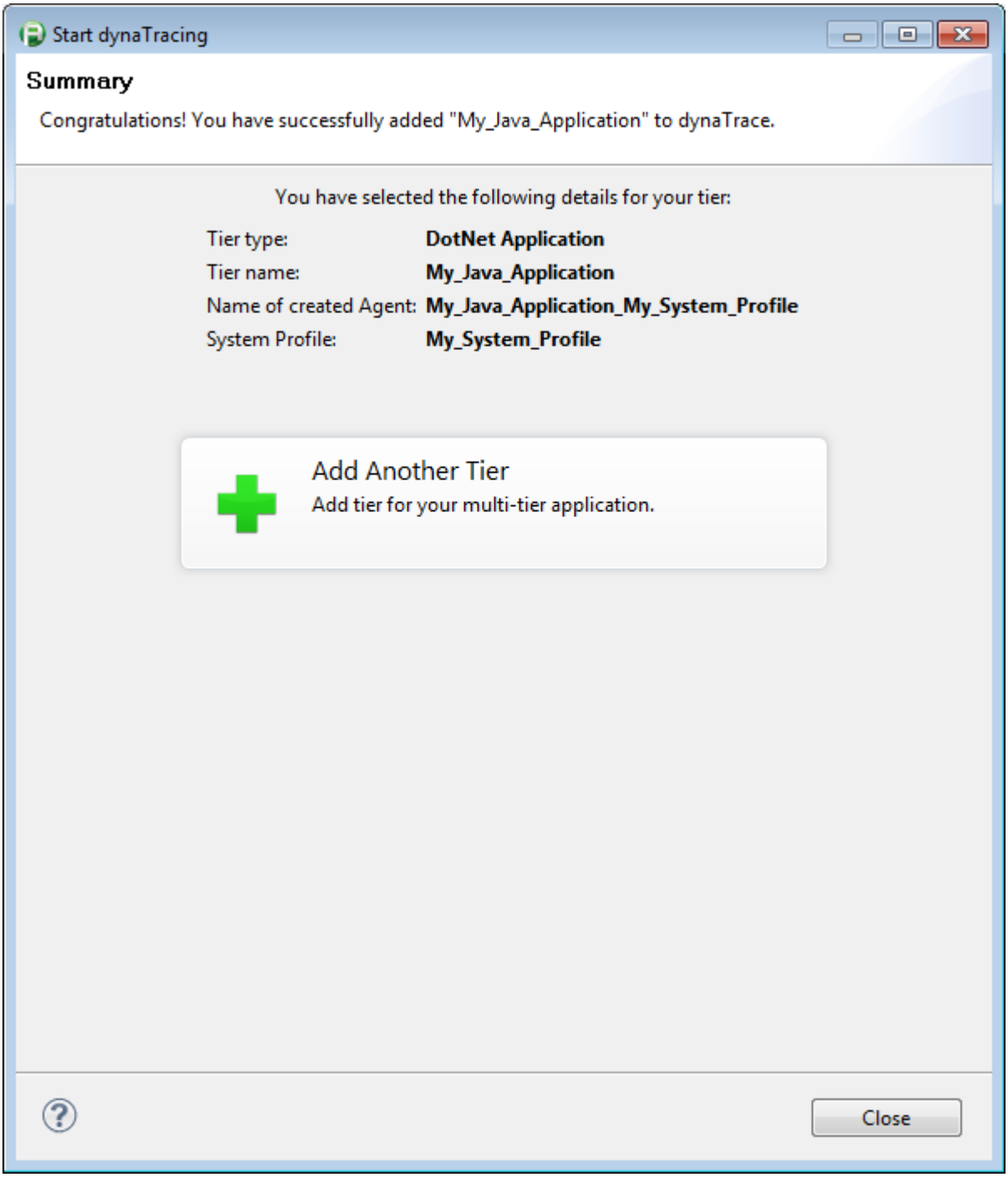
1. Restart the AppMon Web Server Agent Service through the Windows Services.
2. Restart/Recycle your IIS Application Pool through the IIS Manager.

3. Access your application's web site through the browser (ensures that, for example, ASP.NET is loaded).



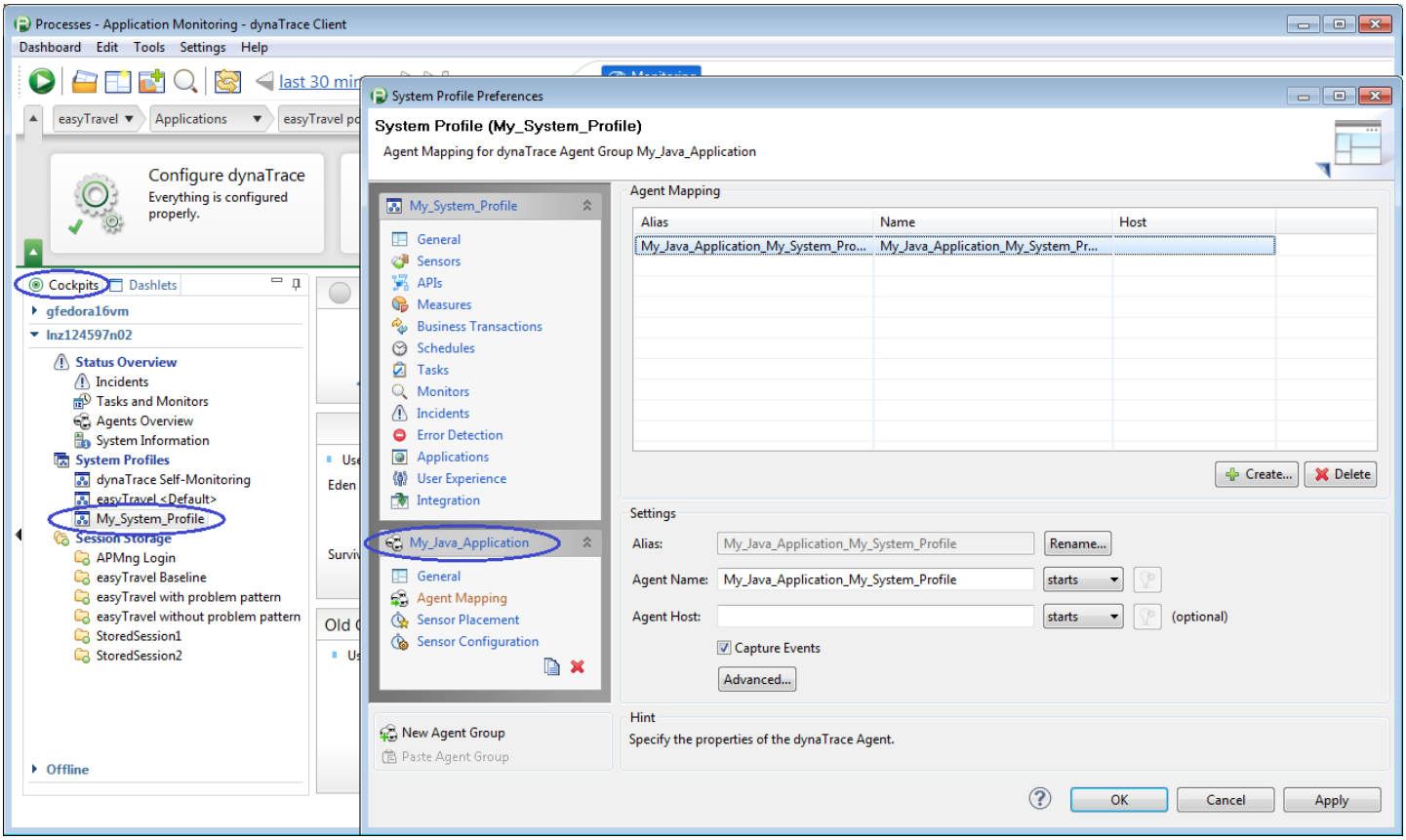
Step 7: Summary of System Profile

In the summary, you can add additional tiers (start again at Step 2) if the wizard was not started with the Application Environment Configuration dialog box.



Step 8: Check the newly created profile

Double-click the newly created System Profile in the **Cockpits** sidebar and check the Agent Groups / tiers.



Edit a System Profile

Looking for the 6.5 version of this page? Open in 6.5 documentation

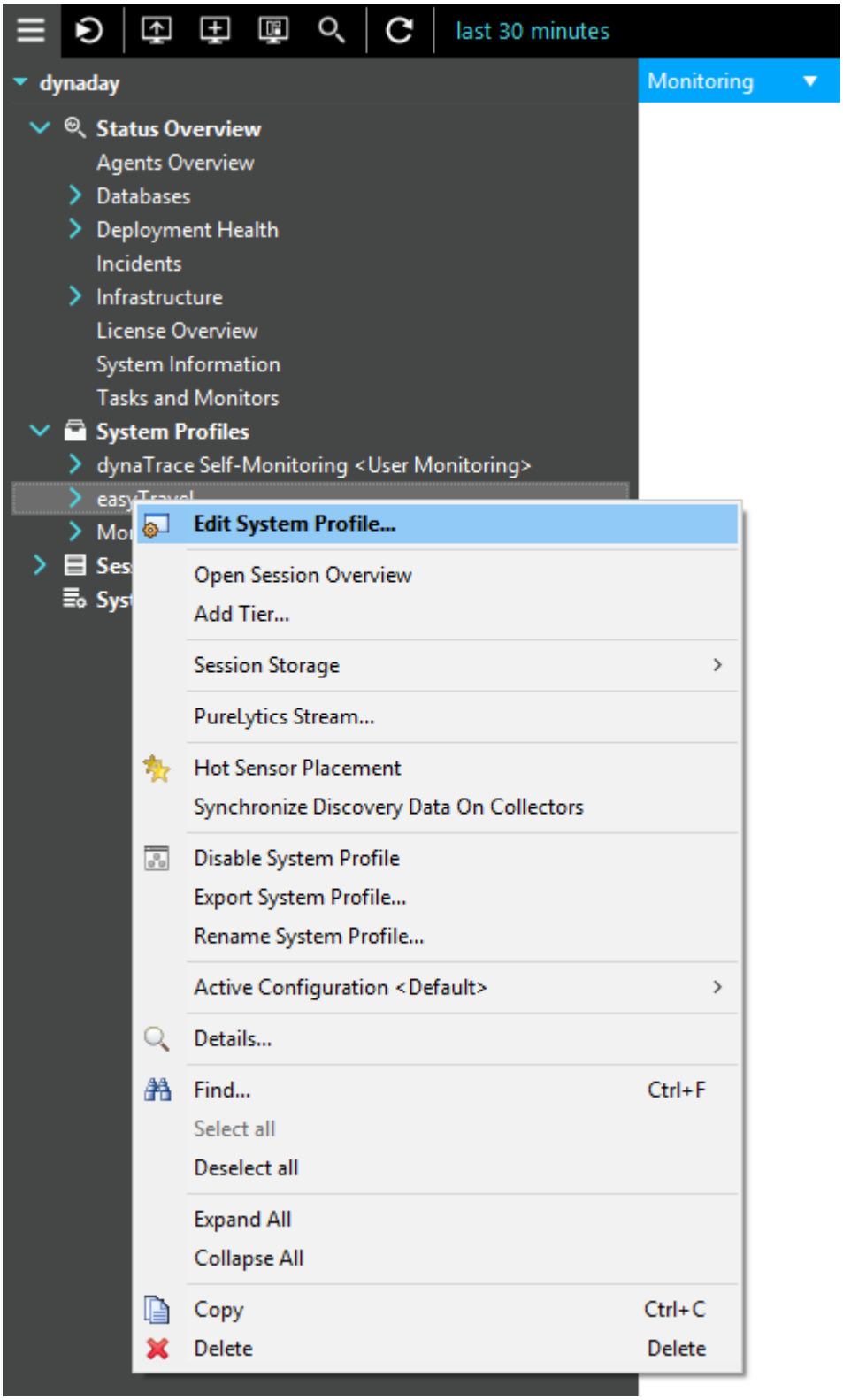
On this page

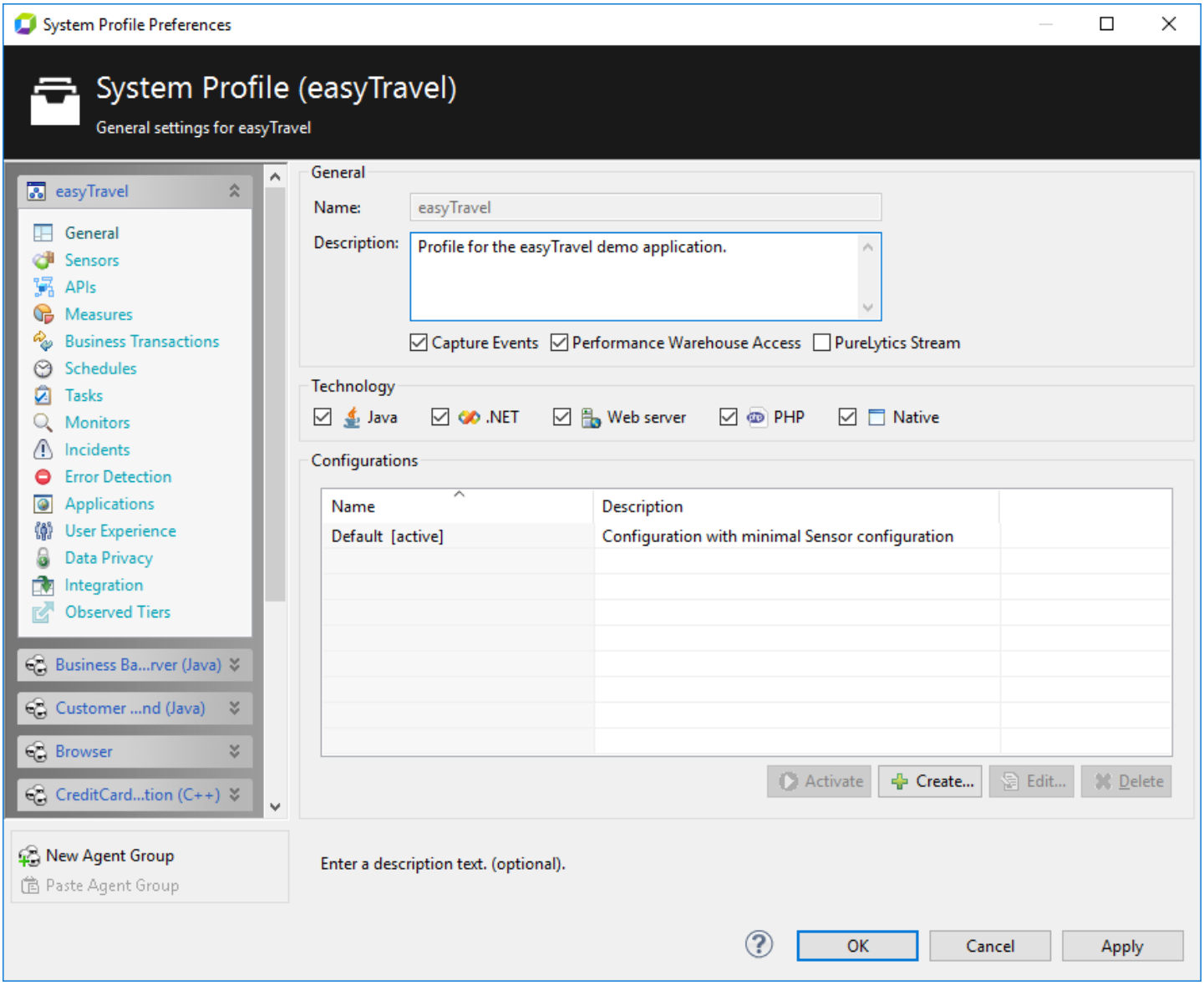
System Profile configuration

Agent Groups

To edit a System Profile, select **Edit a System Profile** from the context menu of a System Profile in the **Cockpits** tab or double-click the System Profile. The **System Profile Preferences** dialog box appears and displays defined sensor rules, subscribed measures, and scheduled tasks or monitors. All configuration information is stored in the System Profile file on the AppMon Server. Changes in this dialog box are applied to all connected systems (AppMon Collectors and AppMon Agents) whenever you press **OK** or **Apply**.

Edit a System Profile


last 30 minutes
dynaday
Monitoring
Status Overview
Agents Overview
Databases
Deployment Health
Incidents
Infrastructure
License Overview
System Information
Tasks and Monitors
System Profiles
dynaTrace Self-Monitoring <User Monitoring>
Edit System Profile...
Open Session Overview
Add Tier...
Session Storage
PureLytics Stream...
Hot Sensor Placement
Synchronize Discovery Data On Collectors
Disable System Profile
Export System Profile...
Rename System Profile...
Active Configuration <Default>
Details...
Find... Ctrl+F
Select all
Deselect all
Expand All
Collapse All
Copy Ctrl+C
Delete Delete



System Profile configuration

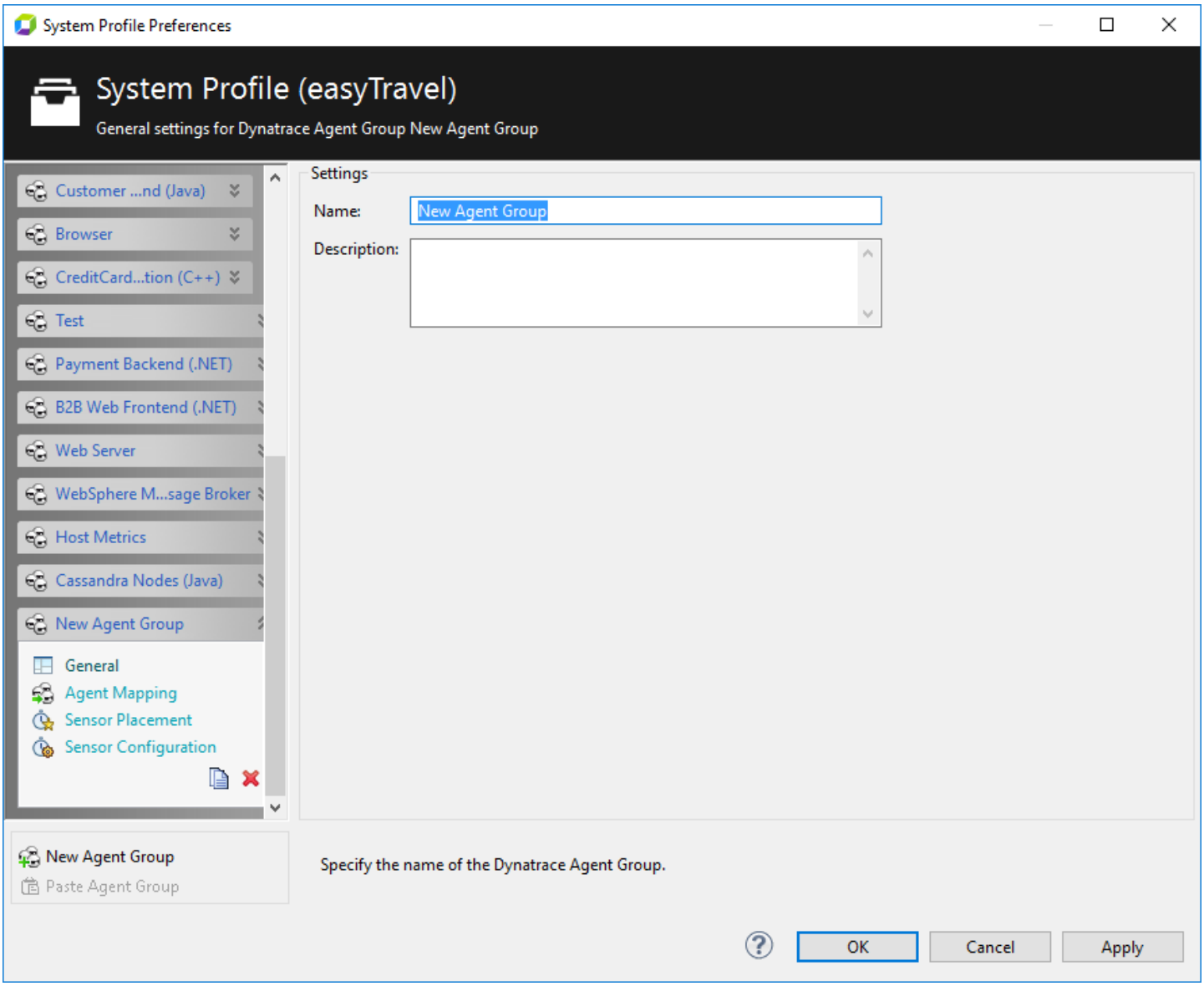
The **System Profile Preferences dialog box** provides the following configuration options per System Profile:

- General: General configuration settings, such as the name and description of the System Profile or supported technologies.
- Sensors: Sensor rule configuration (define sensor groups, method/memory rules, etc.).
- APIs: API configuration.
- Measures: Create and edit measures.
- Business Transactions: Create and edit Business Transactions based on measures.
- Schedules: Create and edit schedules (used for tasks and monitors).
- Tasks: Manage tasks running on different execution targets such as the Collector.
- Monitors: Manage monitors running on different execution targets such as the Collector.

- Error Detection: Configure error detection rules.
- Applications: Configure application detection.
- Incidents: Manage incidents based on measures.
- Test Automation: Configure test automation.
- User Experience: Configure User Experience Management.
- Data Privacy: Configure data privacy.
- Integration: Configure how to integrate other products into AppMon.
- Observed Tiers: Configure data privacy for the System Profile.

Agent Groups

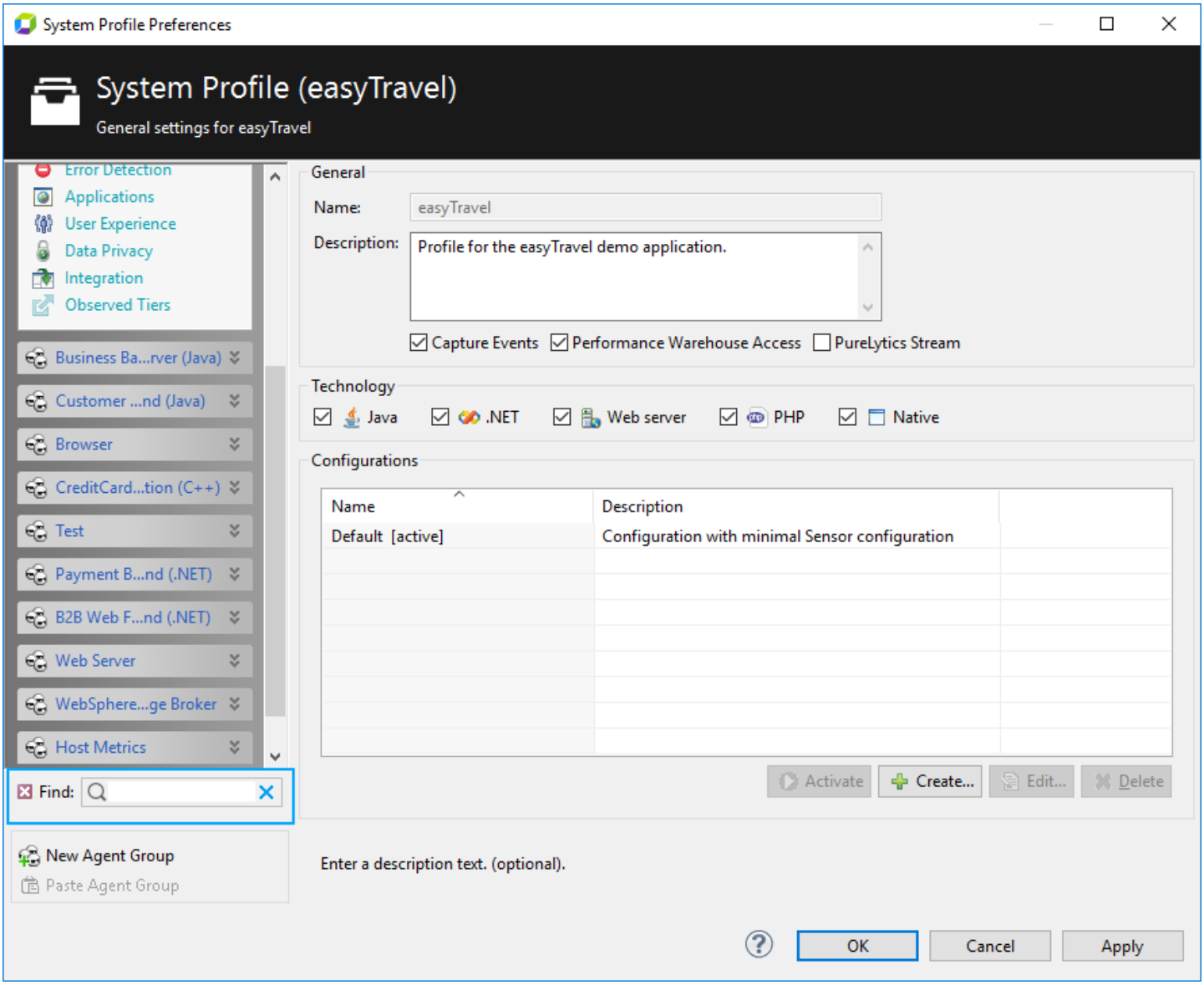
A System Profile may contain any number of Agent Groups.



Each Agent Group provides several configuration sections:

- General
- Agent Mapping
- Sensor Placement
- Sensor Configuration

Press **Ctrl + F** to search and filter Agent Groups. The search is not case sensitive.

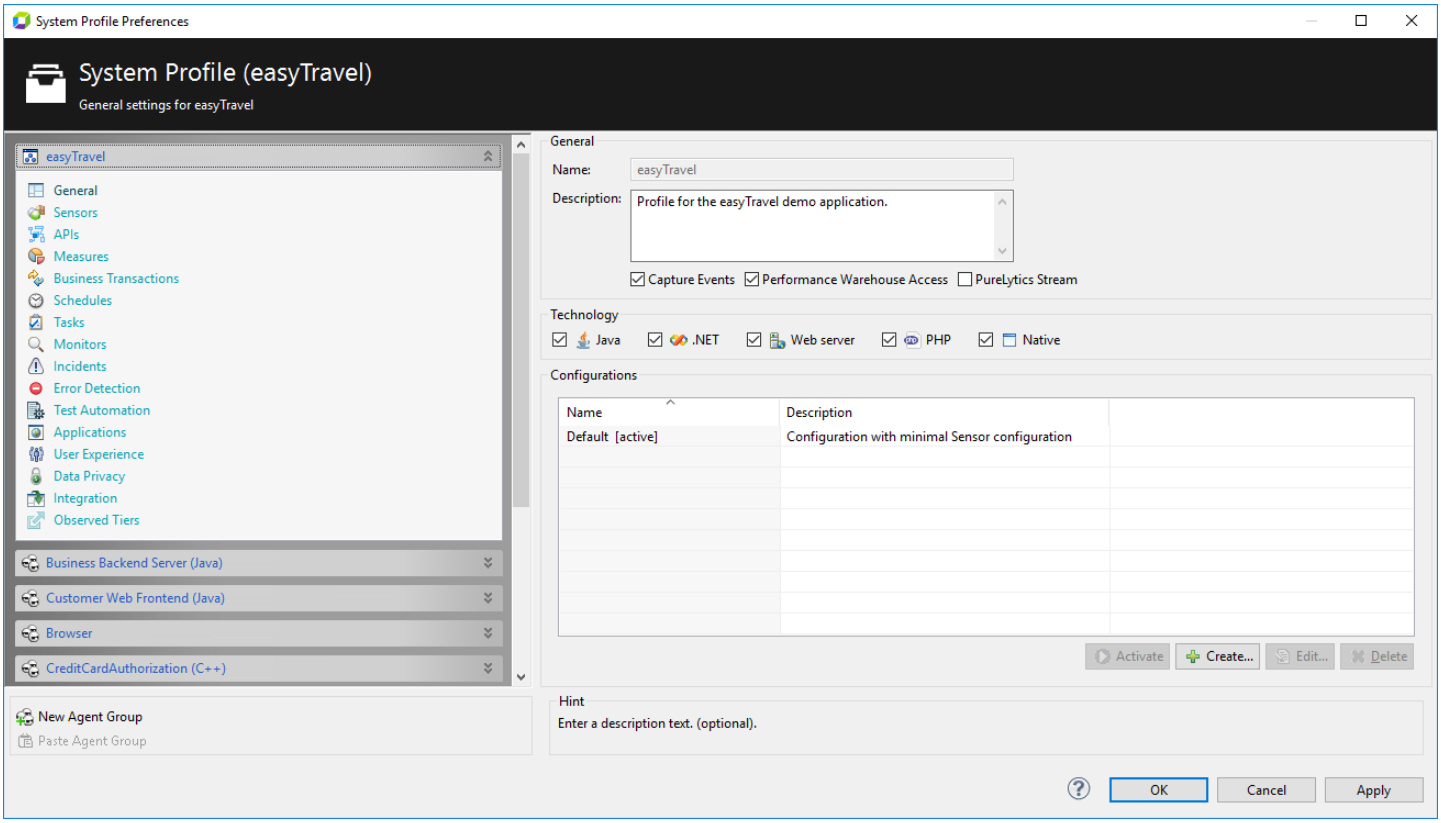


System Profile - General

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Agent Group - Sensor Configuration



Field/Button	Description
Name	A system-wide unique name of the System Profile. The **Cockpit** displays the name as a separate node. It is also used for storing the System Profile on the AppMon Server as an XML file.
Description	Descriptive text of the System Profile. For documentation purposes only.
Capture Events	Toggles event capturing for all AppMon Agents of this System Profile.

Performance Warehouse Access	Defines if the System Profile has access to the Performance Warehouse. Enabling Performance Warehouse access stores data captured by AppMon Agents.
PureLytics Stream	Enables or disables the PureLytics Stream. This feature lets you send real time UEM data from the AppMon Server to an external data source like Elasticsearch so you can you leverage UEM data as part of big data analysis. See PureLytics Stream for more information.
Technology	The following technologies are supported by this System Profile: • **Java**: Java Platform (since JDK 1.4) • **.NET**: .NET Platform • **Browser**: Browser Agents • **Web server**: Webserver Agents • **ADK**: native Agent Development Kit Technologies not supported by the System Profile can't have sensor rules.
Configurations	Create and edit System Profile configurations. Different configurations have separate Sensor configuration sets inside the same System Profile. To activate and deactivate configurations quickly, toggle between different Sensor configurations within the System Profile. There is always exactly one configuration active inside a System Profile. You can't delete the last System Profile configuration. See Sensor Configuration for more information.

System Profile - Sensors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Sensor rule management

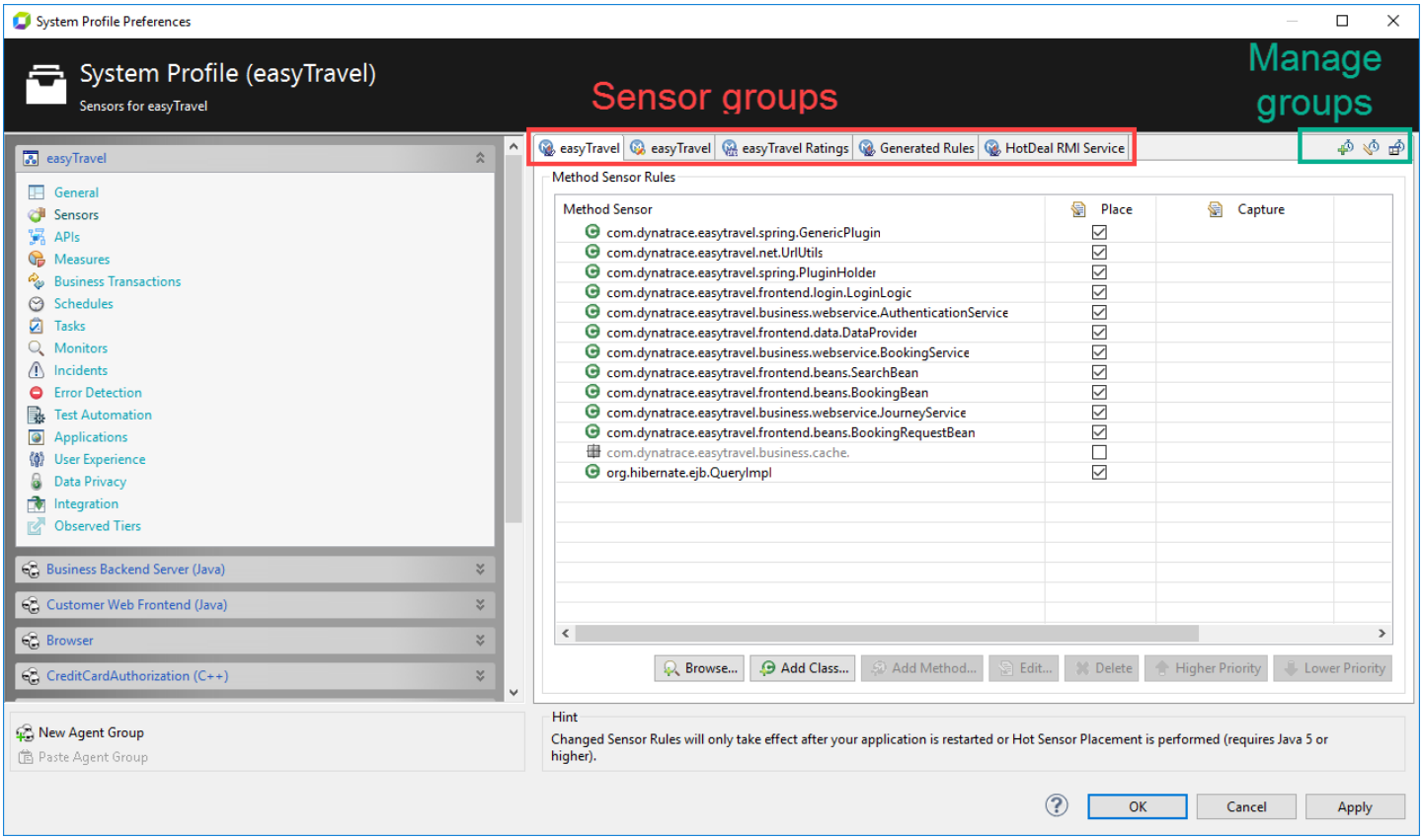
Creating sensor packs

Related pages

Sensor Configuration

Sensor properties

Sensors



Sensor Rules are organized in Sensor Groups inside of a System Profile. Besides Sensor Packs, Sensor Groups are the atomic entities in Agent Group Sensor Placement and Sensor Configuration settings. You can manage changes to whole Sensor Groups instead of single rules. You can also create Sensor Groups from technologies that the System Profile supports.

Sensor rule management

Operation	Description
Browse	Opens the Class Browser. If an AppMon Agent is already connected to this System Profile and classes are loaded, the Class Browser displays the SUD class hierarchy. You can then select packages (assemblies) as well as classes and methods to create new Sensor Rules. See Class Browser for more information.
Add Class	Manually add a Class Rule. May also cover package or assembly rules depending on the Class pattern.
Add Method	Manually add a Method Rule. You must select a Class Rule to execute this. May c over more than one method depending on the method name pattern and argument rules. For example, an empty method name pattern is interpreted as *all methods of a class*.
Edit	Edit the selected Sensor Rule.
Delete	Delete the selected Sensor Rule.
Higher Priority	Move the Sensor Rule up by one. This increases the Rule's priority.
Lower Priority	Move the Sensor Rule down by one. This reduces the Rule's priority.

See Sensor Configuration to learn how to manually define Sensor Rules.

Creating sensor packs

A rule set can be packaged into a globally available sensor pack. To do this, select the requried rules, right click and select **Create Sensor Pack**. The newly created service pack is available in the **Sensor Packs** item of the **Dynatrace Server Settings** dialog box. There you can also export this sensor pack into a `.dtcs` file.

System Profile - APIs

Looking for the 6.5 version of this page? Open in 6.5 documentation

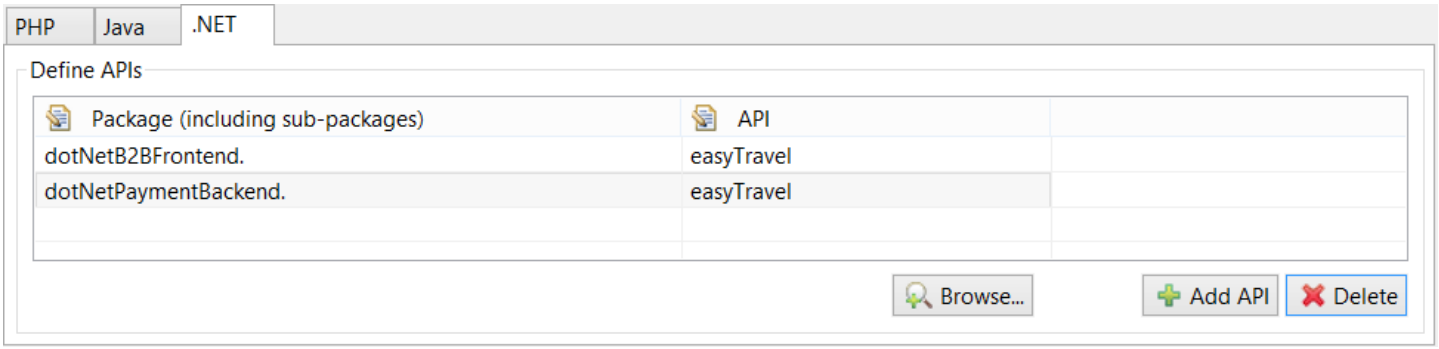
Related pages

Class Browser

APIs

APIs are defined on a per-package basis.

In the AppMon Client, select **APIs** item in the **System Profile Preferences** dialog box, then click the appropriate tab to specify API packages for PHP, Java and .NET. For each API, click **Add API** and enter a package pattern and corresponding API. Enter the APIs manually or click **Browse** to use the Class Browser. See APIs for more information.



Define APIs for PHP in the **PHP API rule configurator** dialog box. The API string can include spaces and other special characters. Patterns entered for Package selects sub-packages of matching patterns. For example, the pattern `com.` also matches with `com.dynatrace`.

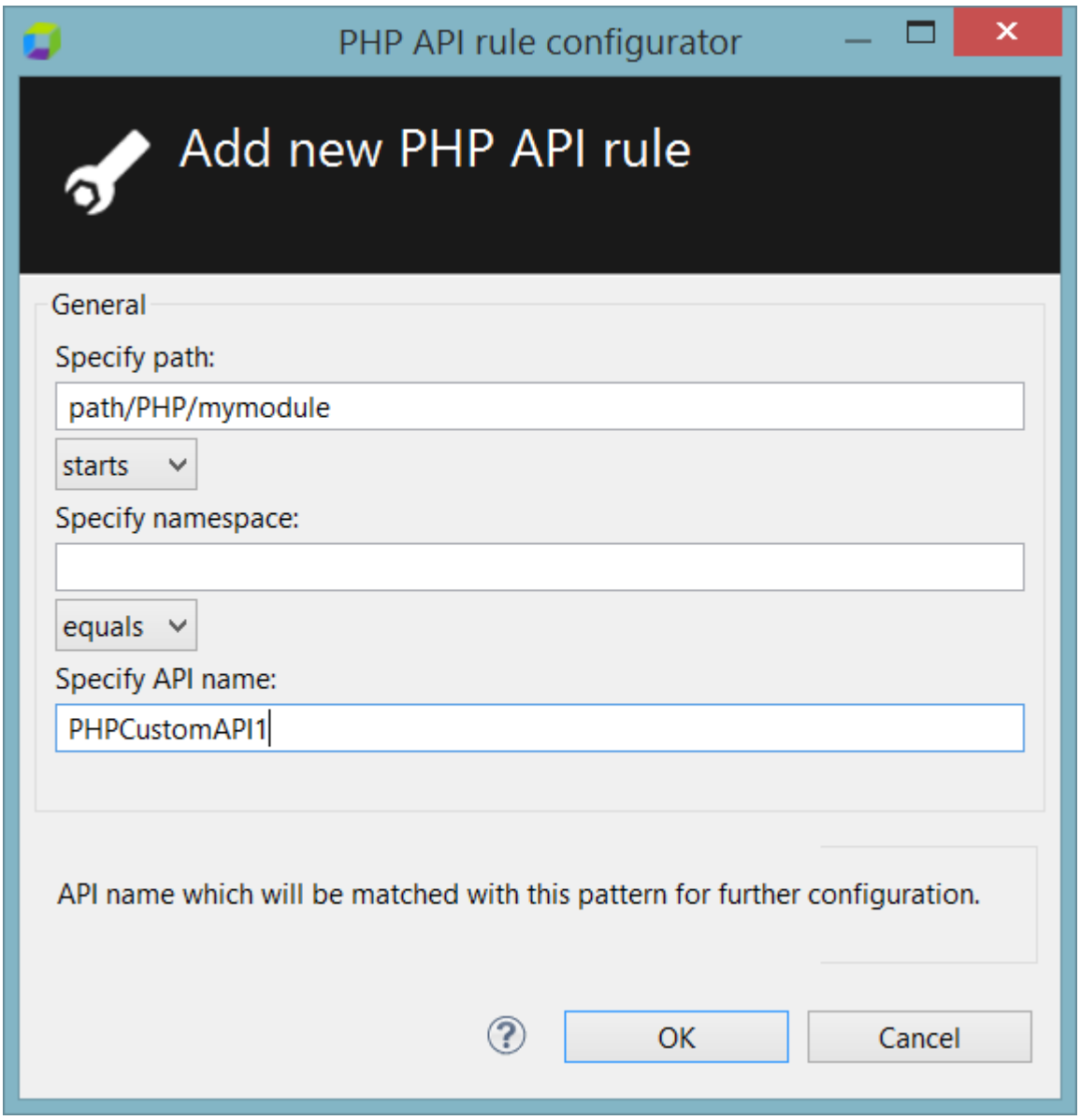
Pattern order is not relevant since AppMon chooses the most specific rules first to avoid errors due to partially overlapping patterns.

Add PHP API rules

PHP API support is available for AppMon 6.3 and later. It identifies and creates APIs for popular plugins, including Drupal, Magento and WordPress. You can also define custom APIs.

In the **PHP** tab, click **Add API** to add rules in the **Add new PHP API rule** dialog box. Enter the path or namespace and set a pattern match for each, and enter the API name. Valid pattern match types include:

- **equals**: The pattern applies if the actual name is equal to the pattern text.
- **starts**: The pattern applies if the actual name starts with the given pattern text.
- **ends**: The pattern applies if the actual name ends with the given pattern text.



System Profile - Measures

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Dynamic Measure Matrix dashlet

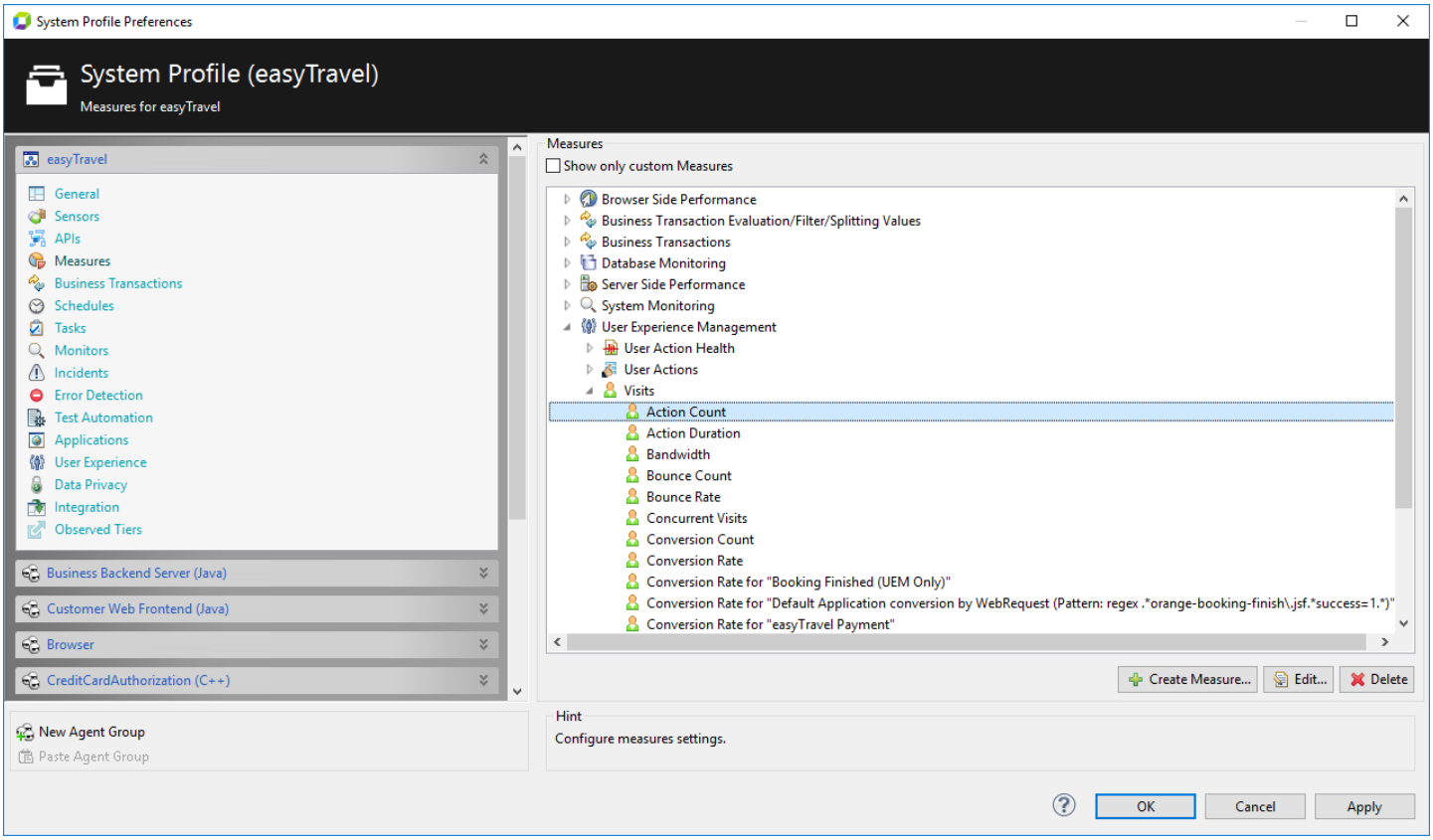
Watch and learn

Creating measures

Managing measures

Measures, measures, measures

View, edit, and define measures for a System Profile. Upon creation of a System Profile a set of default measures is subscribed.



You can copy measures or a group of measures between system profiles. To search for a measure, press Ctrl+F, and type first letters of the measure name. To clear measured values for a measure, or for a group of measures, right click it, and select **Erase Data** from the context menu.

List of measures with description

Select **<System Profile> > Export System Profile > Search for <measures>** to see all the measures.

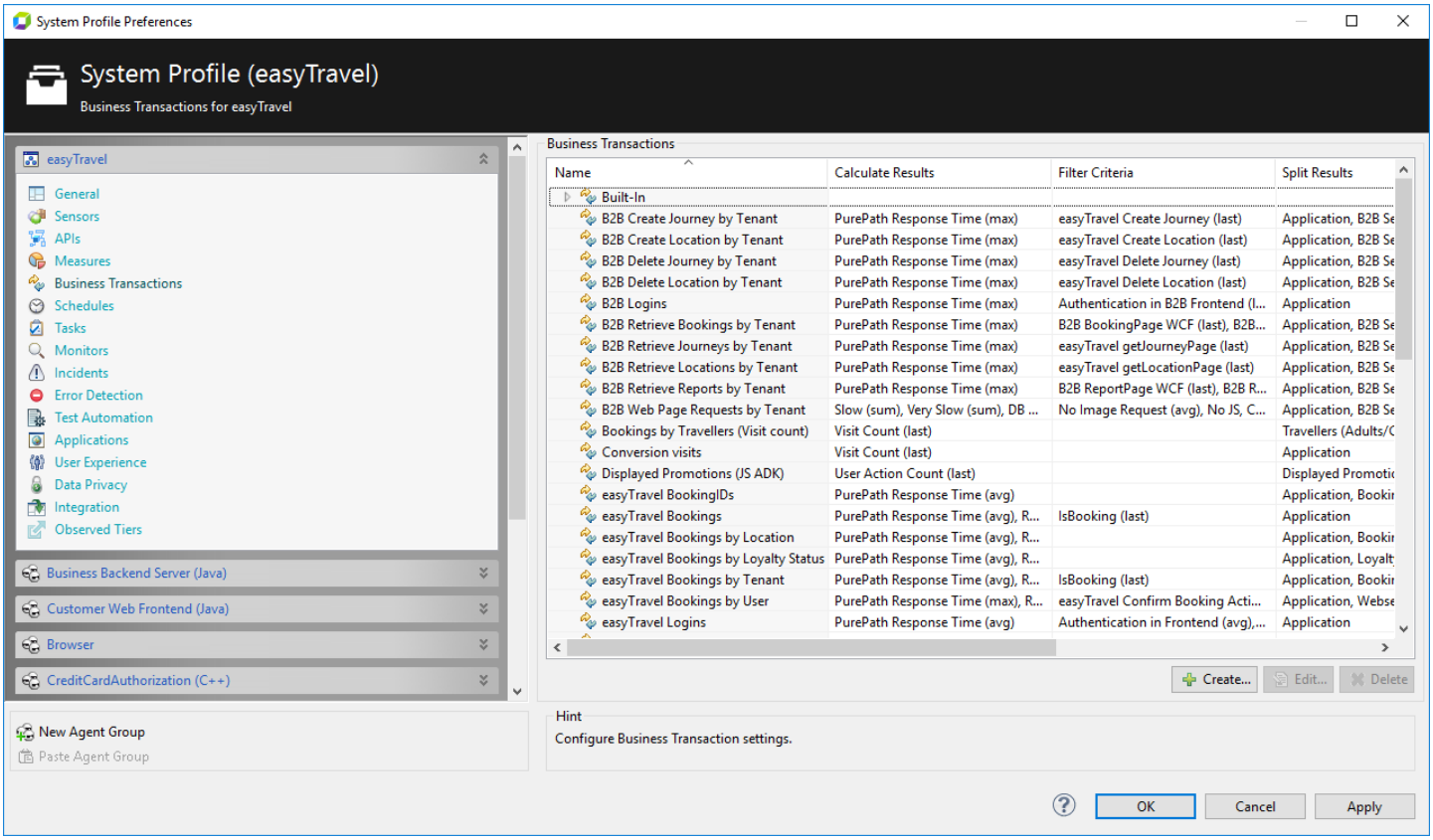
For a better experience, you can apply an XSL Transform to the System Profile XML (see sample `profile.xsl` with instructions). View Monitoring.profile.xml.htm for a sample transform.

System Profile - Business Transactions

See Usage of Business Transactions for Business Transactions details.

> A disabled (gray) entry means the specific business transaction is not be evaluated and does not produce results. Setting Business Transactions active or inactive helps reduce performance impact. Right-click to select or clear the active state.

Configure Business Transactions based on measurements of the System Profile.

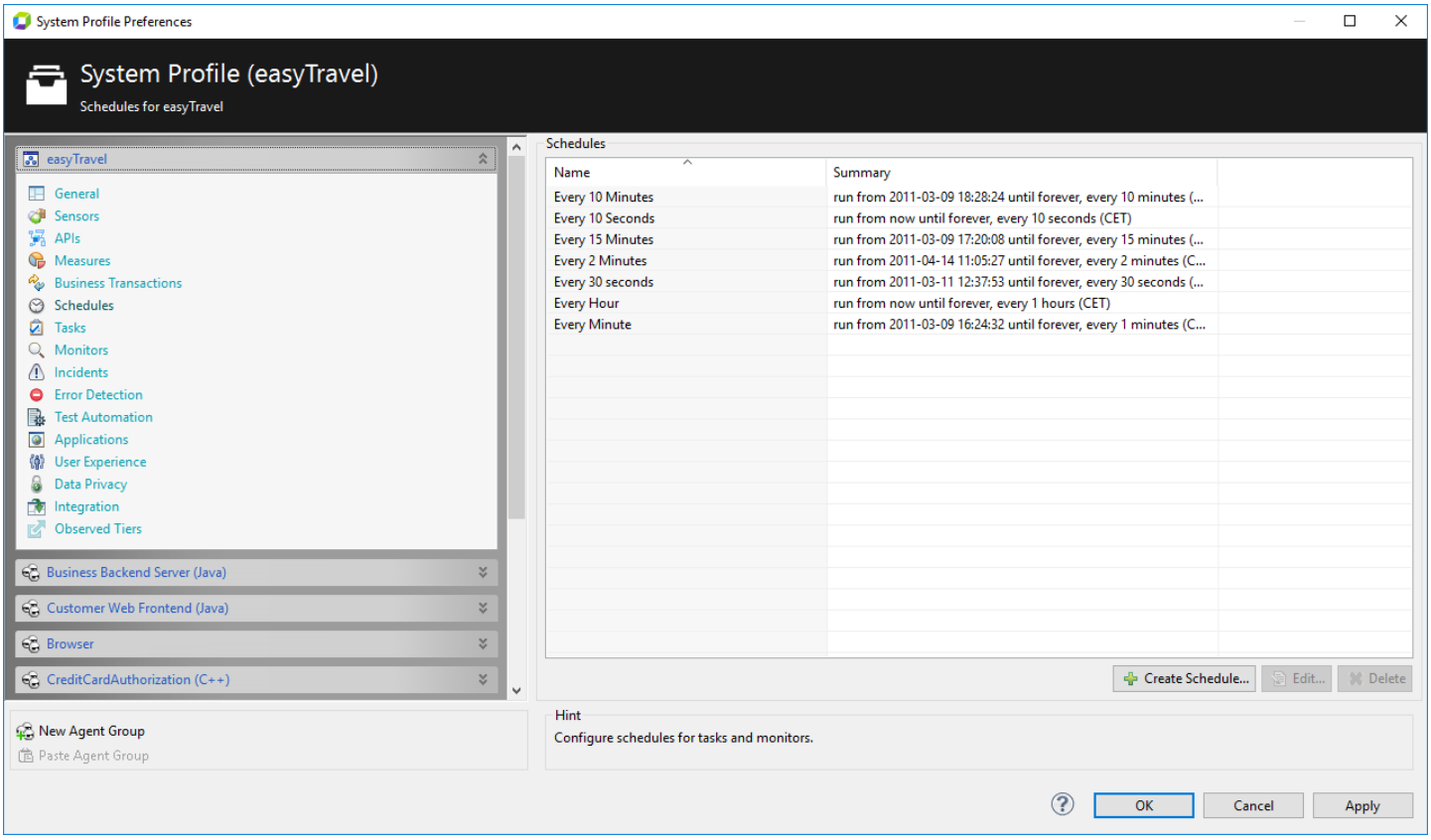


System Profile - Schedules

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Defining a schedule



Operation	Description
Create Schedule	Create a new schedule.
Edit	Edit the currently selected schedule.
Delete	Delete the currently selected schedule.

Defining a schedule

Schedules must have a unique name within the System Profile. It consists of two entities: Schedule Run and Schedule Exclusion.

Schedule Run: Defines when the scheduled action is performed. A Schedule may consist of any number of separate Schedule Runs, although in most cases there is one Schedule Run definition.

Schedule Exclusion: Defines exceptions from the standard runs, to avoid runs on times that cannot be defined easily in Schedule Run definitions. This is typically used to exclude maintenance times or holidays.

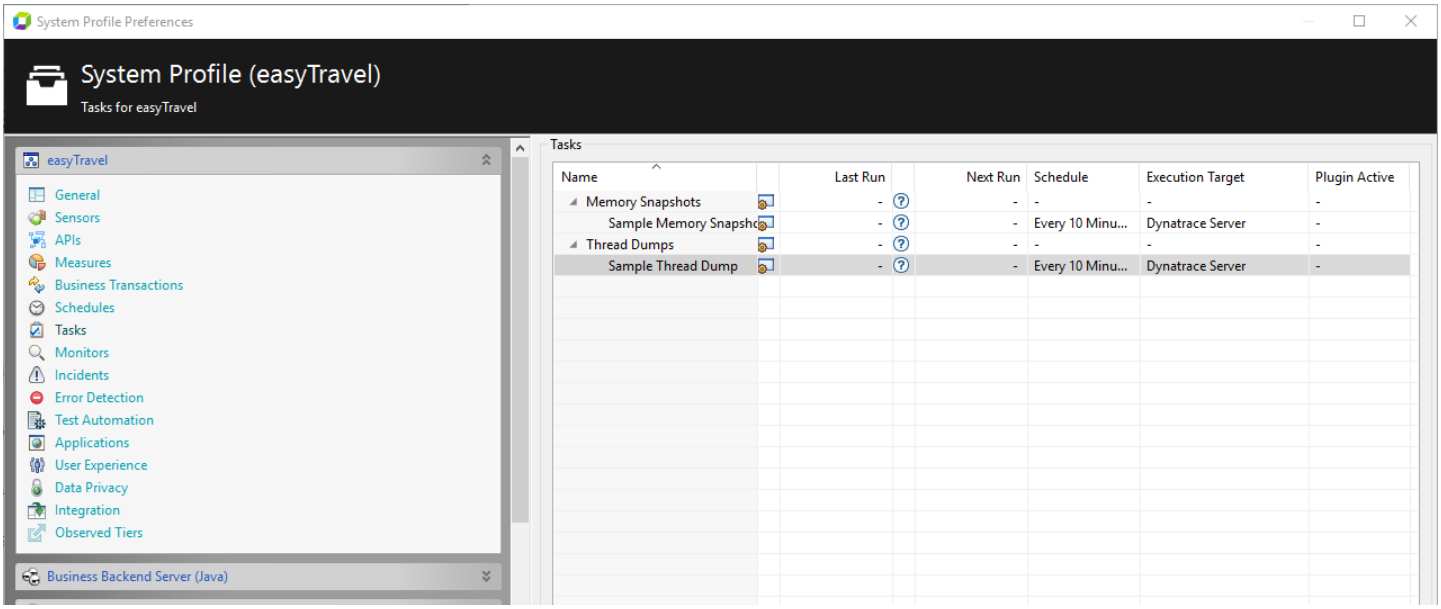
All Schedule Runs and Schedule Exclusions relate to one single time zone that is defined per Schedule. You can use relative time zones, such as the AppMon Server time zone or absolute time zones such as GMT. The minimum interval is 10 seconds.

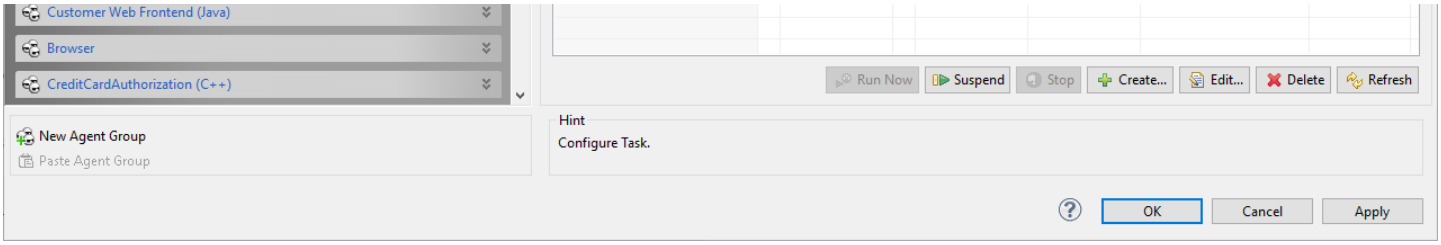
System Profile - Tasks

See Thread Diagnostics and Memory Diagnostics for more information on Memory Dump or Thread Dump Tasks.

Operation	Description
Run Now	Run the selected Task once. Does not schedule the Task for repeated execution if it is stopped.
Resume/Suspend	Schedule the selected Task for repeated execution using Schedule assigned to the Task during Task creation, or suspend the task execution.
Stop	Stop the execution of the selected Task. If the Task is scheduled for repeated execution, this schedule is removed.
Create	Create a new Task based on built-in Tasks and user-defined Plugins. During Task creation, a the Task is assigned a Schedule for repeated execution, as well as an optional execution target and Task parameters depending on the Task type. For example, a Memory-Dump Task requires an Agent group specification.
Edit	Edit the selected Task.
Delete	Delete the selected Task.
Refresh	Refresh the list of Tasks. Updates the state of the displayed Tasks.

Live management of Tasks that exist in the System. This includes actively scheduled Tasks and disabled Tasks.



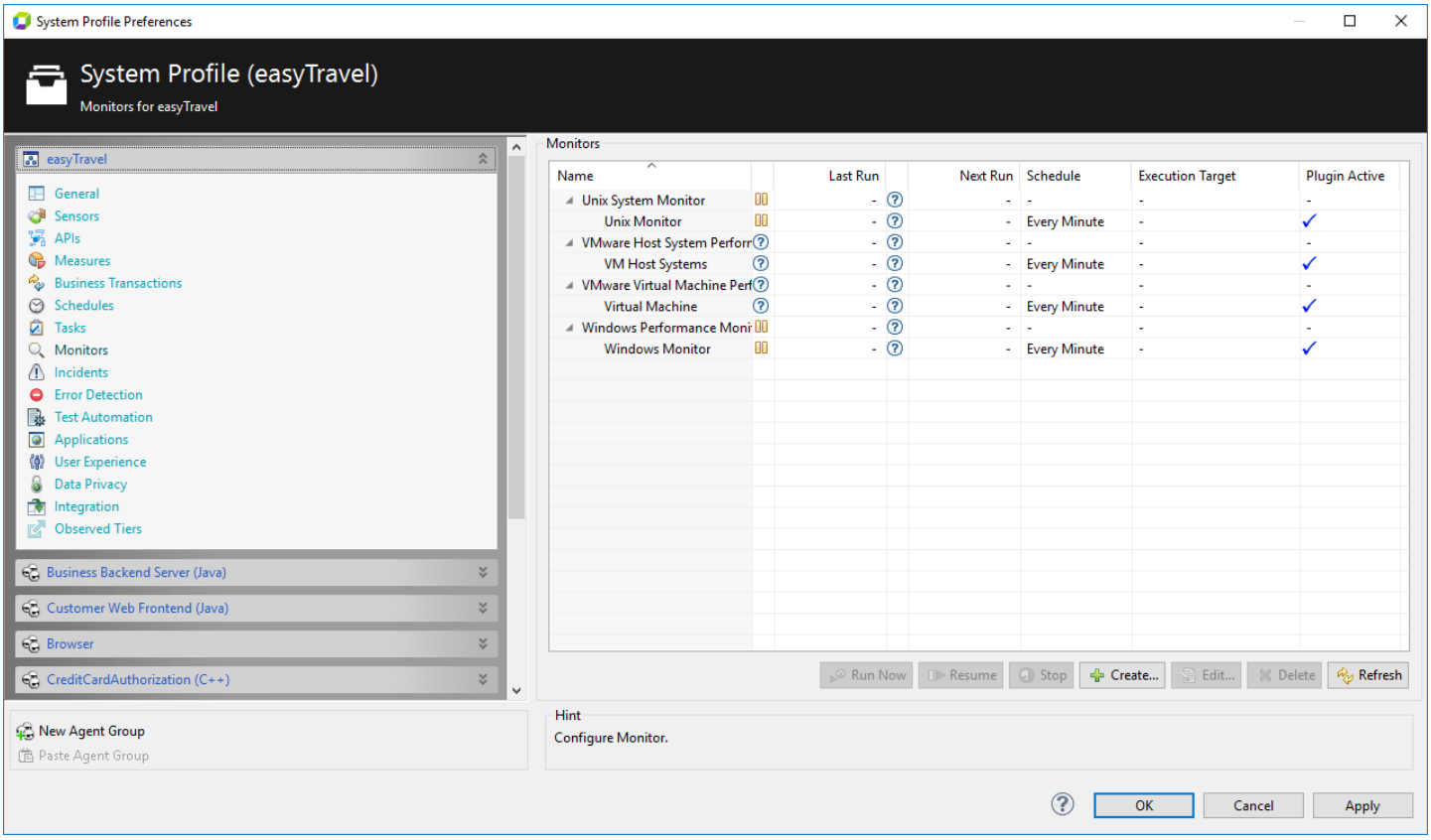

Customer Web Frontend (Java)
Browser
CreditCardAuthorization (C++)
Run Now
Suspend
Stop
Create...
Edit...
Delete
Refresh
New Agent Group
Paste Agent Group
Hint
Configure Task.
OK
Cancel
Apply

System Profile - Monitors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

System, host and availability monitoring



Provides management of Monitors currently existing in the System. This includes actively scheduled Monitors as well as disabled Monitors.

Operation	Description
Run Now	Run the selected Monitor once. Does not schedule the Monitor for repeated execution if it is currently stopped.
Resume/Suspend	Schedule the selected Monitor for repeated execution using Schedule assigned to the Monitor during Monitor creation, or suspend the Monitor execution.

Stop	Stop the currently selected Monitor's execution. If the Monitor is currently scheduled for repeated execution, this schedule is removed.
Create	Create a new Monitor based on built-in Monitors and user-defined Plugins. The Monitor is assigned a schedule during creation for repeated execution, as well as an optional execution target. Monitor parameters depend on the Monitor type. For example, a Web-Transaction Monitor requires an execution script.
Edit	Edit the selected Monitor.
Delete	Delete the selected Monitor.
Refresh	Refresh the list of Monitors. Updates the state of the displayed Monitors.

System, host and availability monitoring

Follow the steps described in Infrastructure Monitoring Walkthroughs to complete the System, Host, and Availability Monitoring configuration.

Get familiar with the measures the following monitors provide, how to complete the configuration, and how to analyze the results:

- Windows Performance Monitor to monitor your Windows environments.
- Unix System Monitoring to monitor Unix-based environments.
- SNMP Monitor to collect metrics via SNMP.
- VMware to monitor Guest, Virtual Machine and Physical Host.
- URL Monitor to monitor availability and performance of your application.
- Web Transaction Monitor to monitor a more complex synthetic web transaction.

See Manage and Develop User Plugins for information on creating your own monitors.

System Profile - Error Detection

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Alerts

Rules

Configure error detection rules

Related pages

Error analysis

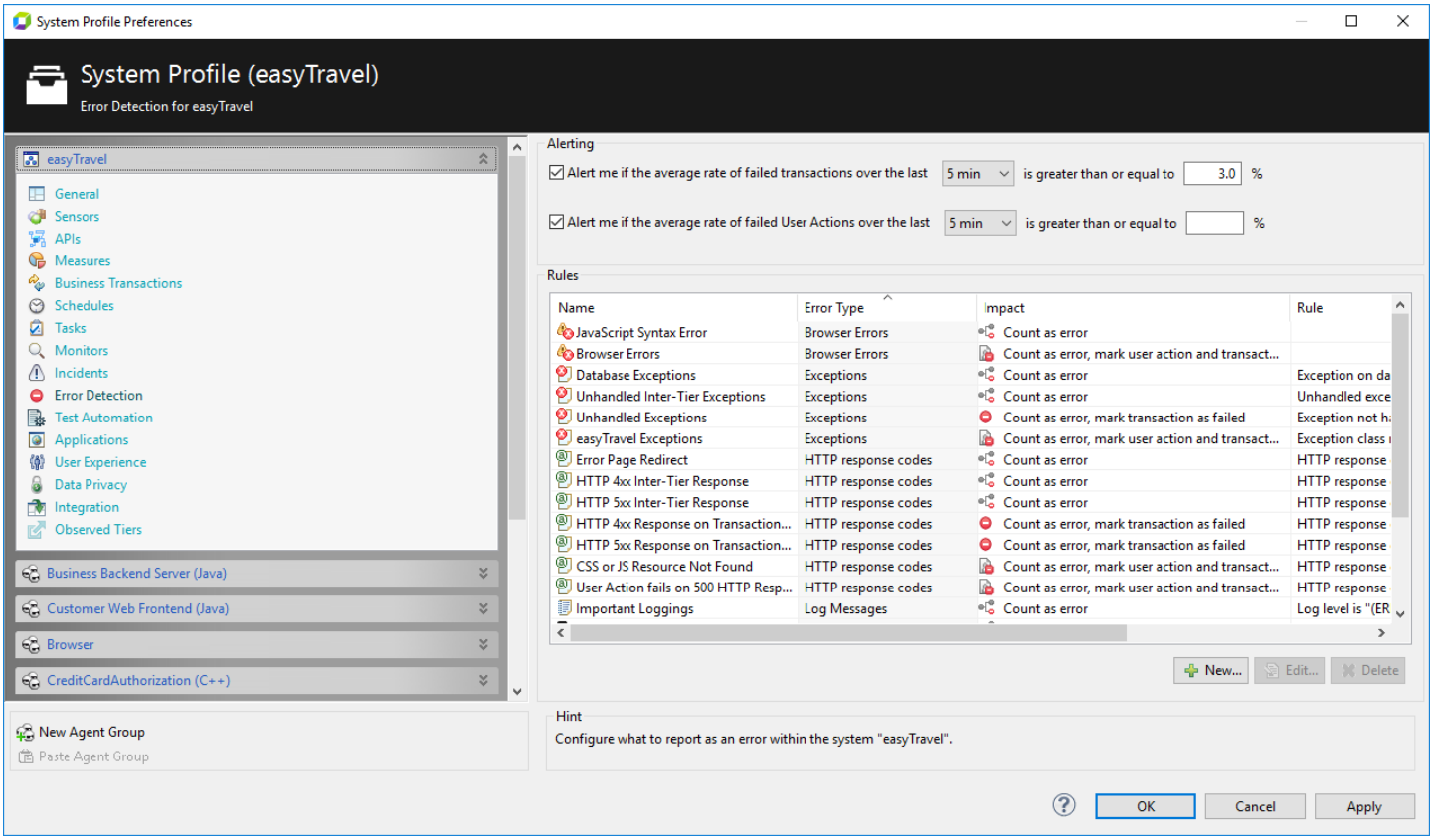
Watch and learn

Error detection

Use the **Error Detection** item of the **System Profile Preferences** dialog box to configure the following items:

- Thresholds for when to alert for overall failed transactions or user actions.
- Rules for what to report as an error and with what impact.

To access, right click your system profile and select **Edit System Profile > Errors Detection**. See Error Analysis for more information.



Alerts

You can configure two kinds of alerts directly from the **Error Detection** tab. These are alerts based on the rate of failed transactions or user actions within a certain time frame. Use incidents based on failure measures for more advanced alerts.

Both alerts (High Overall Failed Transaction Rate and High Overall Failed User Action Rate) are enabled by default with a time range of five minutes and a threshold of 3%, that trigger an e-mail to the Incident Email Group.

Rules

This lists all existing rules that trigger error reports. You can adapt several out-of-the-box rules to match custom requirements or you can create your own rules.

The following Error Detection rules types are based on captured PurePaths data:

- HTTP Response Code Rules
- Logging Rules
- Java/.NET Exception Rules

- Browser Error Rules
- Mobile Error Rules

Each error detection rule defines on of the following impacts:



The error is simply detected and does not have further consequences.



The error indicates that the whole transaction (PurePath) where the error was detected has failed.



The error indicates that the whole PageAction (including the transaction) to which the PurePath belongs has failed.

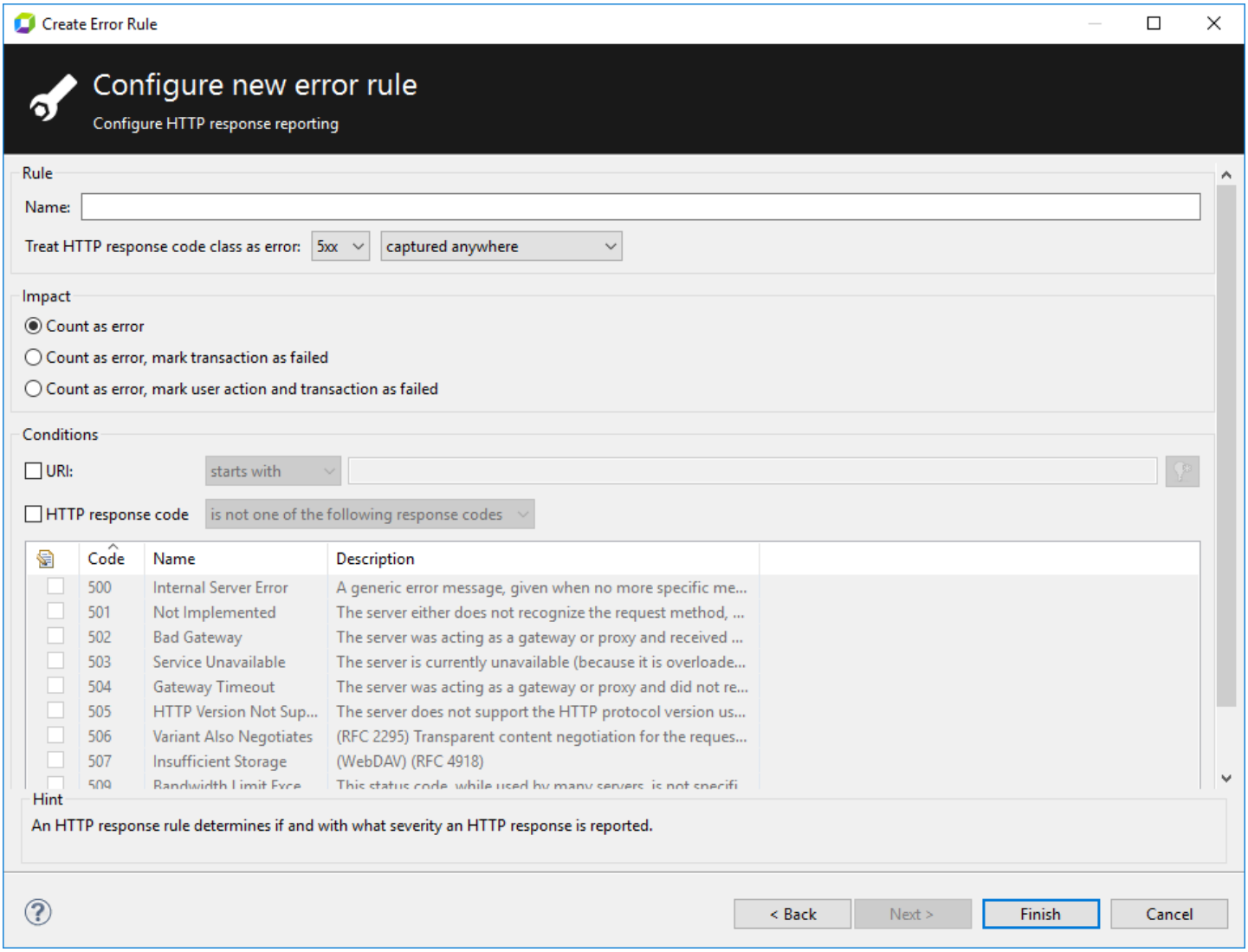
Configure error detection rules

For each rule, specify:

- Type.
- Unique name.
- Impact. See impacts table above.
- Conditions. See detailed descriptions below for your particular rule type.

> **Note**
>
> To trigger an error *all* conditions must be fulfilled.

HTTP response code rules

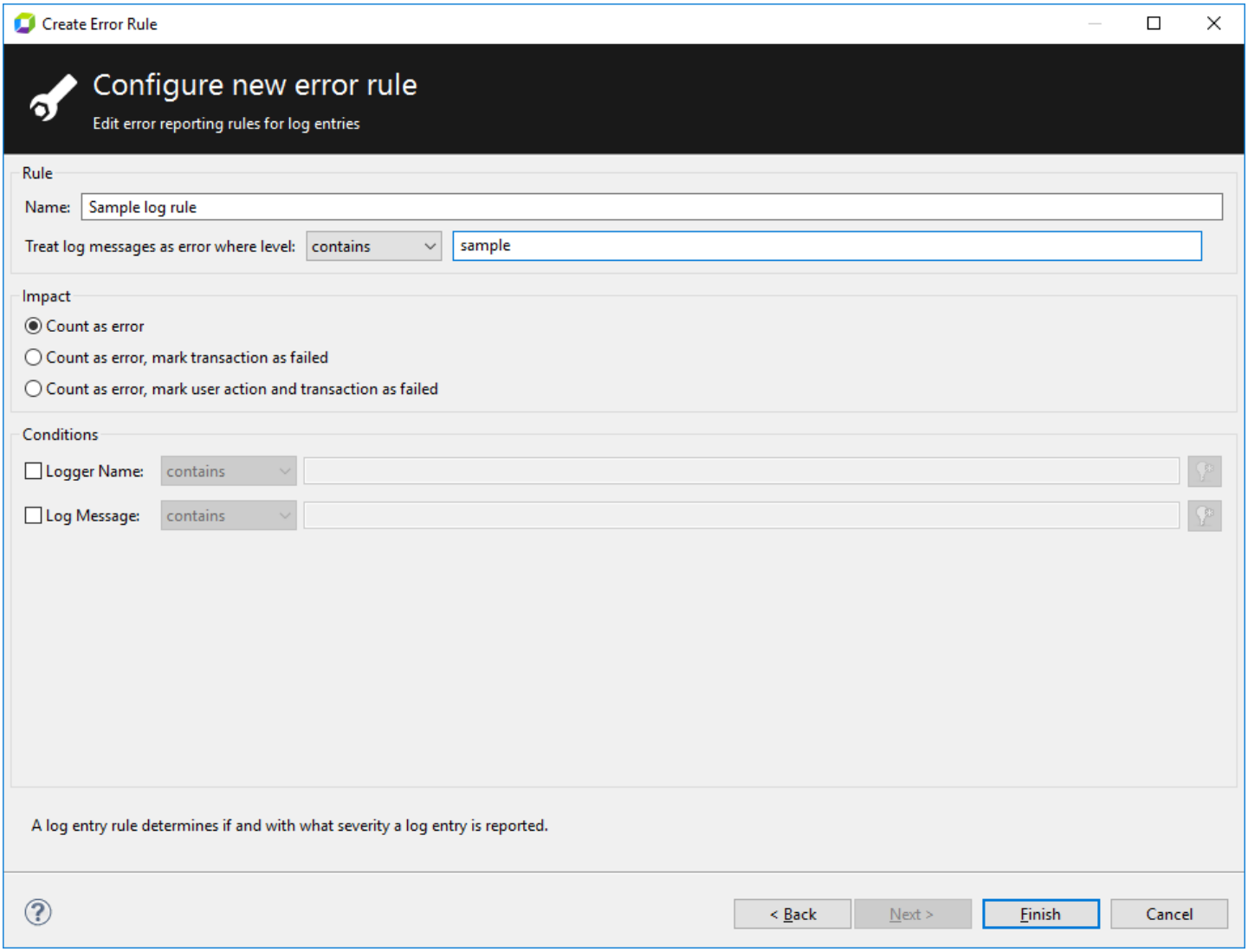


HTTP response code rules are based on data captured by the Servlet Sensor. For each rule in the **Rule** group box, select target HTTP response class, which is classified as an error, and where it should be captured. You can use response code captured on the web server, on transaction entry, or all of them.

The following conditions can be evaluated for HTTP response code rules:

- **URI**: Limits the rule to a specified URI or URI pattern.
- **HTTP Response Code**: Includes or excludes specific HTTP response codes.
- **Redirect Target (class 3xx only)**: Limits the rule to detect errors only if the request has been redirected to a specified target or target pattern.
- **Referrer (class 4xx only)**: Limits errors detection for requests coming from a specified referrer or referrer pattern.

Logging rules

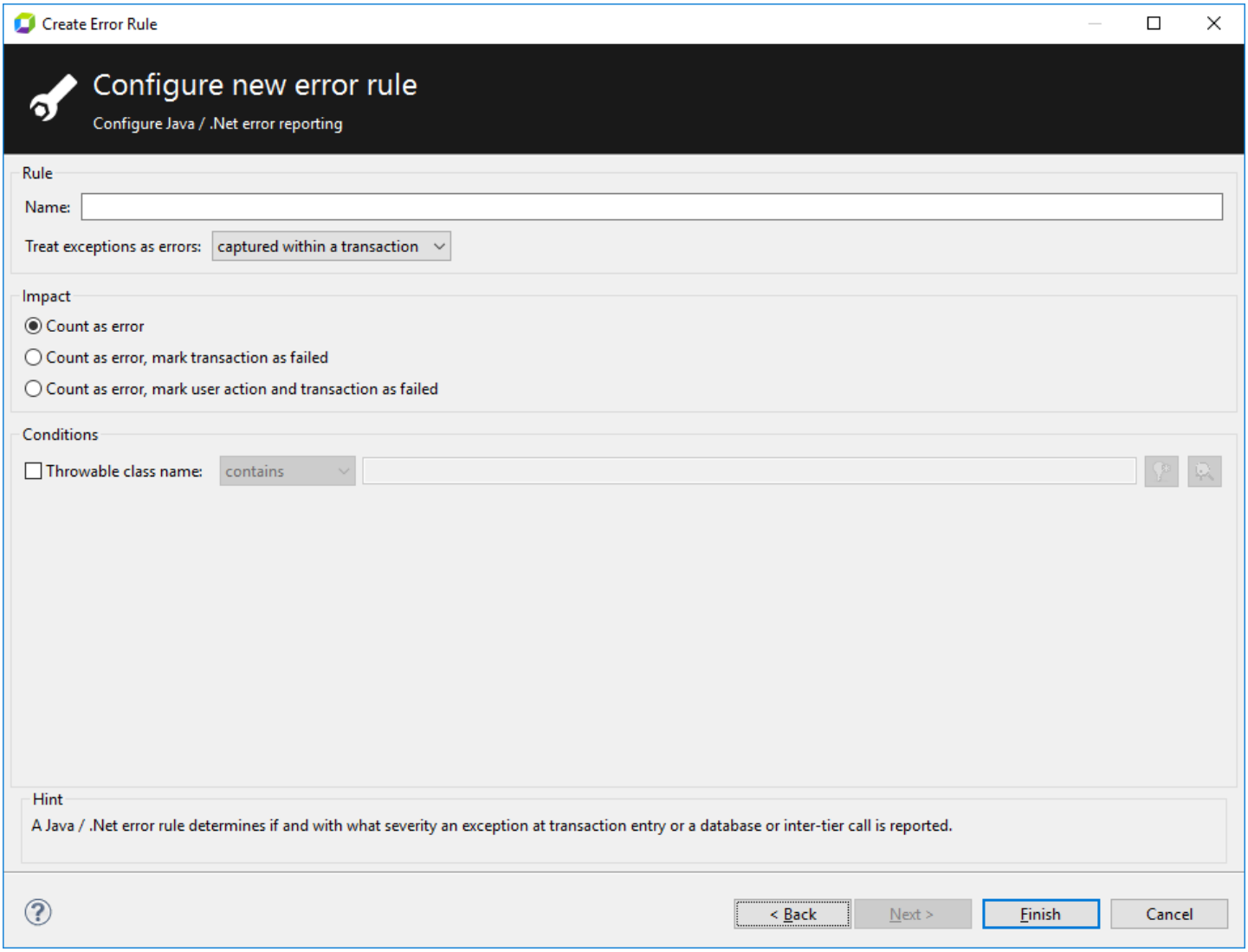


Logging rules are based on captured log statements. You can restrict rule usage to a particular log level. Specify it in the **Rule** group box. All other levels are ignored. For the selected log entries additional conditions can be applied:

- **Logger Name**
- **Log Message**

You can specify a particular logger or message, or templates for them.

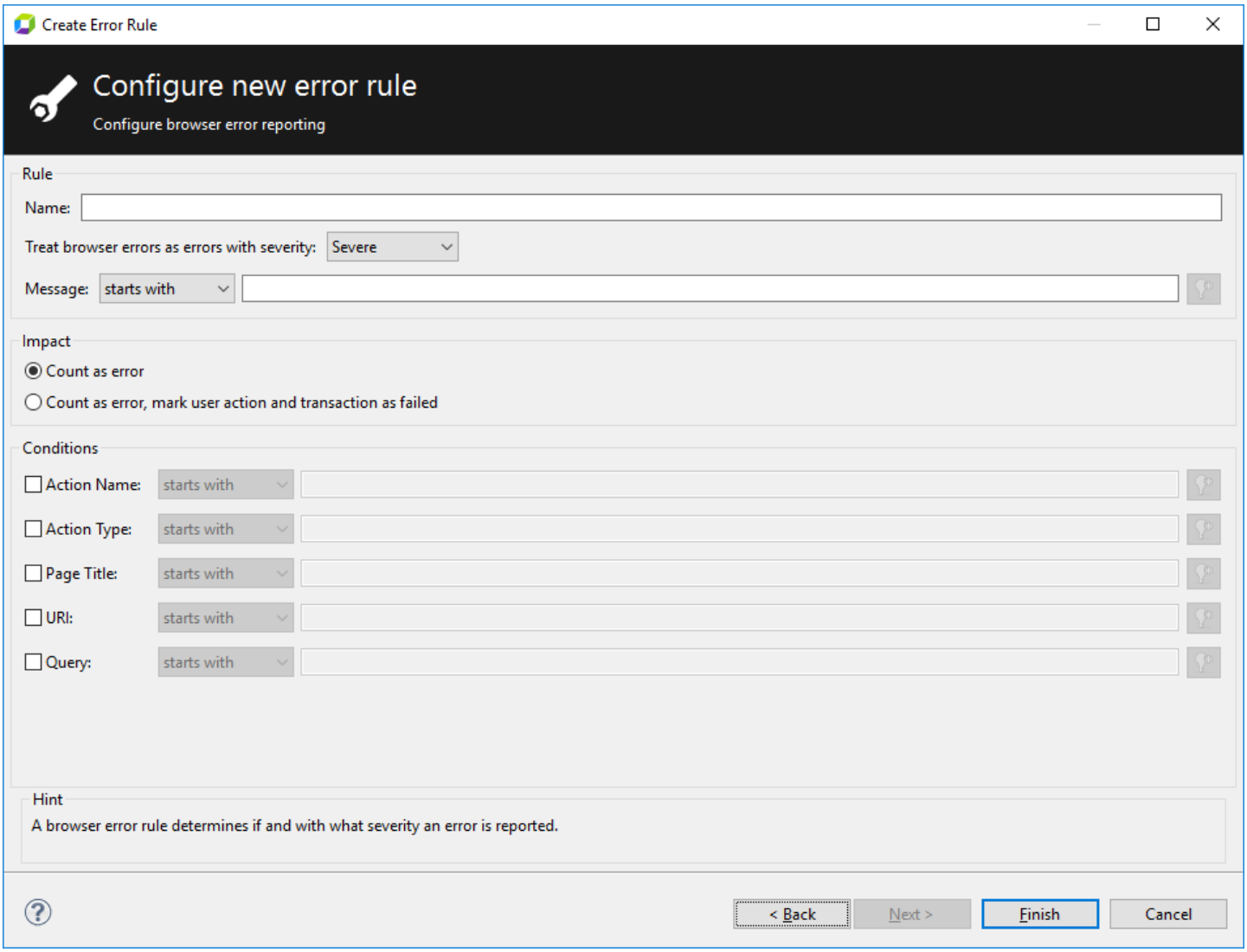
Java/.NET exception rules



Java/.NET exception rules are based on captured Java/.NET exceptions. Use the **Rule** group box to select where the exceptions, which are classified as an error, should be captured for each rule. The following options are available:

- **Captured within a transaction**: All exceptions captured by the Java or .NET exception sensor are treated as an error. For these exceptions, you can specify an additional condition or pattern for the applicable **Throwable class name**.
- **Leaving a transaction**: When the entry point of a transaction (PurePath) returns by throwing any exception this rule matches. Capturing the exception by a sensor is not required to detect such exceptions.
- **Occurring on inter-tier calls**: Matches when a call is made from tier A to tier B within a transaction and the call-initiating method on tier A or the first method on tier B throws an exception.
- **On database calls**: Matches when a database call throws any exception, captured by one of our database sensors.

Browser error rules

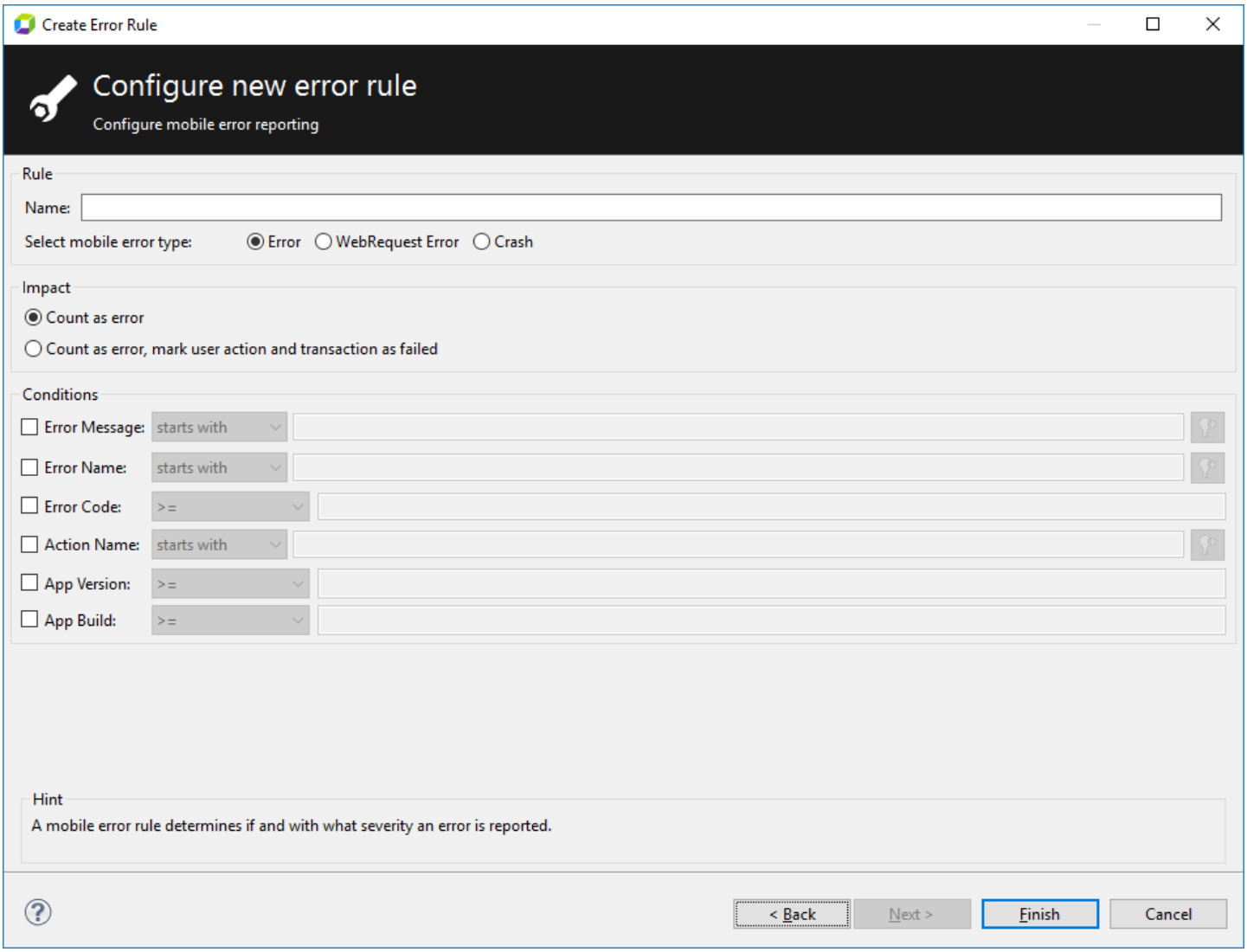


Browser error rules are based on the JavaScript errors, detected on the client side by the User Experience Sensor. Use the **Rule** group box to select browser error severity, which is classified as an error, for each rule. All other browser errors are ignored. You can also restrict rule usage to a particular messages. You can apply additional conditions for the selected browser errors:

- **Action Name**
- **Action Type**
- **Page Title**
- **URI**
- **Query**

For each of them you can specify a particular entry, or a template.

Mobile error rules



Mobile error rules are based on error events sent by the Mobile App ADK. Use the **Rule** group box to select which mobile error events should be classified as an error for each rule. Each of them has its own set of conditions:

Conditions for the **Error** type:

- **Error Message**: Do not set it if you want to match an error code.
- **Error Name**
- **Error Code**: Do not set it if you want to match an error message.
- **Action Name**
- **App Version**
- **App Build**

Conditions for the **WebRequest Error** type:

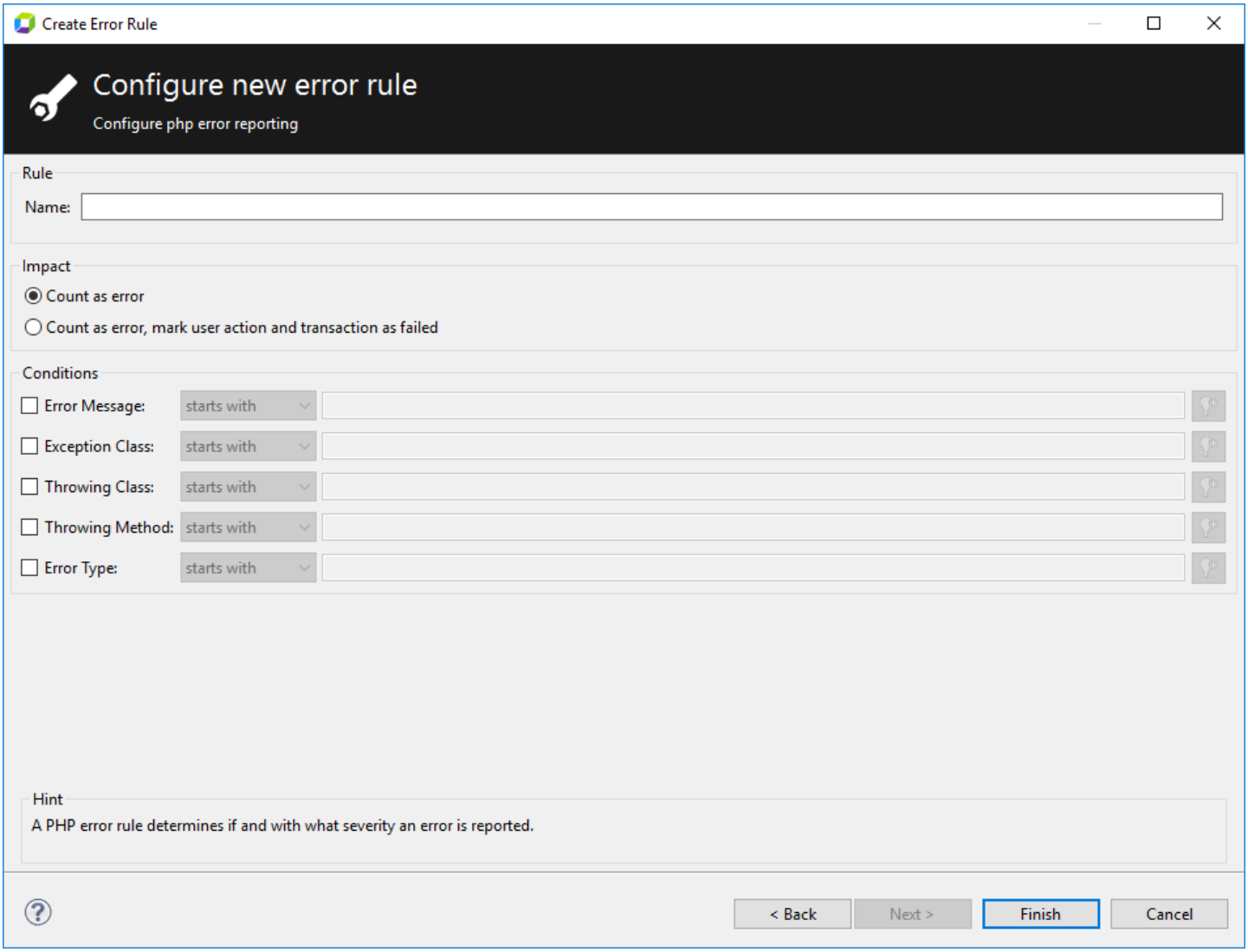
- **URL**
- **Error Message**: Do not set it if you want to match an error code.
- **Response Code**: HTTP status code of failed web request.
- **Action Name**
- **App Version**: `CFBundleShortVersionString` for iOS and `versionName` for Android apps.
- **App Build**: `CFBundleVersion` for iOS and `versionCode` for Android apps.

Conditions for the **Crash** type:

- **Reason**
- **Exception Name**
- **App Version**: `CFBundleShortVersionString` for iOS and `versionName` for Android apps.
- **App Build**: `CFBundleVersion` for iOS and `versionCode` for Android apps.

For each of them you can specify a particular entry, or a template.

PHP error rules



PHP error rules are based on errors detected on the client side by the PHP Sensor. The following conditions are applied:

- **Error Message**
- **Exception Class**
- **Throwing Class**
- **Throwing Method**
- **Error Type**

For each of them you can specify a particular entry, or a template.

System Profile - Applications

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

How to define boundaries for logical applications

Using logical applications

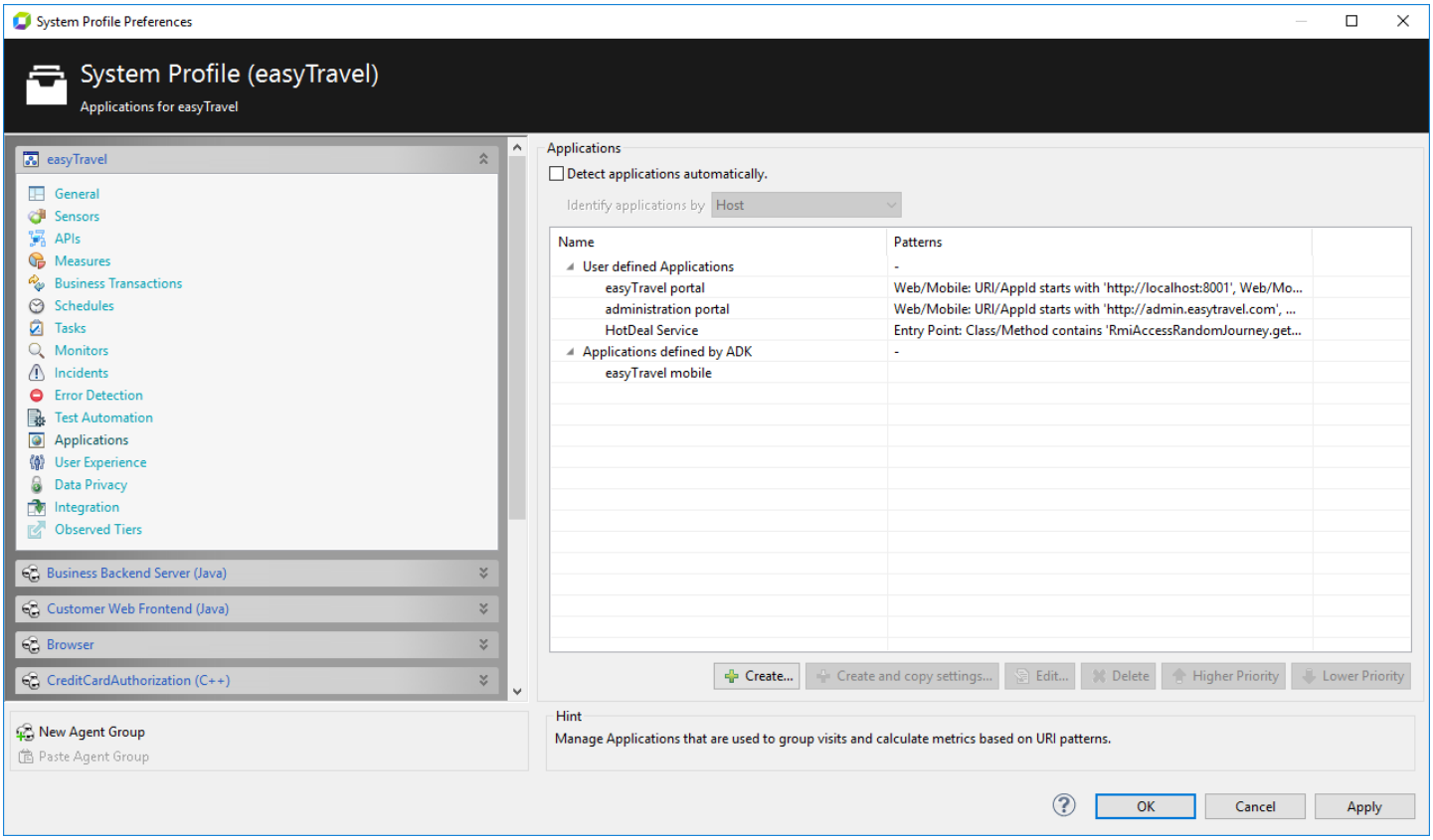
Watch and learn

Defining logical applications

Use the **Applications** item of the **System Profile Preferences** dialog box to manage logical applications within a single System Profile.

To access it, right click your system profile, and select **Edit System Profile > Applications**.

For example, if you host your public website, customer portal and blog on the same server and thus have them in one System Profile, you may still want to monitor and manage performance of these individually by manually grouping them by sub-domain.



How to define boundaries for logical applications

Auto-detected vs. user-defined logical apps

You have the option to do one of the following:

- **Let AppMon automatically detect applications** by selecting the **Detect applications automatically** check box and identifying applications based on host and / or web application ID.
- For Java and Web Server Agents, you can **manually define logical applications** based on URI sub-domain patterns like `blog.dynatrace.com`, `community.dynatrace.com`, ports used, or mobile app name. You can also define logical applications by using PurePath properties like entry point or messaging.
  You can find configured patterns in the easyTravel System Profile, and copy settings from auto-detected applications.

Note

Auto-detection mapping is done on the Server, but mappings created manually by defining logical applications is done on the Agent. The manual mapping is only implemented for Java and Web Server Agents. Manually defining logical applications for PHP only works if there is an instrumented Web Server tier before the PHP tier.

Manually defined logical applications

Logical applications are defined by patterns, described below.

Entry Point pattern

The fully qualified method name of the entry point is compared with the Entry Points patterns. The topmost matching pattern applies. This method has priority over Messaging and Web/Mobile patterns.

Note

Only the PurePath entry point is used, no arbitrary nodes or sub-paths.

Messaging pattern

The queue manager name or destination name field detected by a Messaging Sensor is compared to the Messaging patters. The topmost matching pattern applies. This method has priority over Web/Mobile patterns.

Web/Mobile pattern

The web request URL or AppId, detected by a Web Request Sensor, is compared to the Web/Mobile patterns. The topmost matching pattern applies.

In instrumented apps, the full URL of the web request is split into Host and URL fields. For expample, the `https://www.dynatrace.com/support/doc/appmon/` web request will be split to `www.dynatrace.com` host and `/support/doc/apppmon/` URL.

Rule evaluation order

The evaluation order of the different mechanisms is:

1. Entry point: only analyzed on the PurePath entry point; has priority over everything else.
2. Messaging: analyzed on the whole PurePath, topmost node wins; has priority over Web Request.
3. Web request: analyzed on whole PurePath, topmost node wins.

If multiple applications match with the same technology priority due to overlapping patterns, the topmost application as ordered in System Profile / Applications / User-defined Applications is used.

See Logical Application for more information on applications, configuration, and more use case scenarios.

Creating patterns

To create a pattern for logical application:

1. In the AppMon Client, right click the System Profile and select **Edit System Profile > Applications**.
2. Click **Create**.
3. In the **Name** edit box, type the name for your pattern.
4. Select the pattern type, the entity for comparison, and the comparison rule from the appropriate lists.
5. Specify the pattern rule.
6. Click **+** to add the rule to the pattern.
7. If needed, add more rules.
8. Click **Create**.
9. By default the newly created pattern has the lowest priority. Specify the priority of the pattern by using the **Higher Priority** and **Lower Priority** buttons.

Using logical applications

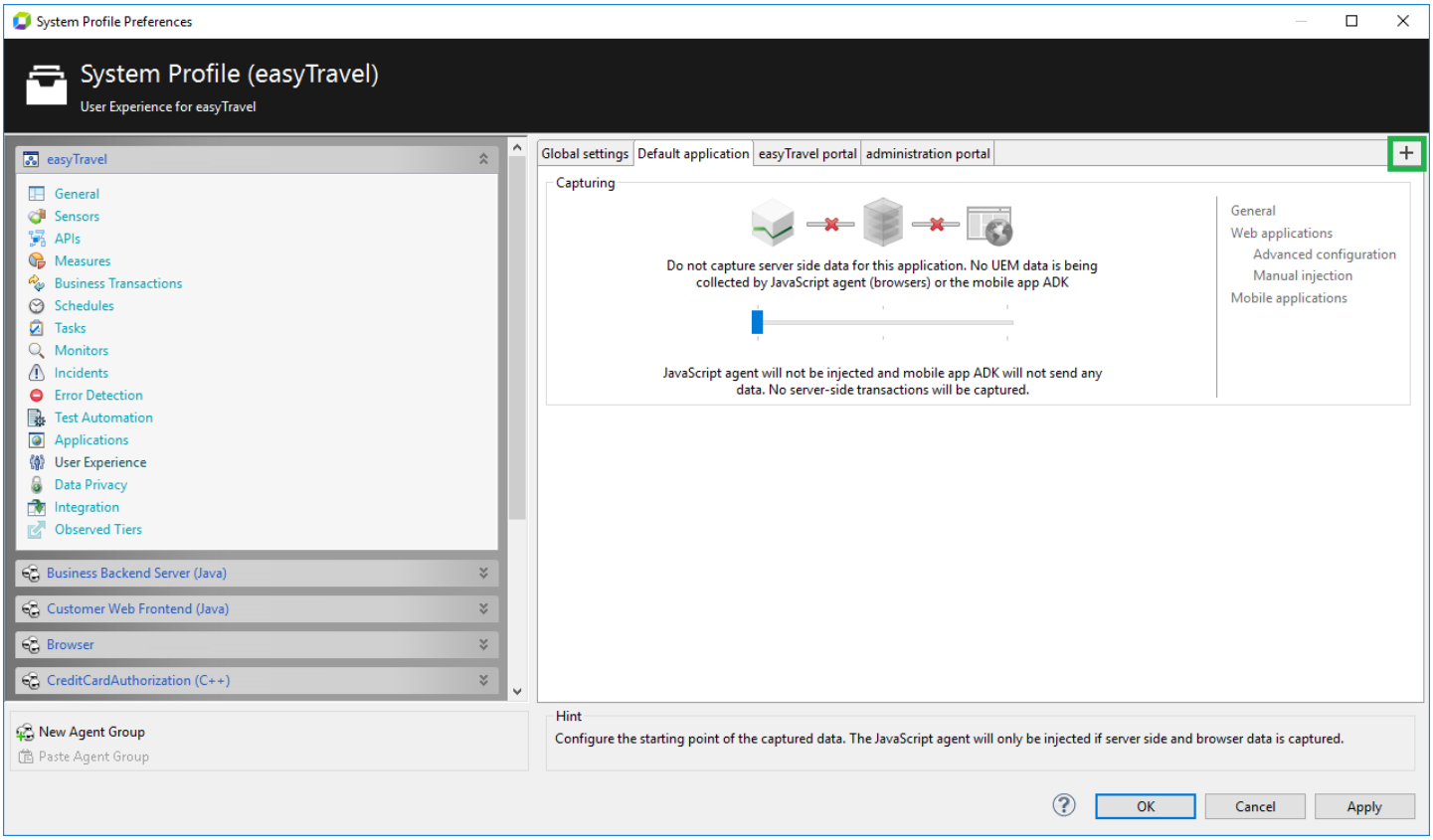
Set data capturing mode per application

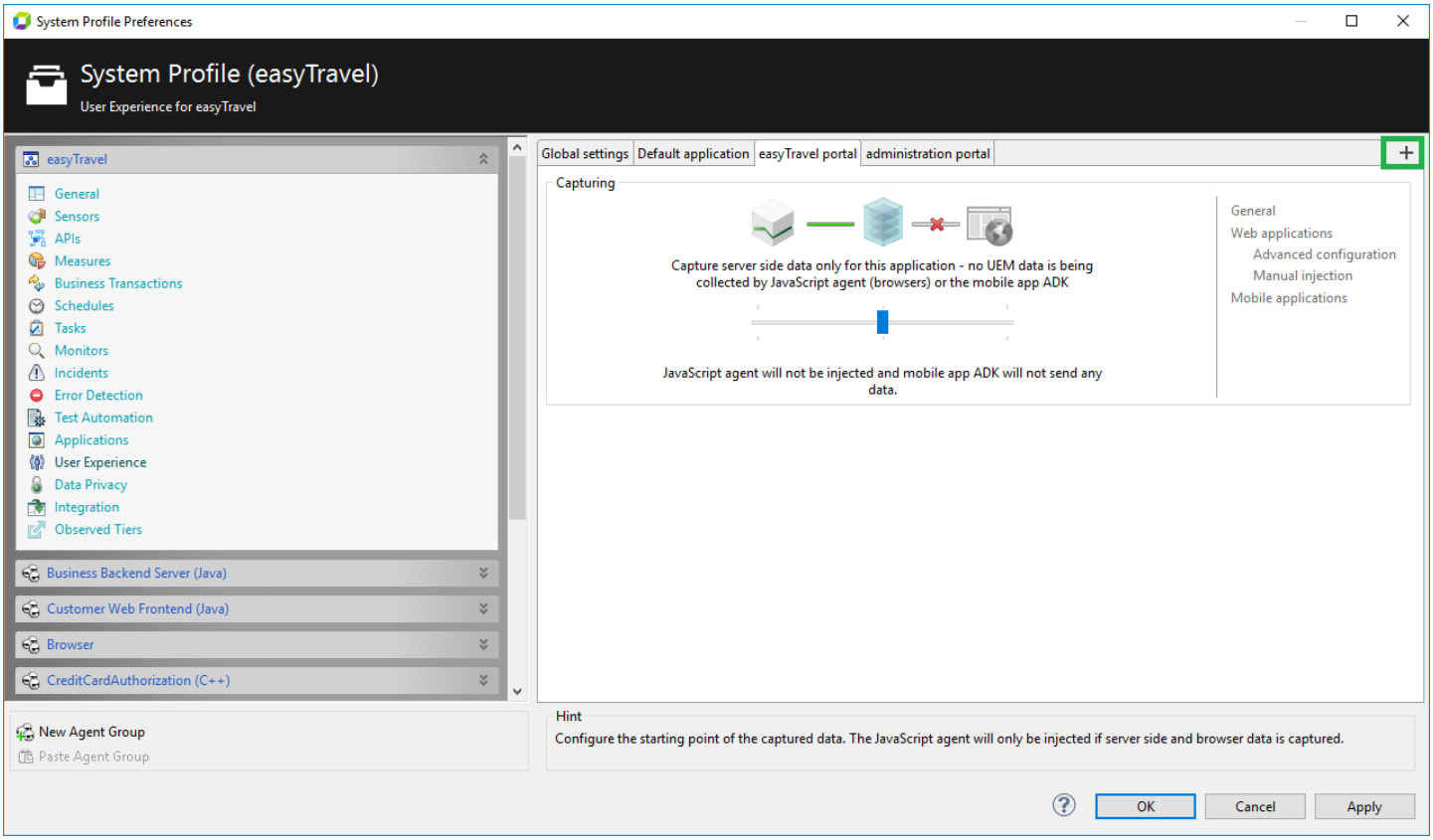
With (logical) applications defined you can adapt for each one what data are being captured. The screenshots below show three applications defined.

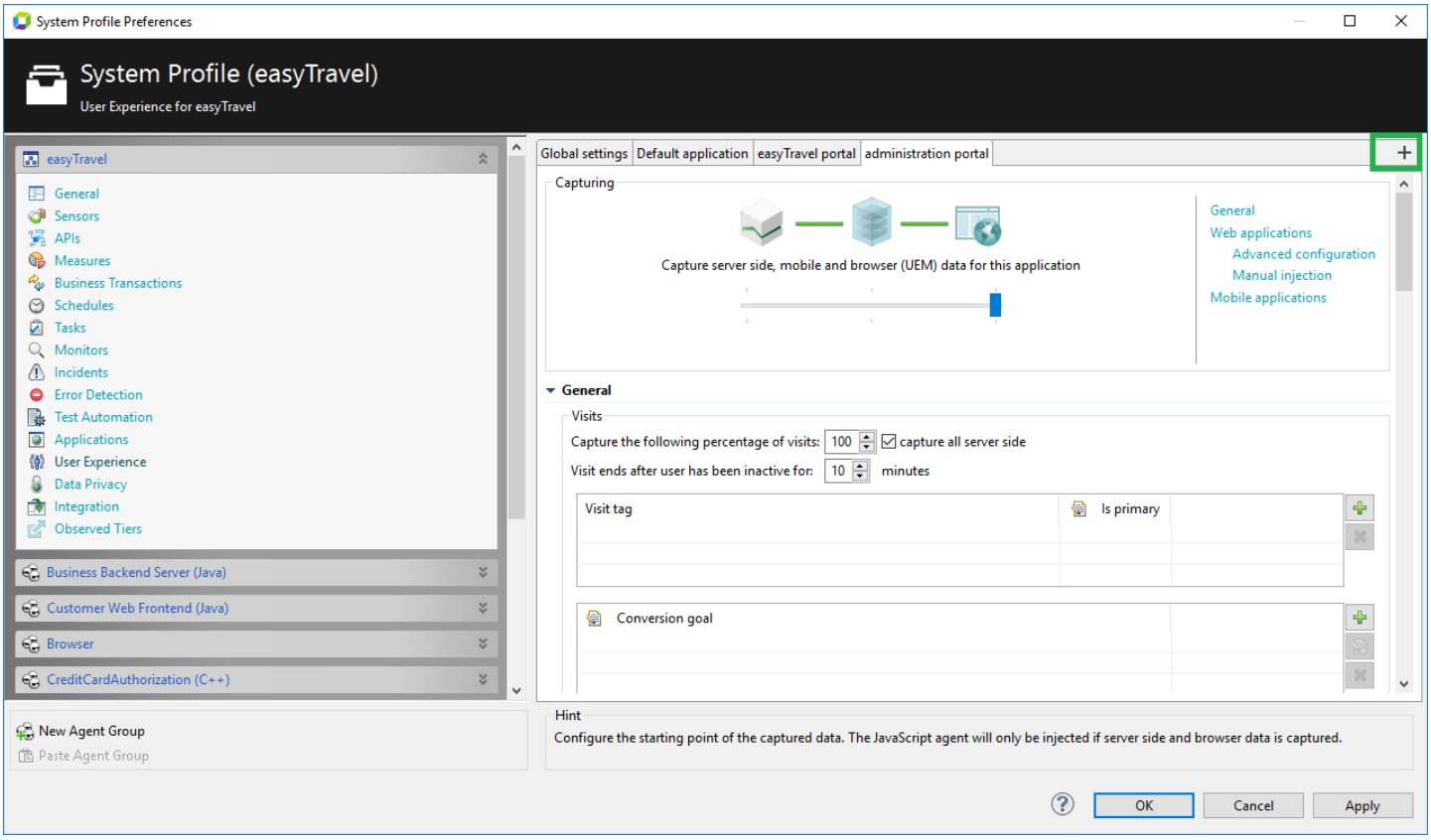
- Two are user-defined logical ones discriminated by URI.
- One is the easyTravel mobile app.

In the **User Experience** item:

- For the administration portal, the setting is to capture all data, including server-side and client-side data.
- For the easyTravel portal, only web and application-server-side data are captured.
- For the default application no data are set to be captured. Because there is no extra setting for easyTravel mobile, the default (no capturing) applies. To create an extra tab for different settings for easyTravel mobile, click the plus icon.




System Profile Preferences
System Profile (easyTravel)
User Experience for easyTravel
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Test Automation
Applications
User Experience
Data Privacy
Integration
Observed Tiers
Business Backend Server (Java)
Customer Web Frontend (Java)
Browser
CreditCardAuthorization (C++)
Global settings
Default application
easyTravel portal
administration portal
Capturing
Capture server side data only for this application - no UEM data is being collected by JavaScript agent (browsers) or the mobile app ADK
JavaScript agent will not be injected and mobile app ADK will not send any data.
General
Web applications
Advanced configuration
Manual injection
Mobile applications
New Agent Group
Paste Agent Group
Hint
Configure the starting point of the captured data. The JavaScript agent will only be injected if server side and browser data is captured.
OK
Cancel
Apply


System Profile Preferences
System Profile (easyTravel)
User Experience for easyTravel
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Test Automation
Applications
User Experience
Data Privacy
Integration
Observed Tiers
Business Backend Server (Java)
Customer Web Frontend (Java)
Browser
CreditCardAuthorization (C++)
Global settings
Default application
easyTravel portal
administration portal
Capturing
Capture server side, mobile and browser (UEM) data for this application
General
Web applications
Advanced configuration
Manual injection
Mobile applications
General
Visits
Capture the following percentage of visits: 100
capture all server side
Visit ends after user has been inactive for: 10
minutes
Visit tag
Is primary
Conversion goal
New Agent Group
Paste Agent Group
Hint
Configure the starting point of the captured data. The JavaScript agent will only be injected if server side and browser data is captured.
OK
Cancel
Apply

Note

Applications detected by web application ID cannot be used for application-specific settings, because this detection mechanism is not implemented on Agents, but only on the AppMon Server side.

Measures by application

You can use the defined logical applications to split measures and restrict calculation to selected apps. To do this:

1. In the AppMon Client, right click you system profile and select **Edit System Profile > Measures**.
2. Select the required measure and click **Edit**, or click **Create Measure** to create a new one.
3. Click the **Details** tab.
4. To split measures, in the **Measure Splitting** field, select the **Create a measure for each application** checkbox.
5. To restrict calculations, in the **Restrict Measure Calculation** field, select the **calculate only for selected applications** radio button, and select the required applications.

Filter dashboard content by application

You can filter content of any dashboard by your logic application. Do to this, click the **Filter by Application**



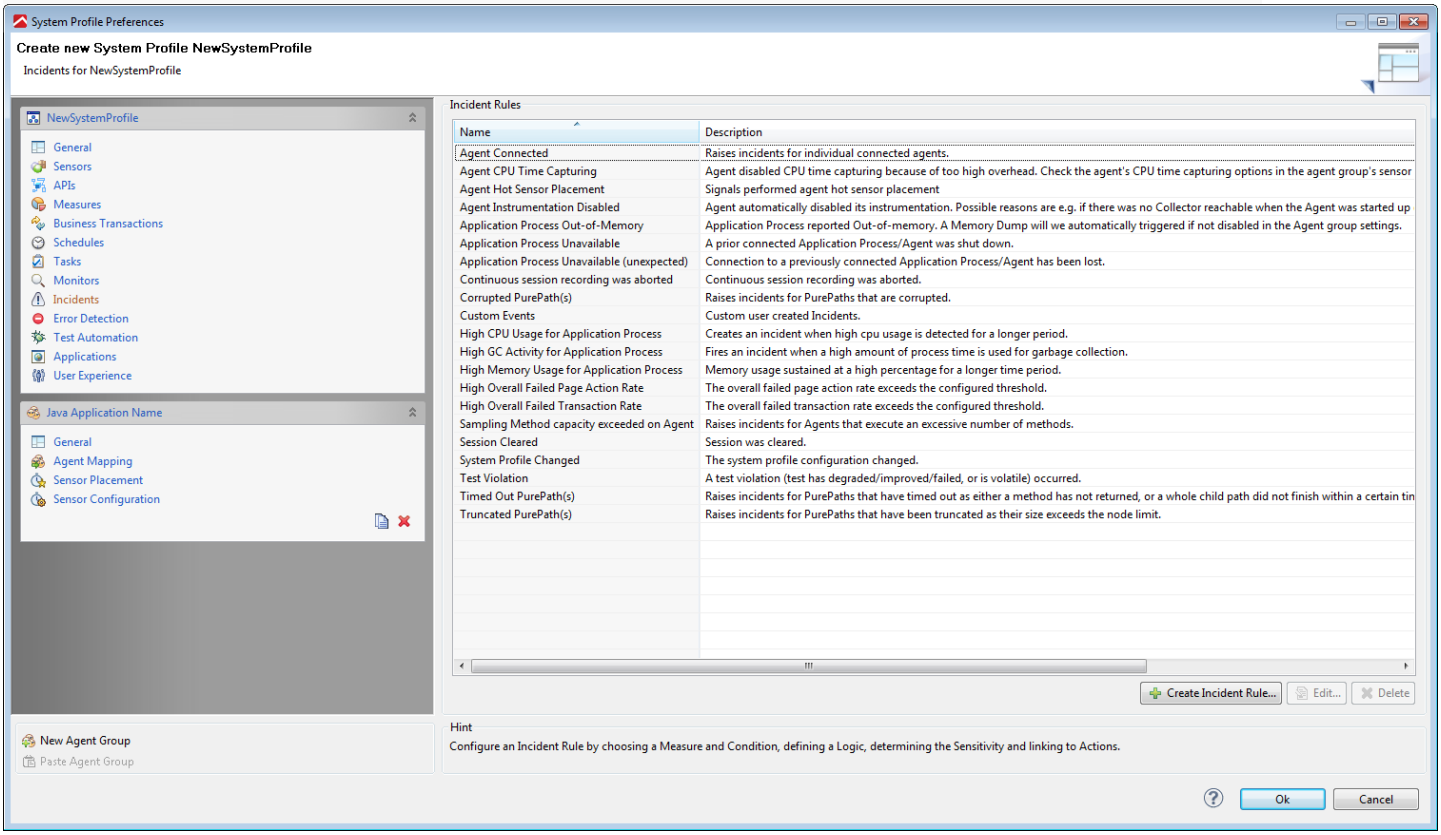
icon, and select the required applications.

System Profile - Incidents

See Incidents and Alerting for details on configuring incident rules for a System Profile. See Incidents Overview Dashlet for details on managing incident information across all System Profiles.

A disabled (gray) entry indicates the incident rule is inactive, so no incident is fired if the rule is violated. Right-click to check or clear **active**, or use the space bar to toggle the active state.

Operation	Description
Create Incident Rule	Create a new Incident Rule based on measures of the System Profile.
Edit	Edit the currently selected Incident Rule.
Delete	Delete the currently selected Incident Rule.



System Profile - Test Automation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Test automation explained

UI-driven browser test

Section	Description
Enable Browser/UI-Driven Tests	Select this check box to allow UI-driven browser testing. See the section on Configuration in UI-driven browser test for details.
Configure UEM application for Tests	Select the test application and click **Configure** to configure all required UEM settings. See the section on Configuration in UI-driven browser test for details.
Degraded Runs	• **Number of upper limit violations**: The minimum number of consecutive violations of the upper limit for a test until it is considered degraded. • **Notification**: Send email notifications if a test has degraded.
Improved Runs	• **Number of lower limit violations**: The minimum number of consecutive violations of the lower limit for a test until it is considered improved • **Notification**: Send email notifications if a test has improved.
Volatile Runs	• **Volatile %**: The minimum percentage of volatility for a test to consider it volatile. • **Notification**: Send email notifications if a test is volatile.
FailedRuns	• **Notification**: Send email notifications if a test has failed.
Corridor Settings	• **False Positive %**: The maximum percentage of false positives. Increase this value to be notified earlier for changes, decrease it to reduce the number of notifications caused by false positives. Minimum value is 0.00001, Maximum value is 10. • **Number of Runs**: The overall number of test runs required to calculate false positive percentages. • **Calibration Runs**: The number of runs taken as calibration runs. No incidents or notifications are sent until this minimum number of runs is reached.
Notification Settings	Opens additional settings for the test notification incident.

System Profile - User Experience

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Global settings

Capturing settings

General

Web Applications

Related pages

User experience management

Perceived render time

W3C resource timing metrics

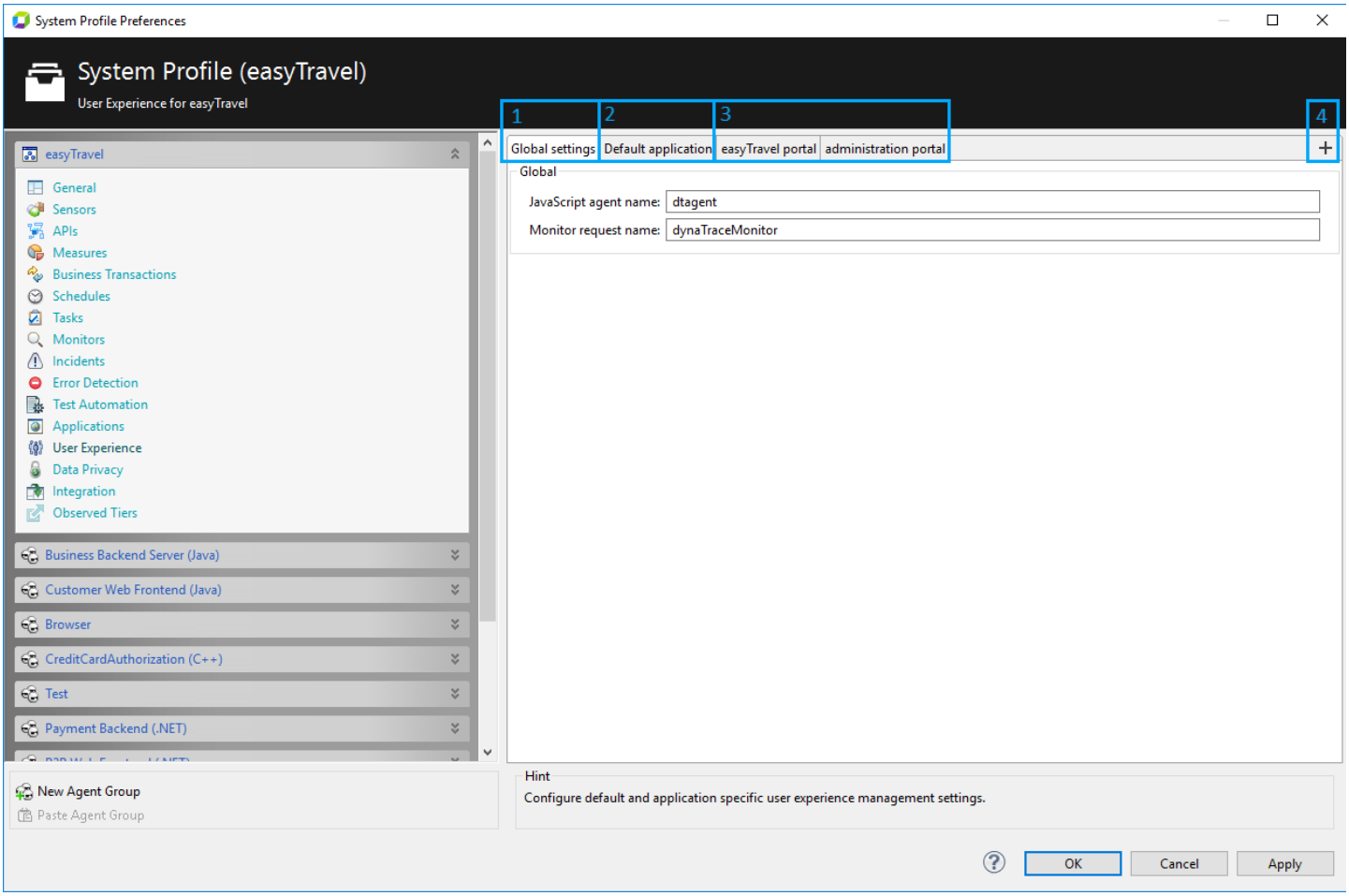
CDN content analysis

Action groups advanced configuration

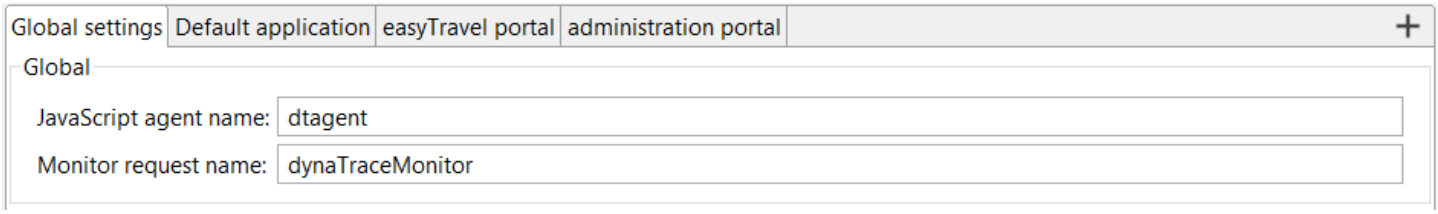
System Profile - Applications

Use the **User Experience** item of the **Edit system profile** dialog box to configure UEM with AppMon. UEM is available for web and mobile native applications. Besides the Global settings (1 in the following graphic), you can configure UEM settings to apply to all applications using Default Application tab, or configure settings in an application-specific tab by adding a specific application, which creates a seperate tab for the application. This lets you customize UEM behavior for each application. For example, if you have two applications and only one of them is using `Angular.js`, you can enable the `Angular.js` sensor for this particular application only.

New system profiles contain a single configuration for the Default application (2). Click the **Add specific application** button (4) to copy the configuration for a specific application (3). Any application that does not have a specific configuration uses the Default application configuration.



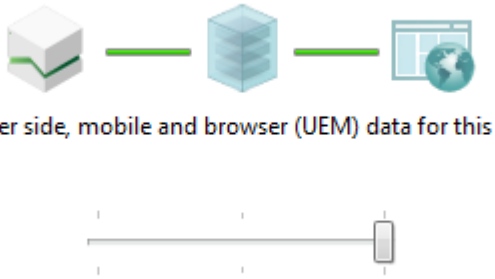
Global settings



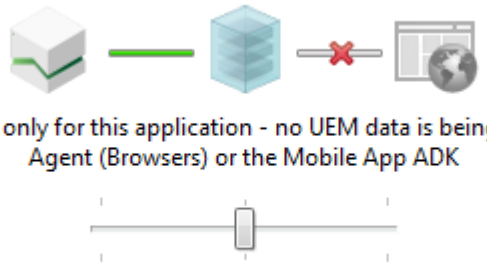
Setting	Description
JavaScript agent name	Specifies the name of the JavaScript Agent file that gets injected. The default is **dtagent**. Usually there is no need to rename it.
Monitor request name	Specifies the endpoint of the monitor request from **dynaTraceMonitor**. Usually there is no need to change it.

Capturing settings

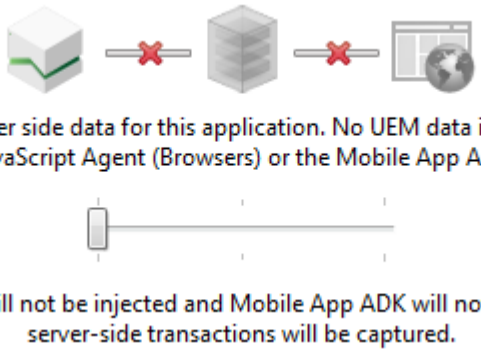
Capturing settings determine the data that is captured: UEM and server-side data, just server-side data, or neither. Drag the slider to change the level of events to capture.



UEM is turned on (default). Web server agents automatically inject the JavaScript Agent tag.



UEM is turned off. Web server agents do not inject the JavaScript Agent tag and send shut down message to manually injected JavaScript Agents and mobile agents.



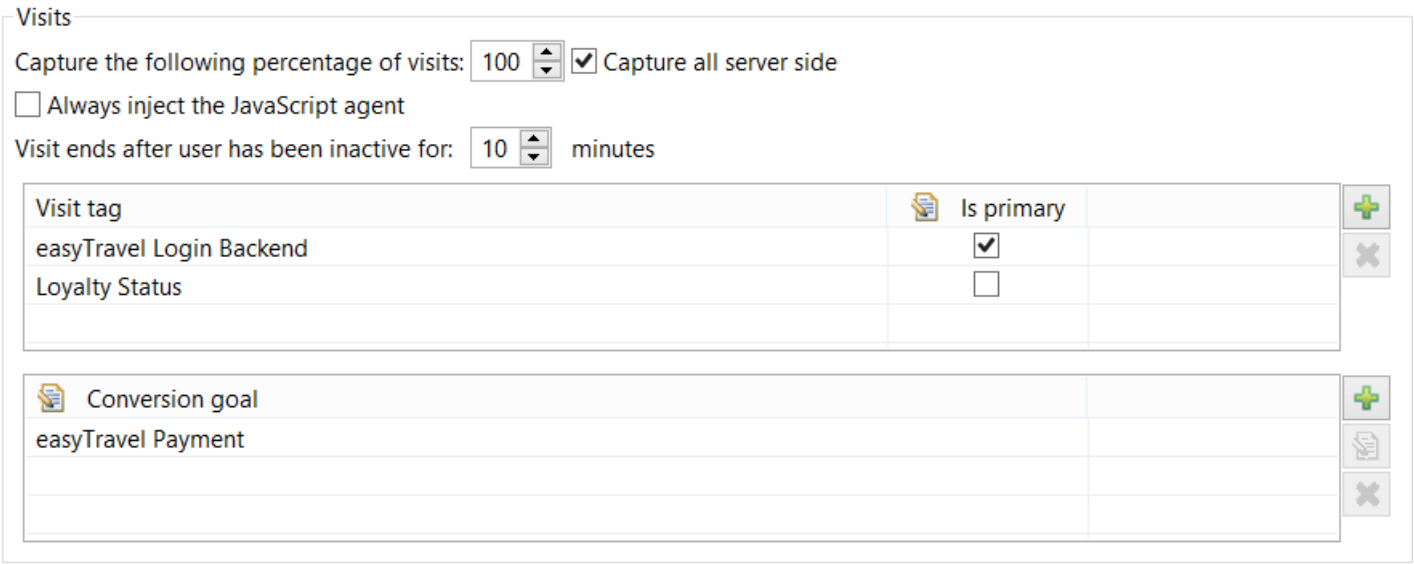
UEM is turned off. Web server agents do not capture web requests. Shut down any manually injected agents. PurePaths are not captured for any incoming web requests.

General

The General section includes settings that affect visit and user action creation.

Visits

General

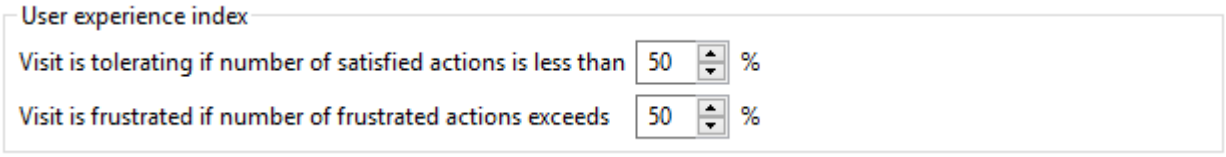


Setting	Description
Capture the following percentage of visits.	Enter a percentage of visits to be captured. For example, if you enter 10% AppMon captures only each 10th visit to your web site. Typically there is no need to change from the default 100%.
Capture all server side	Select this check box to capture server-side PurePaths even when the corresponding visit is not captured.
Always inject the JavaScript Agent	Select this check box if you want the always inject the JavaScript Agent. The monitor signal handles visit percentages when this check box is selected.
Visit ends after user is inactive for X minutes	Lets you configure the time until a visit times out. Visit-based business transactions are calculated and stored in the performance warehouse only after a visit times out. A visit ends after 8 hours or if they contain 1000 user actions. New user actions then create a new visit.
Visit tag	Lets you use the value of a measure as a visit tag. For example, if the user name is stored in a session attribute, you can use a **Web Requests - Session Attribute Value** measure to tag visits with the username. All measures from the **Business Transaction Evaluation/Filter/Grouping Values** category can be used. You can configure any number of visit tags, but one tag must be configured to act as the primary visit tag search availability. Select the **Is primary** check box for the visit tag you want as primary. All other tags then act as additional tags and are not displayed in the visit details. The tag is set only once per visit, so once a valid tag value is determined, it does not change with the re-occurrence of the measure.

Conversion goal	Lets you define conversion goals used to count visits as converted. To define a conversion, use a URI web request matching pattern, a method occurrence, or any measure trigger. Configuring conversion goals is especially useful with e-commerce business applications where the conversion rate (spectator or buyer ratio) should be calculated. For example, if your web-site calls an `orderById` method whenever a user orders an item, you can define a corresponding **Methods - Invocation** measure that identifies a conversion. After you define at least one measure, you can use the **Visits - Conversion Rate** measure (which indicates the conversion rate of any of the conversion measures) and consequently, the conversion count of your visits. There is a separate conversion rate measure created for each measure that is assigned as a conversion measure that only indicates the conversion rate of this specific conversion measure. Default names of these measures are **Conversion Rate for <conversion measure name>**, but you can rename them as you want.

User Experience Index

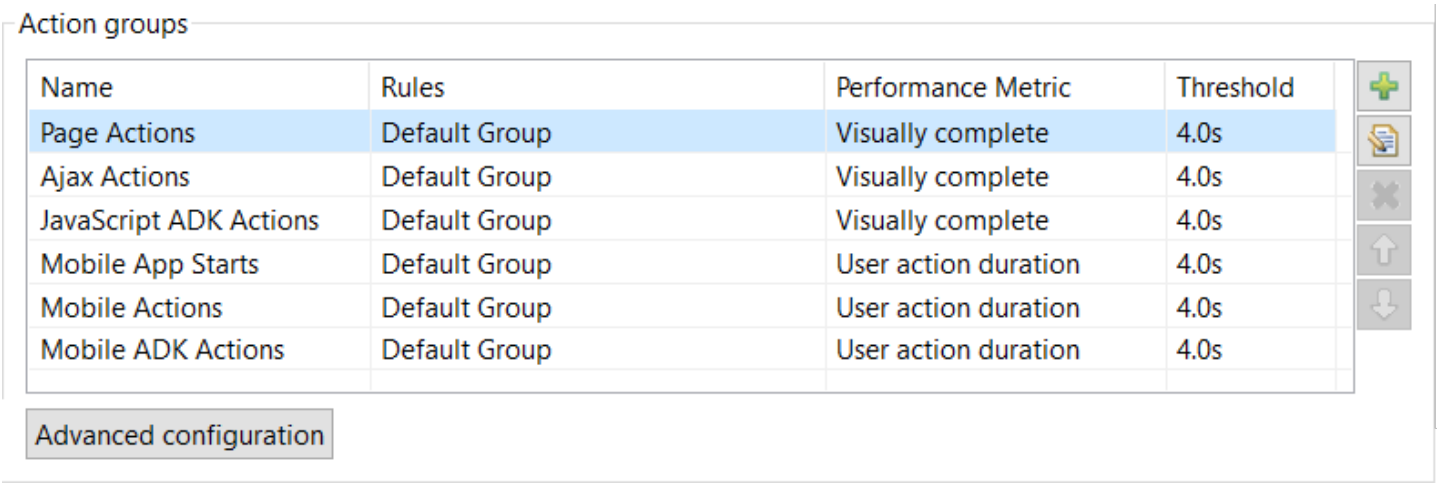
Use this section to customize rules that calculate the user experience index of visits.



Setting	Description
Visit is tolerating if number of satisfied actions is less than	If the percentage of satisfied actions in the visit falls below this threshold, then the visit is marked as Tolerating. The default is 50%.
Visit is frustrated if number of frustrated actions exceeds	If the percentage of frustrated actions in the visit meets or exceeds this threshold, then the visit is marked as Frustrated. The default is 50%.

Action groups

Action groups are a good way to define individual performance baselines for different types of user actions. A new system profile contains six predefined action groups, one for every action type. You can also define custom action groups, such as one that groups all search actions into a custom group defined by a specific threshold. Every user action is assigned to one group by evaluating the action group list from top to bottom.



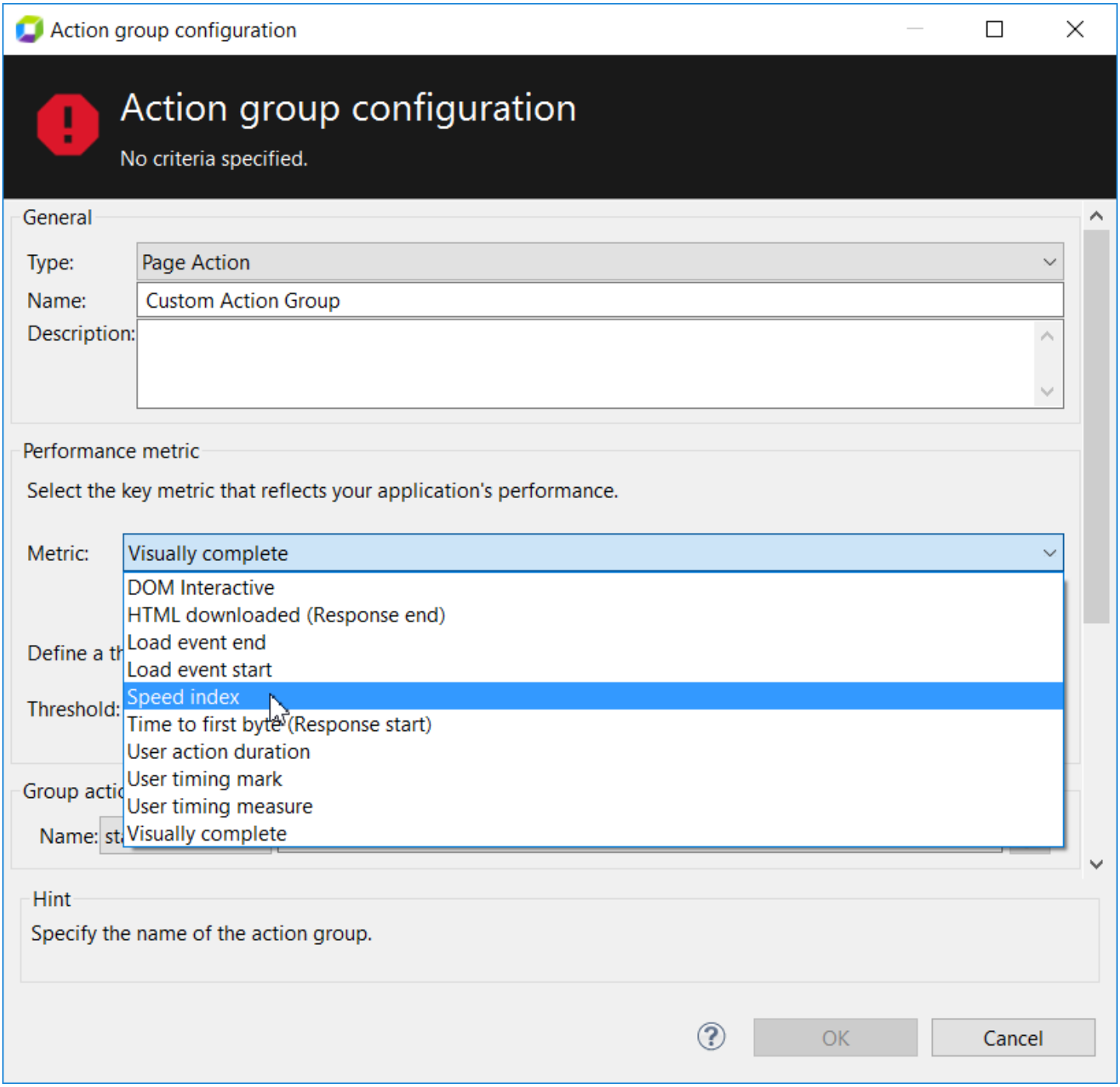
Predefined action groups with individually configurable performance baselines

Action group configuration

Configure action groups by flexible criteria and set a specific baseline for each. Both predefined Customizable action group types (Page, AJAX, JavaScript ADK, Mobile App, Mobile, and Mobile ADK) and any action groups you create can be customized. Group actions are:

- Based on their name.
- Starting or ending on a specific page by title or URL.

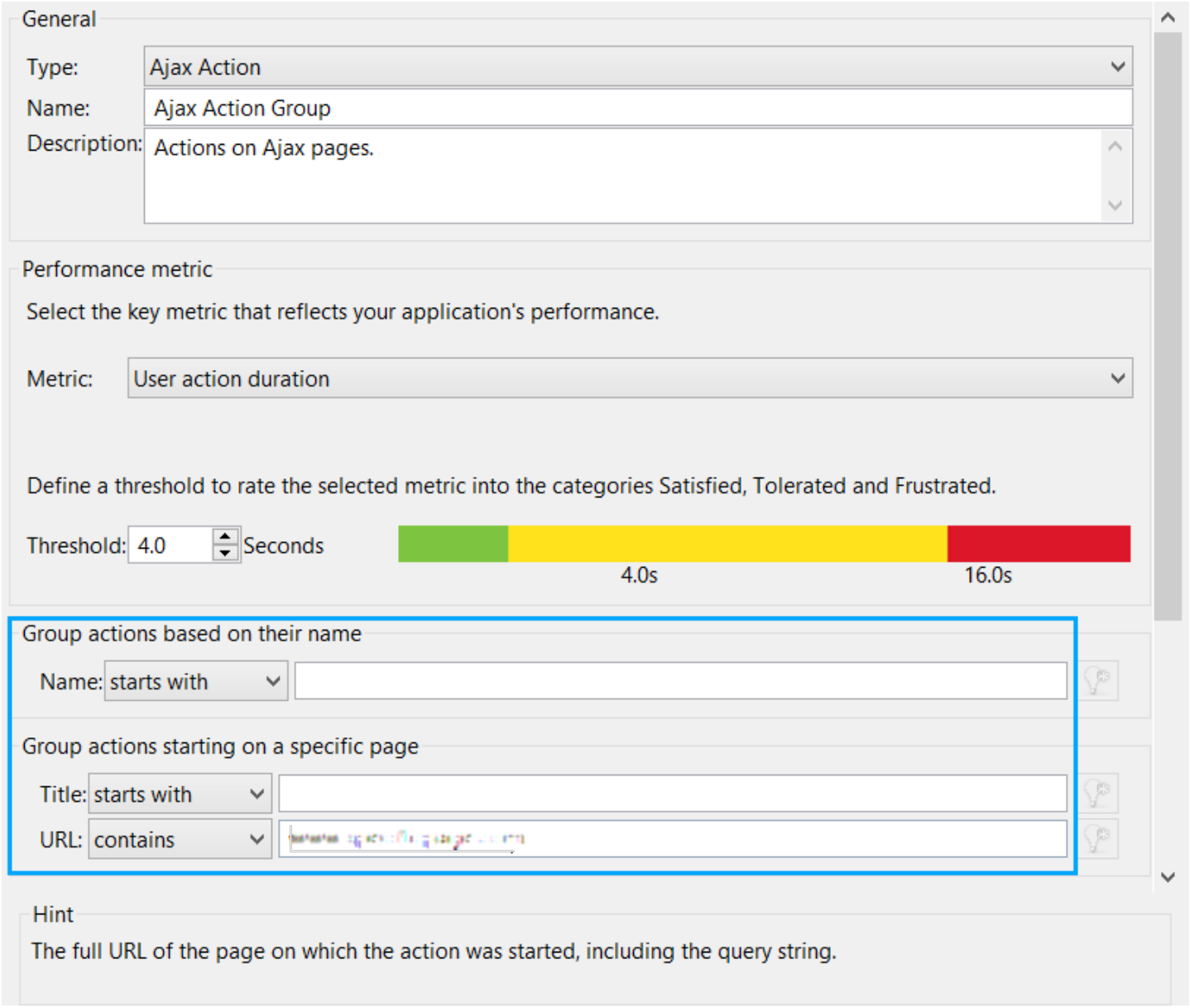
The baseline is the key performance metric used to determine Apdex rating to apply to each action in the each action group. You can set the performance metric for custom action group, or change the performance metric use3d for any predefined action group. To do this, select the action group to configure and click the **Edit** button in the Action Group button bar. Then in the Action group configuration dialog box, select the desired metric from the list. Only the performance metrics applicable to the action group type appear in the list.

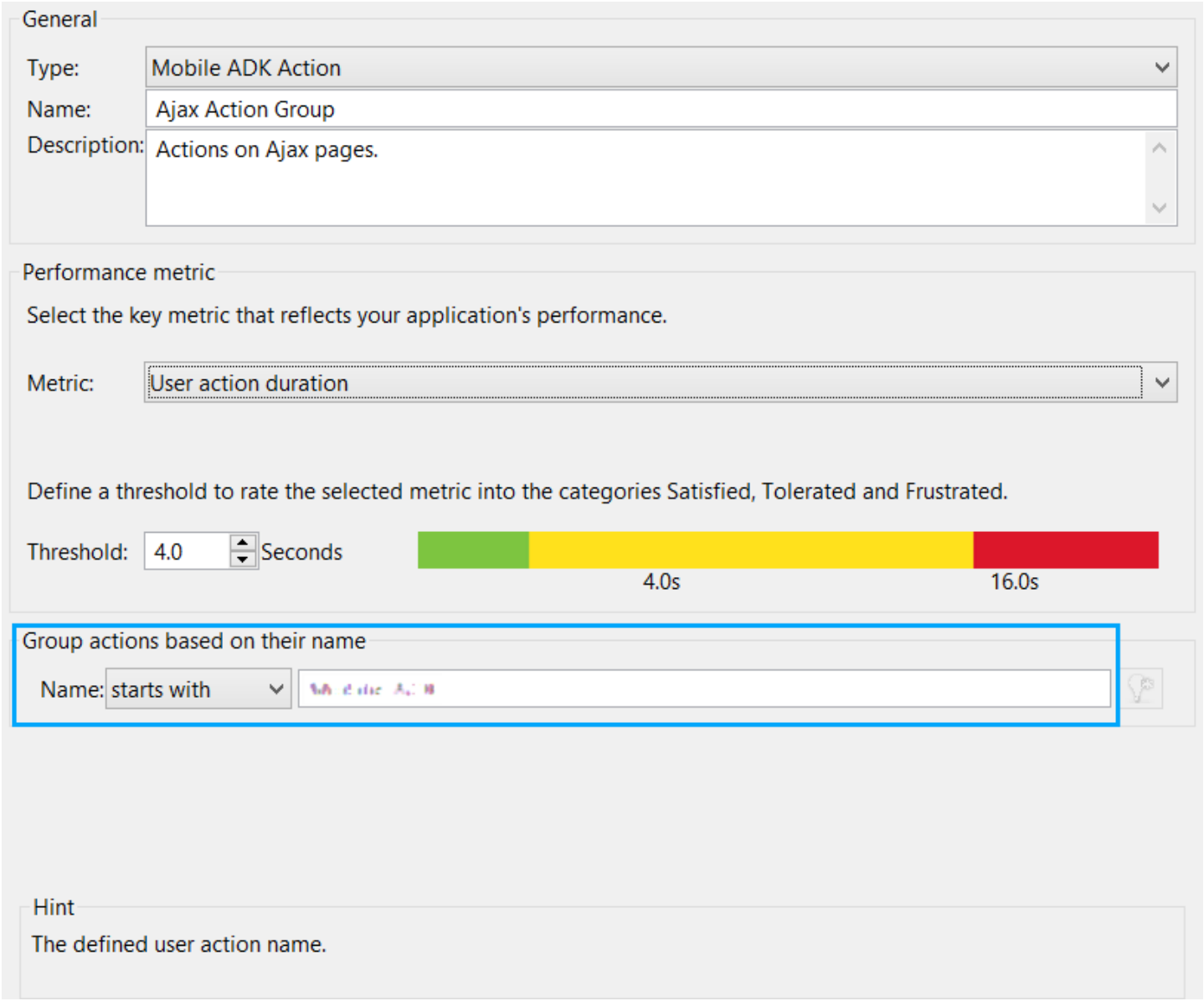


Set the key metric used to determine the Apdex rating for a predefined or custom action groups

The performance metric displays in user actions data in in both the client and AppMon Web. See

You can also optionally differentiate between web and mobile web. as shown in the following images, Actions for the Ajax action type can be grouped by XHR URL. Actions for the mobile ADK action type can only be grouped by name.


General
Type:
Ajax Action
Name:
Ajax Action Group
Description:
Actions on Ajax pages.
Performance metric
Select the key metric that reflects your application's performance.
Metric:
User action duration
Define a threshold to rate the selected metric into the categories Satisfied, Tolerated and Frustrated.
Threshold:
4.0
Seconds
4.0s
16.0s
Group actions based on their name
Name:
starts with
Group actions starting on a specific page
Title:
starts with
URL:
contains
Hint
The full URL of the page on which the action was started, including the query string.

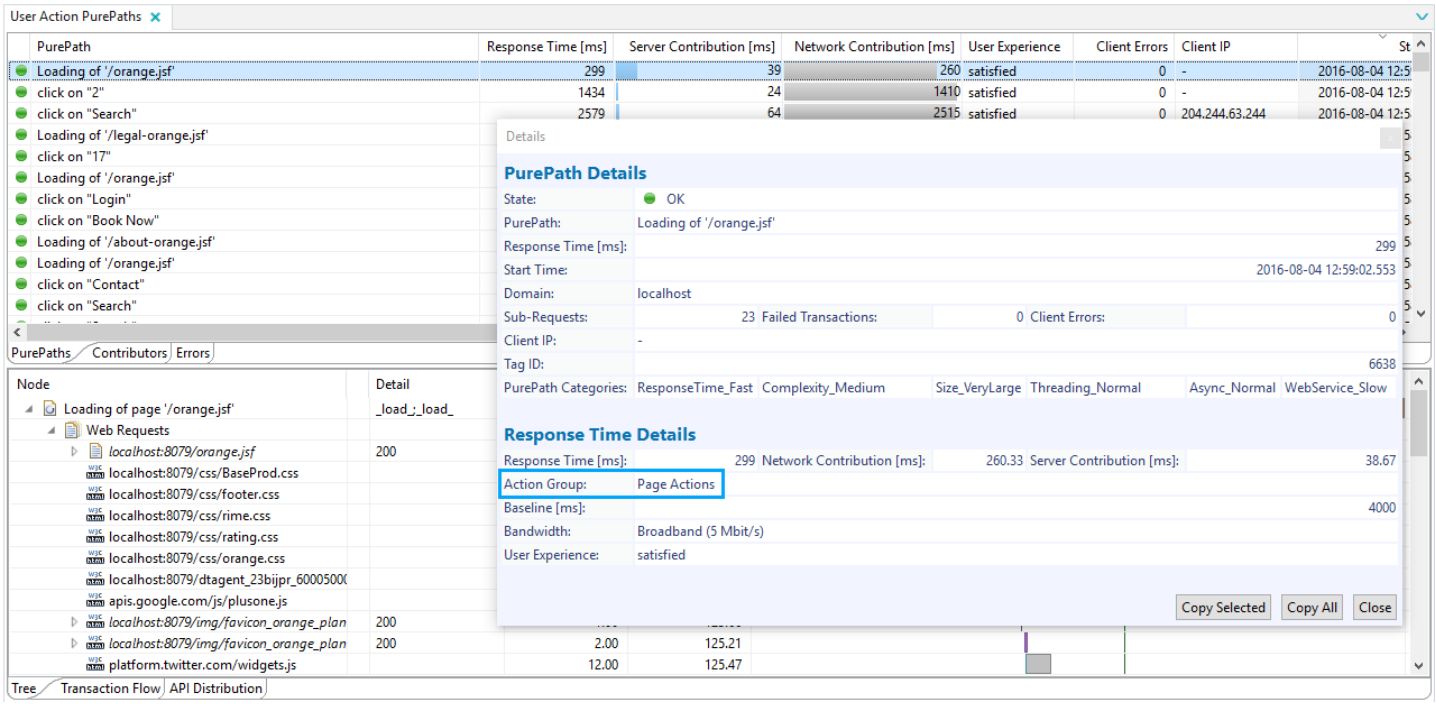


Action group advanced configuration

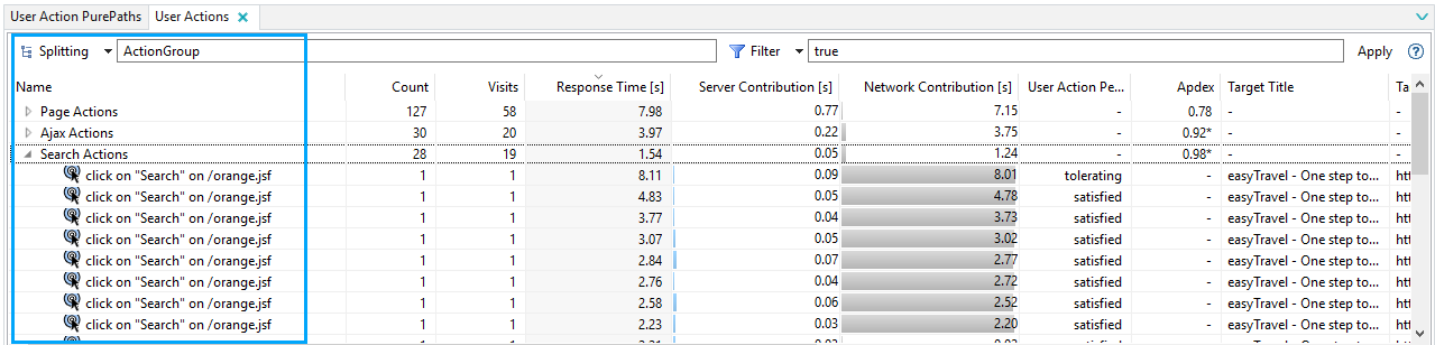
You can make additional configuration changes that apply across all of the selected profile's defined action groups. See Action groups advanced configuration to learn how.

Using multiple performance baselines in action groups

Action groups and individual baselines appear in the User Action PurePath dashboard details.



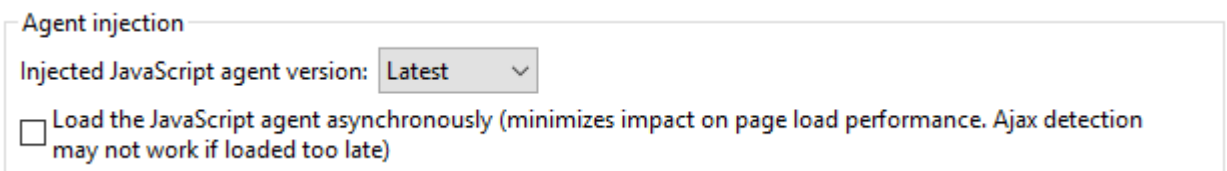
On the User Actions dashboard, actions display by action group



Web Applications

See JavaScript Agent for more information on how AppMon UEM works on websites.

Agent injection



Setting	Description

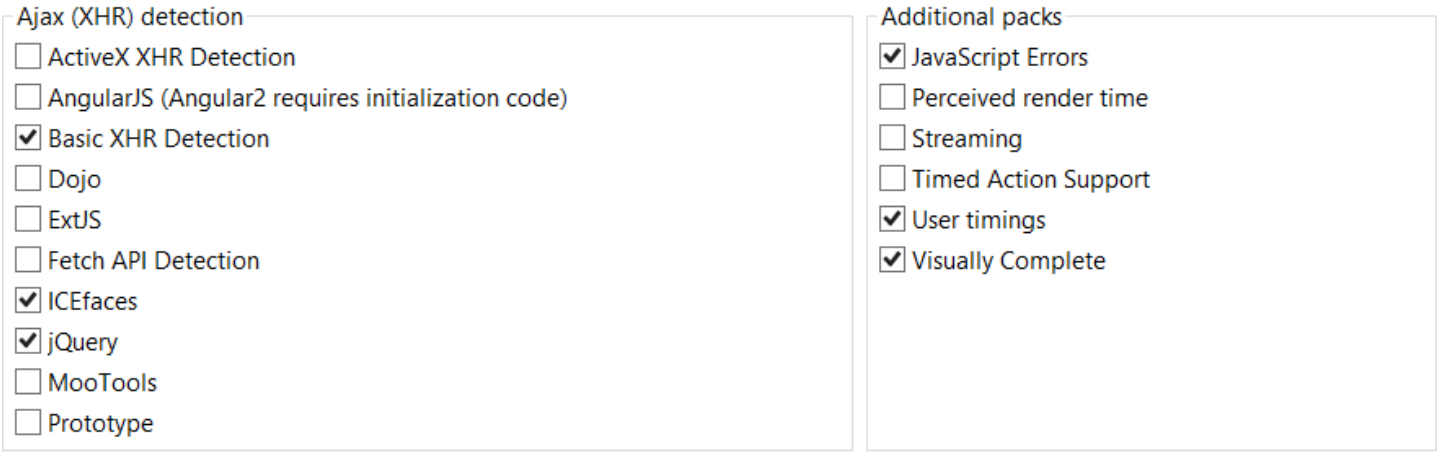
Injected JavaScript agent version	The setting lets you specify the latest delivered JavaScript Agent, or use a specific version. This setting lets you update AppMon without having to immediately change your target websites' included libraries. The delivered version depends on the connected agent. The latest possible JavaScript Agent version is the version of the connected agent, even if a higher one is configured. This ensures compatibility to the web server agents.
Load the JavaScript agent asynchronously	Choose this setting to append the `defer` attribute to the JavaScript Agent script tag, thus downloading the agent does not block the page from loading. Asynchronously loading decreases the agent's ability to capture content and hook into libraries. You can improve the amount of captured data if the agent is still the first script to be executed on the page and defer all other scripts, especially Ajax libraries or scripts that use `XMLHttpRequests`.

Ajax (XHR) detection and additional packs

Ajax (XHR) detection

Use Ajax (XHR) detection to select modules for capturing actions of different JavaScript frameworks that use `XMLHttpRequest` to perform requests.

Basic XHR Detection is a generic wrapper for all requests performed with `XMLHttpRequest`. Selecting this module captures the Internet Explorer `ActiveXObject`. Because it has more overhead than specific framework wrappers, using the specific wrappers whenever possible is recommended.



Some modules for specific frameworks require modifying the framework `js` file. This is necessary to trigger framework instrumentation directly after the framework is loaded.

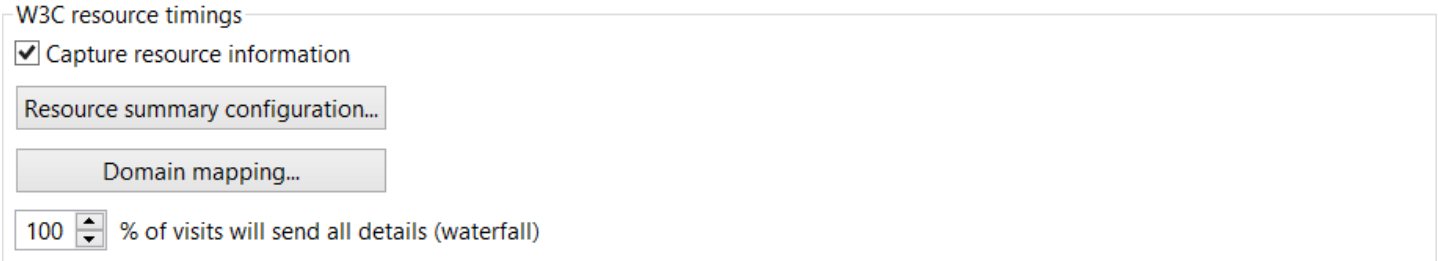
Manual XHR Framework Instrumentation contains a selection of the various code snippets to use.

Additional packs

This list contains settings for capturing additional information:

Setting	Description
JavaScript Errors	Captures JavaScript errors. For AppMon 6.3 and later, you can disable capturing. Stack traces are captured when enabled.
Perceived render time	This module calculates the perceived render time. This is the time it takes for all visible images to completely load, depending on the browser window size. When activated, perceived render time is calculated for each user action. See Perceived Render Time for more information.
Streaming	Captures information about HTML5 video and audio tags embedded into a webpage. Manual instrumentation is required for videos in Flash or Silverlight.
Timed action support	XHRs are often sent via `setTimeout` by JavaScript frameworks. Select this check box to detect actions that trigger XHRs.
User Timings	Captures timings measured by the browser's `performance.mark` and `performance.measure` calls.
Visually Complete	Enables support for both the visually complete metric and the speed index metric. This setting is enabled by default in newly created system profiles. A page is "visually complete" when all the visual components have been rendered. It is the time when a user considers that the page is ready to interact with. It doesn't consider, for example, any Ajax request that could be going on behind the page. Visually complete can be captured for page load and for Ajax actions. See Visually Complete metric for more information. The speed index is the average time at which visible parts of the page are displayed. It is useful for comparing the user experience of various pages. The lower the number, the better the user experience of the page. Speed index is only available for page load actions.

W3C Resource Timings



]

Use the W3C Resource Timings section to enable detailed resource information capturing, and to capture all or a percentage of visits details to produce a resource waterfall chart for PurePaths using every page load action. Use waterfall and captured W3C resource timing data to help optimize the performance of your web application.Waterfall charting appears on the User Action PurePaths dashboard. See Web Performance Optimization for UEM and W3C Resource Timing Metrics for more information.

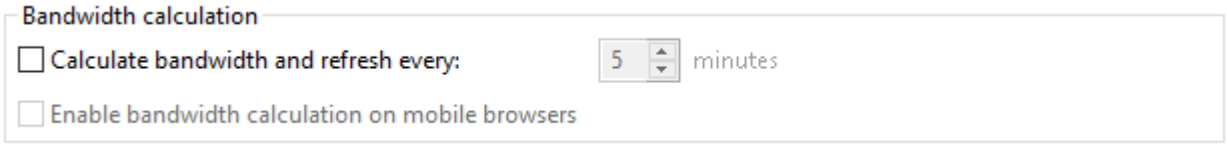
Setting	Description
Capture Resource Information	With UEM support enabled, the JavaScript Agent captures summary information on content embedded within a page. Resource summary information is available for the following: • **First party**: Any resource loaded from the application domain are assigned to the first party category. • **CDN**: New system profiles have a list of preconfigured CDN domains. Resources from these domains are assigned to the CDN category. • **Third party**: Any resources from other domains are categorized as third party. See the W3C Resource Timings section in Third-Party Content Analysis for more information.
Resource Summary Configuration	You can limit or extend the number of resources the JavaScript Agent captures for those actions that do not have full reports generated for them. Click **Resource Summary Configuration** in the **W3C Resource Timings** section to configure settings. These settings let you configure capture details on your slowest third party resources for each specified domain, and specify the slowest domains that are to report details back to the server. See Resource Summary Configuration for more information.
Domain Mapping	See CDN Content Analysis for more information on mapping domains.
% of visits will send all details (waterfall)	You can also configure the JavaScript Agent to capture all details to produce a resource waterfall chart for every page load action. Set the percentage of visits that should deliver the full resource waterfall. For example, setting this to 10% sends all details for every 10th visit. See the section Seeing full waterfall details in Web Performance Optimization for UEM for more information.

Note

Capturing resource information may cause overhead, especially on browsers that do not support the Resource Timing API.

Bandwidth calculation

This enables bandwidth-detection in the JavaScript Agent. For information on bandwidth calculation, see User Experience Management - Bandwidth.



Setting	Description
Calculate Bandwidth and refresh every	Interval at which the bandwidth calculation repeats for a visit. The smaller the value the more accurate the bandwidth if it changes regularly. However, this also produces more overhead. For proper calculation, you also need to enable capture of web request/response sizes. See UEM FAQ for detailed info.

Enable bandwidth calculation on mobile browsers	Enables bandwidth calculation for mobile devices. Bandwidth calculation needs to download extra data. This overhead is often undesired on mobile devices. The option is only available if bandwidth calculation is enabled.

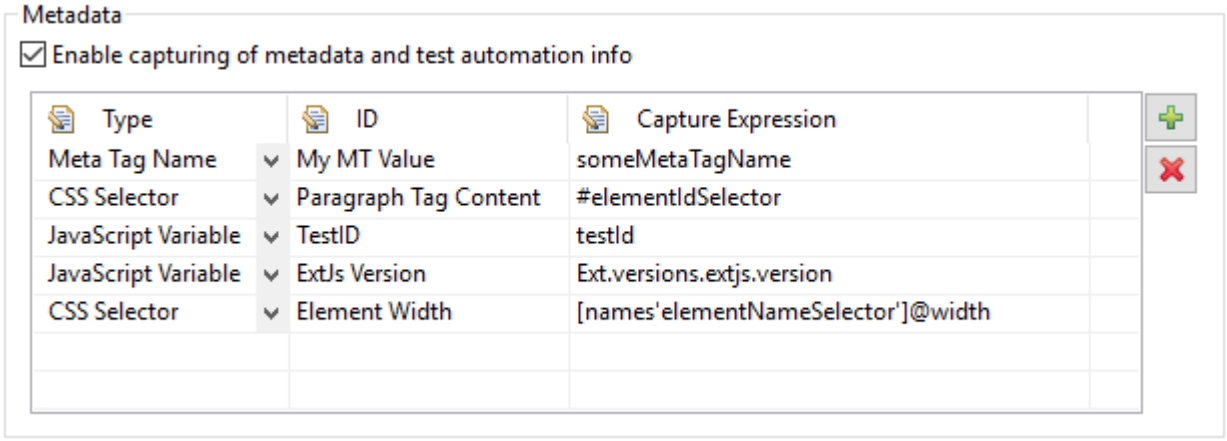
Version

Specifying your web application version number sets it for each visit's default value. Setting this value in the JavaScript ADK overrides this setting, and also lets it be retrieved using a measure. Click in the **Edit** button for the **Retrieve the app version with** field to specify a measure using the Select Measures dialog box.



Metadata capturing

Use the metadata sensor to capture CSS selectors, meta tags and global JavaScript variable values. The JavaScript Agent attempts to capture all configured values immediately before sending a user action, and attaches all available values to the user action. Once captured, you can subscribe measures using this metadata for visit tagging, application version, visit conversion, and other purposes.



```
    <script type="text/javascript">
        var testId = "testIdValue";
    </script>

    <meta name="someMetaTagName" content="metaTagValue"/>
    <script type="text/javascript" src="/ajax/resource/_scripts/extjs/6.0.0/ext-all.js"></script>
</head>
<body>
    <div id='div1' name="elementNameSelector" width="30px"></div>
    <div id='div2' name="elementNameSelector" width="40px"></div>
    <div id='div3' name="elementNameSelector" width="70px"></div>
    <div id='div4' name="elementNameSelector" width="200px"></div>

    <p id="elementIdSelector">info to capture</p>
</body>
</html>
```

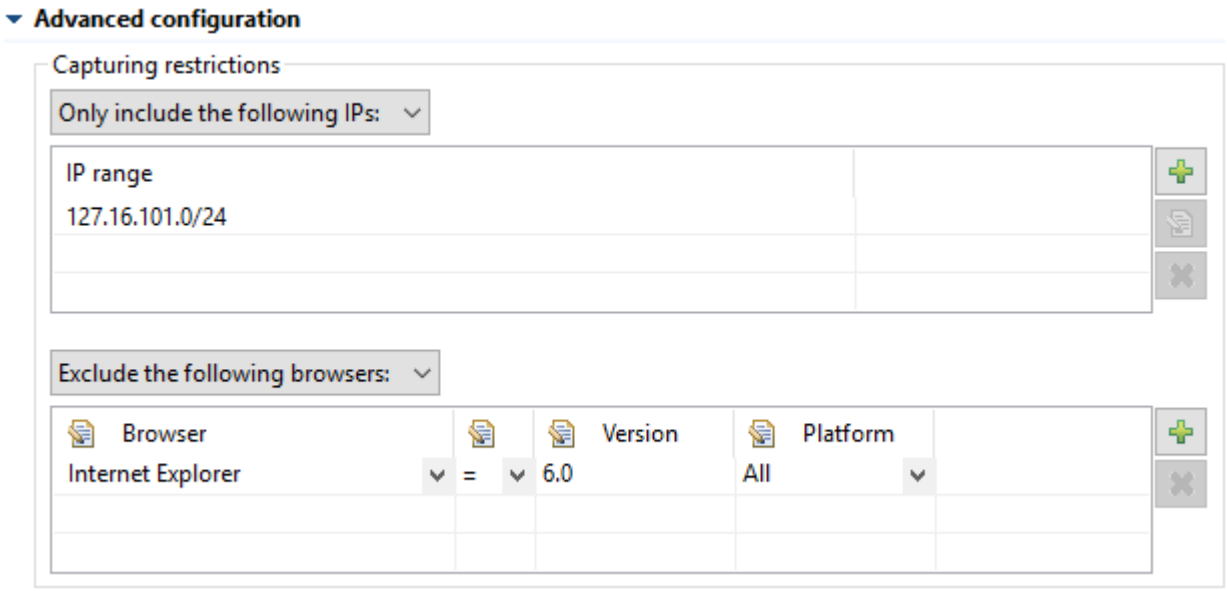
With capturing enabled, you can grab any number of values for these types. The captured string truncates if longer than 100 characters.
Select the type and specify an ID. You can then use this ID for Measures and Business Transactions. Finally, specify a *capture expression (identifier)* by which the value is retrieved:

- Meta Tag Names: Get the *content* attribute of the specified *meta tag name*.
- Meta Tag Names:
  - Get the *content* attribute of the specified *meta tag name*.
- CSS Selectors:
  - Specify a tested (with querySelector) **CSS selector**. JavaScript's querySelector captures the value, so only browsers that support querySelector report this. When testing this with querySelector, you may need to escape or modify some characters to work properly, such as **double quotes** being changed to **single quotes**.
  - In addition to using a CSS selector to locate a particular element, you can also retrieve an element **attribute** value. To do so, add @ followed by the attribute name. The **innerText** or **textContent** of the element is retrieved if no attribute is specified.
  - If the CSS selector query results in multiple elements, only the first element is captured.
- JavaScript variables:
  - Provide a **globally** available JavaScript variable name.
  - The variable must use dot notation for any Object property reference, for example `someObject.property` and not `someObject[property]`. Use brackets only when capturing Array values, such as 'someArray[2]'.

Captured values appear as Metadata Details when you hover over a user action PurePath root node or select Details from its context menu.

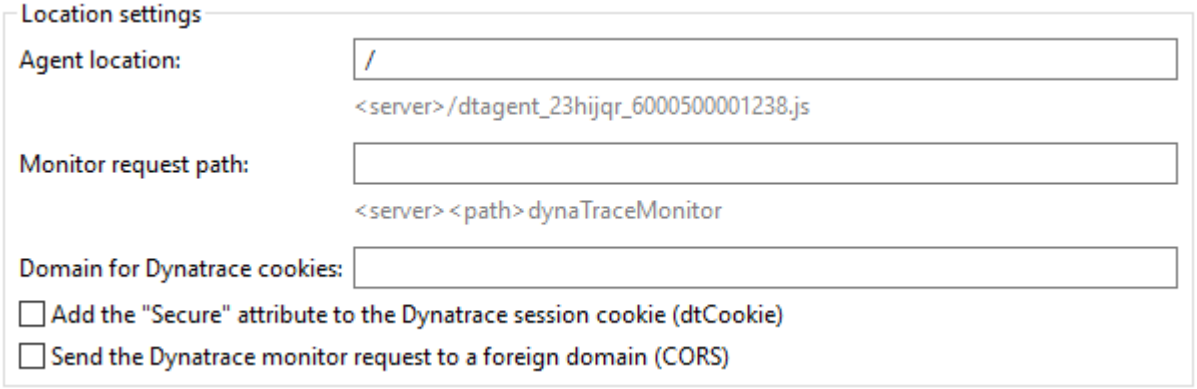
Advanced configuration

Capturing restrictions section



Setting	Description
Include/Exclude IPs (IP range)	You can configure ranges of IP addresses to exclude or include from User Experience Management. For example, if you want to exclude all visits of your web site from the local network you can add the network mask of your local network to the list of excluded addresses.
Include/Exclude Browsers (Browser range)	Include or exclude certain browsers and / or versions.

Location settings

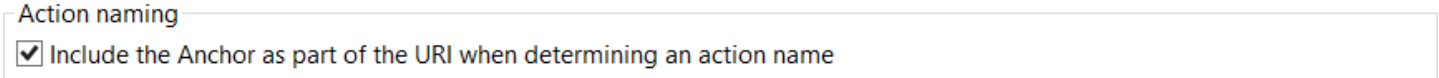


Setting	Description
Agent location	The path used when injecting the JavaScript Agent into your HTML code. For example, if you enter leave the default JavaScript Agent Name (`dtagent`) in the Global Settings horizontal tab, then th looks something like this: `<script type="text/javascript" src="/**myapp/** dtagent620_b_1` use this option to ensure that the client requests the JavaScript Agent from a URI where it reaches

Monitor request path	The URI that the JavaScript Agent uses to report gathered performance data back to AppMon via X
	The Monitor Request Name defined in the **Global Settings** tab (by default `/dynaTraceMoni`
	the JavaScript Agent. This should be a path that reaches an instrumented server. Requests to this
	forwarded to the AppMon Server. When you set the monitor request path, the signal is directed to
	you use load balancers, which can only handle dedicated URLs. For example, the page:
	`https://community.dynatrace.com/community/display/DOCDT62/Syst`
	`+User+Experience` loads the Agent from the following URLs for the different configurations:
	• **`<empty>`**: `https://community.dynatrace.com/community/display/DC`
	• **`"dynatrace"`**:
	`https://community.dynatrace.com/community/display/DOCDT62/d`
	• **`"/dynatrace"`**: `https://community.dynatrace.com/dynatrace/dynaT`
	• **`"https://community.dynatrace.com:8020/dynatrace"`**
	`https://community.dynatrace.com:8020/dynatrace/dynaTraceMon`

Domain for AppMon cookies	Overrides on which domain the AppMon cookies *dtCookie*, *dtPC*, and *dtLatC* should be set. **Note** The agent might not use cookies correctly if this setting is not configured properly. This can similar behavior.
Add the “Secure” attribute to the AppMon session cookie (dtCookie)	Useful for web sites that employ the HTTPS protocol for secure communications, this setting marks `dtCookie` with the W3C-standard “Secure” attribute. This attribute instructs the web browser to connection.
Send AppMon monitoring request to foreign domain (CORS)	The Same Origin Policy only allows XHRs that are sent to the same server delivering the web page Server Agent to send the monitor signal to a different server. To resolve this, modern browsers (IE FireFox 3.5 and later, Safari 4 and later, Opera 12 and later) support CORS, which lets the web se be delivered to a 3rd party site. Enable this here.

Action Naming



Setting	Description
Include the Anchor as part of the URI when determining an action name	Select this check box to include the Anchor as part of the URI when determining an action name.

Manual Injection

If you can not use automatic JavaScript Agent injection, for example because it is not supported for your web or application server, you can inject a JavaScript snippet that automatically loads and injects the JavaScript Agent and keeps configuration current.

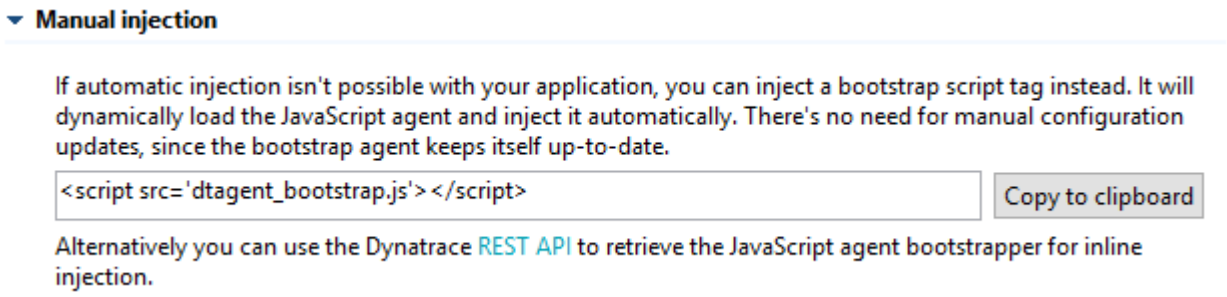
REST API - AppMon 6.5

Use the AppMon REST API to retrieve the snippet for your application and inject it as the first script into your page. The API is designed to be used during the build process of a web application and provides vital functionality the agent needs to instrument a website. Note that it is not recommended to request the snippet directly via a script tag, since this would unnecessarily decrease load times and require user privileges and direct access to the AppMon server. The API call looks like this:

```
https://<Dynatraceserver>:8021/api/v1/profiles/<systemprofile>/applicati
```

Bootstrapper - AppMon 6.3

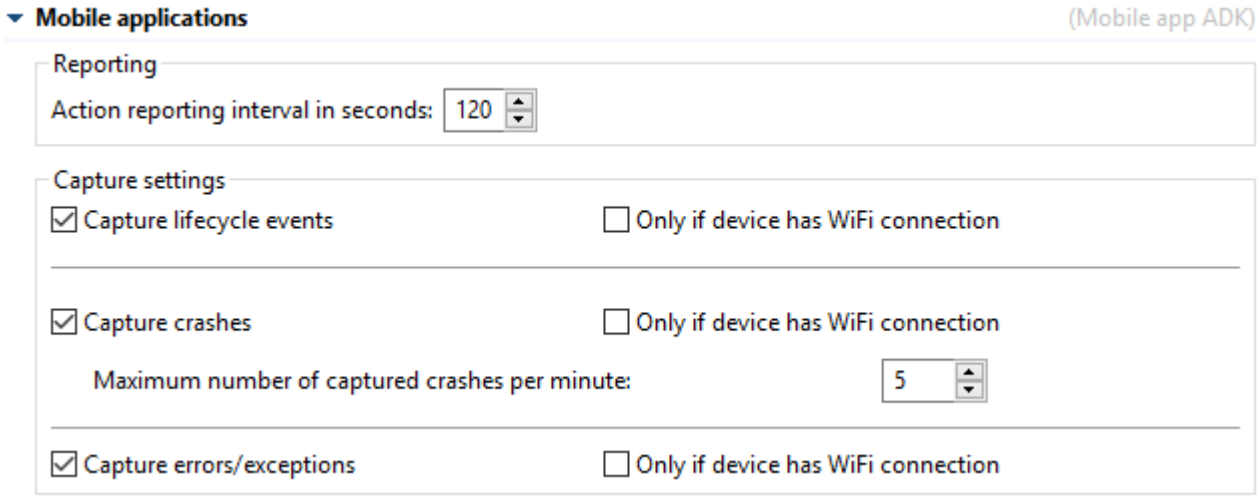
Enter the *bootstrap tag (see screenshot below)* pointing to an instrumented web or Java application server. The JavaScript Agent then builds automatically based the System Profile capture settings and are served dynamically. With AppMon 6.5, you can also retrieve the bootstrapper using the REST API link if the JavaScript *Agent* version is configured to be earlier than 6.5.



For the statically-served Agent corner case, use the REST API to download the JavaScript agent for your current configuration. For AppMon 6.3 and earlier, the filename must not be changed and the script tag must have a specific structure. See JavaScript Agent for additional details.

```
https://<Dynatraceserver>:8021/api/v1/profiles/<systemprofile>/applicati
```

Mobile applications



Setting	Description
Action reporting interval in seconds	Defines the delay until data captured with the mobile agent is sent to the AppMon Server. The delay is adjustable in 10-second steps between 10 seconds and nine minutes. The captured data is also sent when the mobile app is suspended or terminated, or if it is programmatically flushed via the corresponding mobile agent API.
Capture lifecycle events	Determines whether or not instrumented lifecycle actions generated by the mobile agent are sent. You can optionally select **Only if device has WiFi connection for this setting**.
Capture crashes	Determines whether or not to capture app crashes and send detailed crash reports to the AppMon Server. In order to prevent getting spammed with thousands of crash reports of different devices, in case of a defective crashing app, you can limit the number of handled crash reports per minute (five is the default). You can optionally select **Only if device has WiFi connection for this setting**.
Capture errors/exceptions	Determines whether or not the errors and exceptions reported via the `reportError` methods in the mobile agent are sent to the AppMon Server. You can optionally select **Only if device has WiFi connection for this setting**.

CDN content analysis

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

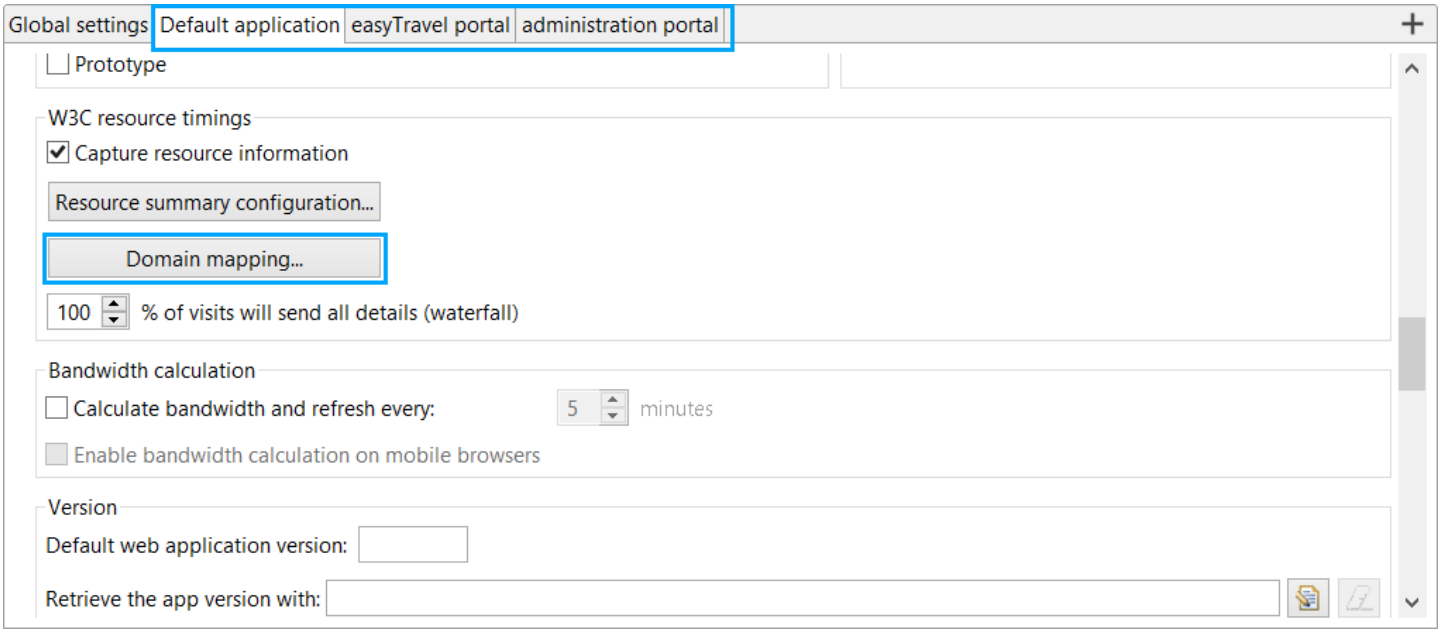
Third-party content analysis

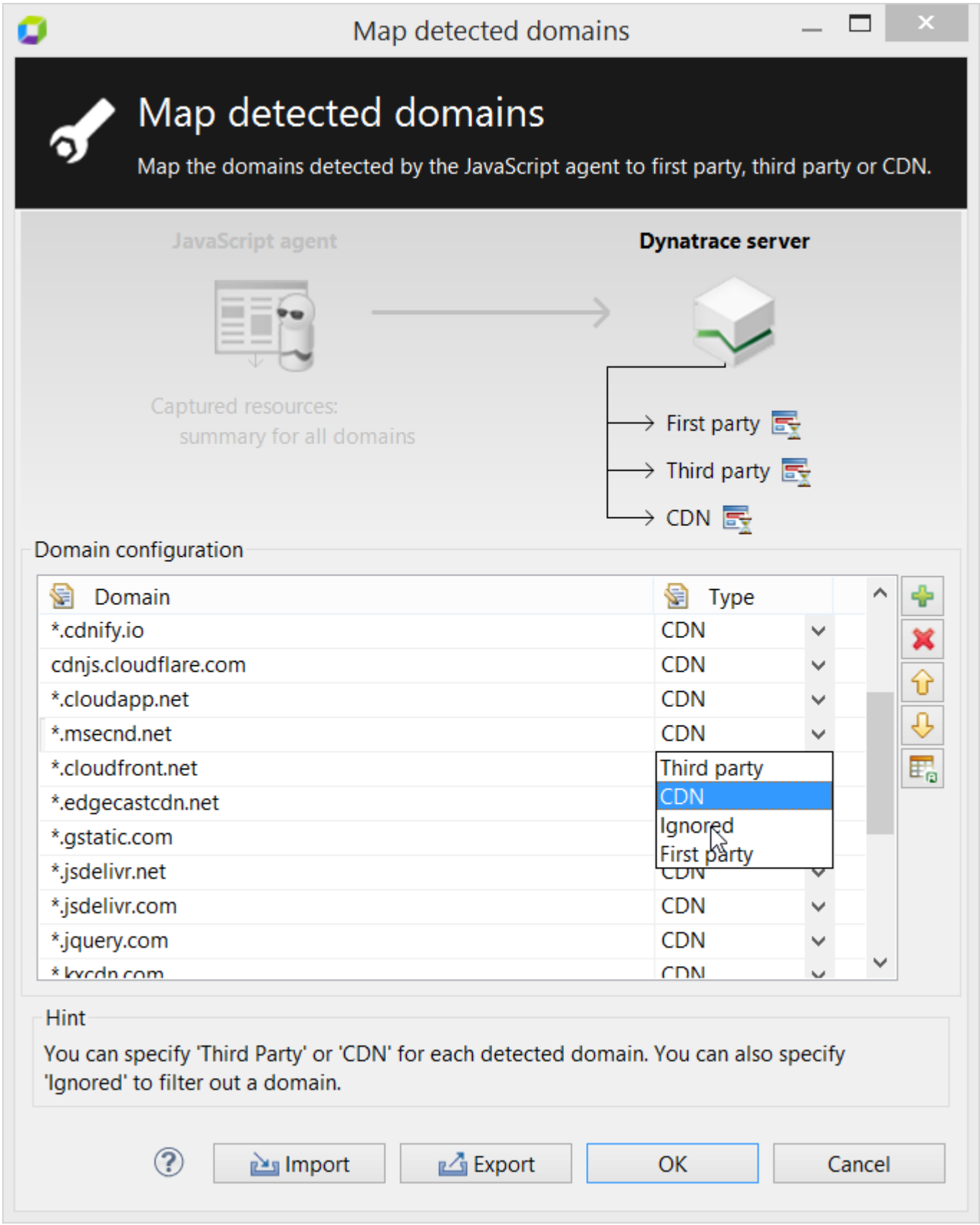
User experience management

Action groups advanced configuration

Overview

The JavaScript Agent detects CDN content the same way as it detects third-party content. However, you can map certain domains captured by the JavaScript agent to CDN instead of third party or first party for either all applications or specific applications in a System Profile. To do this, choose **System Profile Preferences > User Experience**, then select either the Default Application or a specific application tab and click **Domain mapping** in the **W3C Resource Timings** section.

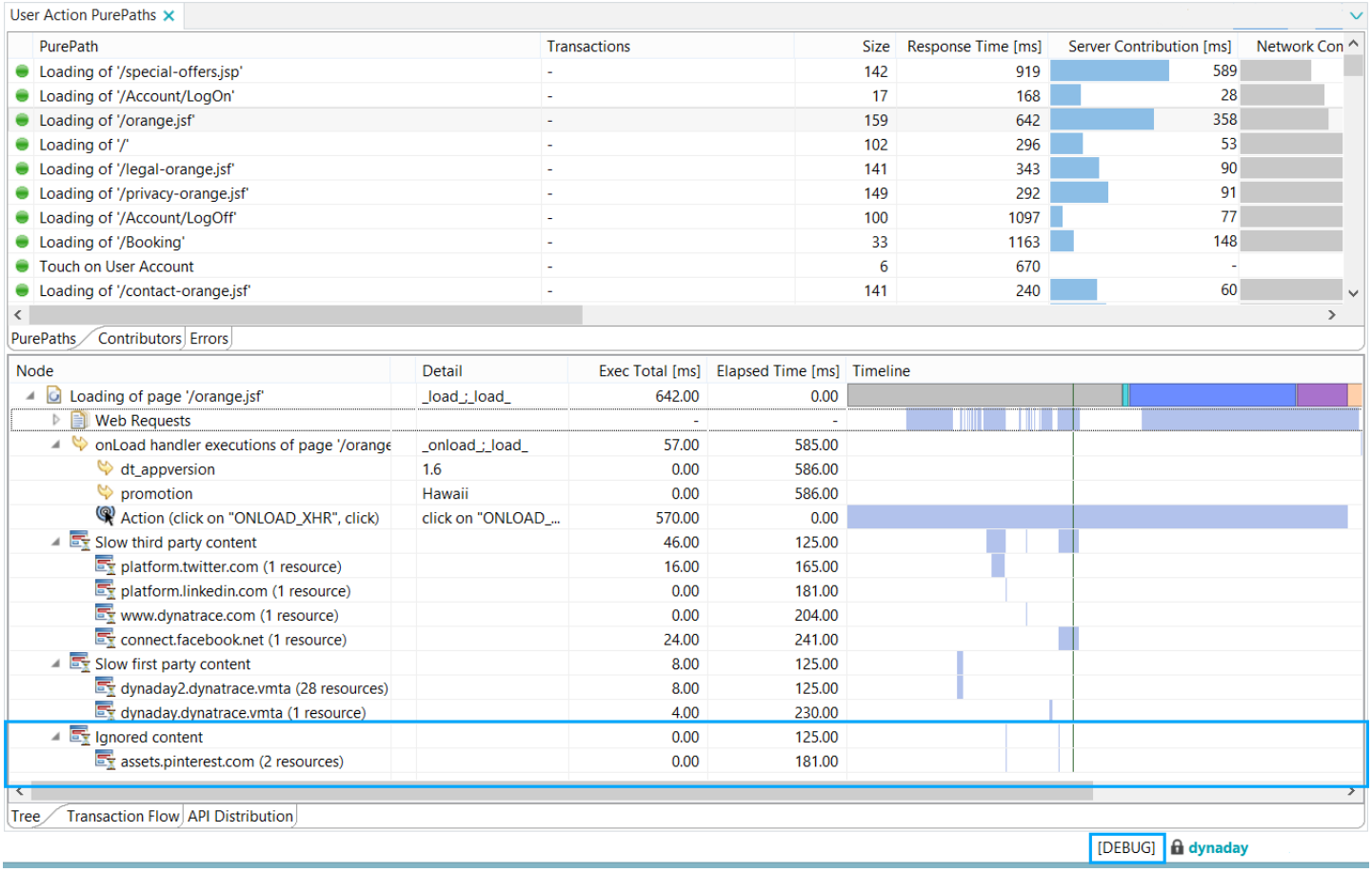

Global settings
Default application
easyTravel portal
administration portal
Prototype
W3C resource timings
Capture resource information
Resource summary configuration...
Domain mapping...
100
% of visits will send all details (waterfall)
Bandwidth calculation
Calculate bandwidth and refresh every:
5
minutes
Enable bandwidth calculation on mobile browsers
Version
Default web application version:
Retrieve the app version with:

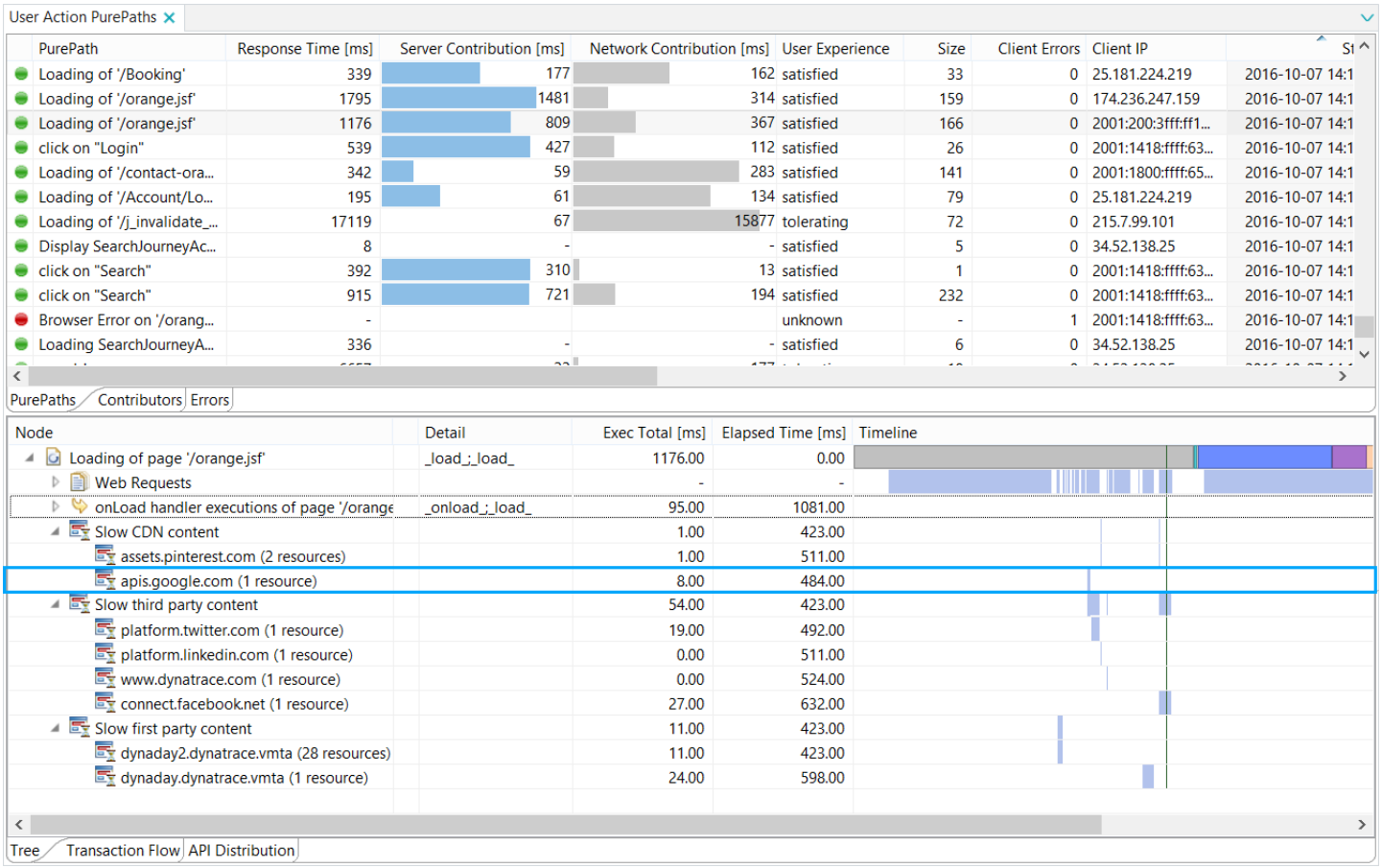


Specify **First Party**, **Third Party** or **CDN** for each configured domain. You can also specify **Ignored** to filter out a domain. Domain mapping impacts several dashlets, measures, business transactions and resource capturing for UEM related metrics.

User Action PurePath view

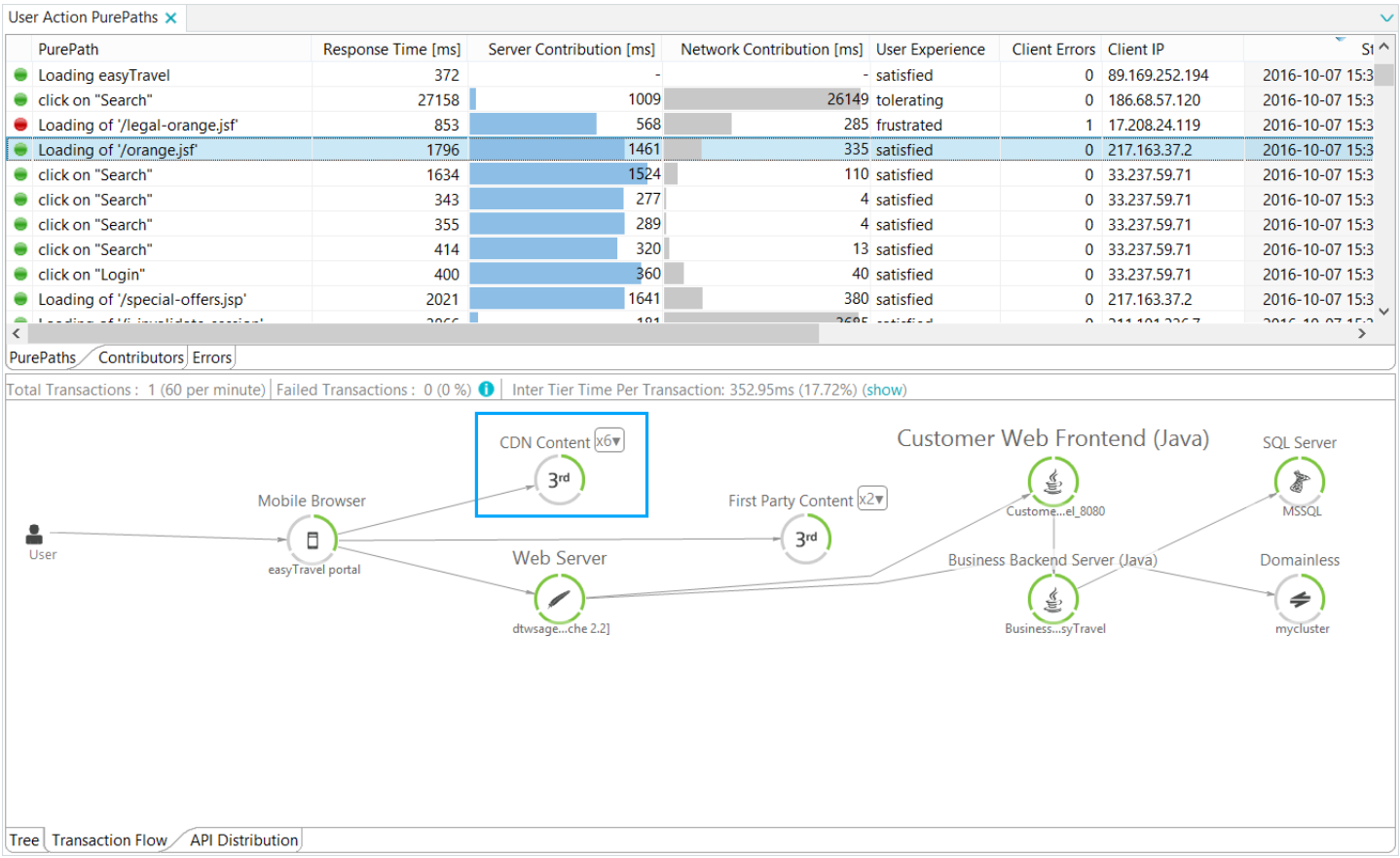
You can verify domain mapping configuration in the **User Actions PurePath** View. If a domain is mapped to CDN, it appears below the **Slow CDN content** node. A domain to **Ignored** is filtered from the table and does not appear in the **Slow third party content** node unless you change the Client to debug mode by pressing **Ctrl + Shift + F9**. The following shows an example.





Transaction Flow

Domains mapped to CDN also display as a new node or part of an existing node in the Transaction Flow, as shown in the following:



CDN measures

CDN busy time

This measure represents the duration of third party content during a user action. Configuration properties of this measure include:

- **Host**: String match that can filter for specific hosts or content providers, such as a set of several domains. You can select several string search types, including regular expressions.
- **Duration type**: Several types of load times are available.
  - **Busy time**: Time when at least one resource was loading.
  - **Average duration**: average duration of all resources.
  - **Shortest duration**: duration of fastest resource.
  - **Longest duration**: duration of slowest resource.

- **Resource type**: It is possible to measure only specific resource types:
  - All
  - Images
  - Scripts
  - Stylesheets (only available with W3C Resource timing)
  - Other resources (only available with W3C Resource timing)
  - Custom resources (resources added via the ADK)

CDN content load time

This measure represents the duration of CDN content during a user action. Configuration properties of this measure include:

- **Host**: Can be used for filtering for specific hosts.
- **Duration type**: Several types of load times are available:
  - **Busy time**: Time when at least one resource was loading.
  - **Average duration**: average duration of all resources.
  - **Shortest duration**: duration of fastest resource.
  - **Longest duration**: duration of slowest resource.
- **Resource type**:
  - All
  - Images
  - Scripts
  - Stylesheets (only available with W3C Resource timing)
  - Other resources (only available with W3C Resource timing)
  - Custom resources (resources added via the ADK)

BT measure User Actions - CDN hotspot

This measure counts the slowest host of a user action when exceeding the configured threshold.

Resource summary configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

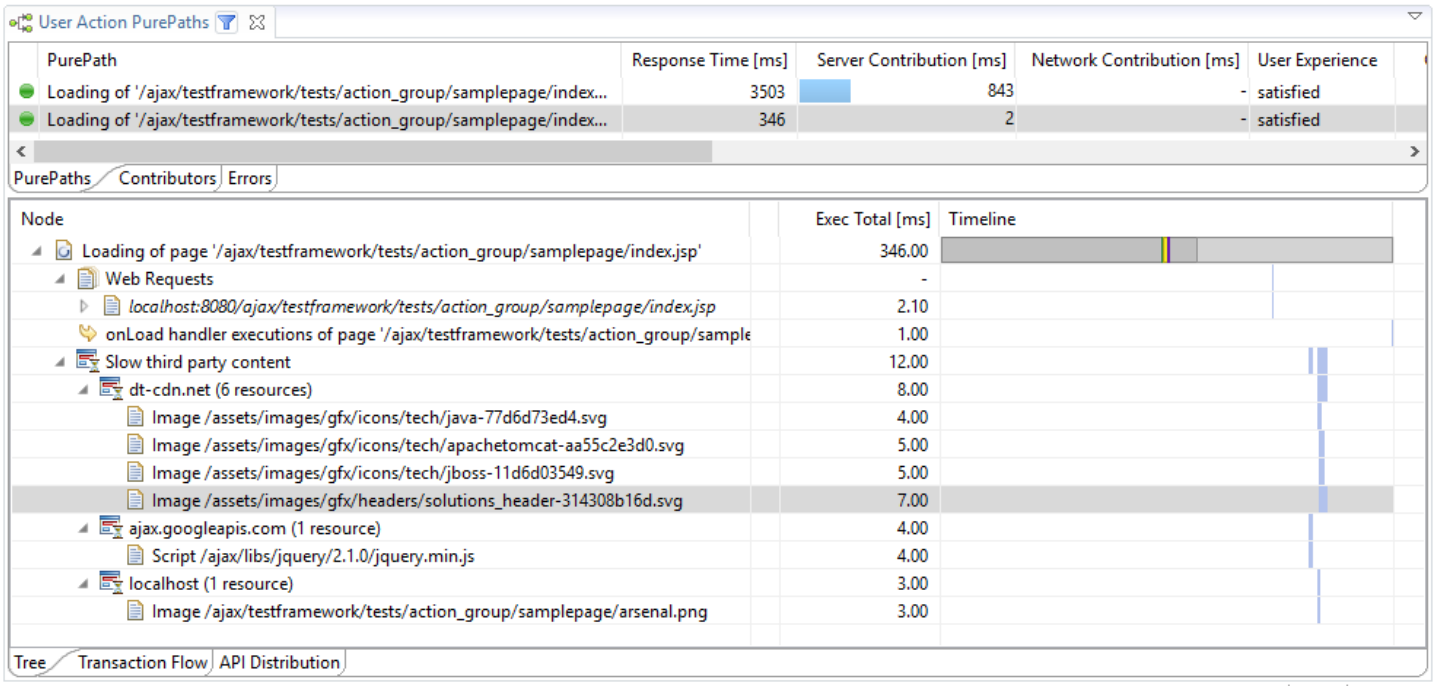
On this page

Summary

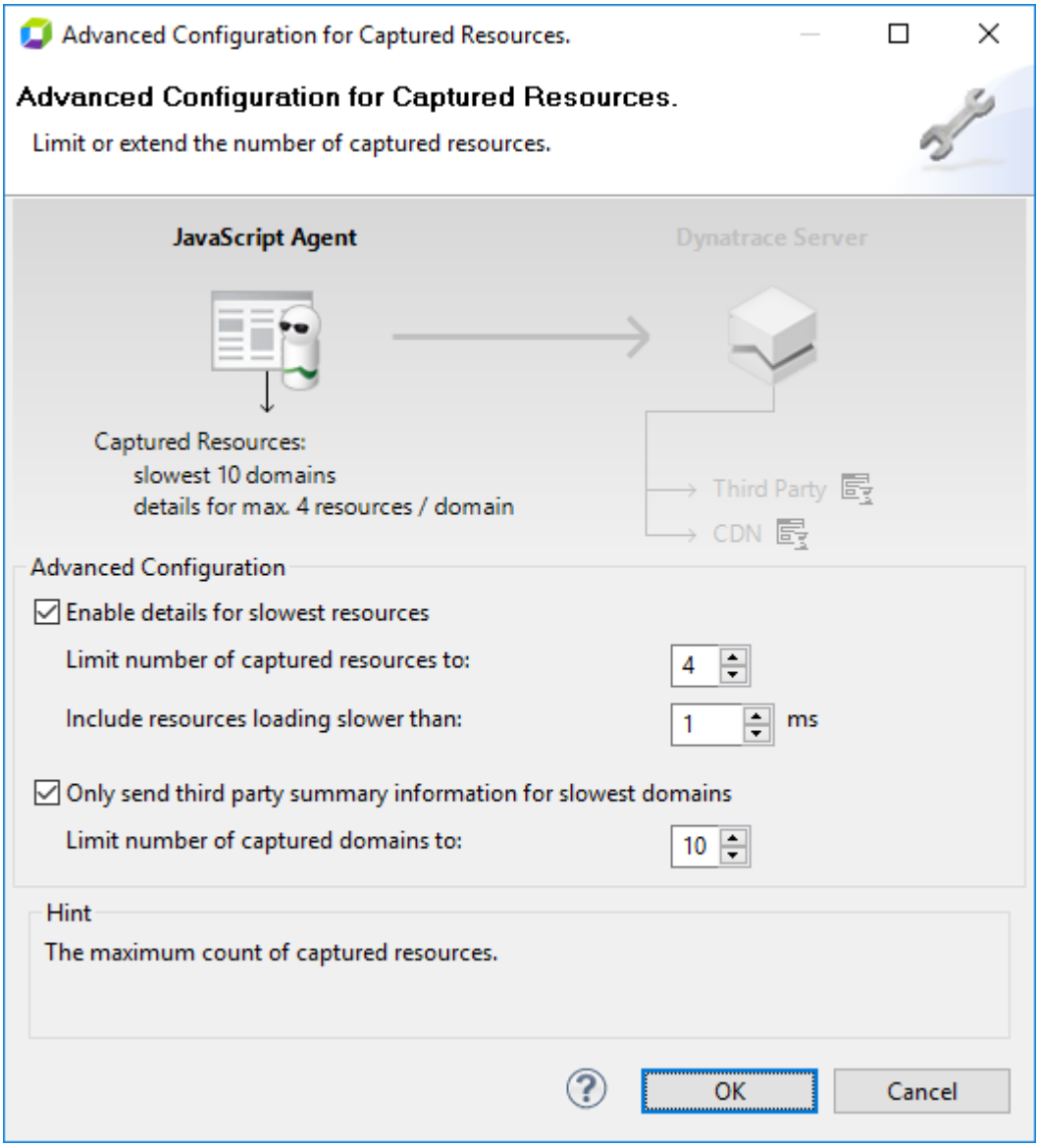
Settings

Summary

The JavaScript Agent detects resources associated with each web page. You can analyze the slowest resources for each captured domain. The User Action PurePath reports the results as shown:



The more resources captured, the more data being communicated from the JavaScript Agent to the server. Be careful as to exactly how much data is captured. Focus on the slowest resources. The **Advanced Configuration for Captured Resources** dialog box lets you control exactly how much data to capture.



Settings

Setting	Description
Enable details for slowest resources.	The User Action PurePath node can report each of the resources that were slow as a node under the domain itself. Select this check box to enable of individual resource reporting. Enabling this check box lists individual resources rather than just providing a summary total in parenthesis.
Limit number of captured resources to:	Use this option to control how many resources per domain are shown. In the example above AppMon has detected six domains from `dt-cdn.net`. However this setting is set to four, so only the four slowest resources are reported under “dt-dcn.net (six resources)”.
Include resources loading slower than:	As a further filter, only resources that took longer than this value to load are reported in the list.
Only send third party summary information for slowest domains.	Enable this option to only capture summary information for the slowest domains. Without this setting selected, all domains are reported.

Limit number of captured domains to:	The number of slowest domains that are to be captured.

Action groups advanced configuration

On this page

Settings

Dashlet metrics applicable to action groups

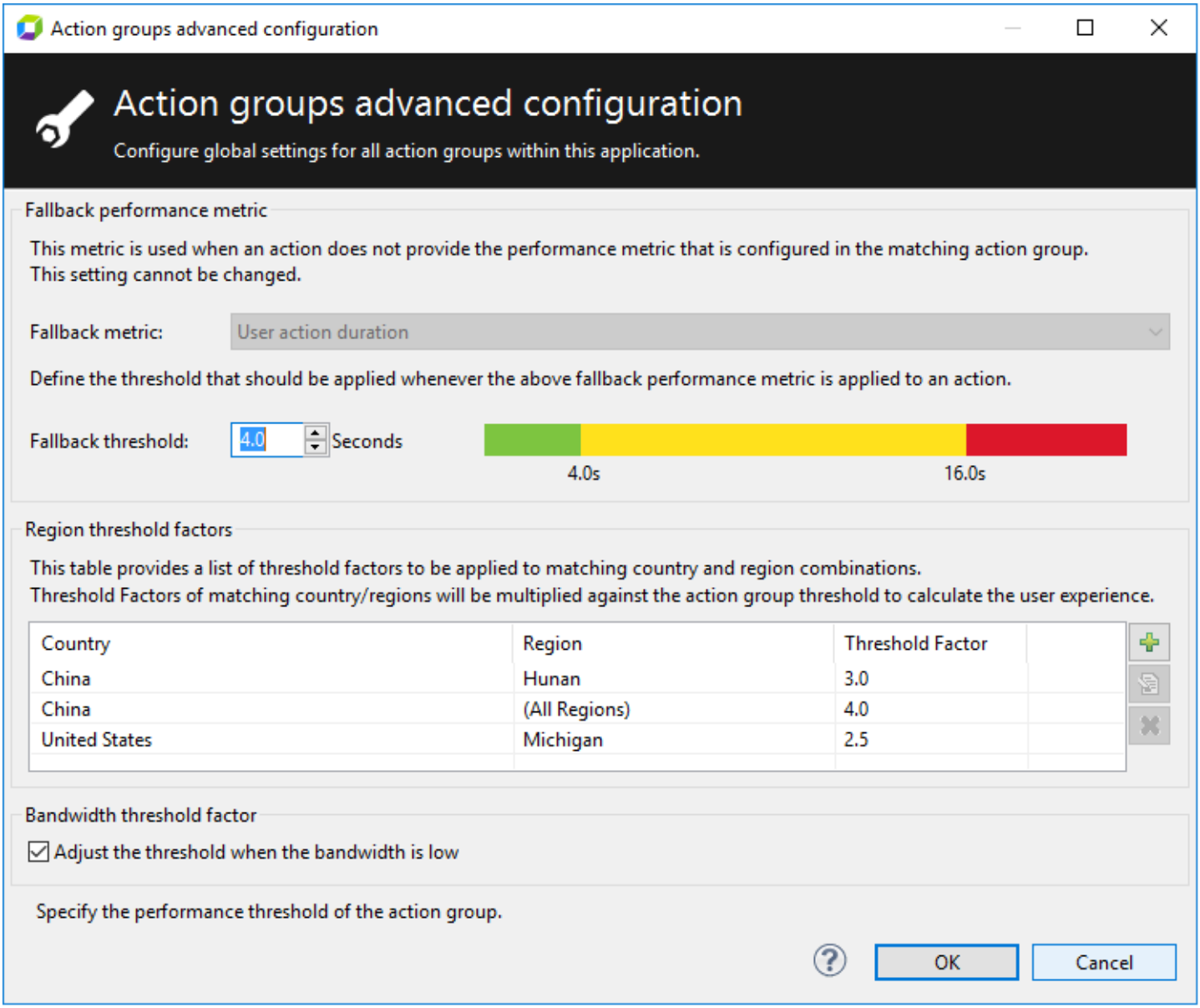
Related pages

CDN content analysis

System Profile - User Experience

Settings

Use the **Action groups advanced configuration** dialog box to make additional configuration changes that apply across all of the selected profile's action groups.



Setting	Description
Fallback metric	Each action group can define its own performance metric. The performance metric determines the Apdex rating. When a defined performance metric is not available for a specific user action, the fallback metric is used instead. Visits, User Action PurePath and User Actions dashlets all display an Apdex rating. There is also a column to display the performance metric that was used. See Dashlet metrics applicable to action groups for an example of how these action group metrics display in the User Action PurePath dashlet. See image below.
Fallback threshold	This is the threshold used when the fallback metric applies. The value sets the threshold for a satisfied visit. When the fallback metric value is less than or equal to this specified fallback threshold, the Apdex rating of “Satisfied” is given.
Region threshold factors	Use Region threshold factors to define thresholds that apply only to a specific region and country combinations. Region thresholds are useful for specific regions where a reduced performance may be expected due to factors such as slow Internet connections. Click the **Create** button next to the Region threshold factors list to define country/region threshold factors.
Adjust the threshold when the bandwidth is low.	AppMon takes low bandwidth into account when calculating the Apdex rating. It calculates a factor that the threshold is multiplied by based on low bandwidth performance. For example, if the bandwidth is slow, AppMon adjusts the threshold to lengthen the time that qualifies as a **Satisifed** visit. This setting is enabled by default for profiles created in AppMon 6.5 and earlier. Select this option to enable this this setting for profiles created with AppMon 2017 May and later.

Add Country Region thresholds factors

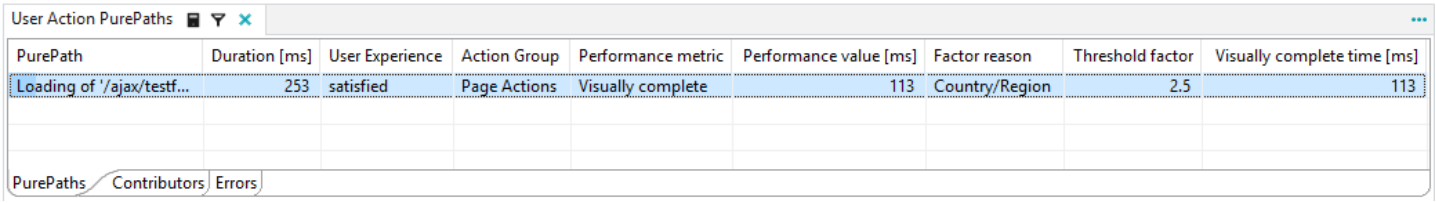
Use the Country Region Threshold factors dialog box to add threshold factors to apply to a specific Country and region. Enter the required threshold factor that each region should use. The threshold factor is multiplied by the performance metric's threshold to determine a new threshold.

For example, the threshold for a specific action group could be set to 4 seconds. However, it could be expected that visits from the Hunan region of China are expected to be slow because users connect only through WiFi. The above setting for China - Hunan multiplies the threshold by a factor (3 in this case). Any performance less than (4 x 3) seconds is marked as "Satisfied".

Setting	Description
Country	Select the country for the threshold factor you are defining.
Region	Select the region within the country from the list. Only regions specific to the selected country appear in this list.
Threshold factor	This is the threshold to be used for the specific country/region definition. This factor is multiplied by the action group threshold to calculate the user experience for the specific country/region setting.

Dashlet metrics applicable to action groups

The following image shows the metrics appropriate for action groups as displayed in the User Action PurePaths dashlet. You can use them to determine how AppMon calculated the Apdex for a specific user action.



Metric	Description
Action Group	The action group used to calculate the Apdex rating.
Performance metric	The performance metric is used to calculate the Apdex rating. It is defined in the appropriate action group. If the defined performance metric is not available when the PurePath processes, any defined fall back performance metric is used. This column shows which performance metric is used.

Performance value	The value of the performance metric that was used.
Factor/reason	The criteria used to determine the performance metric used. More than one factor could be true, in which case the lowest threshold factor applies. For example, if the Country/Region is defined and there is slow bandwidth, only the lowest factor applies. Possible reasons are: **Country/Region**: A Region threshold factor applies. **Low bandwidth**: A factor based on low bandwidth applies. **-**: No factor applies, the action group's performance metric is used. No additional threshold factor applies.
Threshold factor	The factor that the threshold is multiplied by to determine the threshold. This value is `1` if no additional factor applies.

System Profile - Integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Add a new DC RUM appliance

Edit an existing DC RUM appliance

Drill down from DC RUM to AppMon

Troubleshooting

Related pages

Cockpit dashlet

Edit a System Profile

Watch and learn

Integrating with Data Center Real-User Monitoring

Overview

Data Center - Real User Monitoring (DC RUM) provides real-time access to information about performance and usage of key business applications. Based on measurement data provided by **network monitoring** devices, it monitors application performance, with the purpose of detecting, identifying and solving problems. Analysis gives insight into business application performance on transaction and operation level.

In combination with the in-depth analysis capabilities of AppMon PurePath technology, you can quickly find the root cause of problems in its code-level context.

To combine DC RUM data with AppMon, configure a connection to your DC RUM appliances in the **Integration** section of the **System Profile Preferences** dialog box. Enabling integration correlates data collected by DC RUM to AppMon PurePaths.

Supported releases

Integrating DC RUM with AppMon 2017 May requires the following:

- DC RUM version 12.4.10 or later.
- Flash installed on the client's browser. Drilldowns from the DC RUM Central Analysis Server (CAS) require Flash.

Add a new DC RUM appliance

To add a new DC RUM appliance:

1. Right-click your System Profile, and select **Edit System Profile > Integration**.
2. Select the **Import Data from DCRUM Agentless Monitoring Devices listed below** check box.
3. Click **Add**.
4. In the **DCRUM Appliance Configuration** dialog box, provide the following:
   - URL of the DC RUM appliance. Default ports are 9090 (HTTP) and 443 (HTTPS). This field is mandatory.
   - Username and password for the DC RUM appliance. These fields are optional.
5. Click **Test connection** to test the connection of the AppMon server to the DC RUM appliance,
6. Click **OK**.

You must configure at least one DC RUM appliance to activate DC RUM integration for the System Profile you edit. The DC RUM appliance can then be configured for another System Profile.

Edit an existing DC RUM appliance

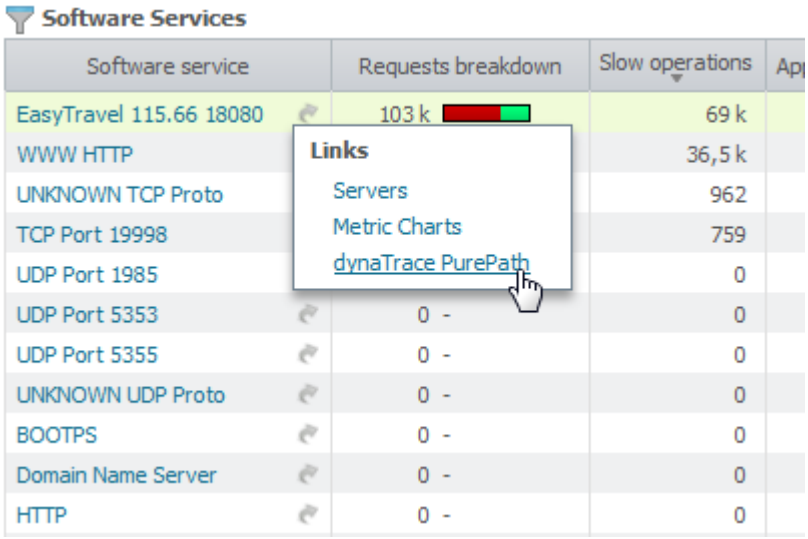
To edit an existing DC RUM appliance:

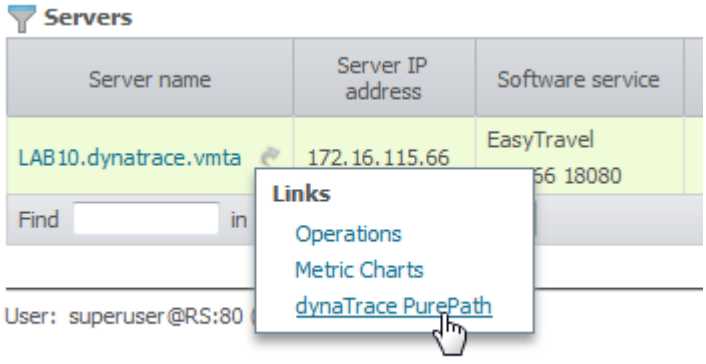
1. Mark the appliance you want to edit in the table.

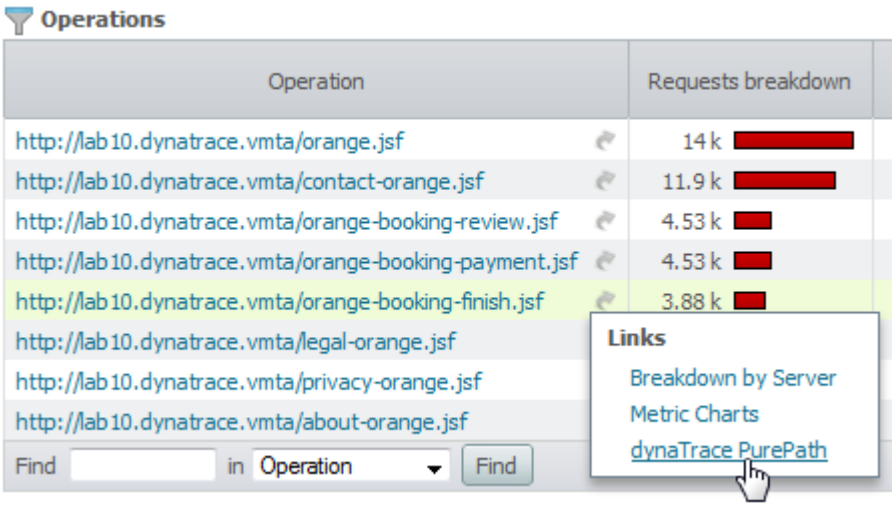
2. Click **Edit**.
3. Edit the URL, username, and password associated with the DC RUM appliance.
4. Click **OK**.

Drill down from DC RUM to AppMon

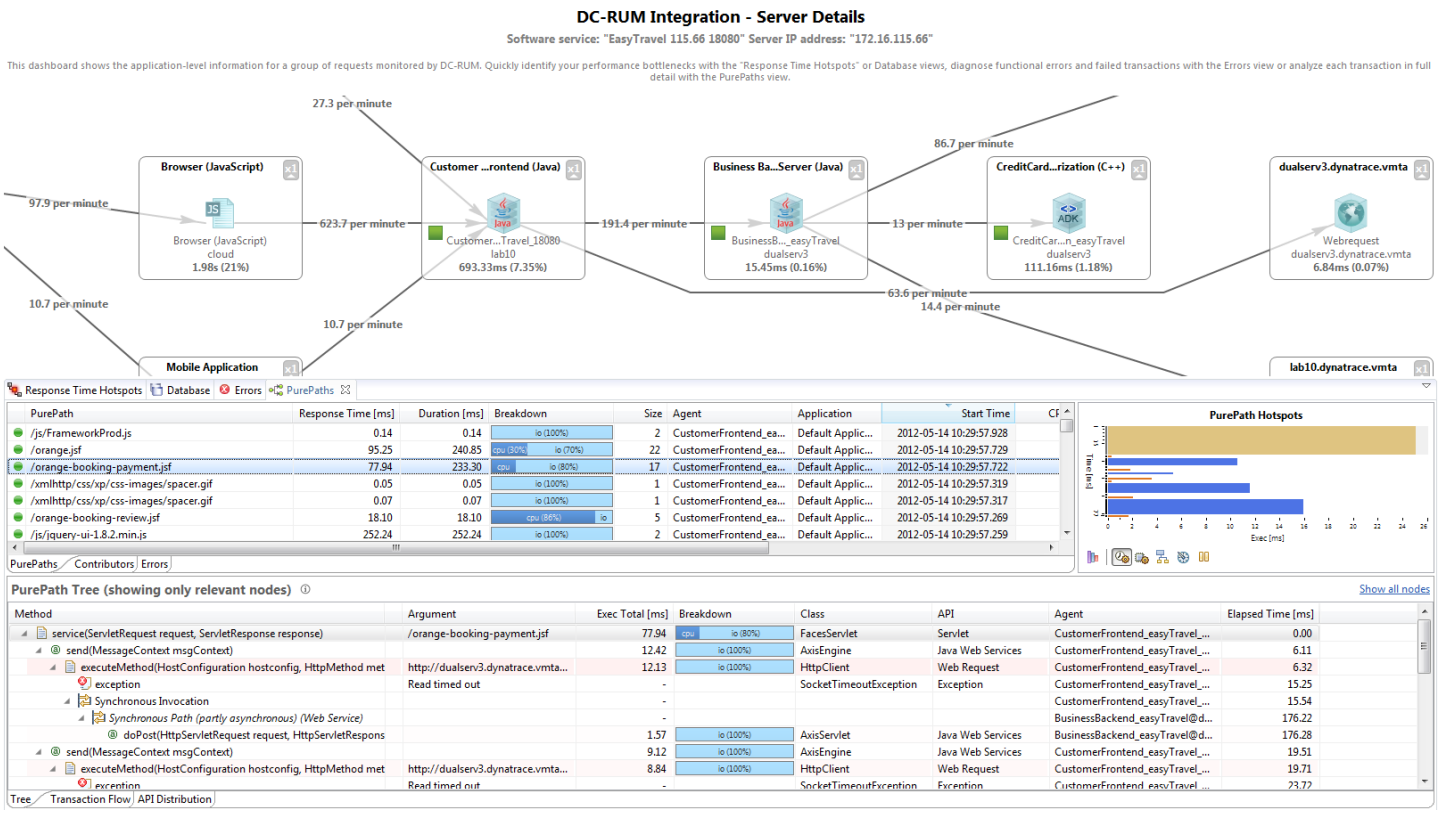
If AppMon and DC RUM are configured to inter-operate, the front end of the CAS offers several drilldown links to AppMon.







Click a drilldown link and the AppMon Client starts and displays a pre-built dashboard.



Dashboard DC-RUM integration

Troubleshooting

You have configured DC RUM Integration but the drilldown from CAS to AppMon does not work. Use the following checklist to troubleshoot the problem.

Verify that Flash is installed in the client's browser

Flash is required for drilldown operation.

Check connection settings

Right-click your system profile, and select **Edit System Profile > Integration**, open your DC RUM Agentless Monitoring Device, and click **Test Connection** to check if AppMon Server can connect to the AMD.

Check DC RUM Record availability

Do this:

1. Create a new dashboard, and set its source to `dynaTrace Self-Monitoring`.
2. Add a new **Chart** dashlet.
3. Click **Add Series** and add the **DC-RUM Records** measure. Press Ctrl+F and search for "DC-RUM" to find it faster.
4. If the value in the chart is always 0, then AppMon isn't receiving data from DC RUM. In this case, check if DC RUM is configured correctly. Contact DC RUM support if necessary.

Check DC RUM Record correlation

To do this:

1. Check if the incoming DC RUM Records are attached correctly to AppMon PurePaths.
    1. Click here to open the **DC-RUM Integration** dashboard using the Client REST Interface and list all PurePaths with DC RUM Records.
    2. Perform the request on the same machine where the AppMon Client is running. Use this URL template:
    http://localhost:8030/rest/integration/opendashboard?dashboardname=DCRUM+Integration&source=live:<YOUR_SYSTEM_PROFILE>&filter=dc:softwareservice?@starts

> **Note**
>
> Remember to replace *<YOUR_SYSTEM_PROFILE>* with the actual name of your System Profile

2. Make sure that the expected PurePaths are listed. Some hints:

   - The filter `filter=dc:softwareservice?@starts` in the URL template above just matches all PurePaths with DC RUM Records, because every `softwareservice` string starts with an empty string.
   - Additionally, you can set some other DC RUM filters. If you want to see only data of a certain timeframe, add a timeframe filter. See REST Filters for more information.

Check DC RUM Monitoring Interval

If all previous checks succeed and the **DC RUM Integration** dashboard is still empty, AppMon may not have data for the specified timeframe.

- Go to the CAS and check the analysis timeframe.
- In the AppMon Client, right click your System Profile and select **Details**. Check the **Recording** field of the **Details dialog box**, to make sure that the data for the corresponding timeframe is stored within AppMon. If not, select more recent timeframe in the CAS before you drill down.

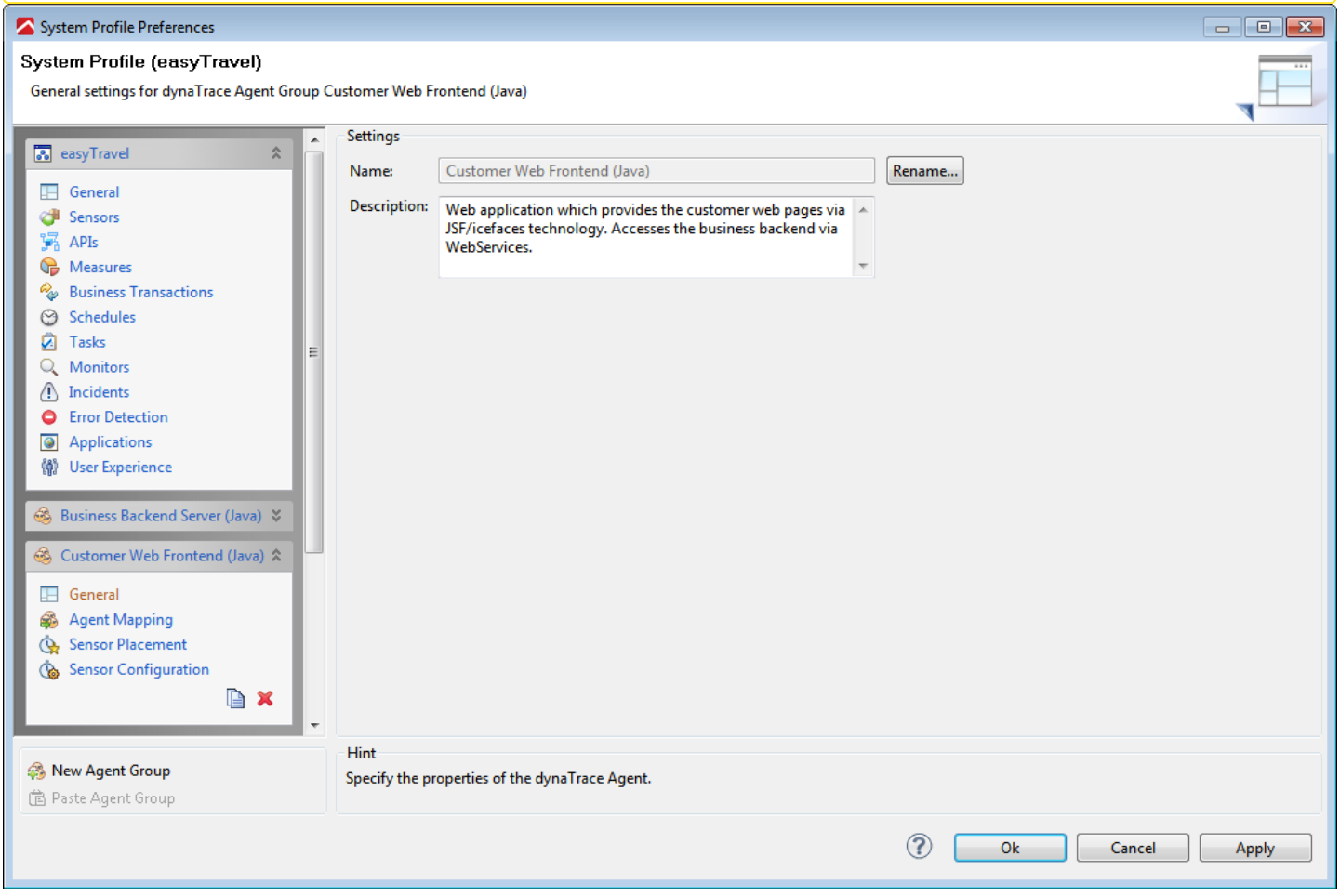
System Profile - Observed Tiers

Observed Tiers is technically related to the Transaction Flow. See Observed Tiers for more information. Access the **Observed Tiers** dialog box from the **System Profile** settings. It tells AppMon about tiers that cannot be automatically detected (such as a load balancer) because they don’t have an Agent, but should appear in the Transaction Flow.

Agent Group - General

Agent Group renaming

Renaming an Agent Group is an immediate action. You can't click **Cancel** and undo it in the System Profile dialog box. To undo a renaming operation, perform a second renaming operation and set the desired name explicitly.



View and Edit the Agent Group's Name and Description

Agent Group - Agent Mapping

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

How agent mapping works

Advanced settings

Related pages

Agent Group - Agent Mapping Advanced Settings

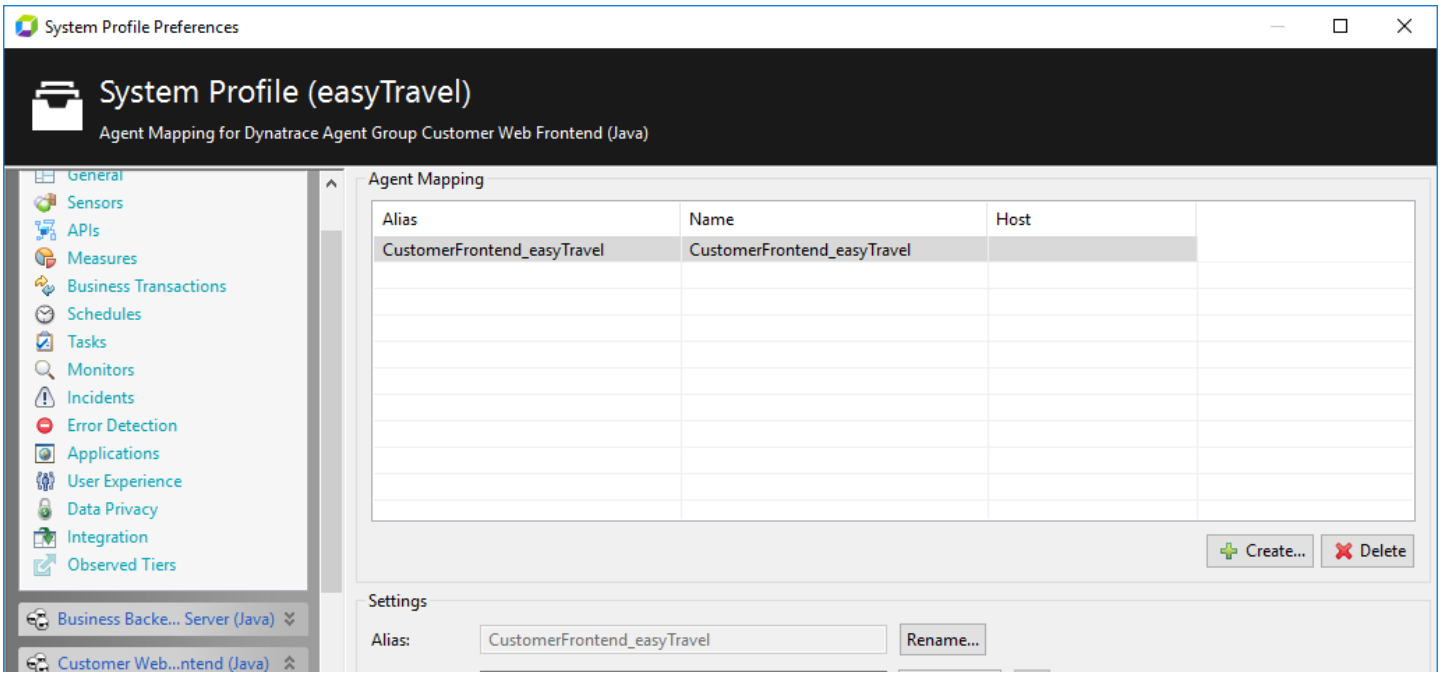
Agent Groups

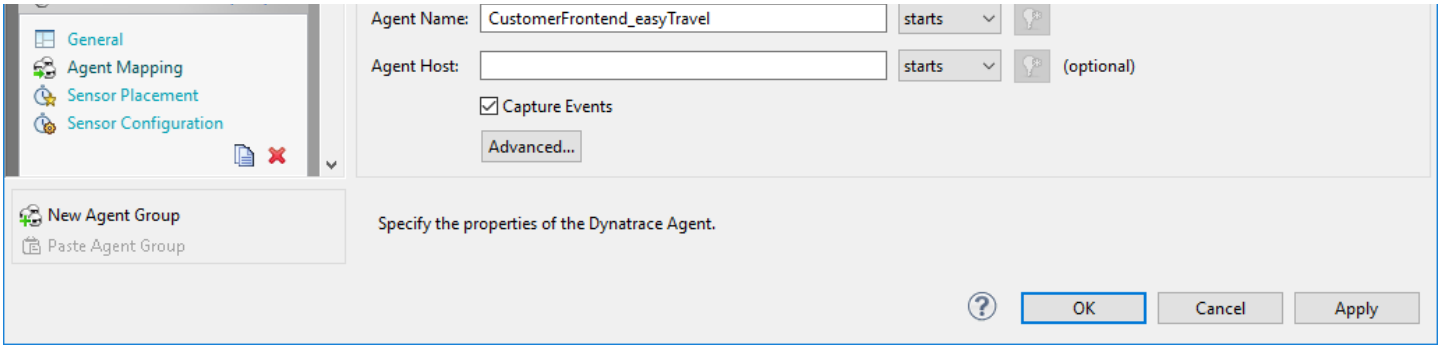
Hot Sensor Placement

Edit a System Profile

Agent timers

Agents Overview dashlet





Define name patterns to match differently named Agents to a specific Agent Group.

The Agent name may be sufficient to assign an Agent to the Agent Group, but in some cases it is useful. For example, using the Agent's host to assign an Agent to an Agent Group.

To match agent names and host names, use `starts`, `ends`, `contains`, `equals` and also `regex`, to specify a Java-style regular expression that is evaluated against the Agent name to decide if the Agent matches this system profile.

How agent mapping works

AppMon searches all System Profiles for matching Agent Mapping definitions if a new Agent connects to the AppMon Server. The following cases may occur:

Case	Description
No matching System Profile found, whether disabled or non-existant.	The AppMon Agent is connected but instrumentation is disabled. An incident is raised: *AppMon Self Monitoring - Ambiguous Agent Mapping*
One matching System Profile found.	The AppMon Agent is connected and the matching System Profile is applied.
More than one matching System Profile found, implying there are overlapping mapping definitions.	The AppMon Agent is connected and the first matching System Profile is applied. An incident is raised: *AppMon Self Monitoring - Ambiguous Agent Mapping*

Advanced settings

You can dynamically configure Agents that match this Agent Mapping. See Agent Mapping Advanced Settings to learn how.

Agent Group - Agent Mapping Advanced Settings

On this page

Access the Agent Mapping Advanced Settings dialog box

Agent Mapping advanced settings

Related pages

Agent Group - Agent Mapping

Agent Groups

Hot Sensor Placement

Edit a System Profile

Agent timers

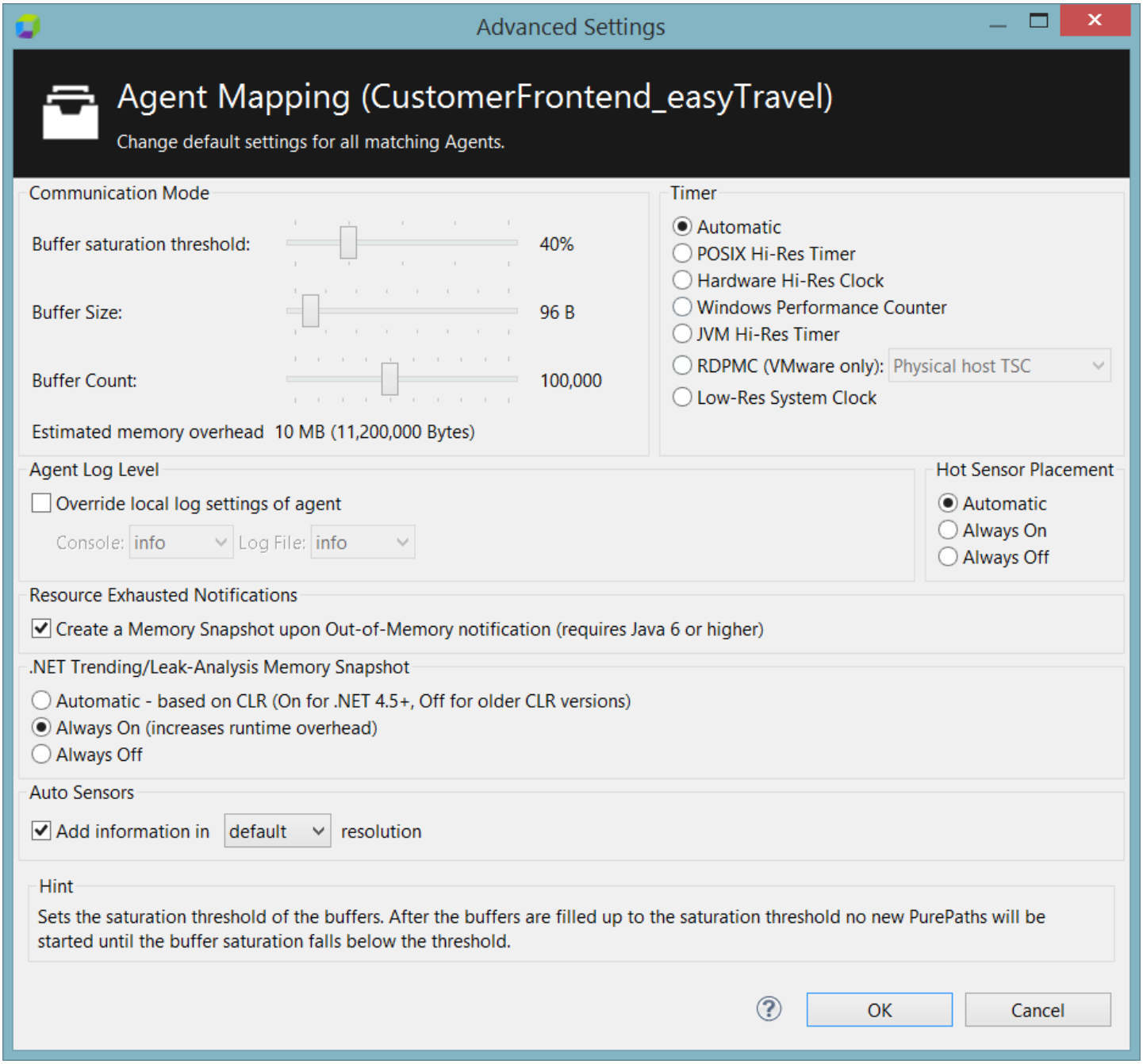
Agents Overview dashlet

Use the **Agent Mapping Advanced Settings** dialog box to dynamically configure Agents that match the Agent Mapping. The Advanced settings let you:

- Adjust communication buffers and specify a timer type that applies to a group of agents.
- Set agent log level.
- Enable hot sensor placement, Leak Analysis Memory Snapshot automatic generation, and Trending / Leak Analysis Memory Snapshot creation for .NET CLRs.
- Adjust auto-sensor resolution for tolerable overhead.

Access the Agent Mapping Advanced Settings dialog box

1. In the desired System Profile, expand the Agent Group, then select **Agent Mapping**.
2. In the **Agent Mapping** dialog box, click **Advanced** to open the **Advanced Settings** dialog box.



Agent Mapping advanced settings

Buffer saturation threshold: If the Agent's communication buffer is filled above this threshold, no new PurePaths are added to the buffer until the buffer empties below the threshold. This increases the amount of correct PurePaths in certain Agent overload scenarios.

Buffer Size and Count: Define the overall size of the Agent communication buffer. Overall size is about equal to size * count.

Timer: Specifies a specific timer type to use for measuring method execution time.

Timers measure method execution time. Different timers provide different accuracy and granularity, based on the OS and architecture.

If a selected timer is not available on the agent platform, the Agent uses the standard run-time (Low-Res System Clock) timer. If changing the timer type, you must restart the Agent for the change to take effect.

Use the Agents Overview Dashlet to see the timer used by a specific Agent. To do this, select the desired Agent in the Agent list. The **HiResClock** and **Timer** properties in the Details pane indicate the type of timer for the Agent.

Select the timer to be used for the Agent. See Agent Timers for descriptions of each timer type.

Agent Log Level: Override the Agent's default log level.

Hot Sensor Placement: Enable/Disable Hot Sensor Placement. If set to **Automatic** the Agent enables Hot Sensor Placement automatically if supported by the underlying VM. Older JVM's do not support Hot Sensor Placement or certain JVM versions contain bugs that prohibit Hot Sensor Placement use.

Resource Exhausted Notifications (only available when **Java** is enabled for System Profile): Enable/Disable automatic generation of a Leak Analysis Memory Snapshot in case a Java 6 VM reports an Out-of-Memory notification. This option is not visible if **Java** is not selected on the **General** pane of the System Profile. Enabling it only affects Java agents.

.NET Trending/Leak-Analysis Memory Snapshot: Enable or disables Trending / Leak Analysis Memory Snapshot creation for .NET CLRs. By default, this is set to **Automatic**. This means that this is determined by the CLR version, but is off by default with any version. **Off** means no Memory Snapshots are possible and no additional overhead due to CLR notifications is necessary for Memory Snapshots. If set to **Always On**, run-time overhead increases even if no Memory Snapshot is created. Memory Snapshots for CLRs can be created. If you change this setting, restart the SUD. For ASP.NET, the AppPool must be recycled. When you see **Agent configuration may cause high runtime overhead** in the Start Center, take the appropriate action.

> **Note**
>
> Enabling this introduces a significant amount of overhead. Garbage collection times increase. Enabling this setting is not recommend in a production environment.

Auto Sensors (Extended PurePath Information): Auto Sensors adjust automatically to ensure they do not incur more overhead than the percentage setting. The amount of detail captured depends on the complexity of the application or environment. You may need to adjust the resolution settings until you see the desired level of detail for a tolerable amount of overhead. You can enable or disable Auto Sensors for the given Agent Group. You can also select the amount of information (known as resolution) provided by Auto Sensors to be stored in PurePaths. A higher resolution provides more detailed PurePaths, but causes more traffic between Agents and the Server.

Auto Sensors must be enabled to have **sync and wait** times.

You can enable or disable Extended PurePath information and adjust tolerable overhead dynamically at run time. See Sensors > Auto Sensor Resolution Settings for overhead and detail numbers and defaults in Editions.

Select the desired resolution level from the dropdown list in the **Add Information in … resolution** option near the bottom of the dialog box. The default setting is **low** for the Production Edition, **medium** for Pre-Production Edition.

The following table shows the typical percentage of overhead for each setting. Note the overhead highly depends on the instrumented application and the level of instrumentation.

Overhead in %	Lowest	Low	Medium	High	Highest
Production Version	1	3	5	10	20
Non-Production	1	5	10	20	50

Changing the Advanced Settings does not require a restart.

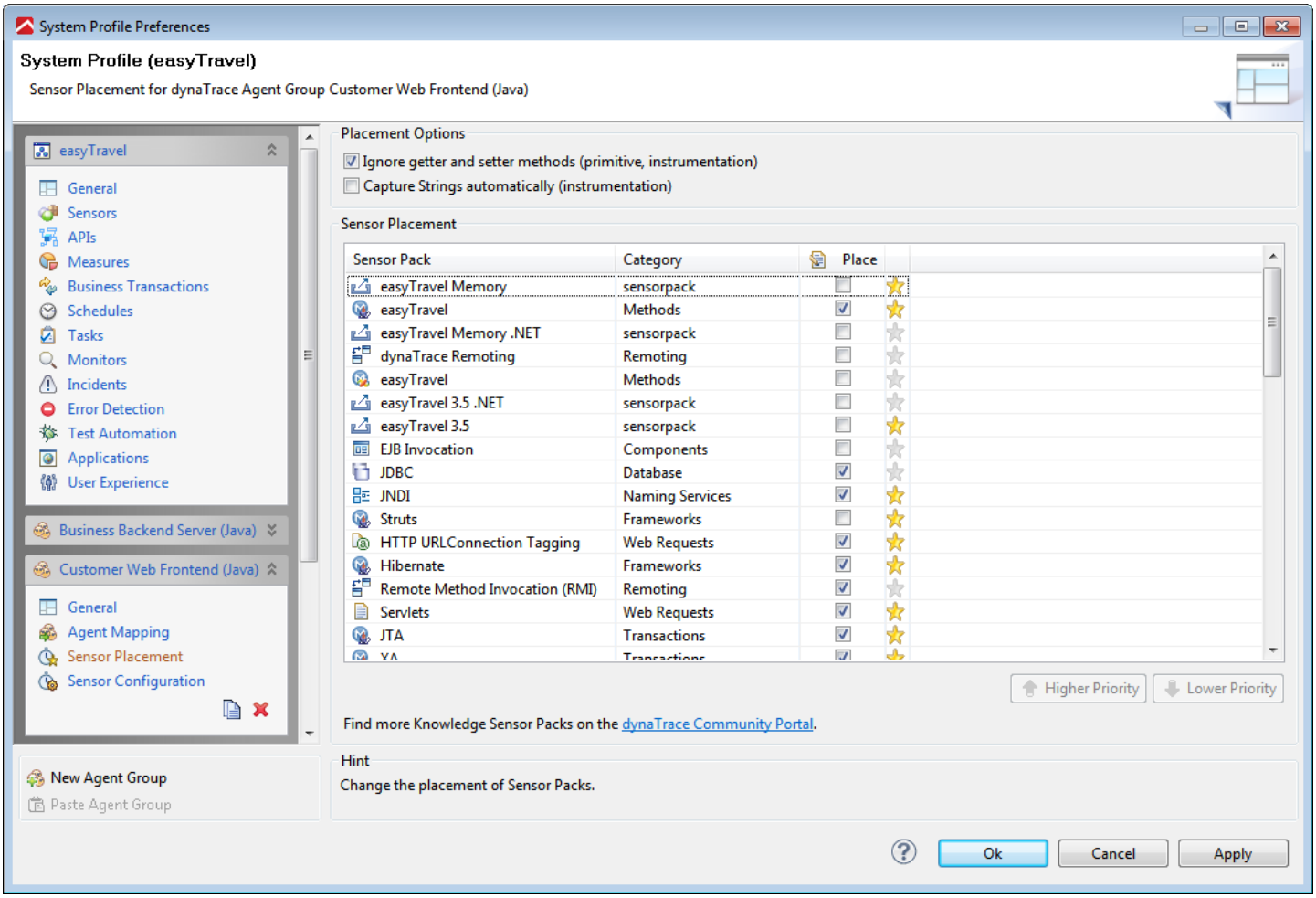
Agent Group - Sensor Placement

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Placement options

Sensor placement



Placement options

Ignore getter and setter methods (primitive, instrumentation)	Do not instrument methods that start with *get* or *set*. This option overrides all manual rules, including explicit Method Rules.
Capture Strings automatically (instrumentation)	Capture the first three method arguments' first string value.

Sensor placement

Define Sensor placement and priority. Hot Sensor Placement is required to apply placement changes to already connected AppMon Agents. If you cannot hot-place a Sensor or Hot Sensor Placement is not supported or enabled, you must restart the respective SUD's. Placed Hot Sensors include 
in the table. See Configuration Change Immediacy Matrix for more information.

> **Note**
>
> Only placed Sensors are configurable and visible in the Agent Group's Sensor Configuration section later.

See Usage of Agent Groups for more information.

Agent Group - Sensor Configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

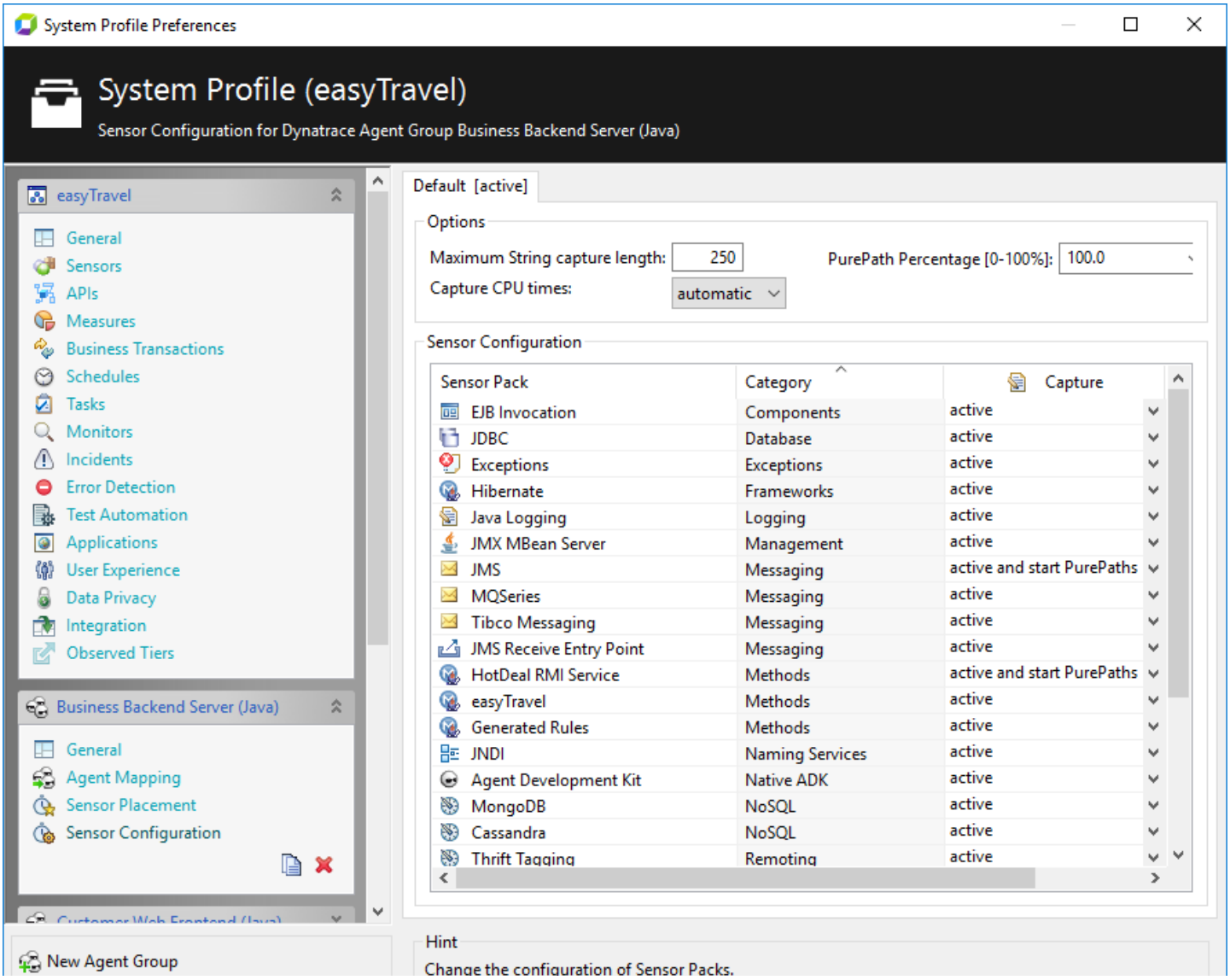
On this page

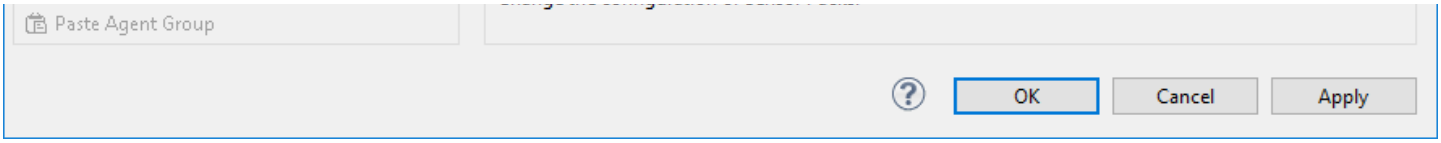
Bulk edit of the sensor properties

Configurations

Options

Sensor Configuration



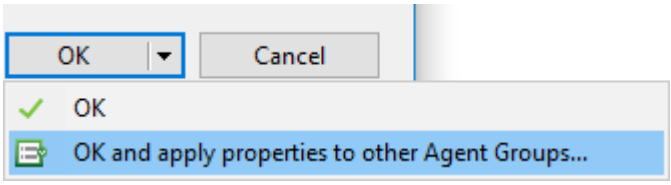


Dynamically Configure Sensors For an Agent Group

All configuration changes applied in this section apply to all connected AppMon Agents. No Hot Sensor Placement or SUD restarts are necessary. See Configuration Change Immediacy Matrix for more information.

Bulk edit of the sensor properties

You can save or change related Sensor configurations. To use this, select **OK and apply properties to other Agent Groups**.



Configurations

Creating additional configurations in the **General** section of the System Profile lets you define multiple Sensor Configuration sets that may be switched manually. You can edit each Configuration separately and store them inside the System Profile.

To switch configurations, use either the System Profiles context menu in the Cockpit or the **General** section of the System Profile.

Options

Option	Description
Max. String Capture Length	Truncate all captured strings to this maximum length. This prevents overhead from excessively long strings.
Capture CPU Times	Enable CPU Time Capturing. If set to **Automatic**, the Agent may disable CPU Time Capturing if estimated overhead is too high.

PurePath Percentage	Reduce percentage of captured PurePaths. For example, setting this to 50% only starts one out of every two PurePaths. This option reduces overhead from excessive PurePaths, but produces less data.
Synchronization Threshold	Enable **Synchronization Diagnosis** to edit this entry. See placement options in Sensor Placement. This threshold determines the time value which indicates a blocking lock.

Sensor Configuration

Set the Capture Settings of Sensor Packs:

Capture Setting	Description
inactive	Invisible inside an existing PurePath.
active	Visible inside an existing PurePath, but this Sensor is not en entry point for new PurePaths.
active and start PurePaths	Visible inside an existing PurePath and start a new PurePath if not yet started.

Click **Properties** to configure **Sensor Properties**:

- JDBC Sensor allows to enable bind value capturing.
- Exception Sensor allows to include/exclude specific Exception classes or capture stack traces.
- JMS Sensor allows to capture message content.
- WebService/Servlet Sensors allow to capture request/response sizes.

See Usage of Agent Groups for more information.

Sensor properties

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

ADO.NET

ASP.NET

CICS Transaction Server

EJB invocation

Java exceptions

Java Logging

Java Web requests

Java web services

JDBC

JMS Messaging

MongoDB

MSMQ

.NET Exceptions

.NET Logging

.NET Web requests

PHP Database

PHP Exceptions

PHP General

PHP Logging

Servlets

Tibco Messaging

User experience

Related pages

Agent Group - Sensor Configuration

Agent Group - Sensor Placement

Configuration change immediacy matrix

System Profile - Sensors

Sensor Configuration

To access sensor properties dialog box, right click your System Profile and select **Edit System Profile > Agent group > Sensor configuration**. Double click the required sensor.

ADO.NET

See ADO.NET sensor.

ASP.NET

See ASP.NET sensor.

CICS Transaction Server

See CICS Transaction Server sensor.

EJB invocation

See EJB Invocation sensor.

Java exceptions

See Exceptions sesnsor.

Java Logging

See Java Logging sensor.

Java Web requests

See Java Web Request sensor.

Java web services

See Java web services sensor.

JDBC

See JDBC sensor.

JMS Messaging

See JMS sensor.

MongoDB

See MongoDB sensor.

MSMQ

See MSMQ sensor.

.NET Exceptions

See .NET Exceptions sesnsor.

.NET Logging

See .NET Logging sensor.

.NET Web requests

See .NET Web Request sensor.

PHP Database

See PHP Database Sensor.

PHP Exceptions

See PHP Exceptions Sensor.

PHP General

See PHP General sensor.

PHP Logging

See PHP Logging sensor.

Servlets

See Servlets sensor.

Tibco Messaging

See Tibco sensor.

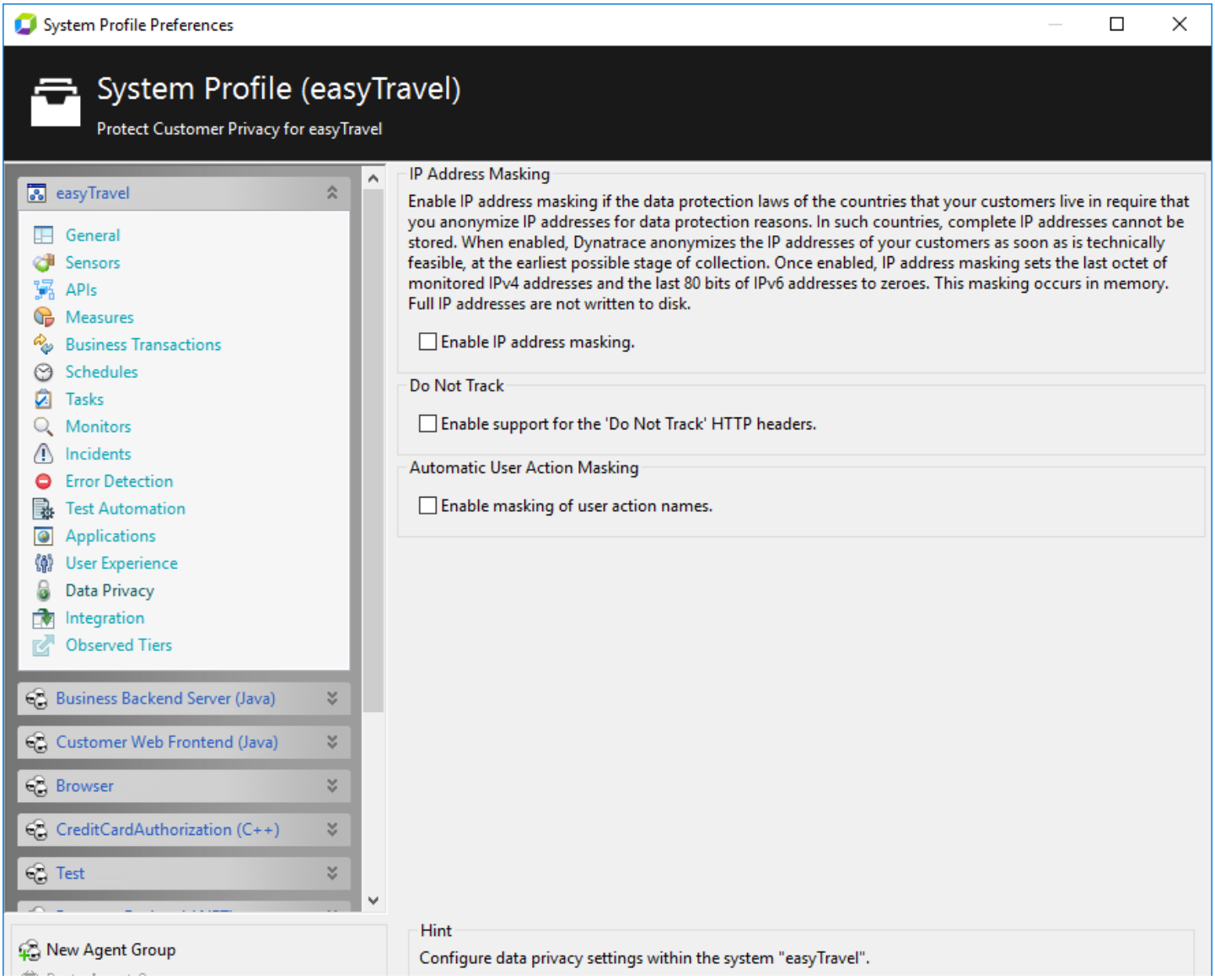
User experience

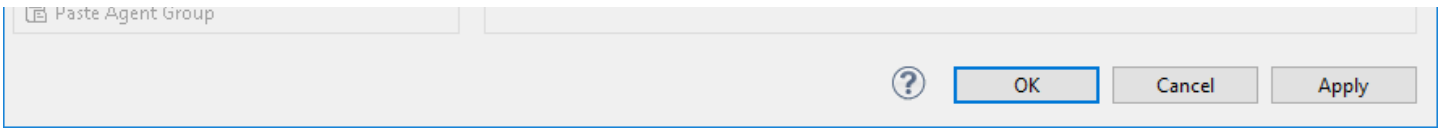
See User Experience sensor.

System Profile - Data Privacy

Method	Description
IP Address Masking	You can provide privacy to your customer's location by selecting this option. Once the IP address is collected by the AppMon Server in memory, it gets masked before being stored. Thus the original full IP address is not saved or displayed anywhere.
Do Not Track support	The `Do Not Track` HTTP header is for users who explicitly wish not to be tracked. This setting prevents page injection page if the header exists.
Automatic User Action Masking	Instead, selecting this option displays the HTML element's tag name. This is useful to prevent capturing private information in HTML elements on the page, such as a button with a user's full name as the text. This setting also suppresses capturing alphanumeric keypress values. This is done at the earliest possible stage of collection, so it it not sent on the monitor signal. You can see this setting indicated on the JavaScript Agent's injected `dtagent` script tag as `uam=true` appended to the `data-dtconfig` string.

This can be done in the following ways:





Provide data privacy for your customers for each system profile.

Rename a System Profile

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

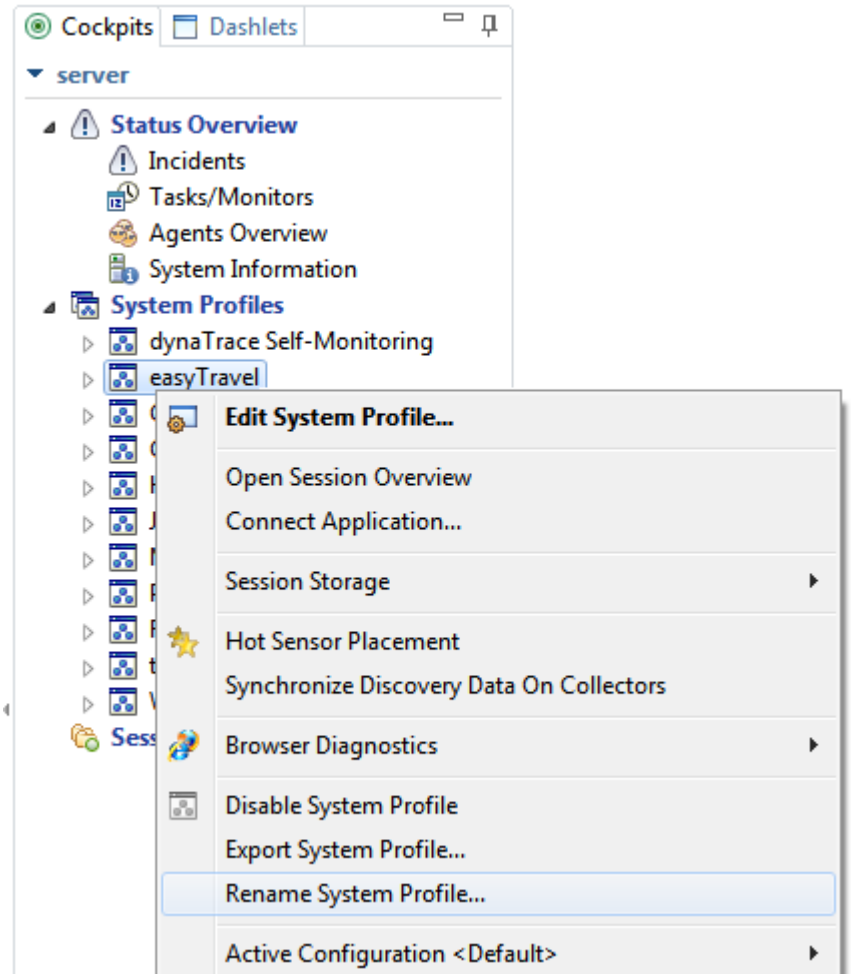
Cockpit dashlet

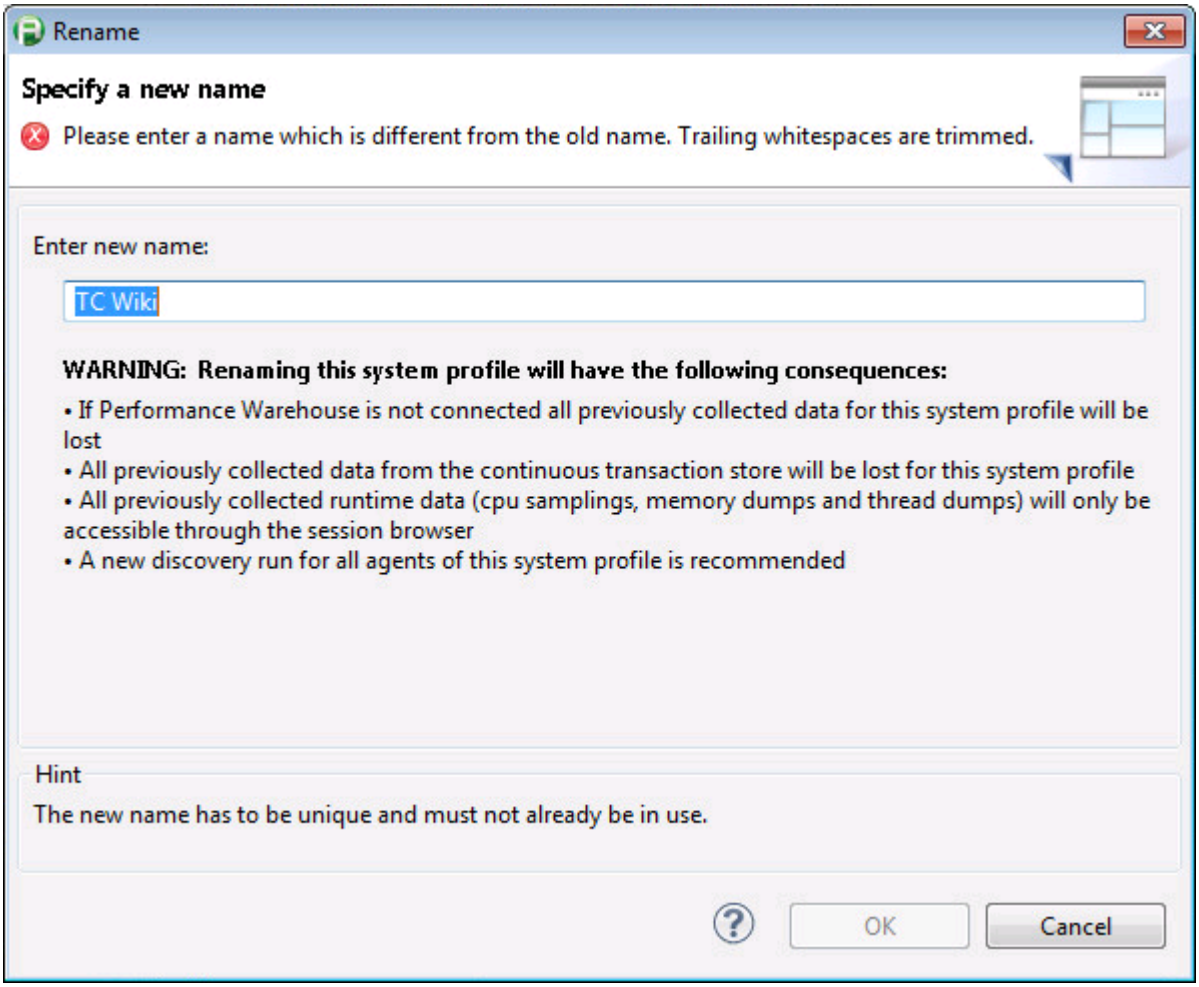
To rename a System Profile to a local XML file, select **Rename System Profile System Profile** from the context menu of a System Profile in the **Cockpits** tab.

Important

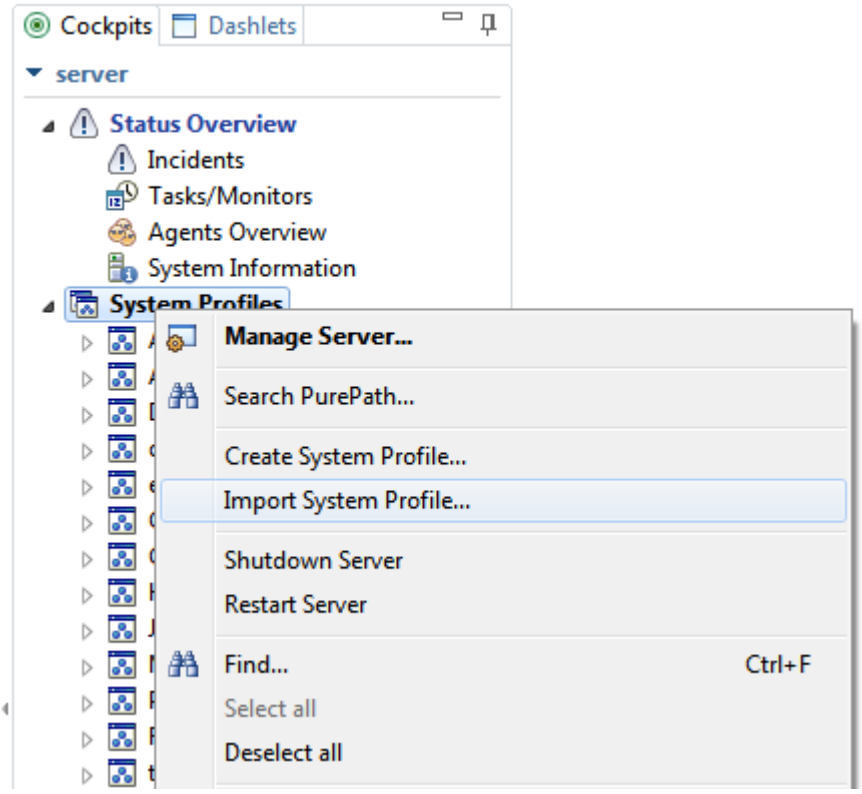
Use caution when you rename a System Profile because the name of the System Profile is used across the whole system. For example, dashboards refer to the System Profile name or tasks that reference the System Profile internally. Dashboards stored on the AppMon Server are updated automatically, and result in reload operations on AppMon Clients that have these dashboards currently opened. Offline dashboards (not stored on the AppMon Server) that refer to a renamed System Profile are invalidated by a rename operation and require a manual update of the dashboard's source or comparison source.

Rename a System Profile


Cockpits
Dashlets
server
Status Overview
Incidents
Tasks/Monitors
Agents Overview
System Information
System Profiles
dynaTrace Self-Monitoring
easyTravel
Edit System Profile...
Open Session Overview
Connect Application...
Session Storage
Hot Sensor Placement
Synchronize Discovery Data On Collectors
Browser Diagnostics
Disable System Profile
Export System Profile...
Rename System Profile...
Active Configuration <Default>


Rename
Specify a new name
Please enter a name which is different from the old name. Trailing whitespaces are trimmed.
Enter new name:
TC Wiki
WARNING: Renaming this system profile will have the following consequences:
• If Performance Warehouse is not connected all previously collected data for this system profile will be lost
• All previously collected data from the continuous transaction store will be lost for this system profile
• All previously collected runtime data (cpu samplings, memory dumps and thread dumps) will only be accessible through the session browser
• A new discovery run for all agents of this system profile is recommended
Hint
The new name has to be unique and must not already be in use.
OK
Cancel

Import a System Profile



Import a System Profile

See Edit a System Profile for more details on System Profile configuration.

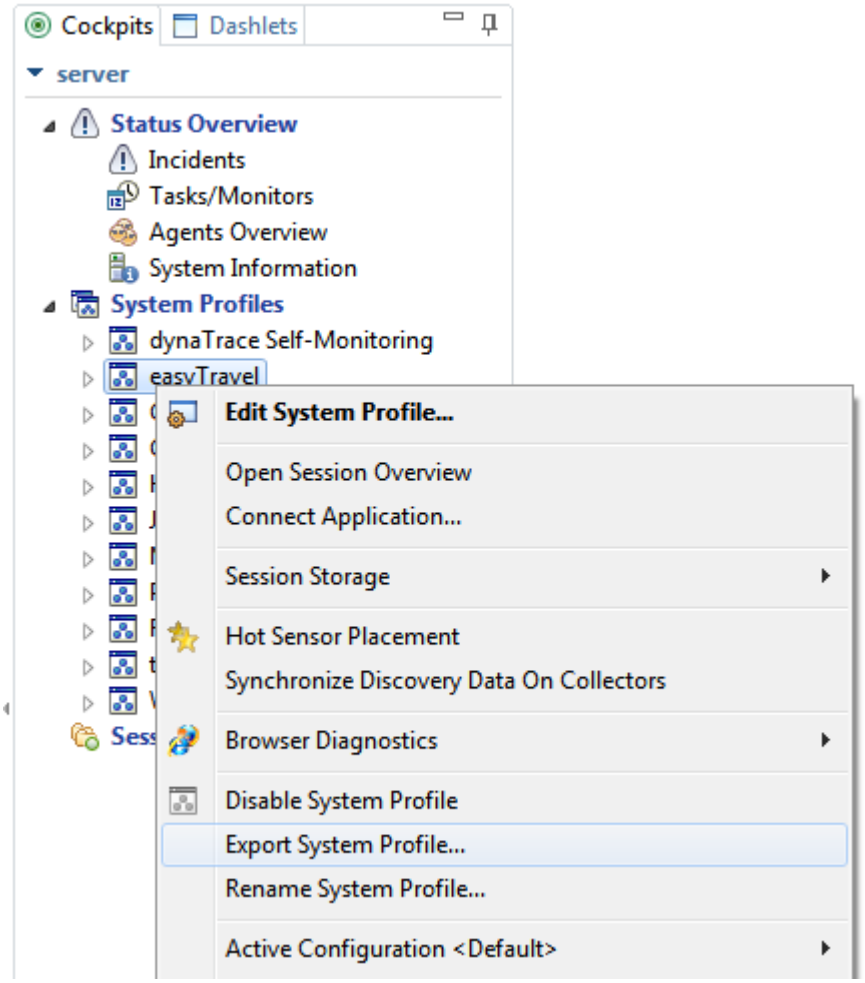
Known problem

Importing a System Profile with a different name is an implicit rename operation that invalidates all stored resource tasks in the System Profile, such as memory, thread, and sampling. Remove the tasks and manually add them again to correct the invalid tasks after you import such a System Profile.

If a license is not applied or the enabled System Profiles limit is met, new System Profiles are automatically disabled after import. As long as the System Profile is not stored on the AppMon Server if you click **OK** or **Apply**, Monitors and tasks cannot be executed within the System Profile.
To import a System Profile, select **Import System Profile** from the context menu of a System Profile in

the **Cockpits** tab. A new **System Profile Preferences** dialog box appears and displays the content of the imported System Profile. The imported System Profile's name is kept if it is unique. Otherwise you must specify a system-wide unique name to create a valid System Profile.

Export a System Profile



Export a System Profile.

Export a System Profile

Known problem

When you export a System Profile with a different name, it is an implicit rename operation that invalidates all resource tasks such as memory, thread, and sampling stored in the System Profile. Remove the tasks and manually re-add them to correct the invalid tasks after you import such a System Profile.

To export a System Profile to a local XML file, select **Export System Profile** from the context menu of

a System Profile in the **Cockpits** tab.

Enable or disable a System Profile

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Edit a System Profile

Cockpit dashlet

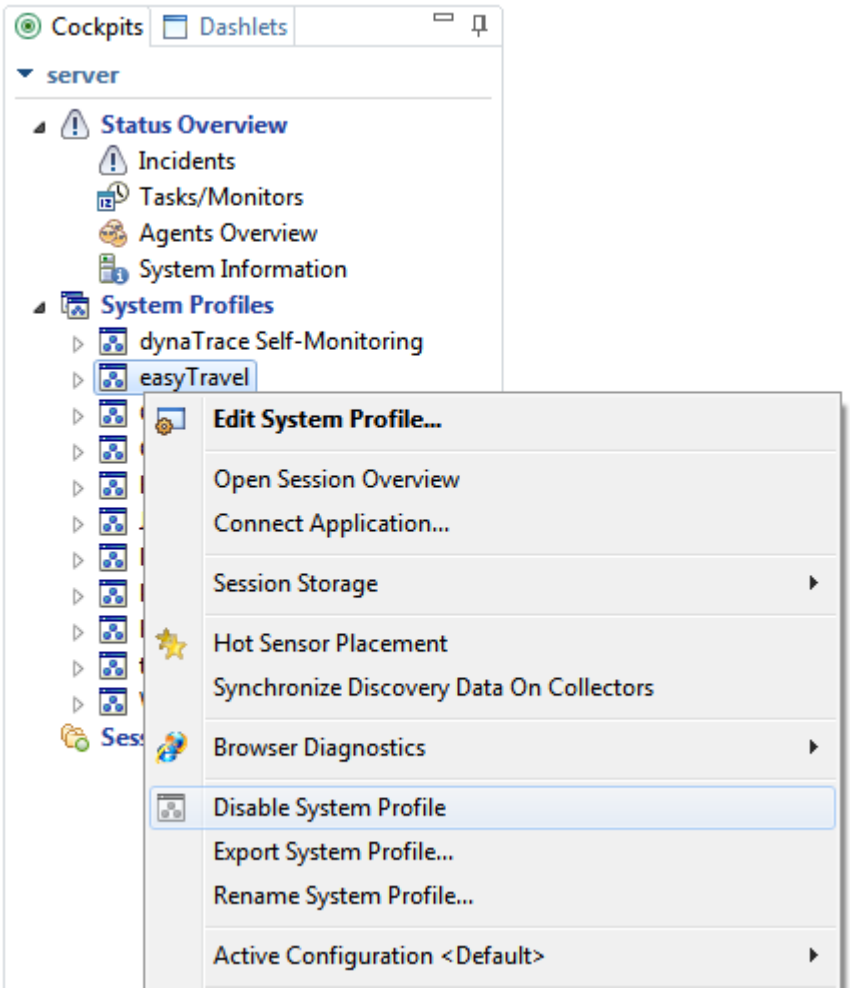
You can enable or disable System Profiles from the context menu of a System Profile in the **Cockpits** tab.

- Disabled System Profiles are still viewable and editable, but you can't use them for mapping AppMon Agents.
- AppMon Agents that are already mapped to a System Profile are not affected if the System Profile is disabled.

Note

If an Agent cannot be mapped to any System Profile, it is connected but not instrumented. No more System Profiles can be enabled if there is no license applied or the enabled System Profile limit is reached.

Enable or disable a System Profile.



Enable or disable a System Profile.

Disable a System Profile

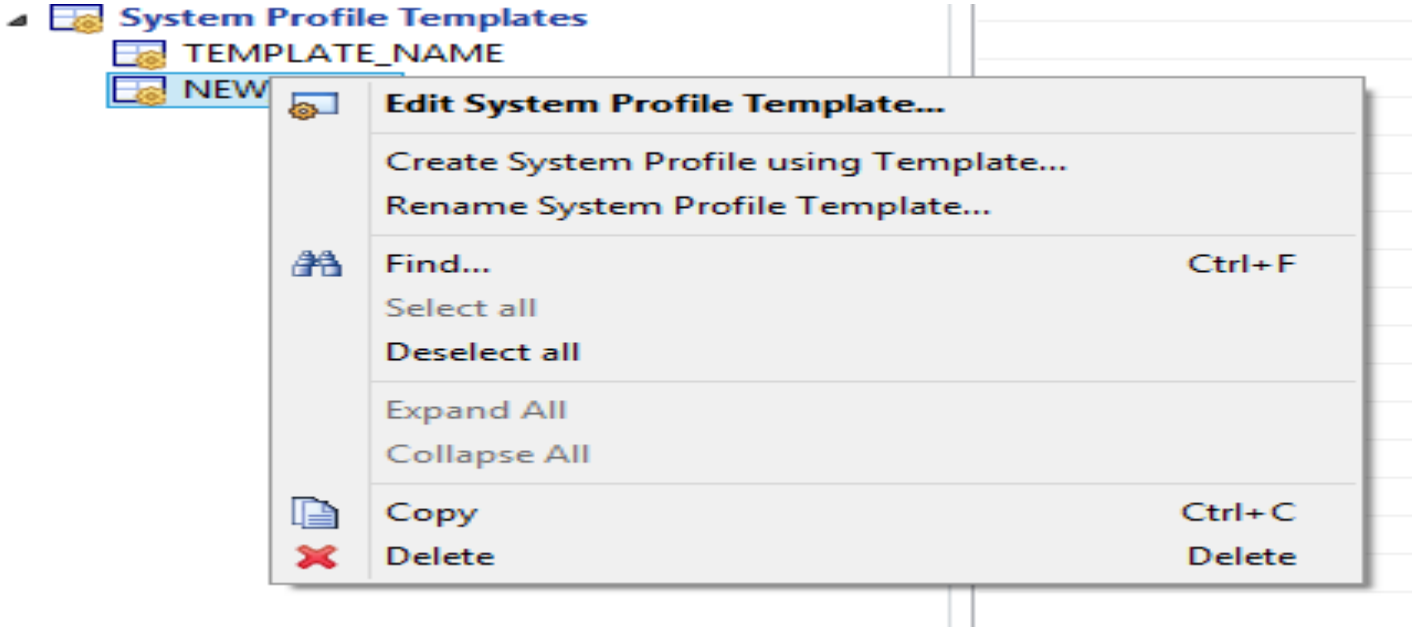
Multiple System Profiles with similar Agent Mappings are confusing. In big deployment scenarios, pick meaningful names for all AppMon Agents and map them into clearly defined Agent Groups. You cannot do this if you copy or import a System Profile.

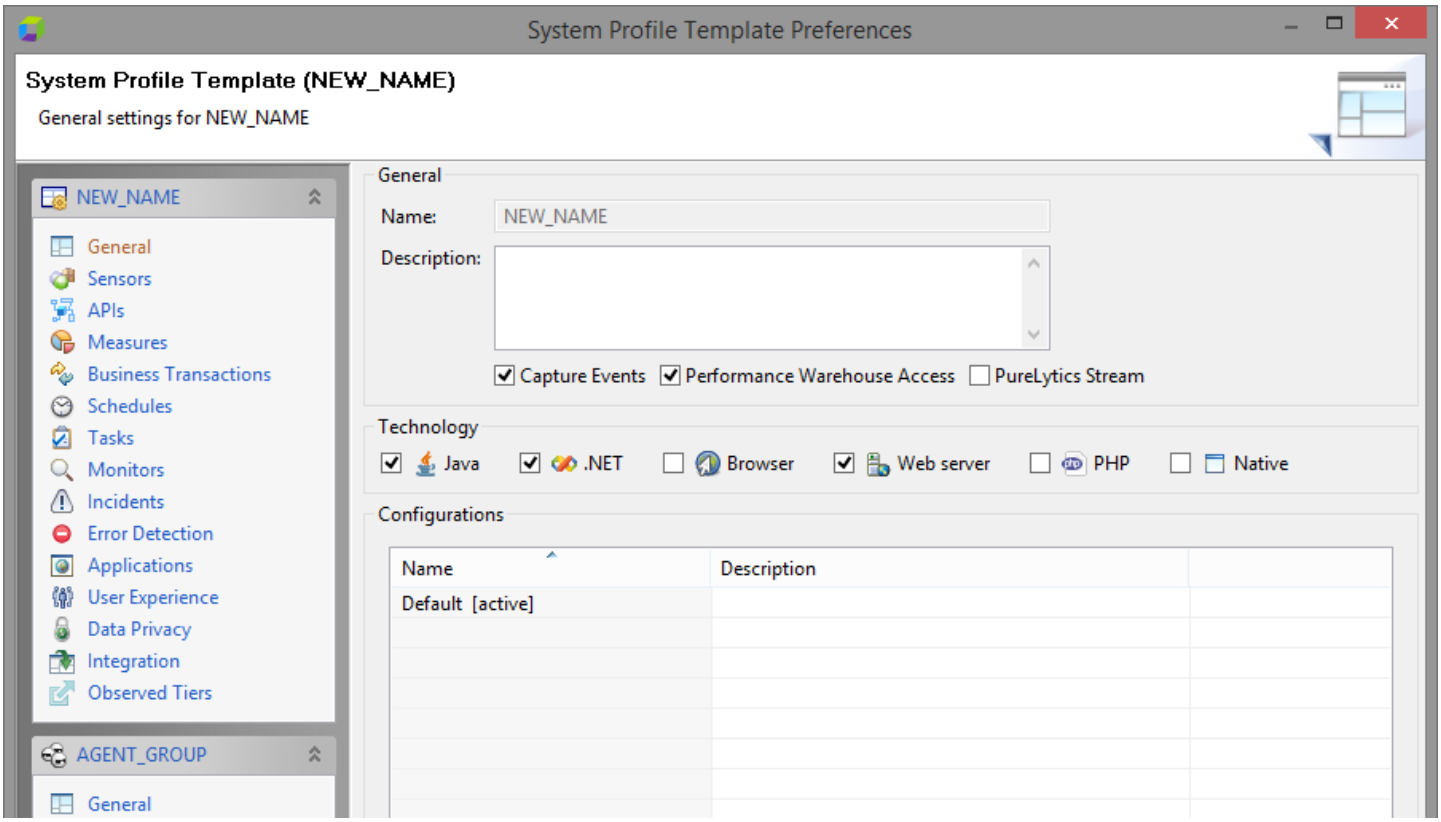
You can disable System Profiles to find configuration problems easier. Disabling skips the System Profile Agent Mapping evaluation for all incoming AppMon Agents. Additionally, any Scheduled Tasks or Monitors are not executed until the System Profile is activated again.

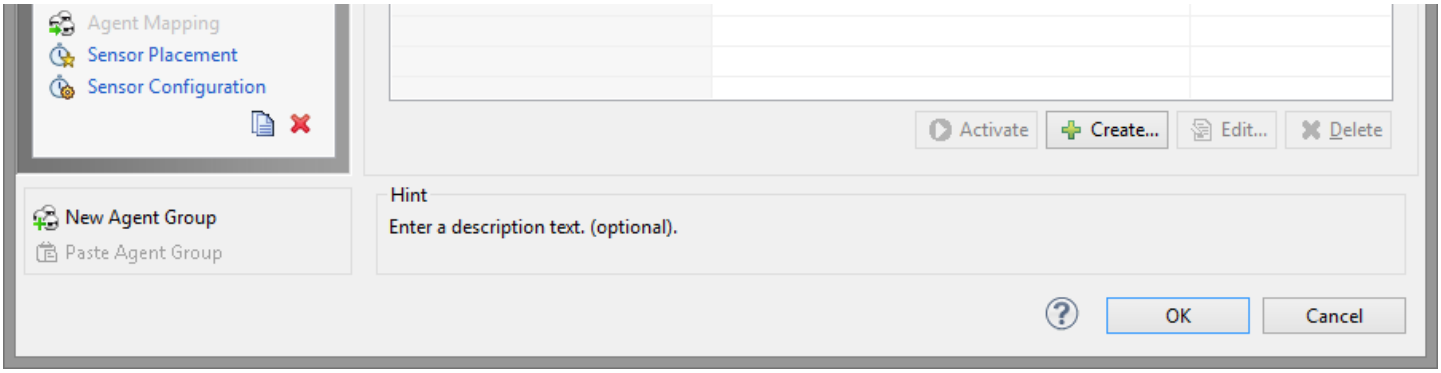
System Profile templates

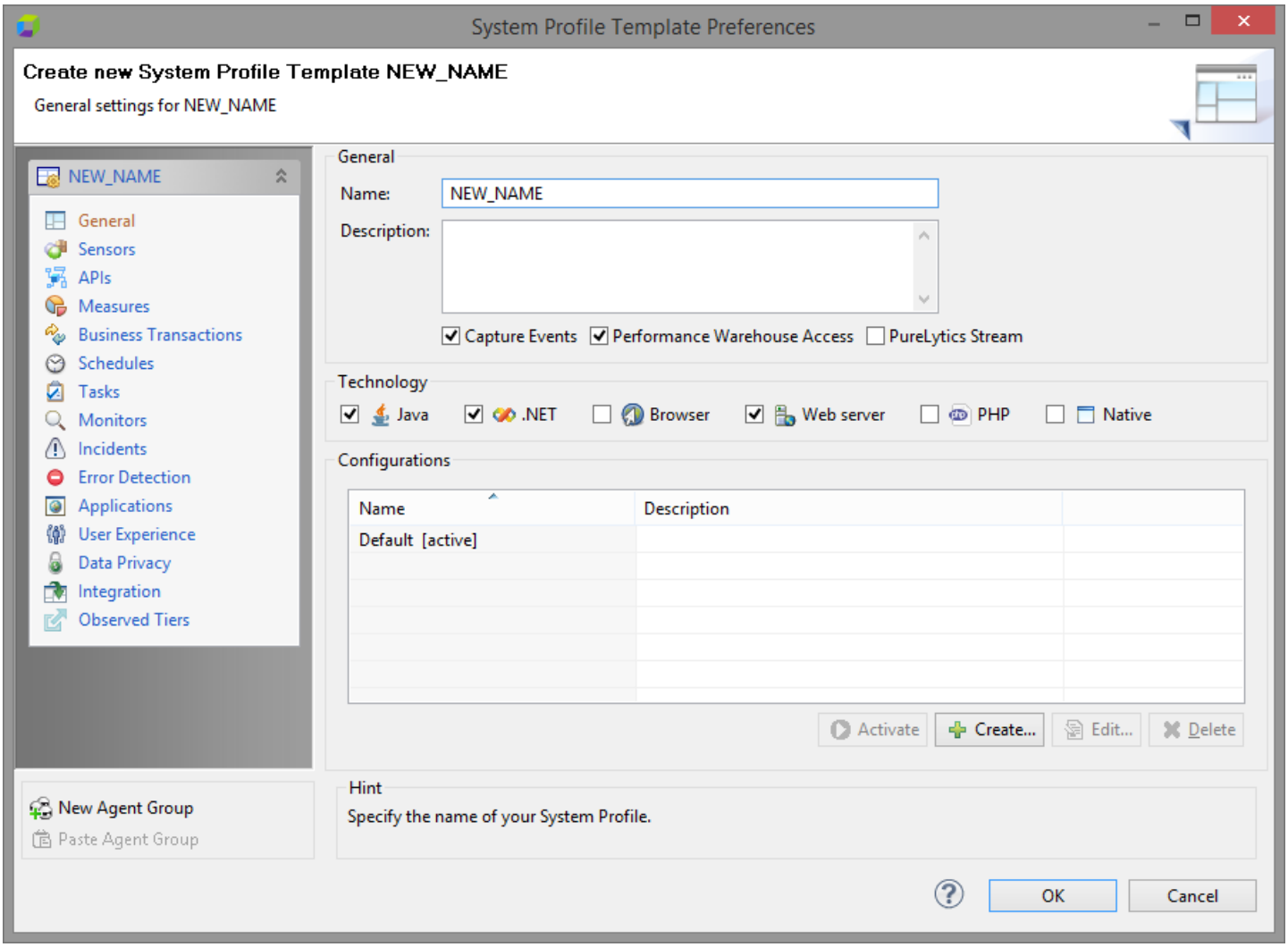
Creating a System Profile out of an existing Template can be done by accessing the Cockpit's menu or by using REST.

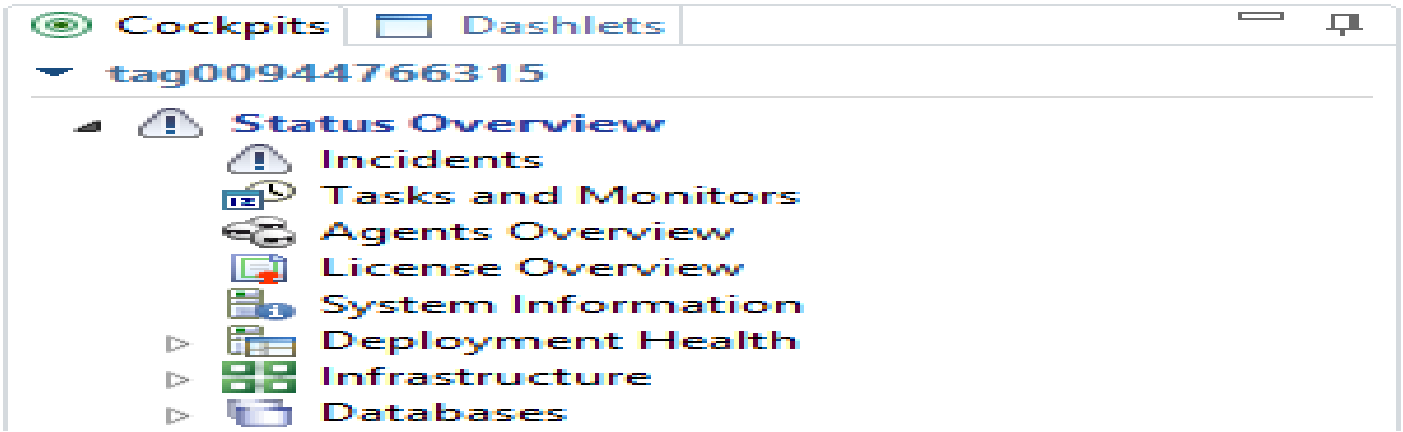
System Profile Templates can be accessed using the Cockpit. Adding an Agent Group to a System Profile Template does not allow the user to create an Agent Mapping as well.






Agent Mapping
Sensor Placement
Sensor Configuration
Activate
Create...
Edit...
Delete
New Agent Group
Paste Agent Group
Hint
Enter a description text. (optional).
OK
Cancel


System Profile Template Preferences
Create new System Profile Template NEW_NAME
General settings for NEW_NAME
NEW_NAME
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Data Privacy
Integration
Observed Tiers
General
Name:
NEW_NAME
Description:
Capture Events
Performance Warehouse Access
PureLytics Stream
Technology
Java
.NET
Browser
Web server
PHP
Native
Configurations
Name
Description
Default [active]
Activate
Create...
Edit...
Delete
New Agent Group
Paste Agent Group
Hint
Specify the name of your System Profile.
OK
Cancel


Cockpits
Dashlets
tag0094476631 5
Status Overview
Incidents
Tasks and Monitors
Agents Overview
License Overview
System Information
Deployment Health
Infrastructure
Databases



Creating a System Profile template

Only administrators can create and modify System Profile Templates. System Profiles can be created based on a given template via REST. System Profile Templates do not have Agent Mappings. When a System Profile is created based on a template the Agent Mappings have to be created manually.

System Profile - Business Hours

On this page

Business Hours for Tasks and Monitors

Business Hours for alert suppression

Related pages

Alert suppression based on Business Hours (REST)

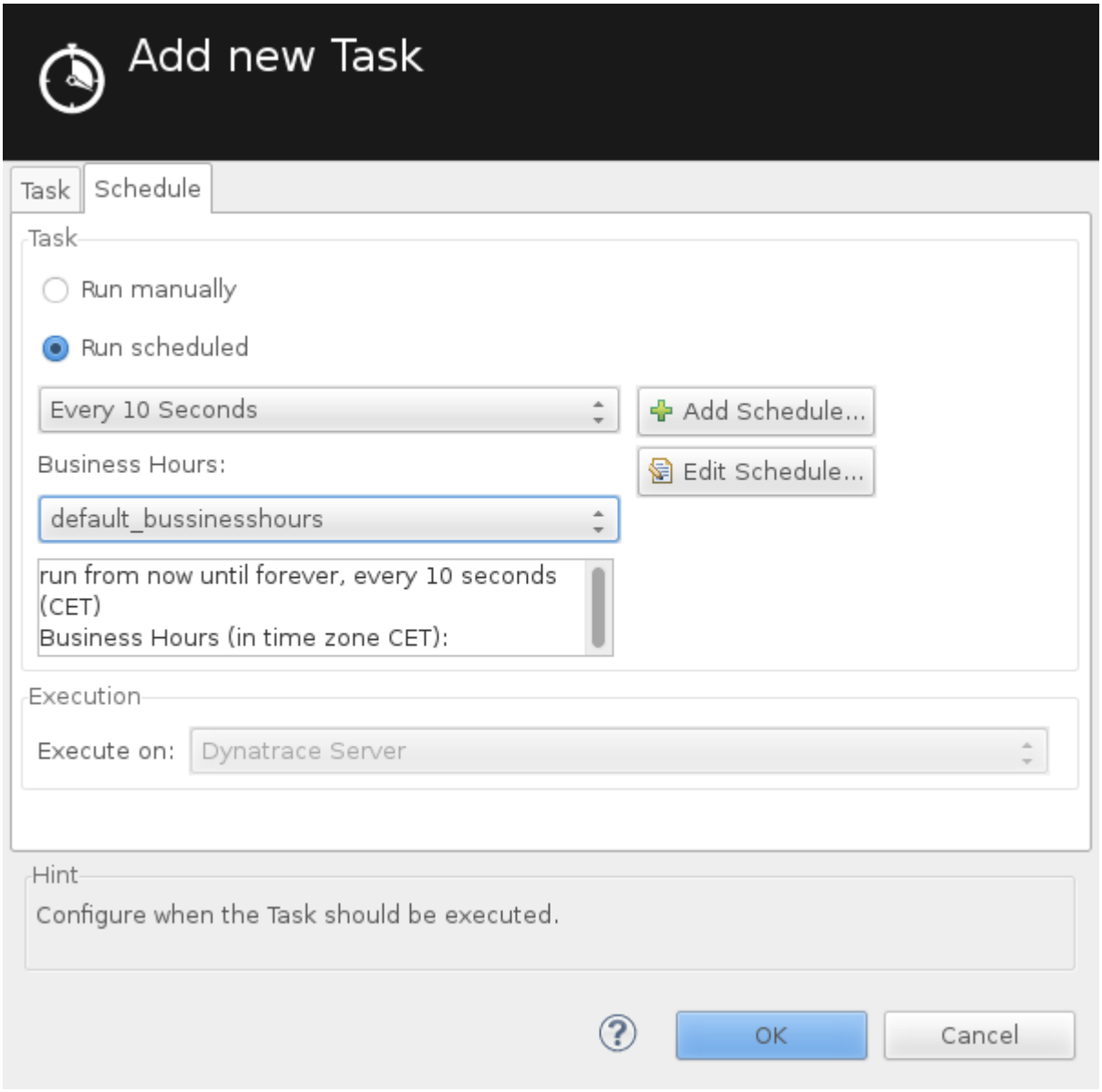
Business Hours define a set of time ranges, which can be used to restrict the scheduling of tasks for System Profiles. Business Hours can be configured through the Server REST API.

The JSON representation used by the REST API for the simple Business Hours definition “Monday to Friday 9am to 5pm” is:

```
{
  "inclusions": [
    {
      "timespan": {
        "start": "09:00",
        "end": "17:00"
      },
      "weekdays": [
        "MON", "TUE", "WED", "THU", "FRI"
      ]
    }
  ]
}
```

This Business Hours definition can then be used to restrict task execution or Incident creation to the defined time range, as explained in the following subsections.

Business Hours for Tasks and Monitors



When selecting a schedule for a Task or Monitor, a Business Hours definition from the same System Profile can be additionally selected to restrict Task execution to time ranges included in the Business Hours. For example, a Task scheduled to run every ten seconds and restricted to the Business Hours with the definition of “Monday to Friday 9am to 5pm” executes every ten seconds during the work day, but is not executed early in the morning, in the evening and on weekends.

Business Hours for alert suppression

Alert suppressions (also known as Incident Downtimes) can be based on a Business Hours definition. During all time ranges **not included** in the Business Hours, no Incident fires. Use the Server REST API to configure Alert suppressions based on Business Hours.

Note

Alert suppressions can only be based on Business Hours defined in the Self-Monitoring System Profile (id: dynaTrace Self-Monitoring) irrespective of the actual System Profiles affected by the suppression.

Sensor Configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Sensor terminology

Sensor rules

Memory sensor rules

Related pages

Class Browser

Configuration change immediacy matrix

Hot Sensor Placement

System Profile - Sensors

Sensor terminology

Sensor Rules: Define the classes and methods that should be instrumented. Sensor Rules can be either:

- **Method Sensor Rules**: These rules relate to the Java/.NET Method Sensor, and collect PurePath method related information.
- **Memory Sensor Rules**: These rules relate to the Java/.NET Memory Sensor, and collect PurePath object allocation information and Selective Memory Dumps.

- **Sensors**: Each Sensor consists of a set of Sensor Rules. Sensors are divided into the following:
  - **Sensor Groups**: Sensors stored and available only in a specific System Profile.
  - **Sensor Packs**: Sensors deployed on an AppMon Server and available in all System Profiles on this AppMon Server. Sensor Packs are created and maintained by AppMon users.
  - **Knowledge Sensor Packs**: Sensor Packs created and maintained by AppMon or by the Dynatrace Community.

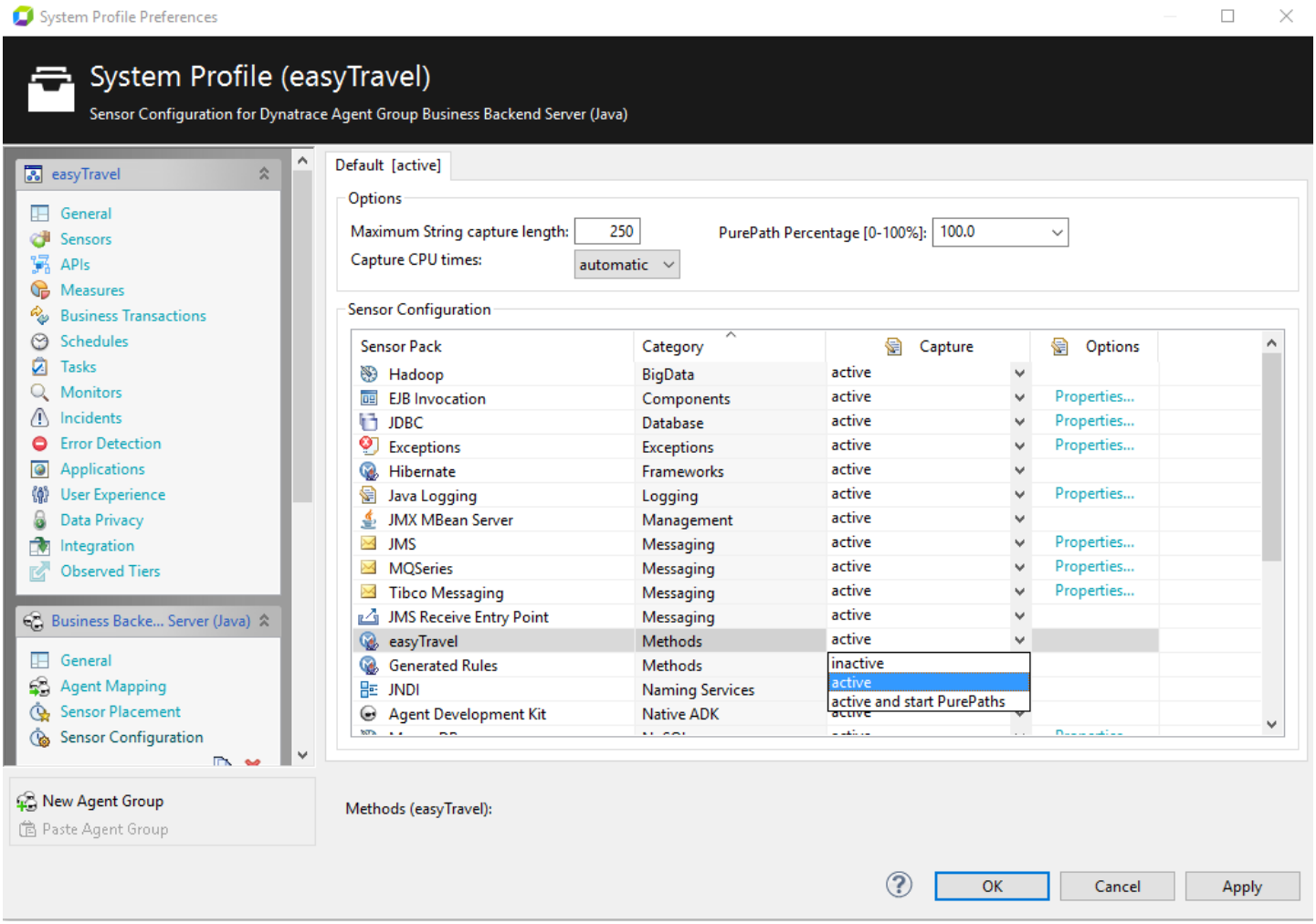
Sensor rules

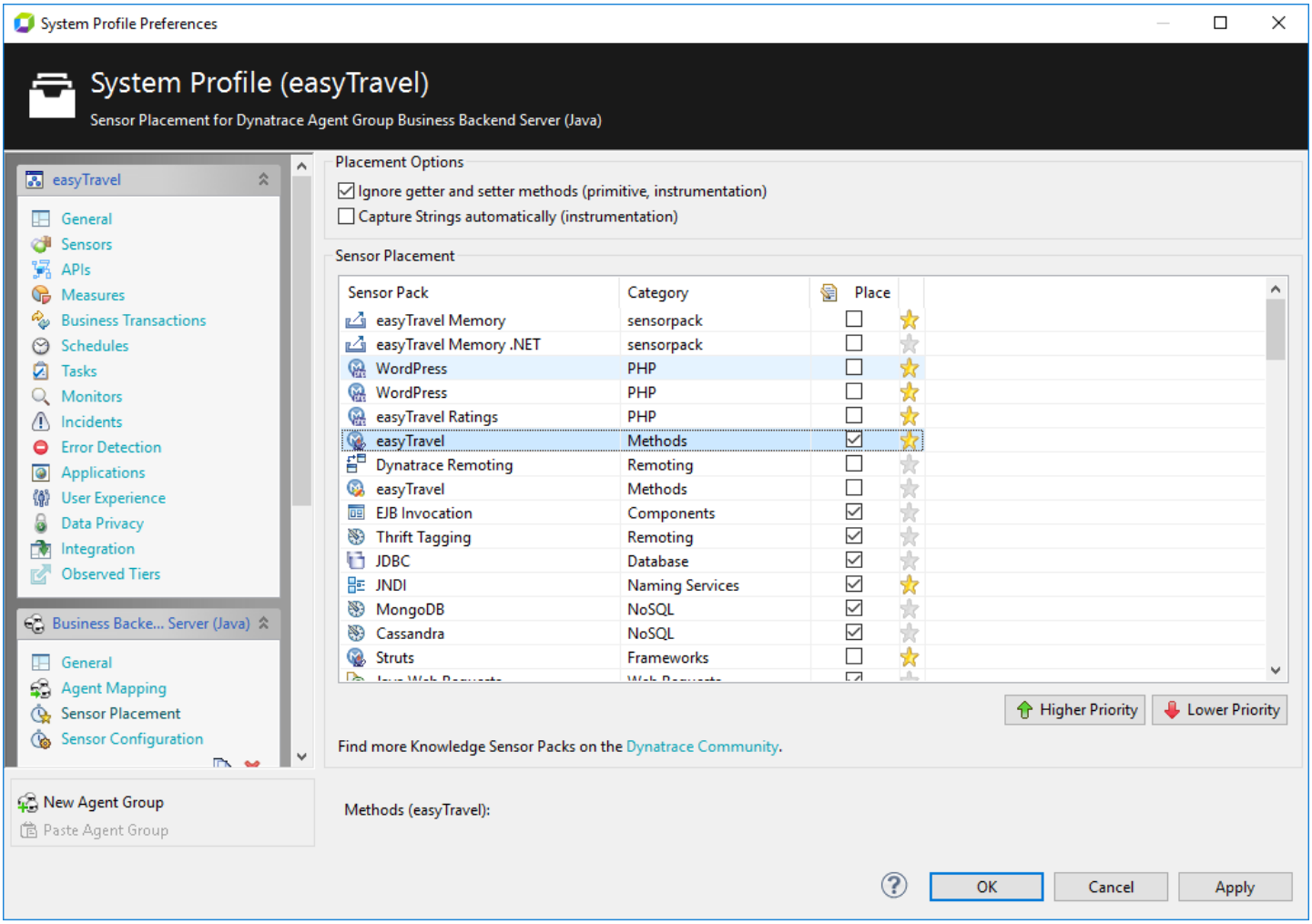
By default, AppMon places Knowledge Sensor Packs on all important component interfaces.

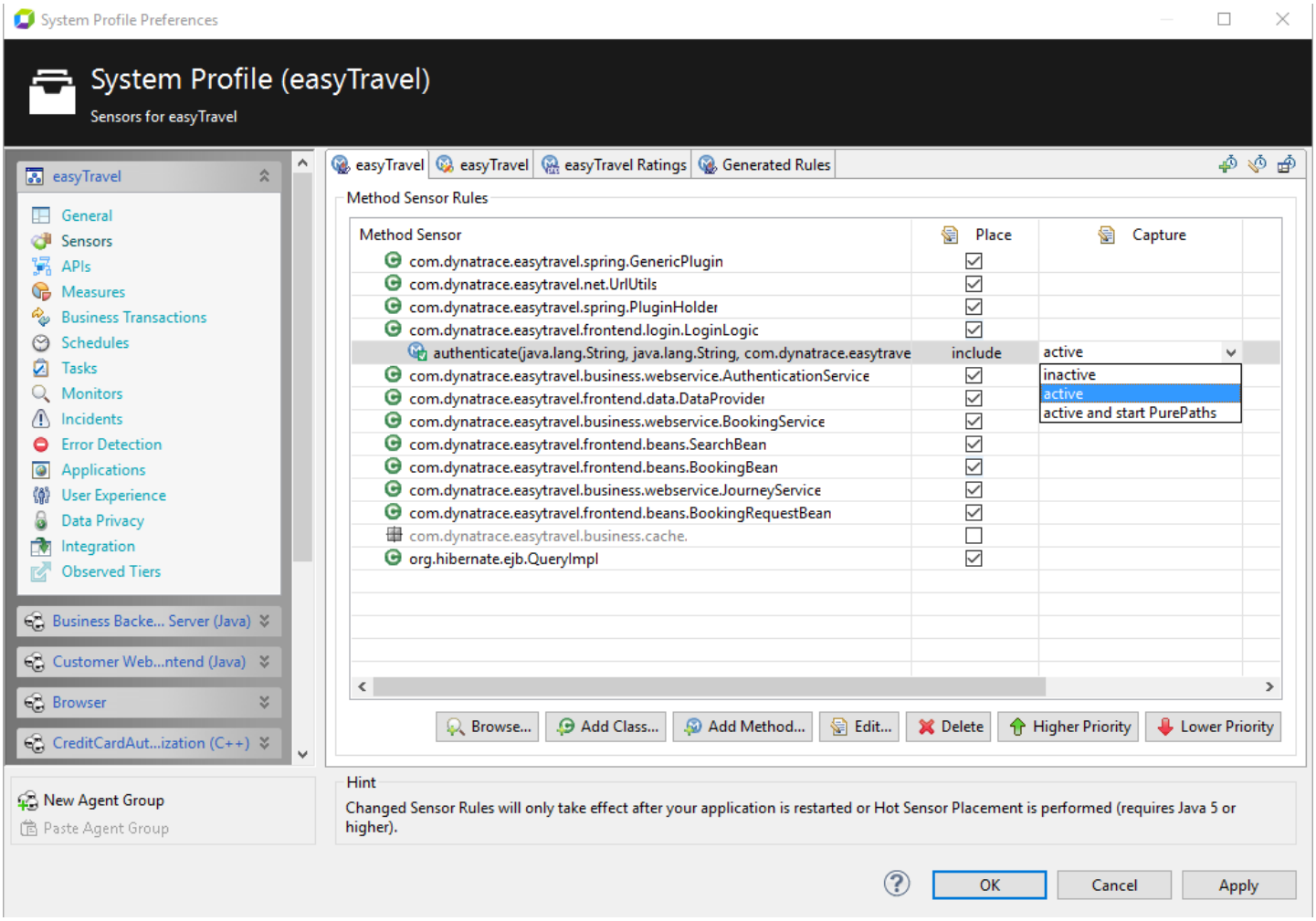
Use Method Sensor Rules to diagnose performance issues deeper in the application at the package/namespace, class or method levels.

AppMon provides a Discovery Run to discover what packages/namespaces, classes and methods are available to instrument. It locates all packages/namespaces of the application. The Method Sensor Rule configuration is used to gather information about specific packages/namespaces, classes and methods. Additionally, AppMon Method Sensor Rules capture specific method arguments and return values which are very useful in diagnosing issues.

System Profile Preferences > Sensors


System Profile Preferences
System Profile (easyTravel)
Sensor Configuration for Dynatrace Agent Group Business Backend Server (Java)
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Data Privacy
Integration
Observed Tiers
Business Backe... Server (Java)
General
Agent Mapping
Sensor Placement
Sensor Configuration
New Agent Group
Paste Agent Group
Default [active]
Options
Maximum String capture length: 250
PurePath Percentage [0-100%]: 100.0
Capture CPU times: automatic
Sensor Configuration
Sensor Pack
Category
Capture
Options
Hadoop BigData active
EJB Invocation Components active Properties...
JDBC Database active Properties...
Exceptions Exceptions active Properties...
Hibernate Frameworks active
Java Logging Logging active Properties...
JMX MBean Server Management active
JMS Messaging active Properties...
MQSeries Messaging active Properties...
Tibco Messaging Messaging active Properties...
JMS Receive Entry Point Messaging active
easyTravel Methods active
Generated Rules Methods inactive
JNDI Naming Services active
Agent Development Kit Native ADK
Methods (easyTravel):
OK
Cancel
Apply


System Profile Preferences
System Profile (easyTravel)
Sensor Placement for Dynatrace Agent Group Business Backend Server (Java)
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Data Privacy
Integration
Observed Tiers
Business Backe... Server (Java)
General
Agent Mapping
Sensor Placement
Sensor Configuration
New Agent Group
Paste Agent Group
Placement Options
Ignore getter and setter methods (primitive, instrumentation)
Capture Strings automatically (instrumentation)
Sensor Placement
Sensor Pack
Category
Place
easyTravel Memory
sensorpack
easyTravel Memory .NET
sensorpack
WordPress
PHP
WordPress
PHP
easyTravel Ratings
PHP
easyTravel
Methods
Dynatrace Remoting
Remoting
easyTravel
Methods
EJB Invocation
Components
Thrift Tagging
Remoting
JDBC
Database
JNDI
Naming Services
MongoDB
NoSQL
Cassandra
NoSQL
Struts
Frameworks
Higher Priority
Lower Priority
Find more Knowledge Sensor Packs on the Dynatrace Community.
Methods (easyTravel):
OK
Cancel
Apply



Single Method Sensor Rules can be specified and grouped into a *Sensor Group*. Method Sensor Rules which are added to a Sensor Group are prioritized by their order. As traffic passes through the Sensor Group the rule that matches first is the first and *only* one applied to that traffic. Method Sensor Rules priority can be specified, also method rules within a class rule can be prioritized.

If a PurePath is not started automatically by a built-in sensor, the *Entrypoint* for recording the PurePath must be specified by setting the **Capture** setting to **active and start PurePaths**.

A Sensor Pack can be generated out of one or more Class Rules or Method Rules with the **Create Sensor Pack** context menu option. The new Sensor Packs listed within the **Sensor Packs** dialog box in the AppMon Server preferences and is available for all Agent Groups in all System Profiles.

The Class Browser provides a convenient way to define Class Rules and Method Rules. See Class Browser for more information.

Sensor Groups are available in the **Sensor Placement** and **Sensor Configuration** dialog boxes for all Agent Groups within the System Profile. Modify Sensor Groups using these dialog boxes in the same way as you would modify predefined Sensor Packs.

Each new System Profile is created with two default Sensor Groups per technology type.

- Default Method Sensor Group
- One default Memory Sensor Group.

See System Profile - Sensors for more information on handling Sensor Groups.

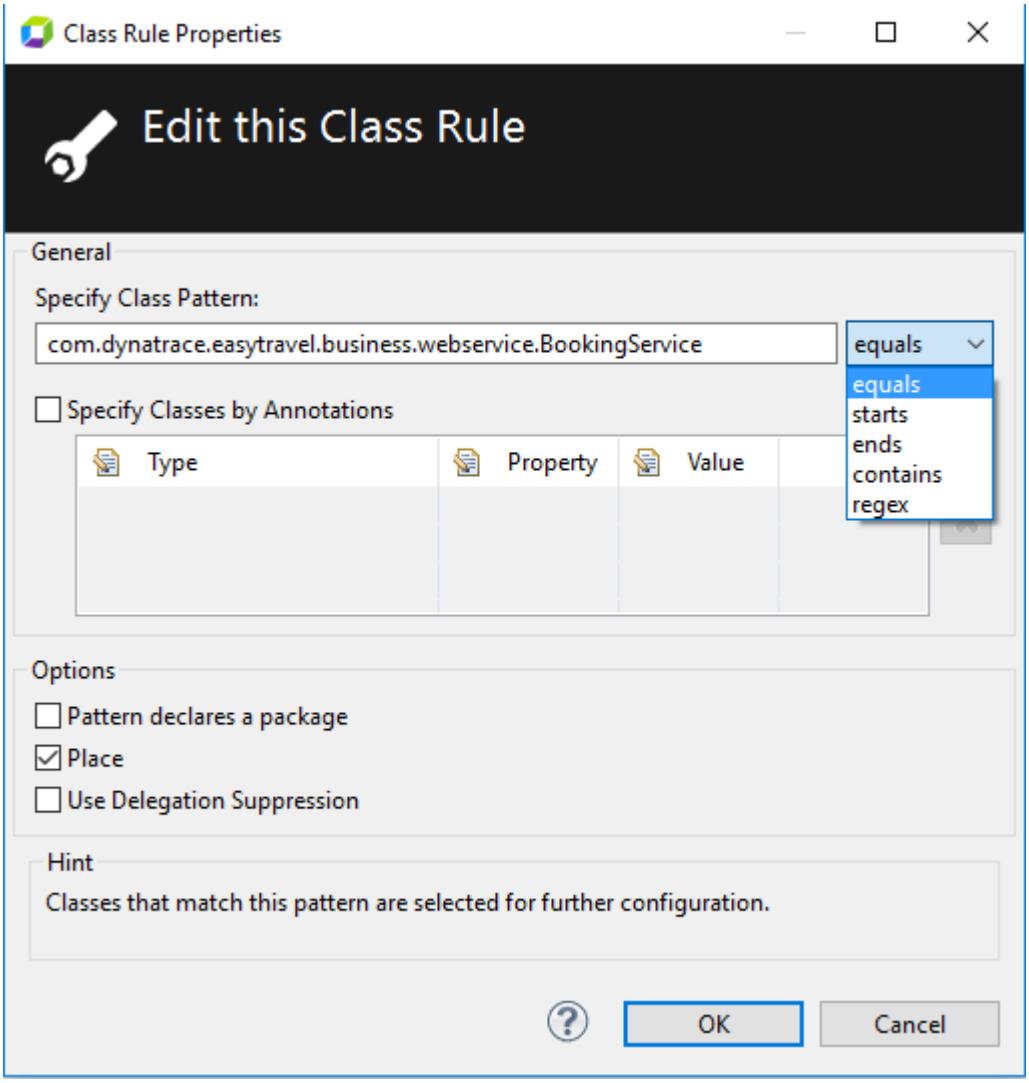
Group Sensor Rules logically into corresponding Sensor Groups for better maintainability.

> **Overhead**
>
> Add only packages/namespaces, classes or methods which are really critical in terms of performance to a Sensor Group. Avoid processing too much non-essential information.

Class rules

A Class Rule defines the container for a number of Method Rules. These Method Rules are evaluated with respect to the Class Rule they are defined within. A Class Rule with an empty pattern affects all classes. A Class Rule whose selection pattern ends with a dot is a package/namespace rule. It matches all classes within the specified package/namespace. The run time technology (Java or .NET) of a Class Rule is defined by add either a Java or a .NET Sensor Group.

System Profile Preferences > Sensors > select class > Edit. Double-click or select **Add Class**.



Class and Method Rules do not necessarily specify concrete names of classes/methods. Instead they use *patterns* which are evaluated when the names of the current class and method are available. These patterns can be specified using one of the following modes:

- **equals**: The pattern applies if the actual name is equal to the pattern text.
- **starts**: The pattern applies if the actual name starts with the given pattern text.
- **ends**: The pattern applies if the actual name ends with the given pattern text.
- **contains**: The pattern applies if the actual name contains the given pattern text.
- **regex**: Allows regular expressions. For example, `ClassName[0-9]+` specifies all classes that start with `ClassName` followed by any number.

Notes

Use a regular expression only when necessary, because it's "costly" on the CPU. Above a `starts` may be good enough if the classes are named efficiently.

Empty Class Rules that result in `<all classes>` are ignored by the instrumentation engine due to possible performance penalties.

General

Specify Class Pattern: Class Rules are specified by patterns that reflect the fully qualified class name and can also define interfaces or packages/namespaces.
Specify Classes by Annotations/.NET Attributes: You can use Java Annotations/.NET Attributes as selection criteria for classes. You can specify one or more Annotations/.NET Attributes per Class Rule.

An Annotation/.NET Attribute specification consists of a mandatory type, which defines the full package/namespace name of the Annotation/.NET Attributes, an optional property, and a value.

A specified Property with an omitted value results in only the Property being is evaluated. If both Property and Value are specified, the actual Values of Annotation/.NET Attribute Properties are compared with the specified Property Values. The specified Annotations/.NET Attributes are conjugated. For example, a Class must provide all Annotations/.NET Attributes specified by a Class Rule to match said Rule.

Switch Evaluation of Annotations/.NET Attributes on and off with the **Specify Classes by Annotations/Attributes** check box. To define a Class Rule that is based only on Annotations/.NET Attributes, leave the Class Pattern blank and switch the comparison mode to "starts".

Options

Pattern declares a package: This check box determines if the Class Rule matches single classes or a whole package/namespace. A Class Rule defined for a package/namespace matches all Classes in the package/namespace.

Note

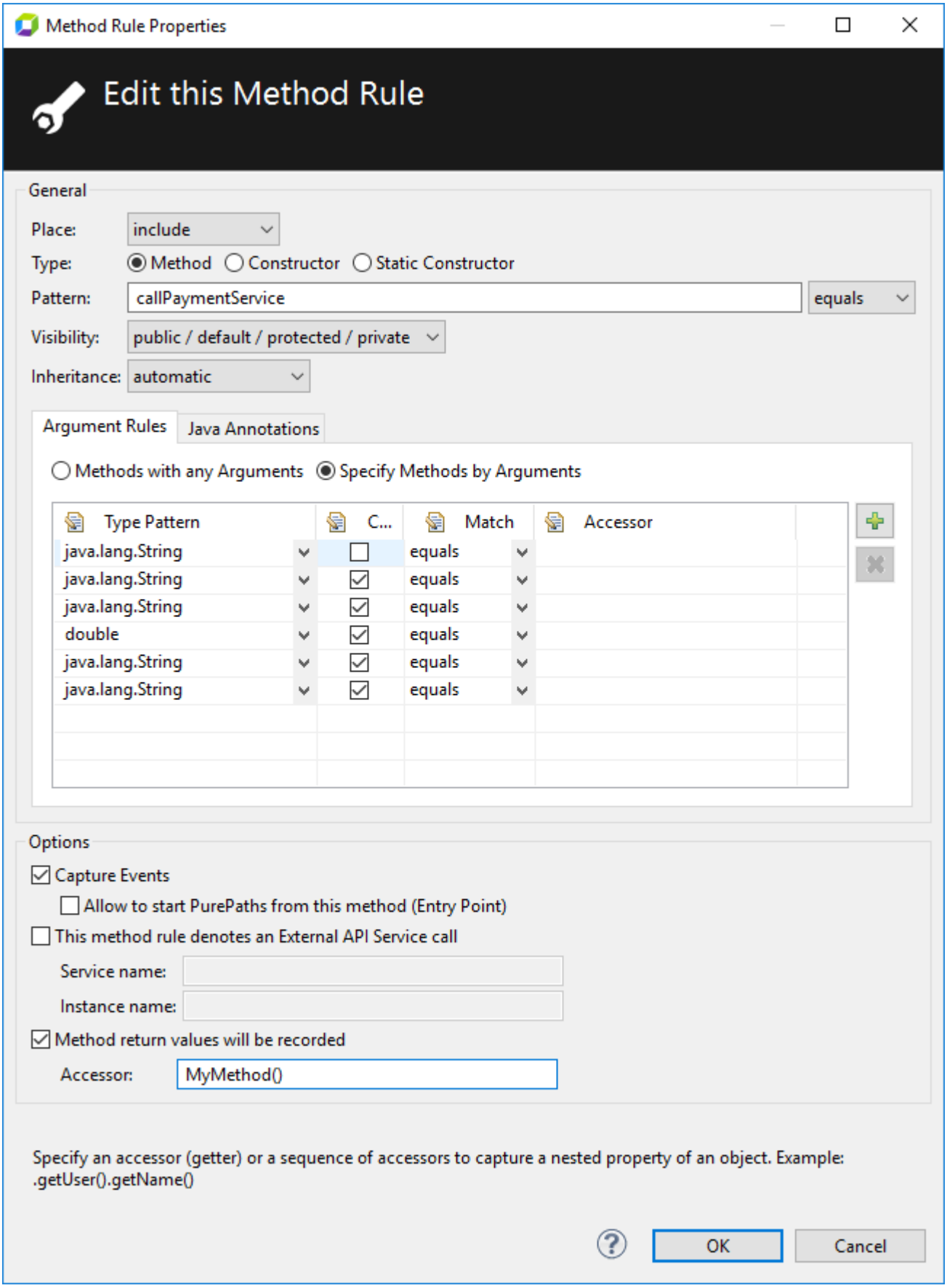
If you specify a package/namespace with mode **equals**, only classes contained in that package/namespace are processed, sub-packages are ignored.

Place: Enable or disable a Class Rule. **Delegation Suppression**: When Delegation Suppression is turned on, the Method Rules contained in that Class Rule are combined to one logical group. Each time one method matching one of these rules calls another method matching these rules, only the first call is shown in the corresponding PurePath. A method such as method read() should be instrumented in one specific class and all of its super-classes (this could be done by using inheritance modes, described below) and this method makes a call to the method read() of its super-classes. As a result, there would be a chain of read() method calls in the PurePath.

To get a clearer PurePath, one could use Delegation Suppression to make all the following calls to read() disappear in the PurePath.

Method Rules

System Profile Preferences > Sensors > select method > Edit. Double-click or select **> Add Method**.



Method Rules define one or more methods and how they should be instrumented. A Method Rule with an empty pattern affects all methods.

Note

Constructors are excluded by default.

General

Type: Defines the type of method that should be instrumented, which can be a method (static or instance), a constructor or a static constructor.
Pattern:The string pattern used to match names of actual methods.
Visibility: Defines the visibility of methods that get instrumented (public, default, protected, private).
Inheritance: When you use this option, you can instrument methods of a specified class and methods of super- or sub-classes. The following modes are available:

- **this method**: Only methods are instrumented where the containing class matches the corresponding Class Rule.
- **inherited methods**: All methods are instrumented where the containing class matches the corresponding Class Rule or the containing class extends a class or implements an interface that matches the corresponding Class Rule. For example, by declaring the method `toString()` of `java.lang.Object` to be instrumented with inherited methods, the method `toString()` of every class gets instrumented.
- **super methods**: All methods are instrumented where the containing class matches the corresponding Class Rule or the containing class is a super class to a class that matches the corresponding Class Rule. For example, by declaring the method `equals()` of `java.lang.String` to be instrumented with super methods, the method `equals()` of `java.lang.Object` also is instrumented.
- **automatic**: If the pattern of the corresponding Class Rule is an interface, the mode inherited methods is used, otherwise this method is applied.

Note

At run time, the complete inheritance hierarchy of a diagnosed system may not be known. Especially when a system is diagnosed for the first time, Method Rules using inherited methods, super methods or automatic may not be correctly applied. In this case, restart the System under Diagnosis to solve the problem because the AppMon Collector remembers the inheritance hierarchy of introspected applications.

Notes

- You can specify Method Rules in a Class Rule, although no methods that match these Method Rules are contained in a matching class. For example, you can create a Method Rule with pattern `substring()` for a Class Rule with pattern `java.lang.Object`. This does not seem to make sense, but when you use this Method Rule in mode "inherited methods", the method `substring()` of `java.lang.String` is instrumented, because `java.lang.String` extends `java.lang.Object`.
- If the inheritance mode "inherited methods" is combined with a pattern matching multiple methods or a blank pattern to select all methods, then the pattern applies to the inheriting classes' methods. For example, a Method Rule with a blank method pattern, selecting all methods, for the class `java.lang.Object`, and the inheritance mode set to **inherited methods** instruments all methods of classes inheriting from `java.lang.Object`, not just the subset of methods inherited from `java.lang.Object`.)
- Use the mode **inherited methods** or **super methods**, so that the methods of the matching class are always included. If the methods of the matching class should be excluded from instrumentation, you must prepend a Method Rule that excludes the methods from the specified class in the **this method** mode.

Arguments: Specify Argument Rules to limit the matching methods to those with the specified argument types. Additionally, you can specify which arguments to capture every time matching methods are called and shown in the PurePath.

Accessor: Deep object access is an extension of Java/.NET method argument or return value capturing. To use deep object access on an argument or the return value of a method, enter an accessor string in the corresponding text field. For arguments, this is the field in the **accessor** column in the method arguments table. For the return value, there is a dedicated text field.

Example

Suppose there are some business transaction methods to be grouped by customer. The methods have signatures like `addToCart(SessionState s, Item newItem)`. Customer information is in a `Customer` object inside the `SessionState` object. Using only method argument capturing without deep object access, you can capture the `SessionState`, but not the `Customer` inside it. Deep object access can define an accessor that specifies how to reach an object deep inside, and capture that object. In this example, a suitable accessor might be `getCustomer().getId()`. If this accessor is defined for the `SessionState s` argument, it is not `s` that is captured, but `s.getCustomer().getId()`.

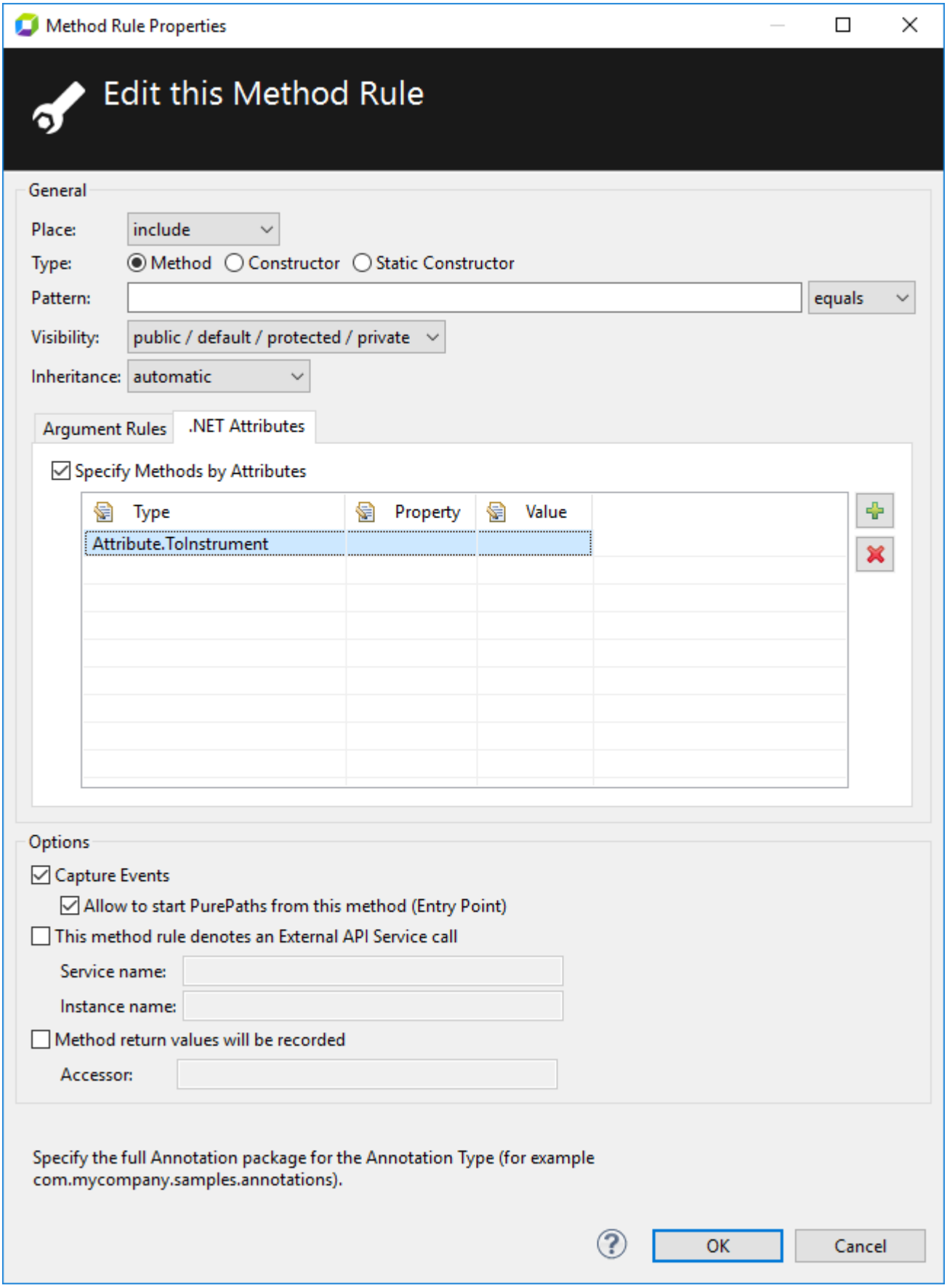
Additional Accessor Details

- You can specify the accessor with or without a leading dot. Both expressions are valid and equivalent.
- The value “accessor failed” returns if the accessor fails because the specified method was not found or the method threw an exception.
- If any object in the chain of access method calls has a value of *null*, the access result is *null*.
- When specifying an accessor, consider potential side-effects, performance impact, and resource consumption that calling the accessor methods may have on your application.

Limitations

- Only one accessor can be defined per argument or return value.
- The methods that the accessor is composed of cannot have any parameters.
- If the argument or return value is an array, deep object access is not possible.
- AppMon does not check whether the specified accessor is a valid method.

Java Annotations/.NET Attributes: Java Annotations or .NET Attributes can also be specified, which results in instrumenting those methods that have these Annotations or Attributes applied. Additionally, you can also specify to only instrument this method if one of the Properties has a certain value. The following example shows a rule that instruments those .NET Methods that have the `ToInstrument` .NET Attribute specified.


Method Rule Properties
Edit this Method Rule
General
Place:
include
Type:
Method
Constructor
Static Constructor
Pattern:
equals
Visibility:
public / default / protected / private
Inheritance:
automatic
Argument Rules
.NET Attributes
Specify Methods by Attributes
Type
Property
Value
Attribute.ToInstrument
Options
Capture Events
Allow to start PurePaths from this method (Entry Point)
This method rule denotes an External API Service call
Service name:
Instance name:
Method return values will be recorded
Accessor:
Specify the full Annotation package for the Annotation Type (for example com.mycompany.samples.annotations).
OK
Cancel

```
namespace TestAttributes
{
    [AttributeUsage(AttributeTargets.Method)]
    class ToInstrumentAttribute : System.Attribute
    {
        public ToInstrumentAttribute()
        {
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            Program p = new Program();
            p.TestMethod1();
            p.TestMethod2();

            Console.ReadLine();
        }

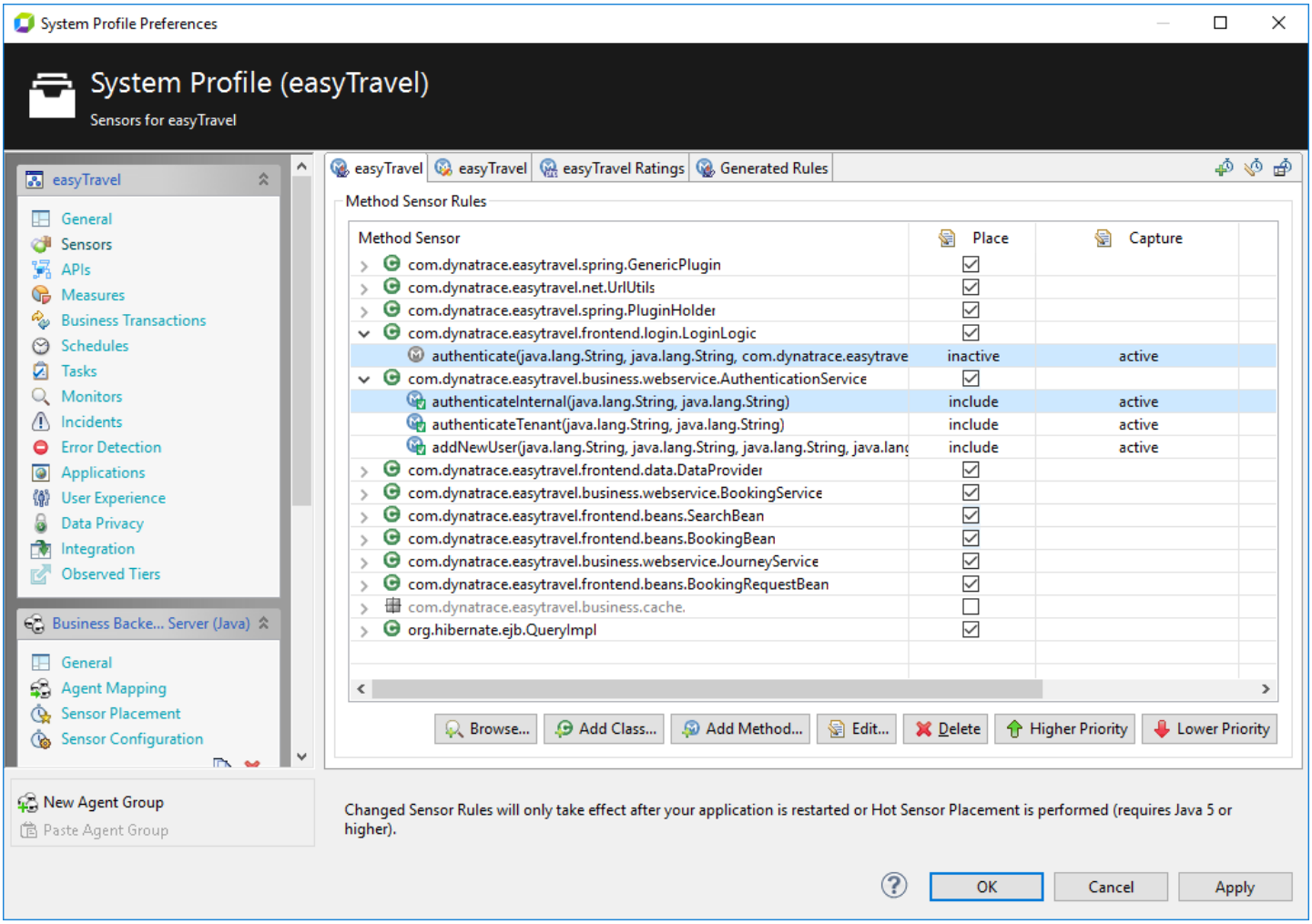
        [ToInstrumentAttribute]
        public void TestMethod1()
        {
            // this method should be instrumented
        }

        public void TestMethod2()
        {
            // this method should not be instrumented
        }
    }
}
```

Settings

Place: The current placement state of a Method Rule can be one of the following:

- **include**: Methods affected by this rule should be instrumented using the given Sensor type.
- **exclude**: Methods affected by this rule should not get instrumented.
- **inactive**: This rule is currently inactive.
- **globalexclude**: Methods affected by this rule should not get instrumented and even include rules in custom Knowledge Sensor Packs are overwritten.



`executePlugins()` methods are instrumented, because the second, lower priority occurrence **in the package** is evaluated by the instrumentation engine and the Rule says Include.

> **Note**
>
> Be careful when use `globalexclude`, because this rule can even override `include` rules defined in custom Knowledge Sensor Packs. The impact of such a rule is not limited to the containing Sensor.

What `globalexclude` *does* and *does not* impact:

Anybody can create Sensor Packs that are installed on the AppMon Server to share it across System Profiles. An example is the **Knowledge Sensor Pack for SharePoint** that you can download on the Community Portal. If you use that Sensor Pack, a global exclusion rule would impact it. However, global exclusion rules do not impact the core functionality of out-of-the-box Sensor Packs.

You can't change out-of-the-box Sensor Packs such as ADO.NET by specifying your own custom exclusion rules. These Sensor Packs instrument certain methods, such as `getConnection` to get a connection string.

Options

Capture Events: Toggle the capturing state of methods instrumented due to this rule. If event capturing is disabled, the Sensors placed due to this rule are disabled. You can change the capturing during runtime.
Allow to start PurePaths from this method (Entry Point): Allows calls to methods that match this rule start a new PurePath, if the call is not part of an already started PurePath.
Method return values will be recorded: Records and shows the return value in the corresponding PurePath every time a matching method is called.

Automatic method sensor rules management

When you create a Method Measure, a method is selected via the class browser. By default, the Method Measure is configured to match exactly on that method. In this case, AppMon manages the Method Sensor Rule automatically throughout the lifetime of the Method Measure. You don't have to configure any Method Sensor Rule.

When a new Method Sensor Rule is created implicitly, Hot Sensor Placement is performed under the following conditions:

- The current user has the permission to perform Hot Sensor Placement.
- The current license is not a *Production Edition* license.
- The Method Measure is a Java method.

When Hot Sensor Placement can't be performed, restart the *Application Process* to make the new Method Sensor Rule effective.

Explicit method sensor rule for custom matching pattern required

When the matching pattern of the Method Rule is configured manually by selecting the **Manually specify matching pattern** check box, no Method Sensor Rule is created for this Method Measure. For the Method Measure to work, you must configure a Method Sensor Rule.

Rule execution order

To create adequate Sensor Groups, it is important to understand the way AppMon evaluates these rules:

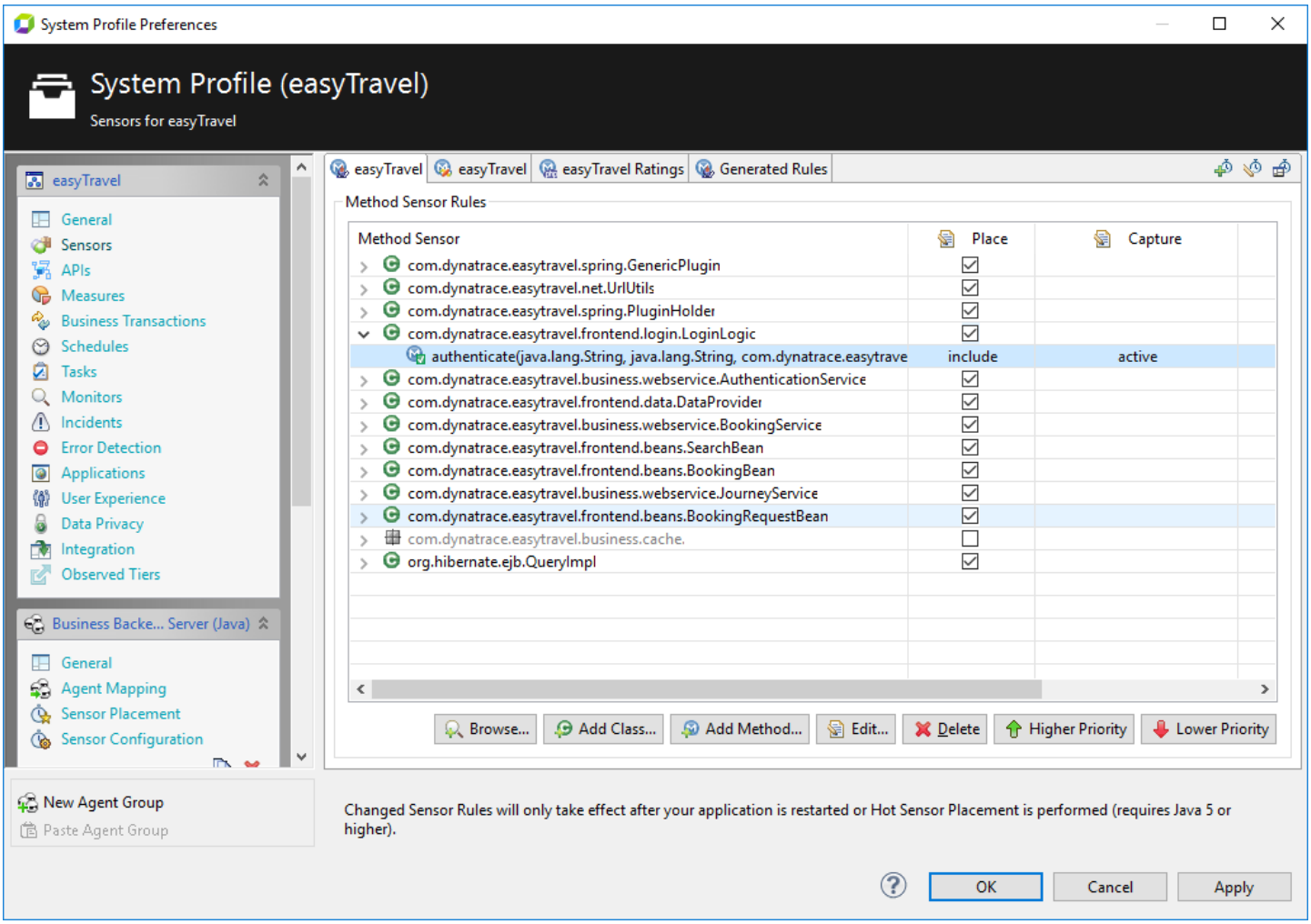
- Sensors are evaluated from the first to the last in their order.
- For every Sensor these steps are applied:
  - All Sensor Rules are evaluated top down.
  - The first matching Sensor Rule is applied, whether it is an inclusion or an exclusion.
  - After the first match in a Sensor, the evaluation for this Sensor is stopped.
- Any global excludes override any includes in all Sensors.

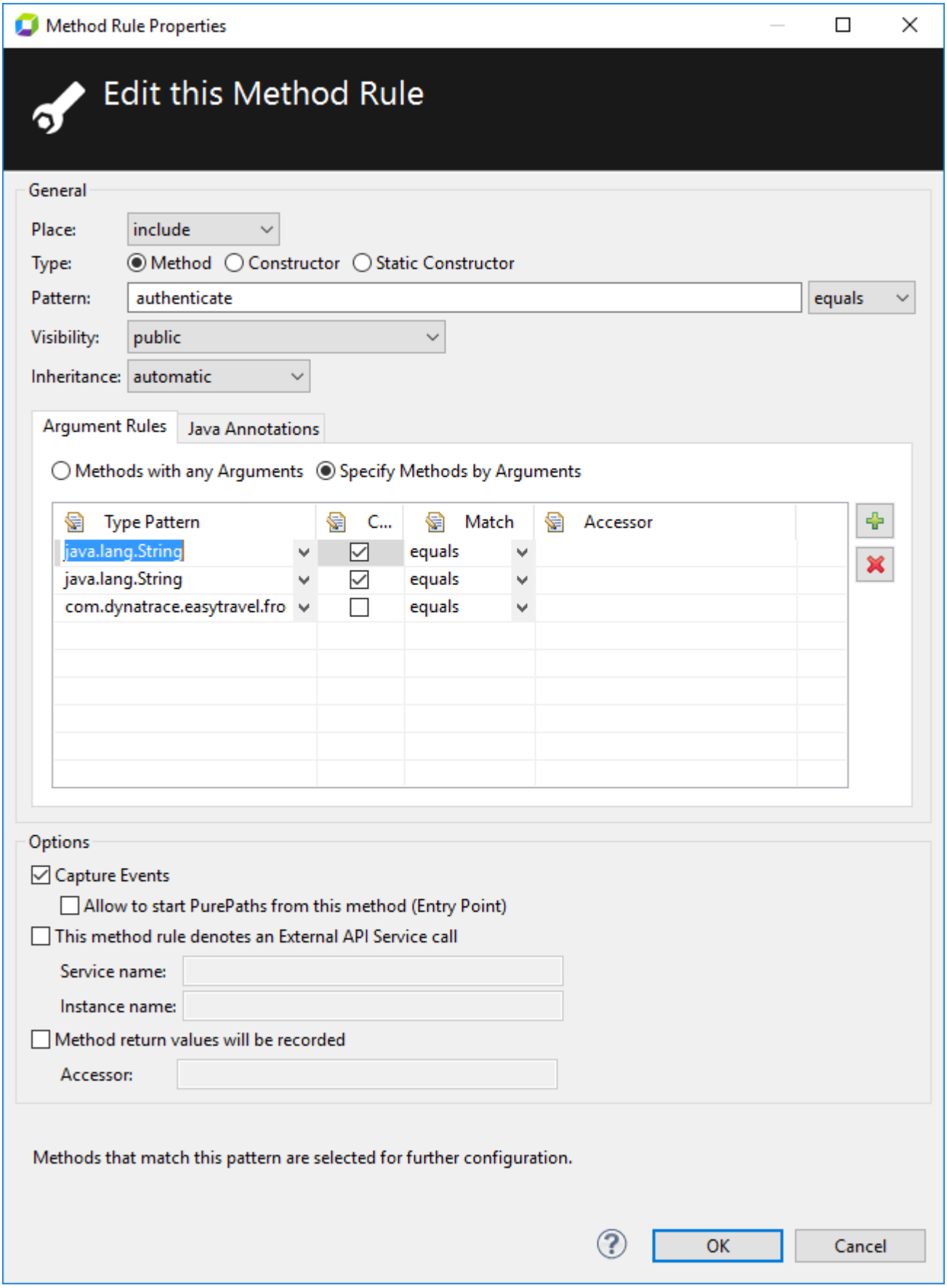
Placement, capture, and argument rules summary

In System Profile <profile_name> > Sensors you decide for which classes or methods Sensors are instrumented. On the next levels you decide for which method and (how many) its arguments data are available.

The following is the **hierarchy for Sensors on the System Profile level**:

- If **Place** is checked for a class, it is instrumented and data **can** be captured.
  - If **Include** is set on the method's Place column, it is instrumented and data **can** be captured.
  - If **Inactive** is set on the method's Place column, the Rule is inactive. If the method appears again lower in priority, then that rule applies. This allows you to test different scenarios.
    - If **Active** is set on the method's Capture column, data is captured for this method. (Active and start PurePaths – additional variant as the name implies).
    - If **Inactive** is set on the method's Capture column, data is not captured for this method, but it's much easier to switch from Inactive to Active than from Not Placed (Not Instrumented) to Placed (Instrumented - byte-code injected). (In Java 1.5+ you alternatively can do Hot Sensor Placement, but that´s much harder on the system than Activate.)
      - If you choose **Method with any Argument** in the **Method Rule** dialog box, data for this method – with any signature – is captured.
      - If you choose **Specify Method by Arguments** in the **Method Rule** dialog box, data for selected arguments of this method – with exactly that signature – is captured.
        - If Capture is checked for an argument, data of this argument is captured.

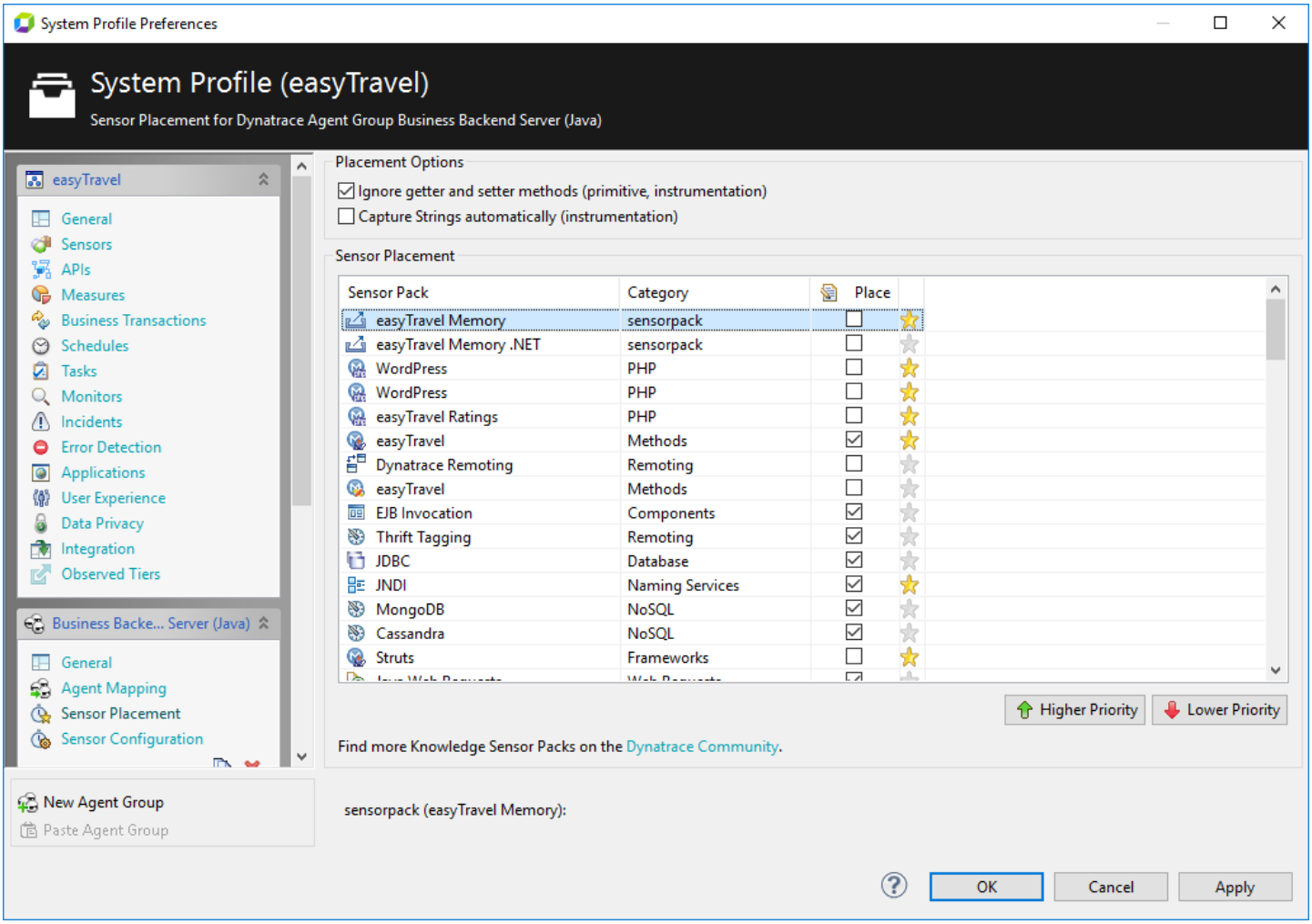


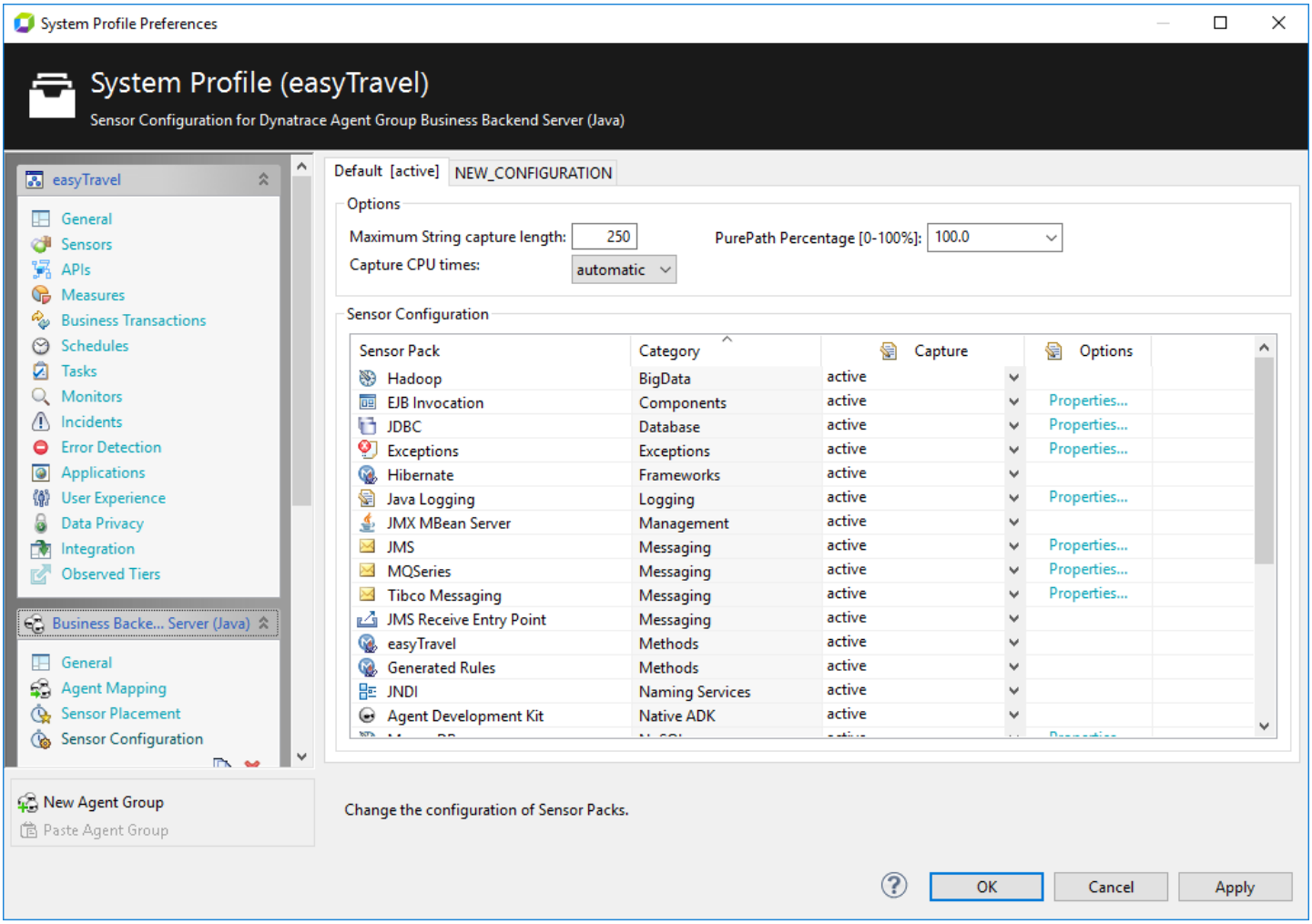


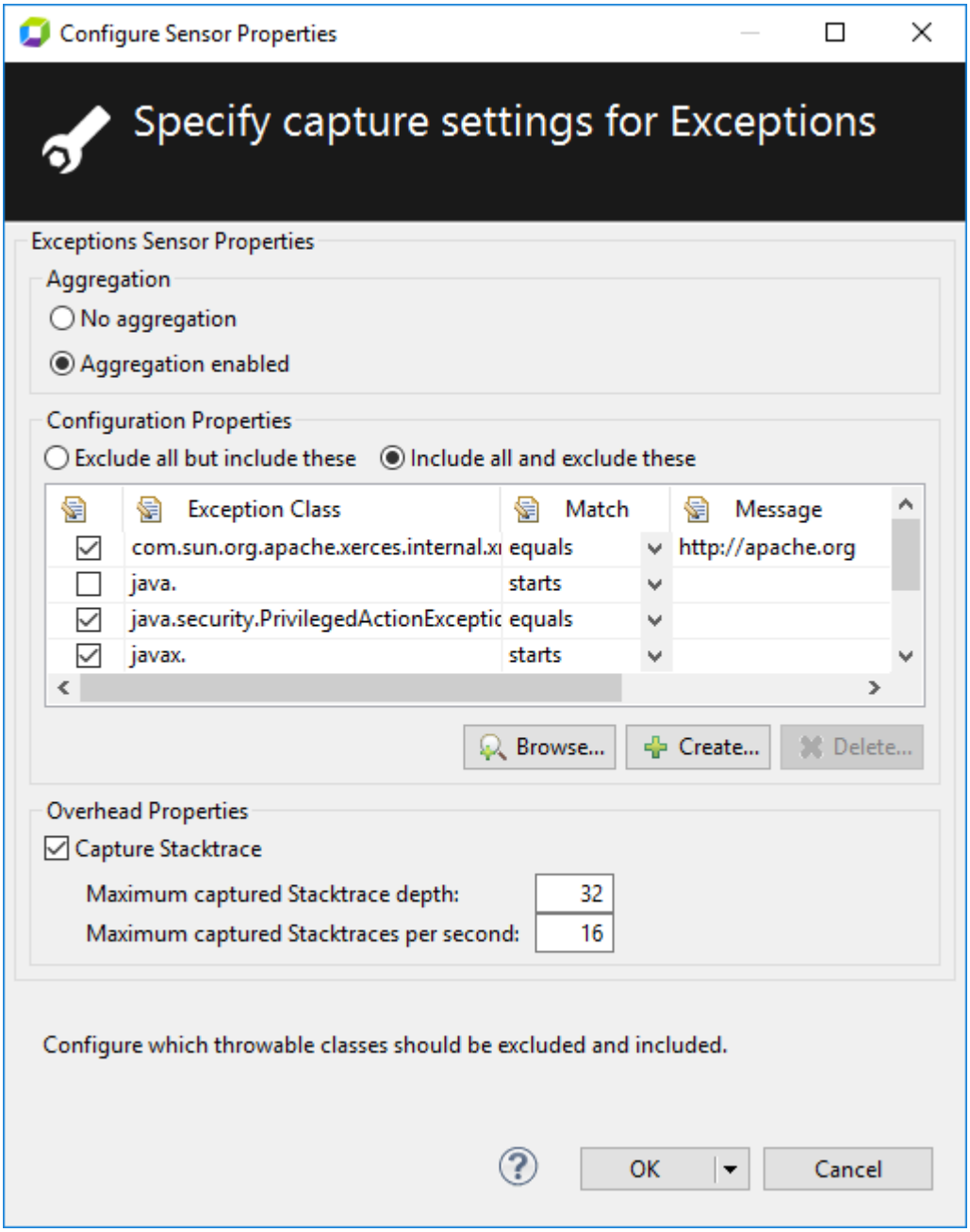
In System Profile <profile_name> > Agent Group <name> you decide which Sensor Packs are instrumented. On the next levels you decide for which method, arguments data will be captured.

The following is the hierarchy for Sensor Placement and Sensor Configuration on the Agent Group (Tier) level:

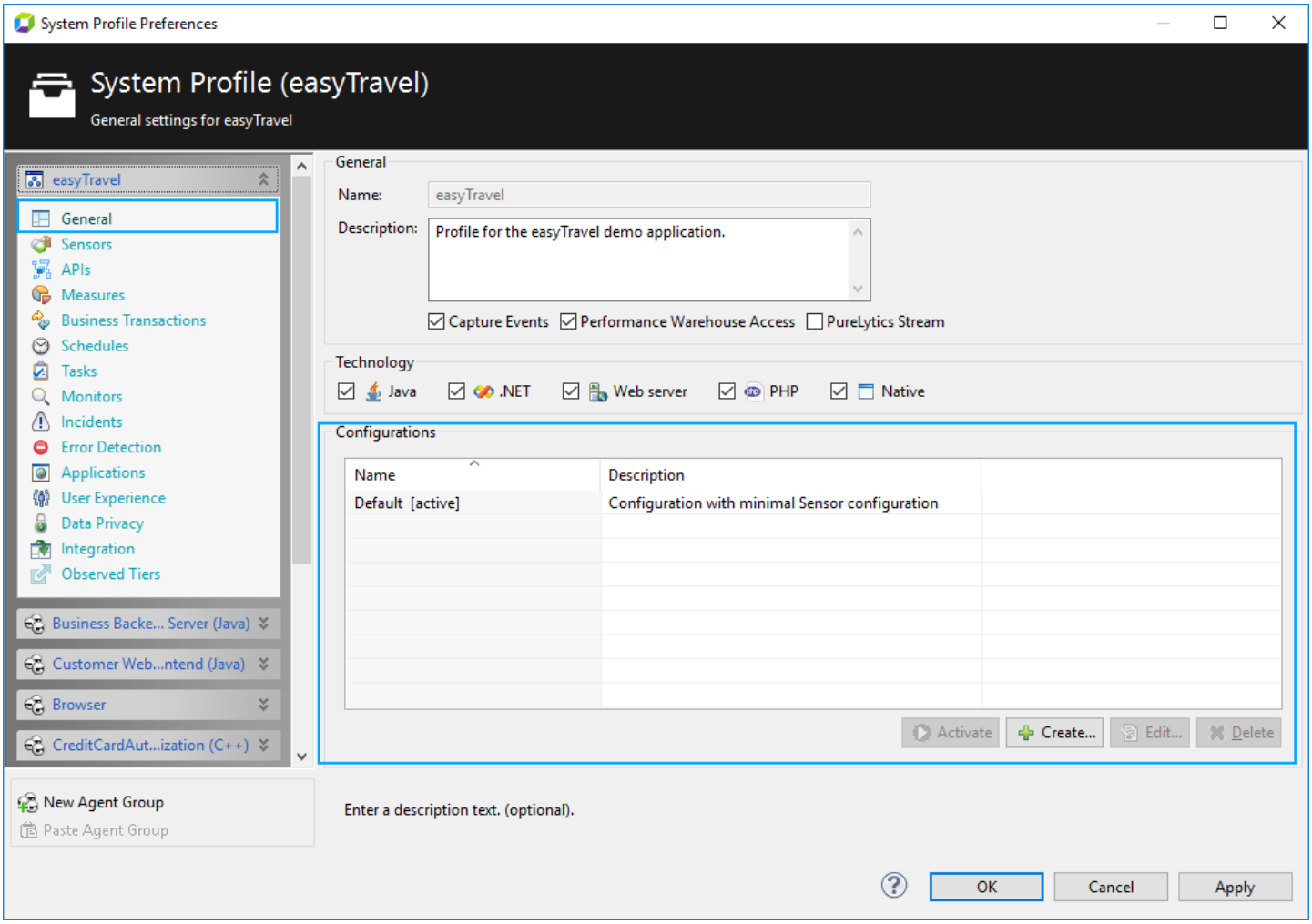
- If **Place** is checked for a Sensor Pack in the Sensor Placement table it is instrumented and data can be captured. A golden star signifies that it is Hot Placeable.
  - If **Active** is set for a Sensor Pack's Capture column, data is captured for this Pack. (Active and start PurePaths – additional variant as the name implies).
  - If **Inactive** is set for a Sensor Pack's Capture column, data is not captured for this Pack, but it is easier to switch from Inactive to Active, than from Not Placed (Not Instrumented) to Placed (Instrumented - byte-code injected). (In Java 1.5+ you can do Hot Sensor Placement, but that's much harder on the system than Activate.)
    - In the Options column you can set Properties for many Packs to better control which data is captured.

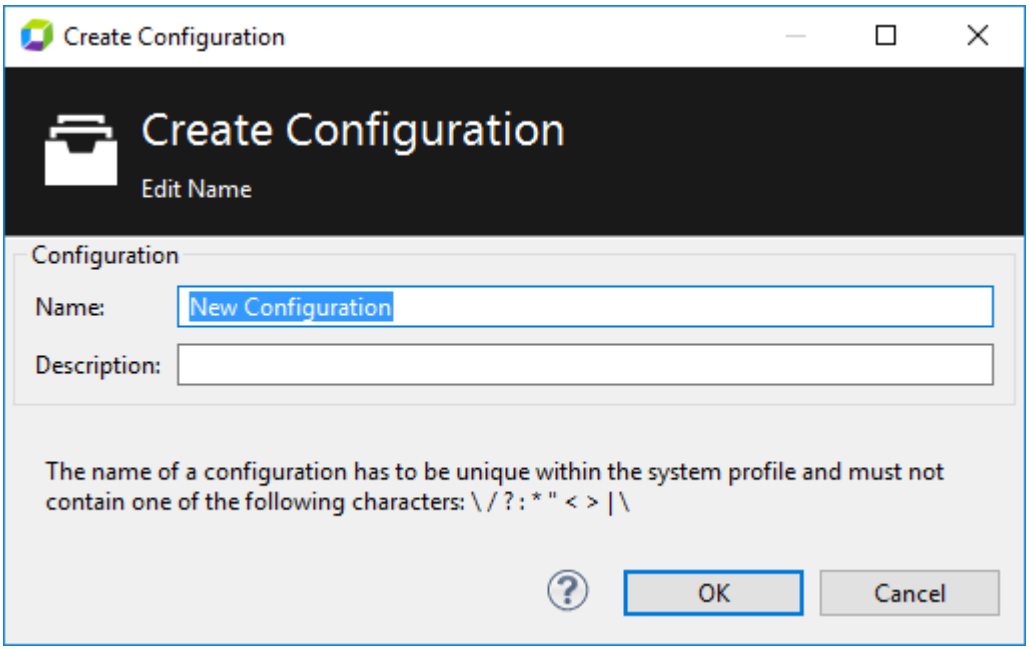


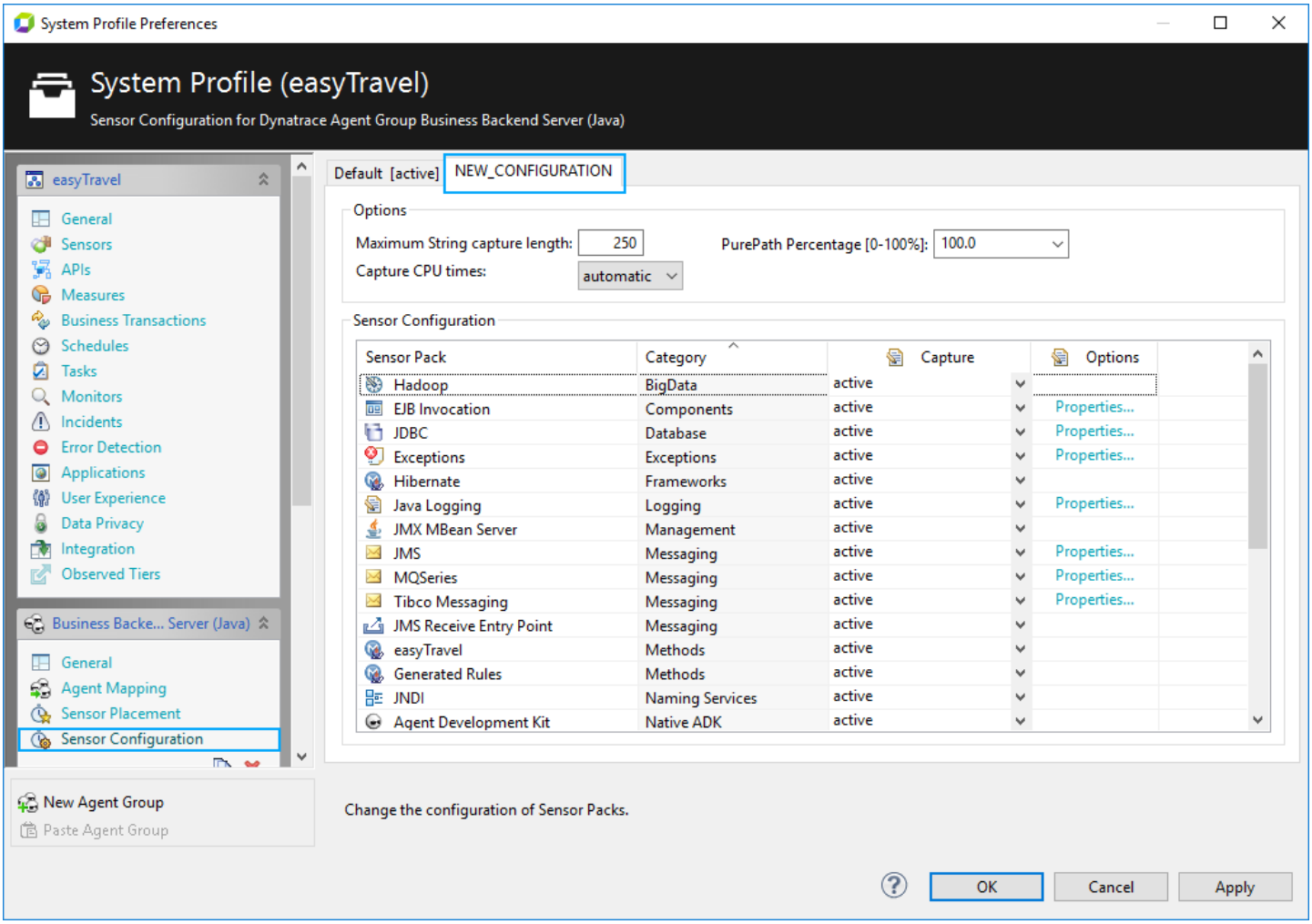

System Profile Preferences
System Profile (easyTravel)
Sensor Configuration for Dynatrace Agent Group Business Backend Server (Java)
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Data Privacy
Integration
Observed Tiers
Business Backe... Server (Java)
General
Agent Mapping
Sensor Placement
Sensor Configuration
New Agent Group
Paste Agent Group
Default [active]
NEW_CONFIGURATION
Options
Maximum String capture length: 250
PurePath Percentage [0-100%]: 100.0
Capture CPU times: automatic
Sensor Configuration
Sensor Pack
Category
Capture
Options
Hadoop BigData active
EJB Invocation Components active Properties...
JDBC Database active Properties...
Exceptions Exceptions active Properties...
Hibernate Frameworks active
Java Logging Logging active Properties...
JMX MBean Server Management active
JMS Messaging active Properties...
MQSeries Messaging active Properties...
Tibco Messaging Messaging active Properties...
JMS Receive Entry Point Messaging active
easyTravel Methods active
Generated Rules Methods active
JNDI Naming Services active
Agent Development Kit Native ADK active
Change the configuration of Sensor Packs.
OK
Cancel
Apply



There can be more than one System Profile Configuration. A tab is created for each Sensor Configuration item in each Agent Group (Tier). See the screenshots below.


System Profile Preferences
System Profile (easyTravel)
General settings for easyTravel
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Data Privacy
Integration
Observed Tiers
Business Backe... Server (Java)
Customer Web...ntend (Java)
Browser
CreditCardAut...ization (C++)
New Agent Group
Paste Agent Group
General
Name:
easyTravel
Description:
Profile for the easyTravel demo application.
Capture Events
Performance Warehouse Access
PureLytics Stream
Technology
Java
.NET
Web server
PHP
Native
Configurations
Name
Description
Default [active]
Configuration with minimal Sensor configuration
Activate
Create...
Edit...
Delete
Enter a description text. (optional).
OK
Cancel
Apply


Create Configuration
Create Configuration
Edit Name
Configuration
Name:
New Configuration
Description:
The name of a configuration has to be unique within the system profile and must not contain one of the following characters: \ / ? : * " < > | \
OK
Cancel



Memory sensor rules

Use AppMon to monitor instantiations of specific classes and deserialization events of instances of specific classes. The classes that are included in Memory Monitoring are selected by Memory Sensor Rules. Memory Sensor Rules are configured similar to Class Rules. The Select Memory dashlet displays the Memory Monitoring data of the selected classes. See Class Rules for more information.

Class Browser

Use the class browser to define Memory Sensor Rules in the same way as Class Rules.

Memory sensor rules

You can specify Memory Sensor Rules for a class or a package/namespace similar to Class Rules, with patterns and Java Annotations as described in Method Sensor Rules. You can active or deactivate the Sensors associated to a Memory Sensor Rule at runtime by toggling the **Activated** check box of the Memory Rule. If you enable the **Include Inherited Classes** check box, memory monitoring Sensors are placed in classes that match the selection pattern of the rule and in all direct or indirect base classes of that class.

Class Browser

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

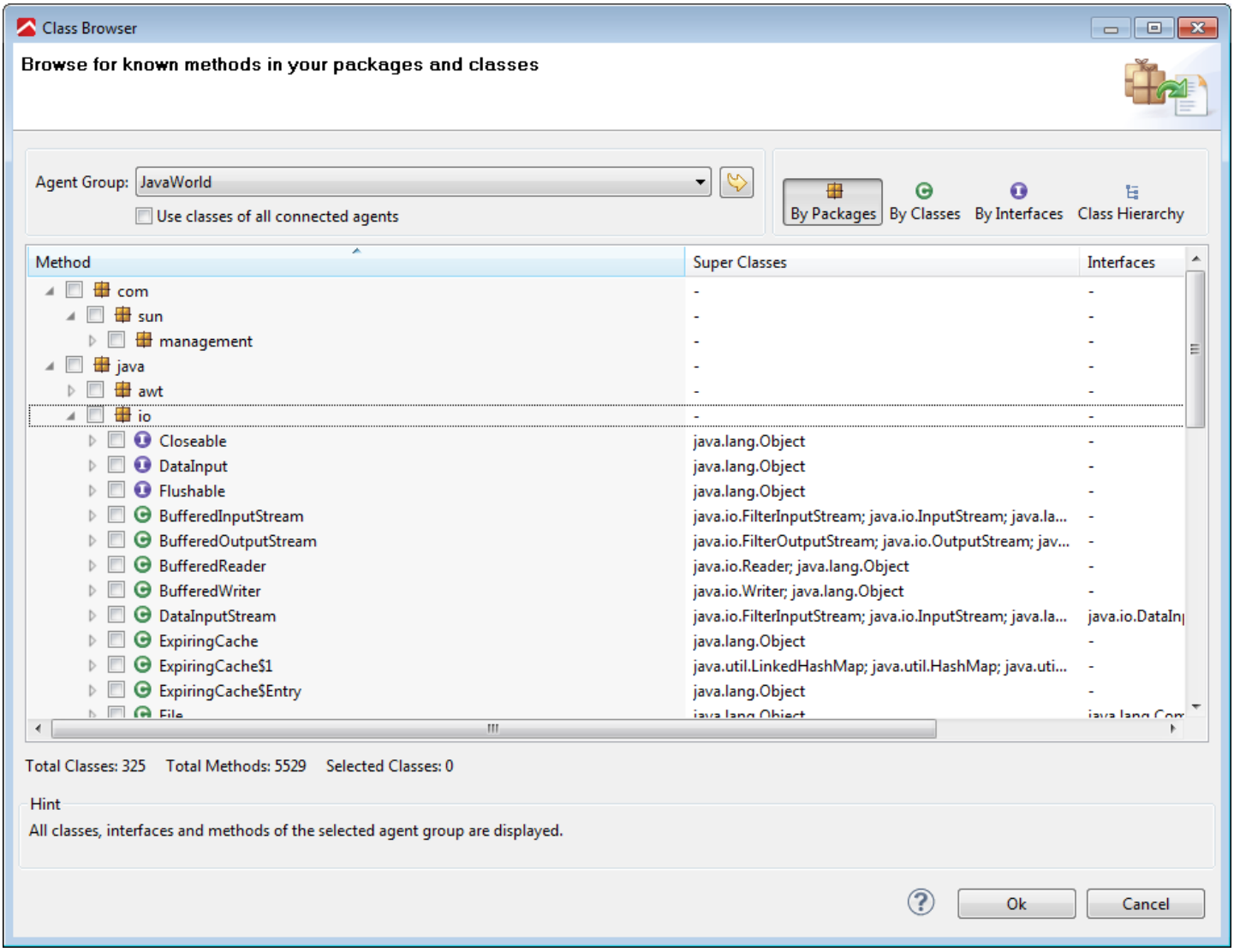
Grouping options

Rule generation

The **Class Browser** provides a convenient way to define class rules and method rules. A tree view shows the package/namespace structure of the classes loaded by the SUDs of a certain Agent Group. If more than one AppMon Collector is connected to the AppMon Server, the **Class Browser** tries to use class data delivered from the one AppMon Collector that appears to have the most current data.

This behavior is sufficient in most cases but there may be situations where the wrong AppMon Collector is used for the user's purposes. In such cases, you can request all data from all connected AppMon Collectors. To do this, select the **use classes of all connected agents** option and refresh the view. The tree view supports the default searching and filtering mechanism. Click in the pane and press **^F**.

> **Important**
>
> When you enable the **use classes of all connected agents** option, a refresh may take several minutes as the data is requested from all connected AppMon Collectors.



Use the Class Browser to see your packages and classes.

Grouping options

Grouping Option	Description
By Packages	Group all classes hierarchically by package/namespace.
By Classes	Flat view: display a simple class list with interfaces sorted on top of the list.
By Interfaces	Group all classes by the implemented interfaces. Classes implementing no interfaces are sorted to the bottom of the list.
Class Hierarchy	Inheritance hierarchy with interfaces sorted on top of the list.

Rule generation

Press **OK** to generate appropriate Sensor Rules. These depend on the selected tree node.

Selected Node	Generated Rule
Package/Namespace	Class rule (class pattern defines package/namespace)
Class	Class rule (class pattern defines single class
Method	Method rule

Depending on the context of the **Class Browser**, you can only select one single tree node, which results in exactly one single generated rule. For example, when you enter the **Class Browser** from the **Exception Measure** configuration dialog box. The **Class Browser** tracks the amount of currently selected classes. It also tracks the selection of a set of well known root packages. Note the **Selected Classes** counter in the **Class Browser** dialog box. If the amount of rules to be generated exceeds a certain level, a warning appears when you click **OK**. The generation of such large amounts of rules is not prohibited though.

Hot Sensor Placement

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Supported JVM versions

Agents overview

Applying Hot Sensor Placement

Related pages

System Profile

Agent Group - Agent Mapping

Configuration change immediacy matrix

Hot Sensor Placement is a technology used to commit changes to an application's instrumentation during run time without the need to restart the application or application server.

Because of technical limitations, Hot Sensor Placement is currently only available for Java Virtual Machines.

Changing a sensor's System Profile configuration impacts the application's instrumentation, for example placing a sensor while AppMon Agents are connected to the System Profile. The change is not applied immediately. To avoid excessive application restarts, use the Hot Sensor Placement feature, which changes the instrumentation while the application is running.

Sensors that support Hot Sensor Placement are marked with a

in the Sensor Placement Section dialog box. If a placed sensor that does not support Hot Sensor Placement (

) changes, an application restart is required.

Supported JVM versions

Hot Sensor Placement is available for all AppMon Agents running under J2SE 5.0 or later. It also applies an Agent Group's Sensor Placement, defined in the activated Configuration, on classes loaded prior to the Configuration change.

JVM support for Hot Sensor Placement

JVM Version	Hot Sensor Placement Supported	Hot Sensor Placement Enabled	Comments
J2SE 1.4.2	no	no	
Sun(Oracle) J2SE 5.0 or newer	yes	yes	
Sun(Oracle) 1.6.0_14 / 1.6.0_15	no	no	Both versions contain bugs that prohibit the use of Hot Sensor Placement.
IBM J2SE 5.0 or newer	yes	no	Disabled by default. You can enable it manually in the Agent Mapping Advanced Settings.

If a JVM does not support Hot Sensor Placement, the AppMon Agent log indicates that the feature is disabled.

```
... warning [native] The capability can_redefine_classes was disabled because of a bug ir
... warning [native] Capability can_redefine_classes not available. Hot Sensor placement
...
... info    [native] Hot Sensor Placement ........ not available
```

Agents overview

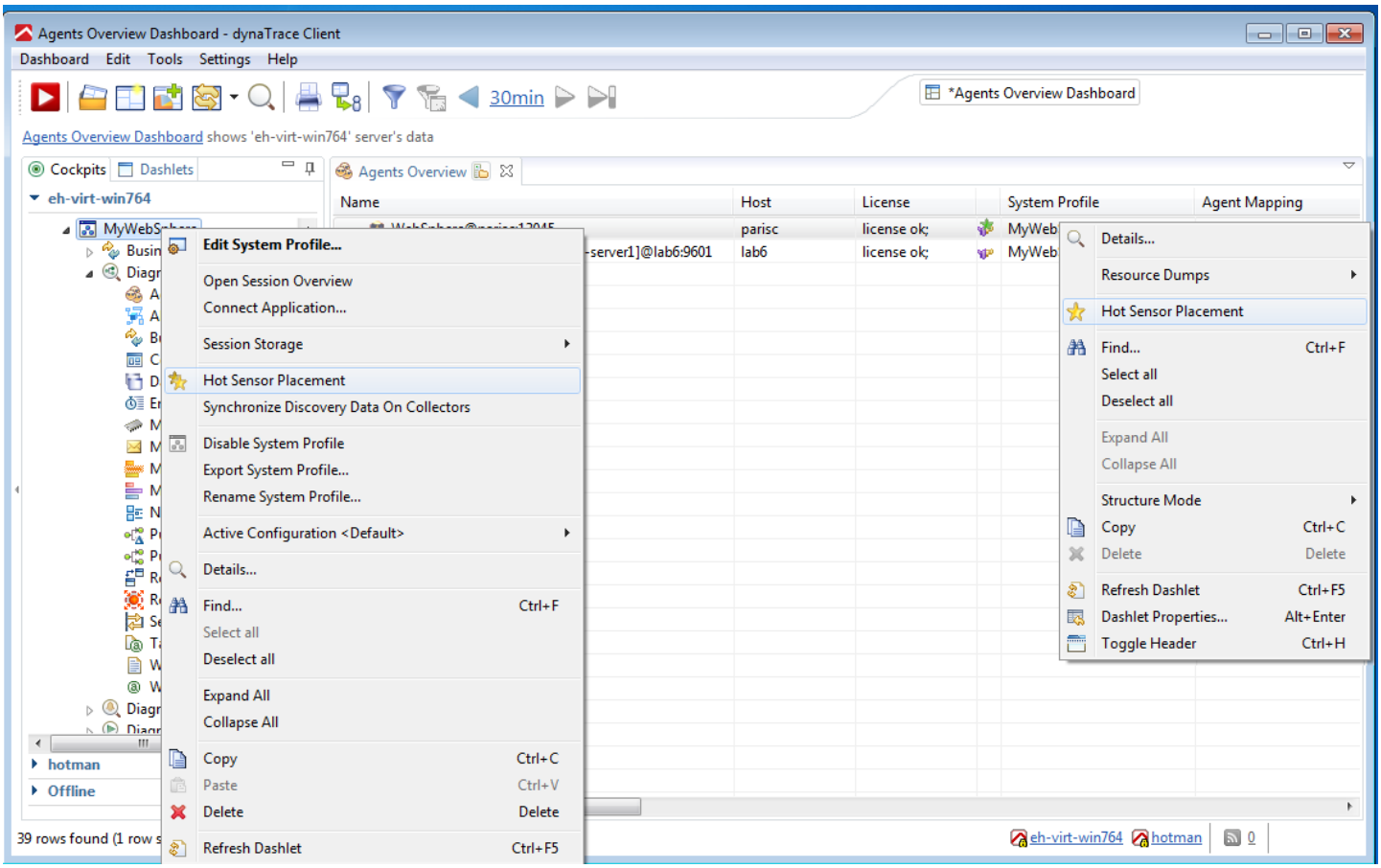
Agents that support Hot Sensor Placement are marked with a badge

in the Agents Overview.



Hot Sensor Placement supported/not supported

Applying Hot Sensor Placement



Hot Sensor Placement menu

Use the System Profile context menu to apply Hot Sensor Placement to all AppMon Agents of a System Profile at once. To apply it to a single AppMon Agent, use the context menu on the Agent in the Agent Overview.

Hot Sensor Placement executes on the method level. All methods on the SUD continue to execute the original code, even during the Hot Sensor Placement, until the Hot Sensor Placement's completion. Methods that start after the Hot Sensor Placement run the new code with the configuration changes applied.

Note

See Configuration Change Immediacy Matrix for more information on available configuration changes and their effects.

Configuration change immediacy matrix

Sensor Pack	Yes	No
EJB Invocation	✓	
Servlets	✓	
Java Web Services	✓	
JNDI	✓	
HTTP URL Connection Tagging	✓	
Hibernate	✓	
JTA	✓	
XA	✓	
JMS	✓	
JMS Receive Entry Point	✓	
Tibco Messaging	✓	
AWT	✓	
Struts	✓	
Exceptions	✓	

Java JDK Logging and Log4J	✓
RMI Tagging	✓
Remote Method Invocation (RMI)	✓
JDBC	✓
JMS Tagging	✓
Tibco Message Tagging	✓
JMX MBean Server	✓
Thread Start Tagging	✓
Executor Tagging	✓
Collection	✓
Map	✓
ASP.NET	✓
ADO.NET	✓
.NET Components	✓
.NET Web Services	✓

.NET Web Service Tagging	✓
.NET Remoting	✓
.NET Remoting Tagging	✓
MSMQ	✓
MSMQ Receive Entry Point	✓
.NET WPF	✓
.NET Windows Forms	✓
.NET WCF	✓
.NET WCF Tagging	✓
.NET Collection	✓
.NET Exception	✓
.NET Logging	✓

Note

Hot Sensor Placement is only available for applications running with J2SE 5.0 or later.

The following configuration table shows which Sensor Packs apply instrumentation changes to the application when a Hot Sensor Placement is performed.

Hot Sensor Placement

Configuration Scope	Change	Immediate	Hot Sensor Placement	Application Restart
System Profile: General	Capture Events	✓		
System Profile: General	Performance Warehouse Access	✓		
System Profile: Sensors	Sensor Rules: capture setting	✓		
System Profile: APIs	API Rules	✓		
System Profile: Business Transactions	Business Transaction Configuration	✓		
System Profile: Tasks	Task Configuration	✓		
System Profile: Incidents	Incident Configuration	✓		
System Profile: Monitors	Monitor Configuration	✓		
Agent Group: Agent Mapping	Capture Events	✓		
Agent Group: Agent Mapping	Advanced: Override local log settings	✓		
Agent Group: Sensor Configuration	Sensor Pack Capture Mode	✓		
Agent Group: Sensor Configuration	PurePath Percentage	✓		
Agent Group: Sensor Configuration	Max. String capture length	✓		
Agent Group: Sensor Configuration	Synchronization threshold	✓		

Agent Group: Sensor Configuration	Sensor Property Configuration	✓		
Agent Group: Sensor Configuration	Active Configuration	✓		
Agent Group: Sensor Configuration	Enable CPU time capturing	✓		
Agent Group: Measures	Measure Subscriptions	✓		
Agent Group: Measures	Thresholds Configuration	✓		
System Profile: Sensors	Method Sensors J2SE 5.0+: other settings		✓	
Agent Group: Sensor Placement	Capture Strings automatically J2SE 5.0+		✓	
Agent Group: Sensor Placement	Ignore getter and setter methods J2SE 5.0+		✓	
Agent Group: Sensor Placement	Place Sensor J2SE 5.0+		✓	
System Profile: Sensors	Method Sensor Rules J2SE 1.4			✓
System Profile: Sensors	Memory Sensor Rules			✓
System Profile: Sensors	Sensor Rules .NET			✓
Agent Group: Agent Mapping	Advanced: Communication Mode			✓
Agent Group: Agent Mapping	Advanced: Buffer settings			✓
Agent Group: Sensor Placement	Place Sensor J2SE 1.4			✓

Agent Group: Sensor Placement	Enable synchronization diagnostics	✓
Agent Group: Sensor Placement	Capture Strings automatically J2SE 1.4	✓
Agent Group: Sensor Placement	Ignore getter and setter methods J2SE 1.4	✓
Agent Group: Sensor Placement	Capture Strings automatically .NET	✓
Agent Group: Sensor Placement	Ignore getter and setter methods .NET	✓
.NET Agent Configuration Tool	Change .NET Agent settings	✓

- Immediate = dialog box is OK.
- Hot Sensor Placement item is clicked in the context menu (for example in the Agents Overview dashlet).
- Application restart.

The following table shows the effort required to make configuration changes:

Client administration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Settings

Services

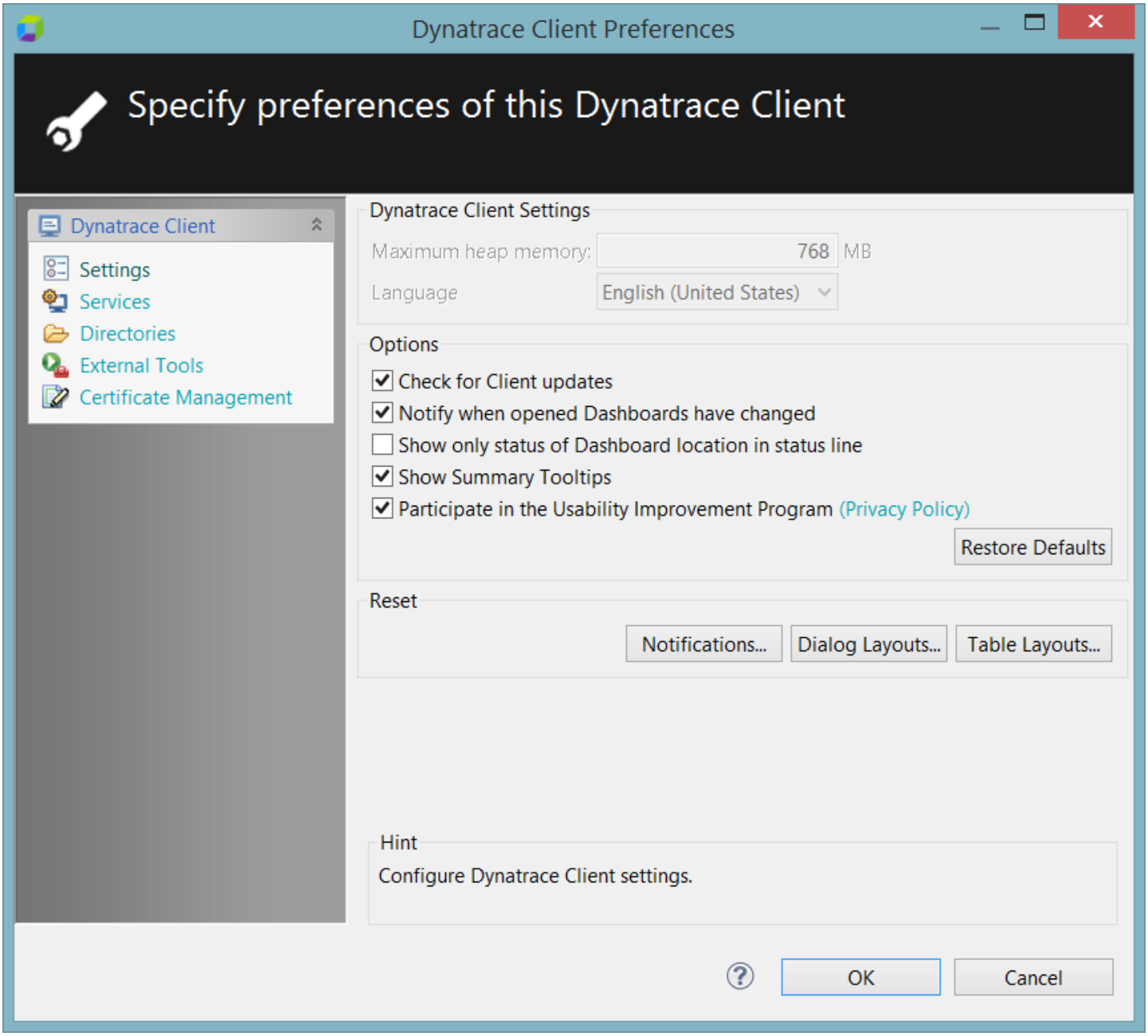
Directories

External Tools

Certificate Management

Use the **Dynatrace Client Preferences** dialog box to configure the UI update interval, display options, IDE integration, proxy, and the AppMon Client directories.

To access the dialog box, select **Settings > Dynatrace Client** in the AppMon Client.



AppMon Client Preferences

Settings

The **Settings** page includes the following configuration options:

- **Maximum heap memory**: Specify the maximum amount of memory the AppMon Client uses. Restart the AppMon Client for settings changes to take effect. The maximum heap memory for an AppMon Webstart Client cannot be adjusted because of security restrictions for the Oracle Java runtime environment.

- **Language**: Select the language for the AppMon Client.
- **Check for Client Updates**: Selecting this check box lets the Client check whether AppMon Client updates are available on the AppMon Server, and updates the Client if updates are available. You must restart the AppMon Client for Plugin updates to take effect.
- **Notify when opened Dashboards have changed**: Display a message in the AppMon Client when an opened dashboard is changed on the server side.
- **Show only status of Dashboard location in status line**: Display the status of the source of the active dashboard. If the active dashboard is offline, the status of all connected AppMon Servers is displayed.
- **Show Summary Tooltips**: Hold the mouse pointer over data to display informational tooltips.
- **Participate in the Usability Improvement Program**: Enable this option to share certain information to help Dynatrace improve the quality of the AppMon Client. Click the Privacy Policy link in the dialog box for details of the Usability Improvement Program.
- **Restore Defaults**: Discard all current modifications to the layout of all AppMon Client windows and restore the original layout.
- **Warning Dialogs**: Restore all Client generated notifications that may be suppressed by selecting **Don't show this message again** in the message window.
- **Dialog Layouts**: Reset the layout and size of all Client windows to the defaults.
- **Table Layouts**: Reset all table layouts to the defaults.

Services

The **Services** page contains configuration options for services and connections:

- **Enable REST Web Services**: When you enable this option, specify the SSL listening port.
- **Allow non-SSL connections**: When you enable this option, specify the non-SSL listening port.
- **Manually configure proxy**: If communication between the AppMon Server and AppMon Client uses an HTTP proxy server, select this check box then specify the **Proxy host** name and port, and the **Authentication** type.

Directories

Use the settings in the **Directories** page to specify the locations of configurable AppMon Client directories:

- Log
- Reports

- Offline Sessions
- Dashboards
- Temp

The default directories for logging, reports, and offline sessions are created during installation. If you change these settings, it does not create the specified directories. The directories must already exist, or the changed setting is rejected.

In this screen, you can also define the **Base path** for your Plugin development tool. See IDE Integration and the relevant subtopics for more information on Plugins.

External Tools

The AppMon Client supports external tools like .NET Reflector and JAD. If you set the paths to a decompiler application, the AppMon Client uses it for decompilation rather than the shipped decompilers.

On the **External Tools** page, you can also set the path for your preferred text editor and Java decompiler.

Certificate Management

Shows private keys and certificates locally stored in the client's keystore. See Certificate Management for more information.

AppMon Web administration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Port configuration

User account configuration

Secure certificate setup

Data display

World map country drilldown

Related pages

Web dashboards

server.config.xml

User permissions and authentication

Certificate management

Configure geographic location

AppMon Web is an HTML5-powered web application that is accessible from the following default secure (HTTPS) URL:

```
https://<servername>:9911
```

AppMon Web is built for viewing on **tablet devices** and **desktop machines** that have a minimum browser resolution of 1024 x 768 pixels (e.g., iPad mini physical resolution 2048 x 1536 vs. browser resolution 1024 x 768). More details for your device can be found here.

AppMon Web is supported and tested with the following browsers:

- Edge 14+
- Firefox 50+
- Chrome 55+
- Safari 10+

Drilldowns to the **AppMon Client** require a running AppMon Client. See the AppMon Client requirements for more details.

Port configuration

You can modify the default AppMon Web port (`9911`) by setting the **webuihttpsport** attribute in the **server.config.xml** file. This file is located in the **/server/conf** folder on the AppMon Server machine. To modify the default AppMon Web port, the AppMon Server must be stopped and restarted.

```
<serverconfig>
        <webui webuihttpsport="9912" webuienabled="true" />
</serverconfig>
```

Learn more about the commonly adjusted settings in server.config.xml.

User account configuration

To access and view AppMon Web, an AppMon user account that is assigned any role is required. See User Permissions and Authentication for more information.

Secure certificate setup

When attempting to log into AppMon Web, your browser may display a security warning. This is because your browser doesn't recognize the AppMon Web security certificate. If this message appears, you have several options depending on your browser type:

1. For all browsers, you can dismiss the security warning message and proceed to log into AppMon Web.
2. For Firefox, Internet Explorer and Safari you can add the Dynatrace certificate to the browser's list of trusted certificates.

3. For Chrome, you can ask your system administrator to install the AppMon Web security certificate in your system's keystore. See Certificate Management and sub-pages for more information.

Data display

Only *live session* data and measure data in the Performance Warehouse appears on web dashboards. *Stored session* data, which is a memory dump, thread dump, sampling, or a former PurePath session that is stored on a hard drive, is not available for display on web dashboards. This data is available for display in AppMon Client dashboards. Learn more about session data.

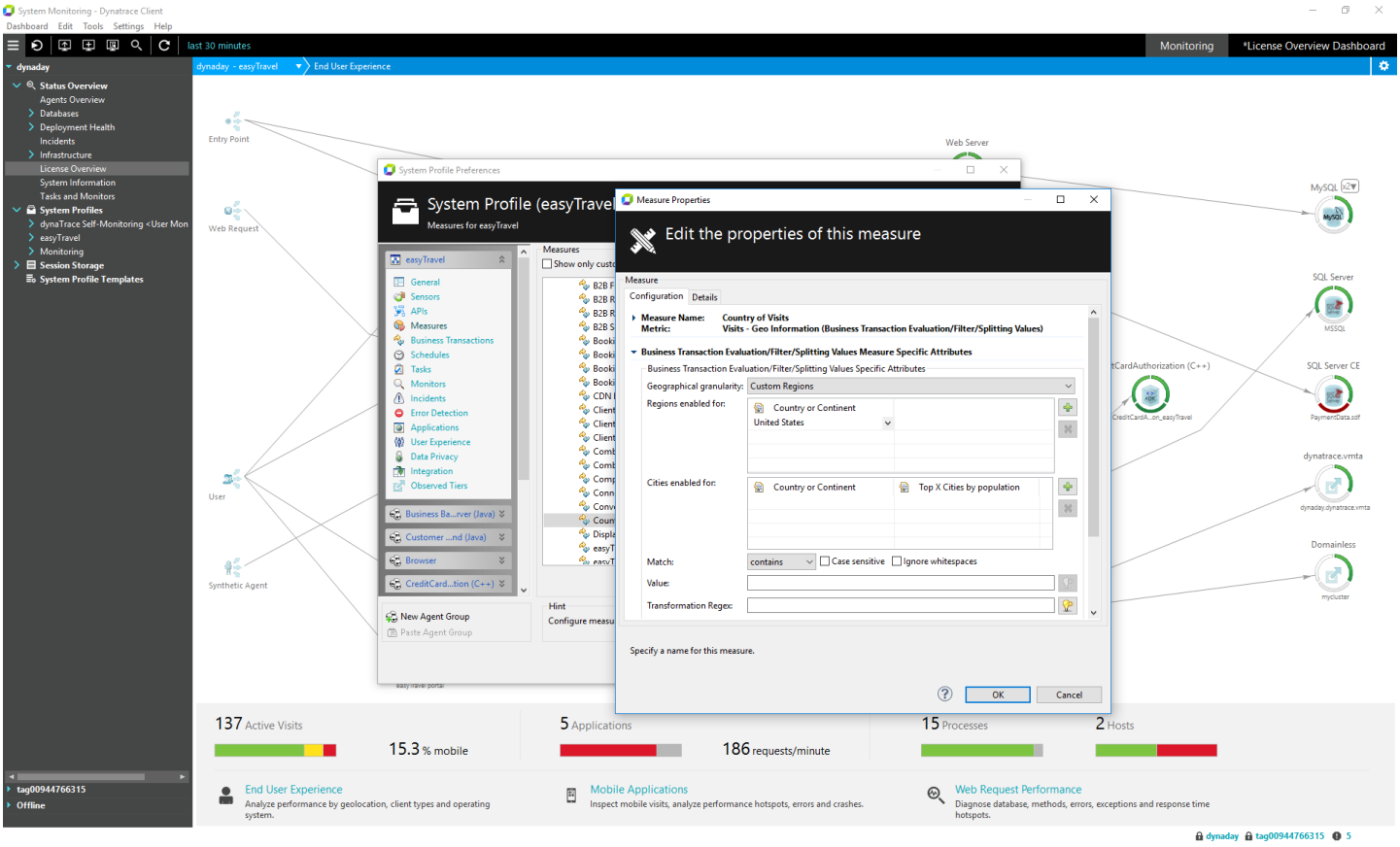
World map country drilldown

The drilldown to specific states and provinces is supported for the following countries provided the **Country of Visits** business transaction splitting value is set up for each system profile:

- Austria
- Australia
- Brazil
- Canada
- China
- India
- Russia
- United States

To set up the **Country of Visits** business transaction splitting value:

1. Access the **System Profile Preferences** dialog box.
2. Choose **Measures**.
3. Expand **Business Transaction Evaluation/Filter/Splitting Values**.
4. Double-click **Country of Visits**, and then add any of the support countries listed above.



See Configure Geographic Location for more information.

User permissions and authentication

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Accounts

Groups

Roles

LDAP

Password security

Related pages

Permissions mapping

Administrators can configure the AppMon built-in security system to protect AppMon installations against unauthorized access or unintentional usage. AppMon uses user accounts, groups, and roles to assign permissions that control access to the product and certain features. Permissions are assigned to roles. Each user group has a role, and thus permissions from this role. To get the permissions the user has to be a part of the group.

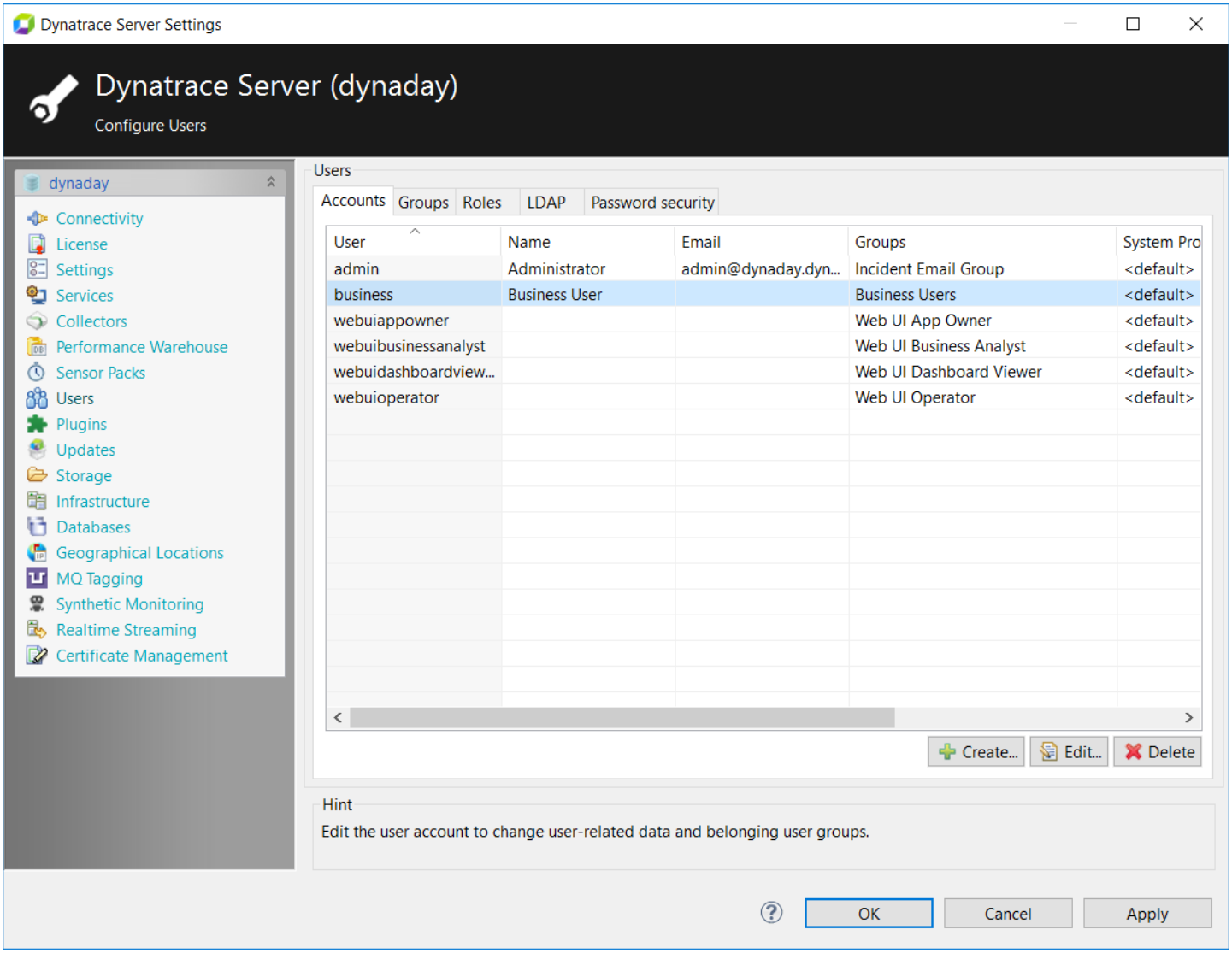
The AppMon security system involves three security areas:

- **AppMon Server Management** for server related permissions.
- **System Profiles** for managing permissions on the level of specific System Profiles.
- **Dashboards** for defining read/write access to shared dashboards.

To manage user accounts through the AppMon Client, click **Settings > Dynatrace Server > Users**.

Accounts

Use the **Accounts** tab to create and edit user accounts.



Set up the account parameters in the **User Preferences** dialog box. To open it, click **Create** to create a new account, or double click an existing account to edit it.

Each user account needs a unique **user ID** and a **password**. The **Full Name** and **Email** fields are optional.

In the **Select User Groups** list, assign the required user groups to the account. If no user group is selected, the user has no permissions. For more information, see the Groups section.

The **System Profiles** and **Dashboards** columns of the **Accounts** tab display the System Profiles and dashboards to which the user has access. If LDAP is used for user authentication, there is no need to create user accounts manually. The AppMon security system creates a local account when a user first logs in with an LDAP account. User accounts have the **Use LDAP Authentication** option set by default. The **Full Name** and **Email** are kept in sync with LDAP. The AppMon security system assigns membership to LDAP groups automatically, according to the LDAP configuration. These groups are dimmed in the **User Preferences** dialog box and you cannot map them manually. However, you can add LDAP accounts to local user groups. For non-LDAP users added to AppMon, there is no difference between LDAP groups and locally defined user groups.

Dynatrace Community credentials

To get the latest Updates, SpeedOfTheWeb data and news within the AppMon Client, you need to enter your Dynatrace community credentials.

System admin account

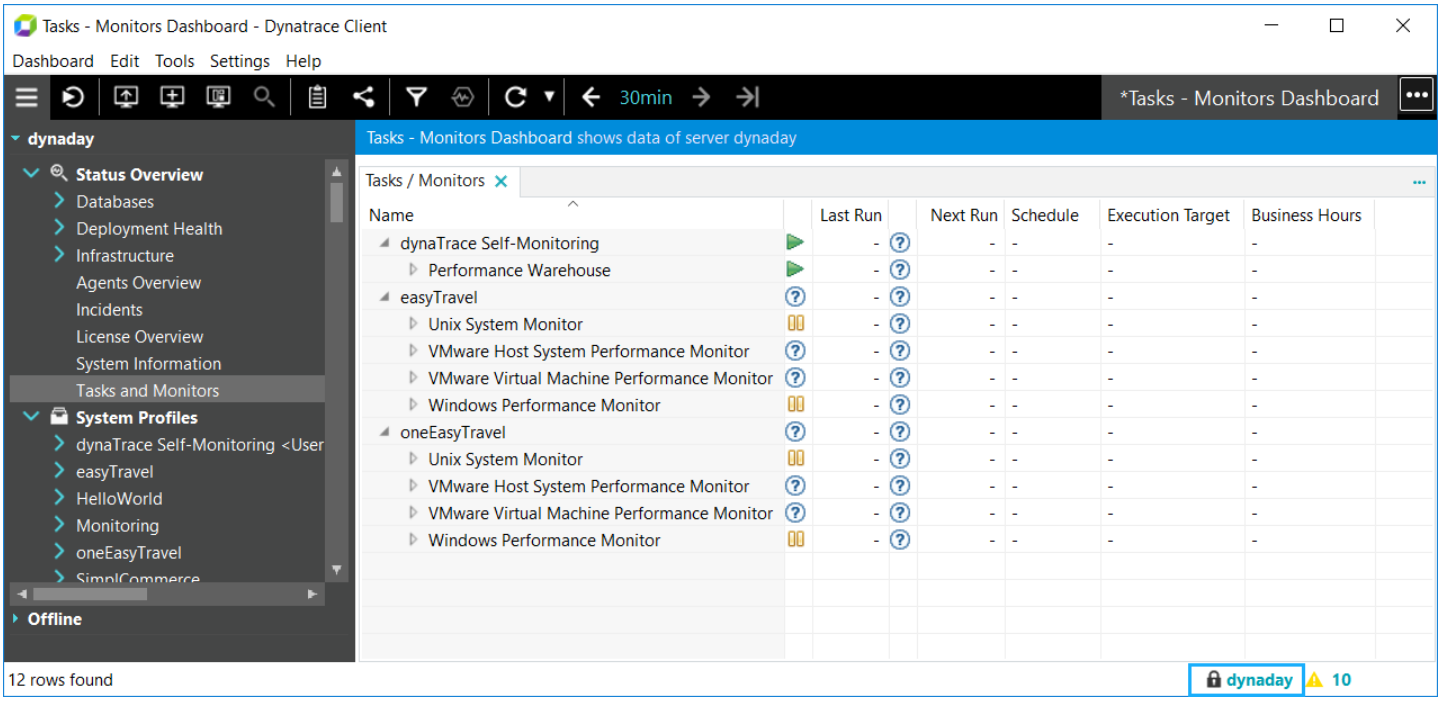
AppMon provides a default administration account that has full access rights granted. This account is available after installation of the AppMon Server, with the username and password both set to `admin`. The admin user cannot be deleted from the system, and another user with this name can't be created. For security reasons, you should change the admin password immediately after installation.

Changing a user's password

> **Note**
>
> You cannot change a user's password through the AppMon Client if the user is authenticated via LDAP.

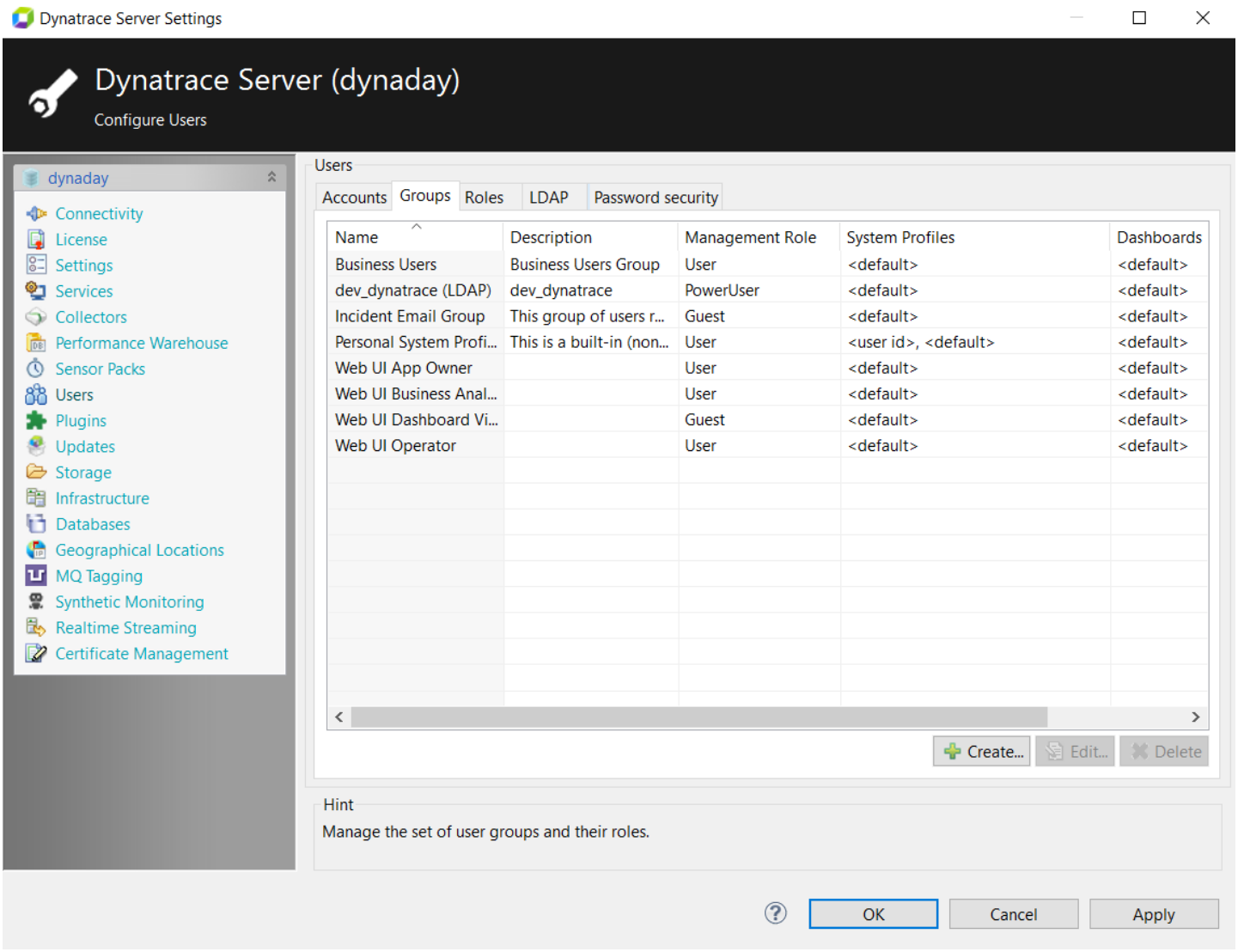
To change the password for the currently logged-in user, do one of the following in the AppMon Client:

- Select **Settings > Change user password**.
- Right-click the server connection icon in the lower right corner of the window, as shown below, and select **Change Password**.



Server Connection icon

Groups



Groups tab

A group defines the roles (and thus the actual permissions) that users can have for AppMon Server management, System Profiles, and dashboards. To assign a user to one or more groups, select the groups in the **User Preferences** dialog box.

When AppMon uses LDAP for user authentication, it is also possible to define LDAP groups. The advantage of using LDAP groups is that the users´ group membership is automatically maintained by the AppMon security system in accordance with the LDAP group memberships. To be recognized as an LDAP group, the AppMon group name must match the LDAP name exactly and you must select the **Is LDAP Group** option in the **User Group Preferences** dialog box. To assign an LDAP group as an AppMon Group, click **Select LDAP Group** in the **User Group Preferences** dialog box. The size of the group list is limited to the page size configuration of your LDAP server (a default of 1,000 rows is usual). Use a filter expression to limit the number of groups.

The **Management Role**, **System Profiles**, and **Dashboards** columns in the **Groups** tab of the **Users** item from the **Dynatrace Server Settings** dialog box show a summary of the profiles and dashboards that are available for each group. For more information, see the Roles section.

Management role

A user group specifies exactly one role to manage AppMon Servers. Roles that are applied for management only grant access to AppMon Server. AppMon ignores other permissions (e.g. for a specific System Profile) contained in this role. To completely deny access to AppMon Server management, select **No Permission**.

System Profiles

To protect System Profiles, AppMon defines roles for them. Unlike management roles, System Profile roles only use permissions that apply to System Profile functionality such as run analyses and create memory dumps. A role's permissions for AppMon Server management are ignored.

Dashboards

Dashboard permissions can have one of two values: **Read** or **Read/Write**. With the **Read** permission, users can view dashboards. Any manipulations are temporary: **Save** and **Save As** operations are denied for the AppMon Server, but you can save on local drives. Users with **Read/Write** permissions have full access to view dashboards, change them, and save them on the AppMon Server.

Roles

A *role* is a set of granted permissions.

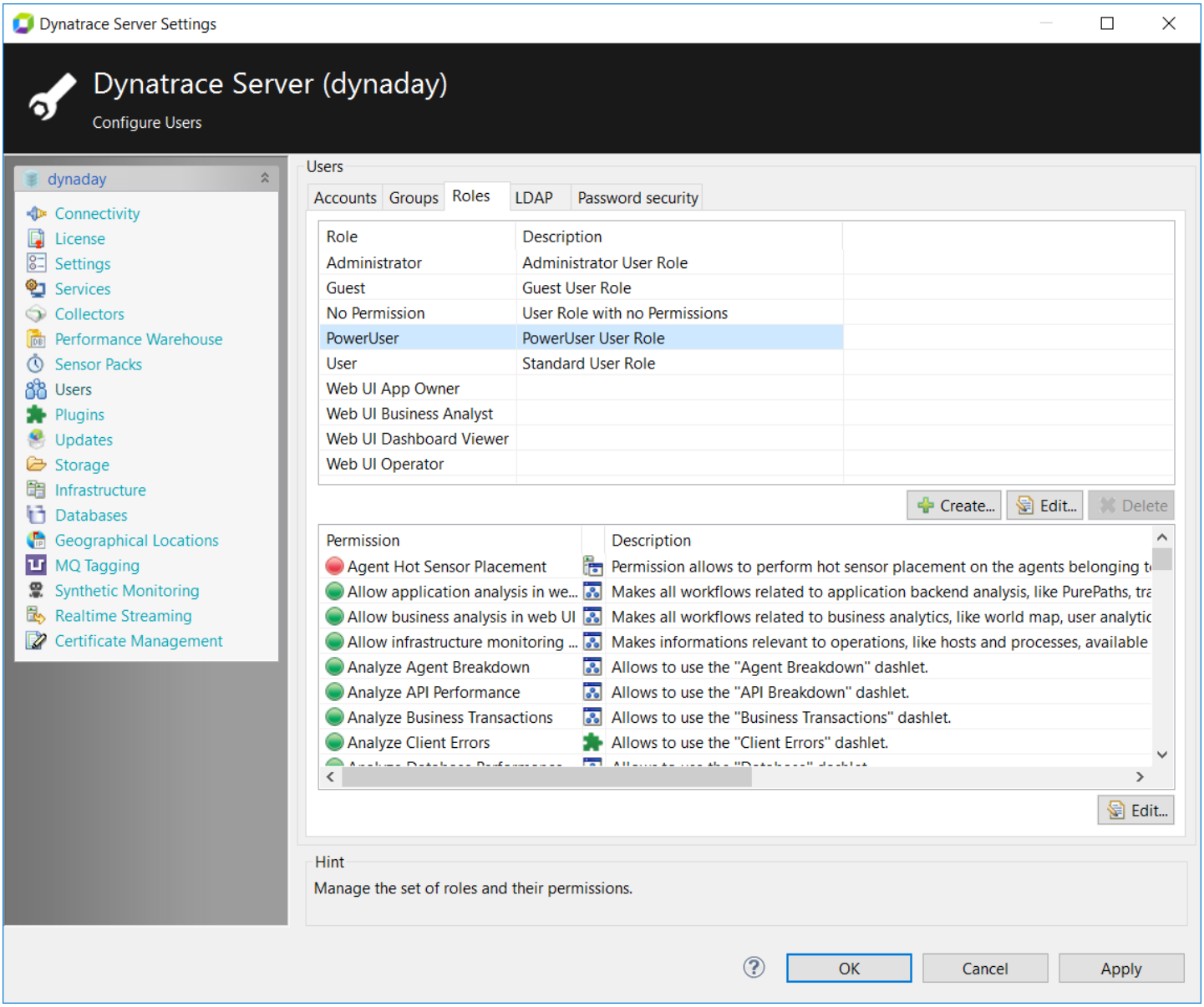
By default, AppMon has these roles for different levels of access control, sorted from less to more powerful:

- No Permission: doesn't have any permissions.
- Guest: allowed to view certain application-related dashboards, but cannot modify them or create new. Doesn't have access for self-monitoring dashboards. Can't modify System Profile and server settings. Typically suitable for business users, who wants to see the data, but doesn't need to perform any configurations.

- User: allowed to view certain application- and system-related dashboards, but cannot modify them or create new. Allowed to work with stored sessions. Can't modify System Profile and server settings. Typically suitable for experienced business users.
- Power User: allowed to view and modify certain application- and system-related dashboards. Can modify System Profile, can't modify server setting. Typically suitable for developers, who should have extended access, but should not affect server settings.
- Administrator: has all the permissions

See Permission mapping for more information about permissions and full list of permissions for these roles.

You cannot delete these roles. The Administrator, Guest, and No Permission roles cannot be edited.



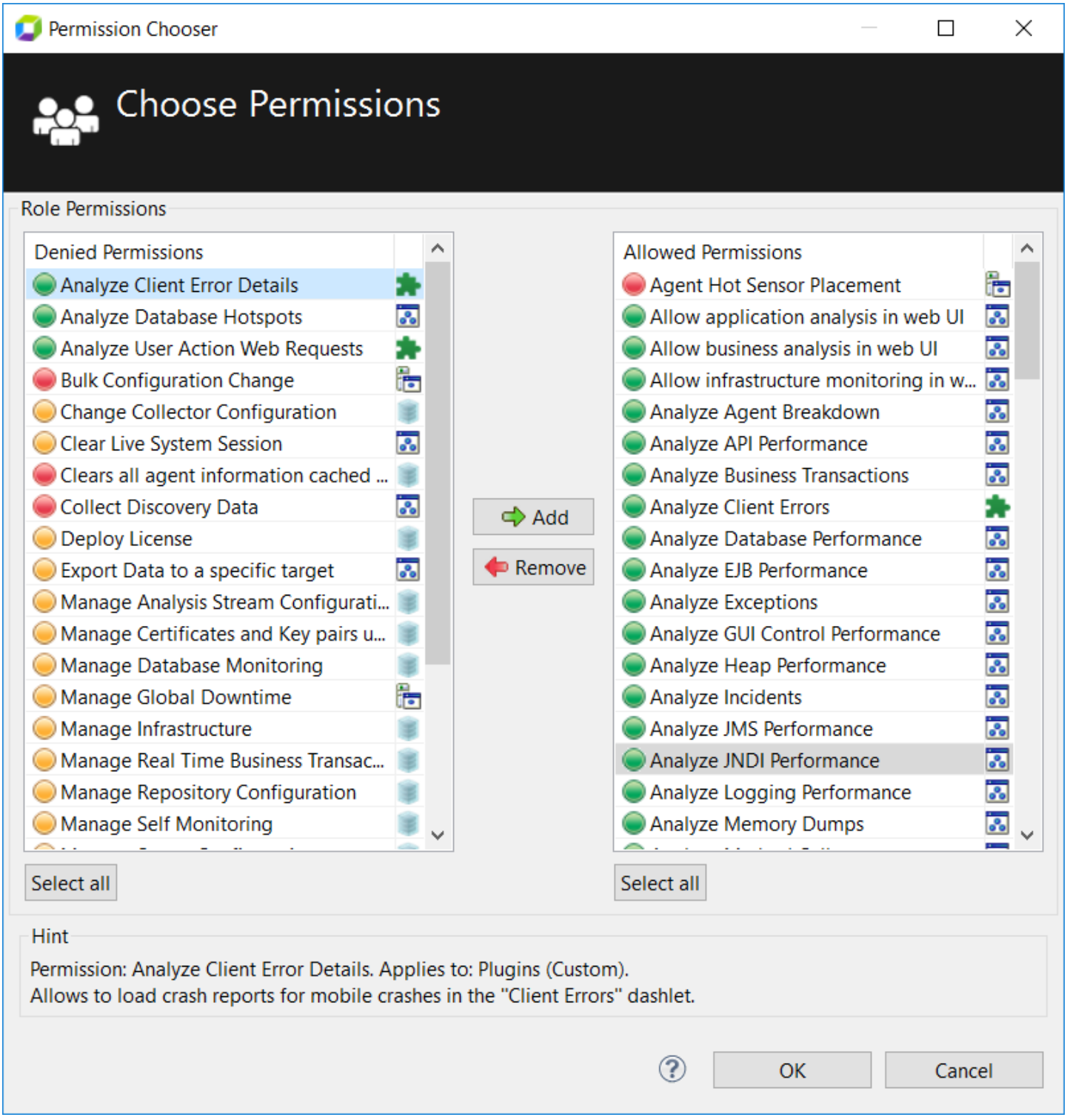
Roles tab

The **Roles** tab of the **Users** item from the **Dynatrace Server Settings** dialog box lists all out-of-the-box and custom roles. Select a role to see the list of permissions granted to the role. The colored icons at the left of the permissions indicate how risky the permission is. For example, a red icon may indicate a permission to change configuration settings, with a potentially significant effect on all users.

If a role's permission can be changed, the **Edit** button below the permissions list is available when you select the role.

To create a custom set of permissions, click **Create** below the roles list and specify a unique name and optional description to create a new role. After you create the role, add permissions to it:

1. Click **Edit** below the Permissions table to open the **Permission Chooser**.
2. Select the desired permissions in the **Denied Permissions** list. To select multiple permissions, hold down the Ctrl key and click the permissions.
3. Click **Add** to move the selected permissions to the **Allowed Permissions** list.
4. Click **OK** to add the allowed permissions to the role and close the **Permission Chooser**.



Permission chooser

LDAP

To support single sign-on solutions, configure AppMon to authenticate users according to credentials stored on an LDAP server.

If a user unknown to the AppMon Server logs in, a local user account is created automatically with the **Use LDAP Authentication** setting selected. The user information (name and email address) is retrieved from LDAP and is subsequently kept in sync. Credentials stored locally at the AppMon Server are ignored; only mappings to local user groups are used from the AppMon Server. User memberships in LDAP groups are verified and updated with current LDAP settings at login.

Note

Default LDAP integration with AppMon includes the LDAP primary group Domain Users, which is specific for the Microsoft Active Directory LDAP system. This may cause login problems with AppMon. To prevent these problems, `<DT_HOME>/dtserver.ini` must contain the following:

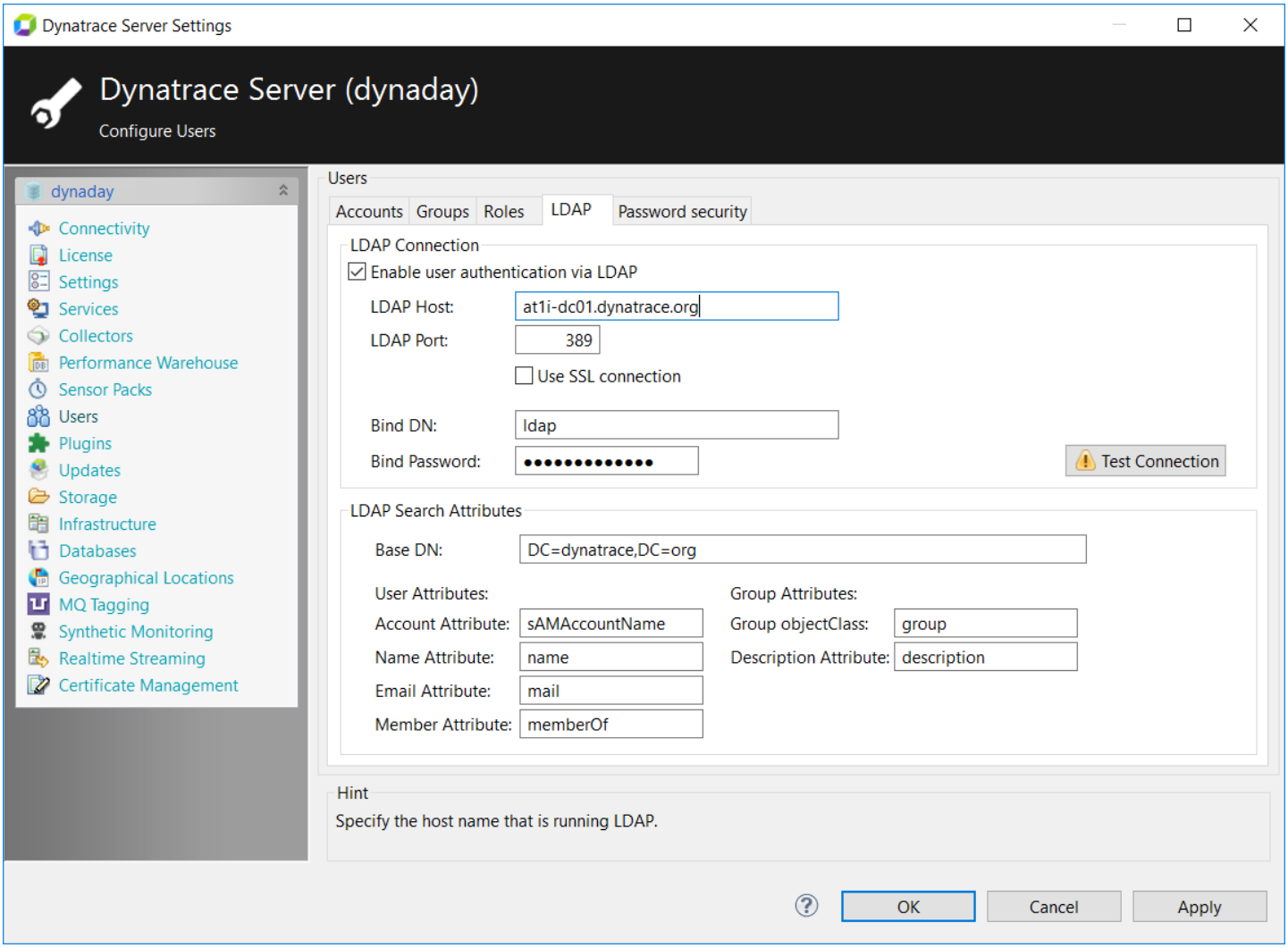
```
-Dcom.dynatrace.diagnostics.includePrimaryGroupForLDAPActiveDirectory
```

To be allowed to log in, a user with a valid LDAP account must be the member of at least one LDAP group that is configured in AppMon. Note that users must be able to access their member attribute to confirm their membership.

However, to allow users to log in without verification of a group membership, add the following to `<DT_HOME>/dtserver.ini`:

```
-Dcom.dynatrace.diagnostics.strictLDAPAuthentication=false
```

Use the **LDAP** tab in the **Dynatrace Server Settings** dialog box to configure the LDAP connection and search attributes.



LDAP tab

Configuring the LDAP connection

To configure the LDAP connection:

1. Enter the **LDAP Host** that runs the LDAP service and the **LDAP Port** (the non-SSL default is 389).
2. If the LDAP server requires it, select **Use SSL Connection** (the default port 636).
3. Specify a **Bind DN** (such as `cn=dtd`, `ou=users`, `dc=company`, `dc=com`) and a **Bind Password** if anonymous search for users is not supported. If **Bind DN** and **Bind Password** are left blank, anonymous search will be used to find users.
4. When you configure an SSL connection, the certificate from the LDAP server may need to be accepted once by the administrator. Click **Test Connection**: if the certificate is unknown, you are prompted to accept it.

Configuring LDAP search attributes

Use the **LDAP Search Attributes** area to map LDAP attributes to AppMon user accounts:

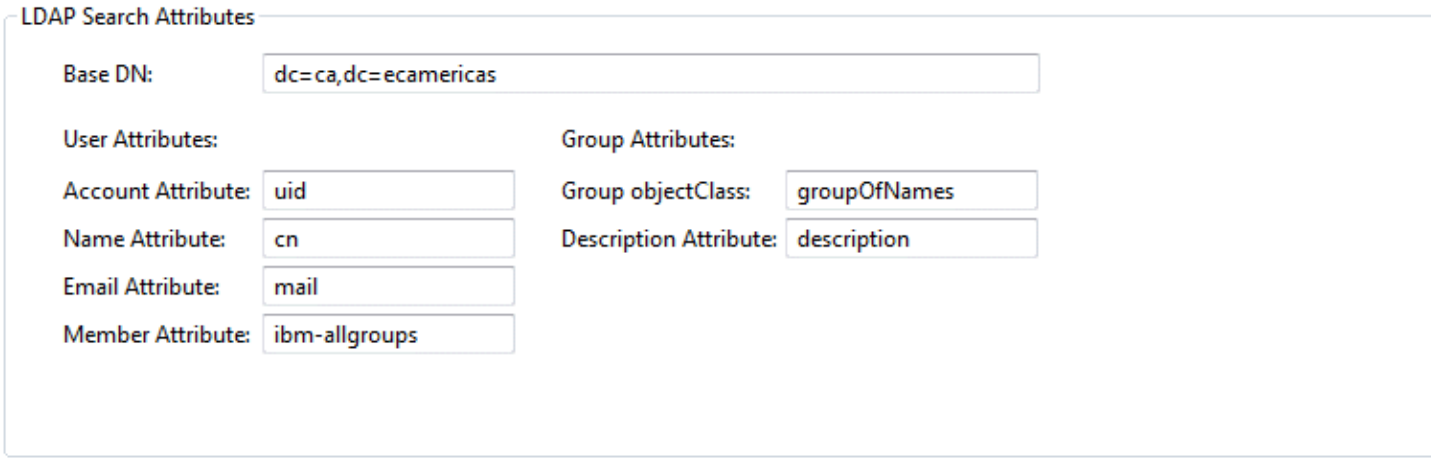
- **Base DN**: The LDAP name of the root node to search for user credentials, such as `ou=users,dc=company,dc=com`.
- **Account Attribute**: The user's login name, for example `uid`, or `sAMAccountName` for ActiveDirectory.
- **Name Attribute**: The user's name.
- **Email Attribute**: The user's email address.
- **Member Attribute**: The group membership for the user account.
- **Group objectClass**: The attribute used to identify groups in the LDAP system.
- **Description Attribute**: The group's description.

These attributes may vary depending on the LDAP server being used. The default settings apply to Microsoft Active Directory.

Use the **Test Connection** button to verify the LDAP connection after you define the search attributes. This check does not verify whether the LDAP search attributes are valid.

IBM Tivoli Directory Server

The following shows an example configuration with according attributes for an IBM TDS.



LDAP performance

The connection timeout to the LDAP server depends on the current network settings. By default, it is the network (TCP) timeout, which is usually a few minutes.

If the LDAP server is down completely, LDAP users cannot log in. However, successful user authentications are cached on the AppMon Server for 10 minutes by default. If the same user logs in within this time frame, no LDAP request is sent again and the user's authentication timeout is prolonged. The user's group membership is not verified or updated within this time frame.

You can configure the authentication timeout by changing the value of the following system property in the `dtserver.ini` file:

```
-Dcom.dynatrace.diagnostics.ldapAuthenticationTimeout=600
```

Password security

To provide an intermediate solution until a more sophisticated user management is implemented, you can define user password security. Setting the **Allow Clients to store credentials** check box requires all users to sign on using their credentials each time they connect to the server. You can set the number of digits, lowercase letters, uppercase letters, special characters, and the minimum length a password must have to be accepted. Old passwords that don't meet the complexity don't have to be instantly changed, but when the password is changed, the complexity requirements apply.

Permissions mapping

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Permissions reference tables

Related pages

User permissions and authentication

To view and manage permissions of a user role, click **Settings > Dynatrace Server > Users > Roles** in the AppMon Client. Select the required role to view the list of its permissions below. To edit permissions list, click **Edit** under permission list.

Consider the severity of risk when granting permissions. Permissions for analyzing data and making changes, such as **Create Dashboard**, are considered low risk. Permissions that affect system performance or make changes at the Server level, such as **Agent Hot Sensor Placement** are high risk. In the AppMon Client following color code is used:


High risk

Medium risk

Low risk

The AppMon Client uses the following icons for usage encoding:


Permission applies to Server Management.

Permission applies to System Profile.

Permission applies to Server Management and System Profile.

Permission applies to Plugins (Custom).

Permissions reference tables

This section lists all AppMon permissions, and which permissions the built-in user roles have.

Click to view the table of available permissions

Risk	Name	Description	Applies to
High	Agent Hot Sensor Placement	Allows to perform hot sensor placement on the agents belonging to the related System Profile.	Server Management, System Profile
Low	Allow application analysis in web UI	Grants access to all workflows related to application backend analysis, like PurePaths, transaction flow, errors in the web UI. Also allows you to view web dashboards.	System Profile
Low	Allow business analysis in web UI	Grants access all workflows related to business analytics, like world map, user analytics, visits, in the web UI. Also allows you to view web dashboards.	System Profile
Low	Allow infrastructure monitoring in web UI	Grants access to information relevant to operations, like hosts and processes, in the web UI. Also allows you to view web dashboards.	System Profile
Low	Analyze Agent Breakdown	Grants access to the **Agent Breakdown** dashlet.	System Profile
Low	Analyze API Performance	Grants access to the **API Breakdown** dashlet.	System Profile
Low	Analyze Business Transactions	Grants access to the **Business Transactions** dashlet.	System Profile
Low	Analyze Client Error Details	Allows to load crash reports for mobile crashes to the **Client Error** dashlet.	Plugings (Custom)
Low	Analyze Client Error	Grants access to the **Client Error** dashlet.	Plugings (Custom)

Low	Analyze Database Hotspot	Grants access to the **Database Hotspots** dashlet.	System Profile
Low	Analyze Database Performance	Grants access to the **Database** dashlet.	System Profile
Low	Analyze EJB Performance	Grants access to the **EJB Analyzer** dashlet.	Sustem Profile
Low	Analyze Exceptions	Grants access to the **Errors** dashlet.	System Profile
Low	Analyze GUI Control Performance	Grants access to the **GUI Controls** dashlet.	System Profile
Low	Analyze Heap Performance	Grants access to the **Memory Allocations** dashlet, to analyze heap performance.	System Profile
Low	Analyze Incidents	Grants access to the **Incidents** dashlet. This permission only allows to view the dashlet. To configure incidents you need the **Manage Incidents** permission.	System Profile
Low	Analyze JMS Performance	Allows to analyze JMS performance in the **Messaging** dashlet.	System Profile
Low	Analyze JNDI Performance	Grants access to the **Naming Services** dashlet.	System Profile
Low	Analyze Logging Performance	Grants access to the **Logging** dashlet.	System Profile
Low	Analyze Memory Dumps	Grants access to the **Selective Memory** and **Total Memory** dashlets. This permission only allows to view the dashlets. To create memory dumps you need the **Create Memory Dumps** permission.	System Profile
Low	Analyze Method Calls	Grants access to the **Methods** dashlet.	System Profile
Low	Analyze Method Hotspots	Grants access to the **Method Hotspots** dashlet.	System Profile
Low	Analyze PurePath	Grants access to the **PurePaths** dashlet.	System Profile
Low	Analyze PurePaths Entry Points	Grants access to the **Entry Points** dashlet.	System Profile

Low	Analyze Remoting Performance	Grants access to the **Remoting** dashlet.	System Profile
Low	Analyze Response Time Hotspots	Grants access to the **Response Time Hotspots** dashlet.	System Profile
Low	Analyze Sampling Sessions	Grants access to the **CPU Sampling** dashlet, to analyze sampling sessions.	System Profile
Low	Analyze Sequence Diagram	Grants access to the **Sequence Diagram** dashlet.	System Profile
Low	Analyze Suspension Performance	Grants access to the **Runtime Suspension** dashlet.	System Profile
Low	Analyze System	Grants access to the **Status Overview** dashlet.	Server Management, System Profile
Low	Analyze Thread Dumps	Grants access to the **Threads** dashlet. Creating thread dumps additionally requires the **Create Thread Dumps** permission.	System Profile
Low	Analyze Transaction Flow	Grants access to the **Transaction Flow** dashlet.	Plugins (Custom)
Low	Analyze User Actions PurePaths	Grants access to the **User Actions PurePaths** dashlet.	Plugins (Custom)
Low	Analyze User Actions	Grants access to the **User Actions** dashlet.	Plugins (Custom)
Low	Analyze User Action Web Requests	Grants access to web requests in the **User Actions** dashlet.	Plugins (Custom)
Low	Analyze User Visits	Grants access to the **Visits** dashlet.	Plugins (Custom)
Low	Analyze Webpage Performance	Grants access to the **Tagged Web Requests** dashlet	System Profile
Low	Analyze WebRequest Performance	Grants access to the **Web Requests** dashlet	System Profile
Low	Analyze WebService Performance	Grants access to the **Web Services** dashlet	System Profile

High	Bulk Configuration Change	Allows to change settings for several System Profiles or AppMon Servers at once.	Server Management, System Profile
High	Change Active System Configuration	Allows to change the active System Profile settings.	System Profile
Medium	Change Collector Configuration	Allows to change Collector settings	Server Management
Medium	Change Incident Configuration	Allows to change the incident configuration, including rules. For being able to change incident rules the permission **Change System Profile Settings** is required, too.	Server Management, System Profile
Medium	Change Incident Configuration	Allows to change the monitor configuration.	Server Management
Medium	Change Plugin Default Configuration	Allows to change the plugin default configuration.	Server Management
Medium	Change System Profile Settings	Allows to change the System Profile settings.	Server Management, System Profile
Medium	Change User Password	Allows to change user's password.	Server Management
Medium	Clear Live System Session	Allows to clear live session data	System Profile
High	Clears all agent information cached by this Collector	Allows to clear Collector's cache.	Server Management
High	Collect Discovery Data	Allows to collect, merge and redistribute the data collected during the Discovery Runs of all connected Dynatrace Collectors. This data is placed in the `server/deployment/imap` directory and is automatically used for new Dynatrace Collectors that are added to the system to avoid the requirement of a Discovery Run.	System Profile
Low	Create Dashboard	Allows to create dashboards.	Server Management
High	Create Memory Dumps	Allows to create memory snapshots. This permission requires the **Analyze Memory Dumps** permission.	System Profile
High	Create Sample	Allows to create CPU samples.	System Pforile

Medium	Create System Profile	Allows to create System Profiles.	Server Management
High	Create Thread Dumps	Allows to create thread dumps.	System Profile
Low	Create web dashboards	Allows to create, import, and clone web dashboards.	Server Management
Low	Decompile Source Code	Allows to decompile the source code.	
Medium	Delete Resource Dumps	Allows to delete memory dumps, thread dumps and CPU samples.	Server Management
Medium	Delete Stored Sessions	Allows to delete stored sessions.	Server Management
Medium	Delete System Profile	Allows to delete System Profiles.	Server Management
Medium	Deploy Bundles for Testing Purpose	Allows to test installed plugins.	Server Management
Medium	Deploy License	Allows to deploy licenses.	Server Management
Medium	Deploy Sensor Packs	Allows to manage sensor packs.	Server Management
Low	Edit Stored Sessions (PurePath Sessions, Memory Dumps, Thread Dumps and Samplings)	Allows to modify stored sessions.	Server Management
Medium	Export Data to a specific target.	Allows to export data.	System Profile
Medium	Export Live Sessions to a local directory	Allows to export PurePaths and time series to your local hard drive.	System Profile
Medium	Export Resource Dumps	Allows to export memory dumps, thread dumps and CPU samples.	Server Management
Medium	Export Stored Sessions to a local directory	Allows to export PurePaths and time series from stored sessions to your local hard drive.	Server Management

Low	Export System Profile of Live Session	Allows to export System Profiles from live session to your local hard drive.	System Profile
Low	Export System Profile of Stored Session	Allows to export System Profiles from stored session to your local hard drive.	Server Management
Low	Find node in a session	Grants access to the **Browser PurePaths** dashlet.	System Profile
Low	Generate Dashboard Reports via Web Service	Allows to create reports of dashboards in AppMon Web.	System Profile
Medium	Import Session to a Dynatrace Server	Allows to import previously stored sessions to a Dynatrace Server	System Profile
Medium	Manage Analysis Stream Configuration	Allows to manage analysis stream configuration.	Server Management
Low	Manage Business Transactions	Allows to modify business transactions	System Profile
Medium	Manage Certificates and Key pairs used to secure Communication between Modules	Allows to manage Certificates and Key pairs used to secure Communication between Modules.	Server Management
Medium	Manage Database Monitoring	Allows to manage database monitoring.	Server Management
Medium	Manage Global Downtime	Allows to manage global downtime.	Server Management, System Profile
Medium	Manage Incident Downtime	Allows to manage incidents downtime.	Server Management, System Profile
Medium	Manage Incidents	Allows to confirm incidents. This permission requires the **Analyze Incidents** to have the access to the **Incidents** dashlet.	System Profile
Medium	Manage Infrastructure	Allows to manage server’s infrastructure	Server Management
Medium	Manage Plugin Bundles	Allows to manage plugin bundles	Server Management
Medium	Manage Real Time Business Transactions Feed	Allows to manage business transactions in real time.	Server Management

Medium	Manage Repository Configuration	Allows to manage Performance Warehouse settings.	Server Management
Medium	Manage Self Monitoring	Allows to edit Dynatrace Self-Monitoring System Profile.	Server Management
Medium	Manage Server Configuration	Allows to edit Dynatrace Server settings.	Server Management
Medium	Manage Tasks and Monitors	Allows to edit configured tasks and monitors.	System Profile
Medium	Manage Updates	Allows to manage Dynatrace Server updates (for example, installation and deactivation of updates)	Server Management
Medium	Manage User Configurations	Allows to manage user accounts.	Server Management
Meduim	Read Confidential Strings	Allows to read Confidential Strings.	Server Management
Low	Read Stored Sessions (including Memory Dumps, Thread Dumps and Samplings)	Allows to read previously stored sessions.	Server Management
Low	Re-analyze Stored Sessions	Allows to process configuration changes for previously stored sessions.	Server Management
Low	Rename Sessions	Allows to rename previously stored sessions.	Server Management
Medium	Restart Agents	Allows to restart Agents, connected to the server.	Server Management
Medium	Serarch for PurePath	Allows to find PurePaths by HTTP response header, as shown in load test tools like SilkPerformer.	Server Management
Medium	Show Database Monitoring	Grants access to the database monitoring.	Server Management, System Profile
Medium	Show Infrastructure	Grants access to the infrastructure view.	Server Management, System Profile
Low	Show Measure data within Charts	Allows to show measured data in charts.	System Profile

Medium	Shutdown or Restart	Allows to shutdown and/or restart the Server.	Server Management
Medium	Shutdown or Restart Collect	Allows to shutdown and/or restart the Collector.	Server Management
Low	Start or Stop Session Recording	Allows to start/stop session recording.	System Profile
Low	Store Live Stream Session	Allows to store PurePaths and time series.	Server Management, System Profile
Low	Test Automation-Accept Change	Allows to accept changes in the **Test Automation** dashlet.	Plugins (Custom)
Low	Test Automation-Change Test Assignee	Allows to change the assignee (the person getting notification emails) for a test in the **Test Automation** dashlet.	Plugins (Custom)
Low	Test Automation-Delete Test	Allows to delete tests from the **Test Automation** dashlet.	Plugins (Custom)
Low	Test Automation-Invalidate Test Execution	Allows to invalidate a test run in the **Test Automation** dashlet.	Plugins (Custom)
Low	Test Automation-Remove Measures	Allows to delete measures from the **Test Automation** dashlet.	Plugins (Custom)
Medium	View System Information Dashlet	Grants access to the System Information dashlet.	Server Management
Medium	View Tasks and Monitors	Grants access to configured tasks and monitors.	Server Management, System Profile
Low	View Web Dashboards	Grants access to the shared web dashboards.	System Profile
Medium	Web Service Interface Access	Grants access to the web services of the AppMon Server.	Server Management

Default permissions for the PowerUser role

Risk	Permission	Usage
Low	Show Measure data within Charts	System Profile

Medium	Search for PurePaths	Server Management
Low	Start or Stop Session Recording	System Profile
Medium	Deploy Bundles for Testing Purpose	Server Management
Low	Analyze Heap Performance	System Profile
Medium	Show Infrastructure	Server Management and System Profile
Low	Analyze PurePaths	System Profile
High	Create Memory Dumps	System Profile
Medium	Manage Incident Downtime	Server Management and System Profile
Low	Analyze Memory Dumps	System Profile
Low	Analyze Sampling Sessions	System Profile
Low	Create Dashboard	Server Management
Low	Decompile Source Code	
Medium	Export Stored Sessions to a local directory	Server Management
Low	Analyze Thread Dumps	System Profile
High	Change Active System Configuration	System Profile

Medium	Manage Incidents	System Profile
Low	Test Automation-Delete Test	Plugins (Custom)
Low	Export System Profile of Stored Session	Server Management
Medium	Delete Stored Sessions	Server Management
High	Create Sample	System Profile
Medium	Change Monitor Configuration	System Profile
Low	Analyze Logging Performance	System Profile
Medium	Show Database Monitoring	Server Management and System Profile
Low	Analyze Incidents	System Profile
Low	Analyze JMS Performance	System Profile
Low	Allow business analysis in web UI	System Profile
Low	Analyze Response Time Hotspots	System Profile
Low	View web dashboards	System Profile
Low	Analyze API Performance	System Profile
Low	Create web dashboards	Server Management

High	Create Thread Dumps	System Profile
Medium	Manage Tasks and Monitors	System Profile
Low	Test Automation-Remove Measures	Plugins (Custom)
Low	Analyze Agent Breakdown	System Profile
Low	Analyze Remoting Performance	System Profile
Medium	Export Live Sessions to a local directory	System Profile
Low	Export System Profile of Live Session	System Profile
Low	Re-Analyze Stored Sessions	Server Management
Low	Manage Business Transactions	System Profile
Low	Analyze System	Server Management and System Profile
Low	Analyze Suspension Performance	System Profile
Low	Analyze Method Calls	System Profile
Low	Allow infrastructure monitoring in web UI	System Profile
Low	Analyze WebRequest Performance	System Profile
Medium	Change User Password	Server Management

Low	Allow application analysis in web UI	System Profile
Low	Read Stored Sessions (including Memory Dumps, Thread Dumps and Samplings)	Server Management
Low	Test Automation-Accept Change	Plugins (Custom)
Low	Analyze Business Transactions	System Profile
Low	Analyze Exceptions	System Profile
Low	Generate Dashboard Reports via Web Service	System Profile
Low	Analyze Webpage Performance	System Profile
Low	Test Automation-Change Test Assignee	Plugins (Custom)
Low	Analyze JNDI Performance	System Profile
Low	Analyze GUI Control Performance	System Profile
Medium	Import Stored Sessions to a Dynatrace Server	System Profile
Low	Test Automation-Invalidate Test Execution	Plugins (Custom)
Medium	Manage Plugin Bundles	Server Management
Medium	View Tasks and Monitors	Server Management and System Profile
Medium	View System Information Dashlet	Server Management

Low	Analyze Database Hotspots	System Profile
Low	Analyze Method Hotspots	System Profile
Medium	Delete System Profile	Server Management
Medium	Change Plugin Default Configurations	Server Management
Medium	Delete Resource Dumps	Server Management
Medium	Change System Profile settings	Server Management and System Profile
Low	Analyze PurePaths Entry Points	System Profile
Medium	Create System Profile	Server Management
Low	Analyze WebService Performance	System Profile
Medium	Read Confidential Strings	Server Management
Medium	Web Service Interface Access	Server Management
Low	Analyze EJB Performance	System Profile
High	Agent Hot Sensor Placement	Server Management and System Profile
Low	Analyze Database Performance	System Profile
Low	Rename Sessions	Server Management

Medium	Change Incident Configuration	Server Management and System Profile
Medium	Export Resource Dumps	Server Management
Medium	Deploy Sensor Packs	Server Management
Low	Store Live System Session	Server Management and System Profile
Low	Find nodes in a session	System Profile
Medium	Manage Database Monitoring	Server Management
Low	Edit Stored Sessions (PurePath Sessions, Memory Dumps, Thread Dumps and Samplings)	Server Management

Default permissions for the User role

Risk	Permission	Usage
Low	Show Measure data within Charts	System Profile
Medium	Search for PurePaths	Server Management
Low	Analyze Heap Performance	System Profile
Medium	Show Infrastructure	Server Management and System Profile
Low	Analyze PurePaths	System Profile
Low	Analyze Memory Dumps	System Profile
Low	Analyze Sampling Sessions	System Profile

Low	Create Dashboard	Server Management
Medium	Export Stored Sessions to a local directory	Server Management
Low	Analyze Thread Dumps	System Profile
Low	Test Automation-Delete Test	Plugins (Custom)
Low	Export System Profile of Stored Session	Server Management
Medium	Delete Stored Sessions	Server Management
Low	Analyze Logging Performance	System Profile
Medium	Show Database Monitoring	Server Management and System Profile
Low	Analyze Incidents	System Profile
Low	Analyze JMS Performance	System Profile
Low	Allow business analysis in web UI	System Profile
Low	Analyze Response Time Hotspots	System Profile
Low	View web dashboards	System Profile
Low	Analyze API Performance	System Profile
Low	Create web dashboards	Server Management

Low	Test Automation-Remove Measures	Plugins (Custom)
Low	Analyze Agent Breakdown	System Profile
Low	Analyze Remoting Performance	System Profile
Medium	Export Live Sessions to a local directory	System Profile
Low	Export System Profile of Live Session	System Profile
Low	Re-Analyze Stored Sessions	Server Management
Low	Analyze System	Server Management and System Profile
Low	Analyze Suspension Performance	System Profile
Low	Analyze Method Calls	System Profile
Low	Allow infrastructure monitoring in web UI	System Profile
Low	Analyze WebRequest Performance	System Profile
Medium	Change User Password	Server Management
Low	Allow application analysis in web UI	System Profile
Low	Read Stored Sessions (including Memory Dumps, Thread Dumps and Samplings)	Server Management
Low	Test Automation-Accept Change	Plugins (Custom)

Low	Analyze Business Transactions	System Profile
Low	Analyze Exceptions	System Profile
Low	Generate Dashboard Reports via Web Service	System Profile
Low	Analyze Webpage Performance	System Profile
Low	Test Automation-Change Test Assignee	Plugins (Custom)
Low	Analyze JNDI Performance	System Profile
Low	Analyze GUI Control Performance	System Profile
Medium	Import Stored Sessions to a Dynatrace Server	System Profile
Low	Test Automation-Invalidate Test Execution	Plugins (Custom)
Medium	View Tasks and Monitors	Server Management and System Profile
Medium	View System Information Dashlet	Server Management
Low	Analyze Database Hotspots	System Profile
Low	Analyze Method Hotspots	System Profile
Medium	Delete Resource Dumps	Server Management
Low	Analyze PurePaths Entry Points	System Profile

Low	Analyze WebService Performance	System Profile
Medium	Read Confidential Strings	Server Management
Medium	Web Service Interface Access	Server Management
Low	Analyze EJB Performance	System Profile
Low	Analyze Database Performance	System Profile
Low	Rename Sessions	Server Management
Medium	Export Resource Dumps	Server Management
Low	Find nodes in a session	System Profile
Low	Edit Stored Sessions (PurePath Sessions, Memory Dumps, Thread Dumps and Samplings)	Server Management

Guest role permissions

Risk	Permission	Usage
Low	Analyze System	Server Management and System Profile
Low	Analyze Suspension Performance	System Profile
Low	Show Measure data within Charts	System Profile
Low	Analyze Heap Performance	System Profile
Medium	Show Infrastructure	Server Management and System Profile

Low	Analyze Method Calls	System Profile
Low	Allow infrastructure monitoring in web UI	System Profile
Low	Analyze WebRequest Performance	System Profile
Medium	Change User Password	Server Management
Low	Allow application analysis in web UI	System Profile
Low	Analyze PurePaths	System Profile
Low	Analyze Business Transactions	System Profile
Low	Analyze Exceptions	System Profile
Low	Analyze Webpage Performance	System Profile
Low	Analyze JNDI Performance	System Profile
Low	Analyze GUI Control Performance	System Profile
Low	Analyze Database Hotspots	System Profile
Low	Analyze Logging Performance	System Profile
Low	Analyze Method Hotspots	System Profile
Medium	Show Database Monitoring	Server Management and System Profile

Low	Analyze Incidents	System Profile
Low	Analyze JMS Performance	System Profile
Low	Allow business analysis in web UI	System Profile
Low	Analyze PurePaths Entry Points	System Profile
Low	Analyze Response Time Hotspots	System Profile
Low	Analyze WebService Performance	System Profile
Low	View web dashboards	System Profile
Low	Analyze API Performance	System Profile
Low	Analyze EJB Performance	System Profile
Low	Analyze Database Performance	System Profile
Low	Analyze Agent Breakdown	System Profile
Low	Analyze Remoting Performance	System Profile
Low	Find nodes in a session	System Profile

See User Permissions and Authentication > Roles for more information.

SAML/SSO configuration

On this page

Prerequisites and limitations

Configuration

Downloads

Related pages

User permissions and authentication

Permissions mapping

Important

This is a beta feature in the AppMon 2017 May. Beta features are publicly available in the release and supported by Dynatrace technical support team. Please keep in mind though, that it is limited and not tested extensively in real world scenarios. Please verify in your test or staging environment, whether it works as expected and fit your needs.

Prerequisites and limitations

- GUI for SSO configuration is not available.
- SSO works only in Windows-based AppMon Clients and the WebUI.
- On Mac, SSO might work but there may be some problems with certificate management and request redirection.
- No certificate verification implemented. The AppMon Server does not check whether certificates were revoked, certificate chains are not checked (WIP).
- AppMon Server login is based on matching groups; so during the SSO the identification provider must send a group name which has a matching group on the AppMon Server side. The user permissions are then based on that matching group.

Configuration

To activate the SAML/SSO feature you have to create two files in the `<DT_HOME>/server/conf` directory:

- `sso.config.xml` for the SAML/SSO configuration
- `idp.xml` for the identity provider configuration metadata

The frontend server reads these files during the startup phase. If the `ssoEnabled` attribute is set to `true` in the `sso.config.xml` file, SAML/SSO is active. If you change these files while server is running, you have to restart the server. You can find two zipped sample files attached to this page.

The sample `sso.config.xml` file contains the following settings:

- Identity provider attributes
- Identity provider attribute names
- SAML service provider attributes
- SAML general attributes

SAML identity provider attributes

Attribute	Description
idpEntityId	The SAML entity ID of the identity provider. It is recommended that a system entity use containing its own domain name to identify itself. If the identification provider exposes m the entity ID is used as a well known URL for the meta data of the entity. For example: `https://my.idp.my.domain/simplesamlphp/saml2/idp/metada`
idpMetadataFilename	The name of the file with the identity provider metadata. This file contains the identity p configuration information which is the basis for sending SAML requests to the service p The file is located in the `<DT_HOME>/server/conf` directory.

Identity provider attribute names

The identity provider sends information about the authenticated user to the service provider. The info may be stored in attributes of various names. For example, one identity provider could send the authenticated user name using the attribute called `userId`, and another provider could use `uid` instead. To be able to read these values the AppMon Server has to know the attributes naming.

Attribute	Description	Default value
userIdAttributeName	The name of the attribute which holds the authenticated user name.	`uid`

groupsAttributeName	The name of the attribute which holds the authenticated user's group assignments. At least one of the group assignments must exist in AppMon, because the AppMon user permissions are based on such a group. The user name sent by the identification provider is meaningless for AppMon.	`eduPersonAffiliation`
fullNameAttributeName	The name of the attribute which holds the authenticated user's full name. If it's empty, the name is not retrieved.	`fullName`
emailAddressAttributeName	The name of the attribute which holds the authenticated user's email address. If it's empty, the email address is not retrieved.	`emailAddress`

SAML Service Provider Attributes

Attribute	Description	Default value
spEntityId	The entity ID of the service provider. The SAML service provider is part of the AppMon Frontend Server and it uses the WebUI port 9911 for communication.	`https:/`
spLoginPath	The path to the AppMon server's login servlet. The fully qualified URL looks like this: `https://<AppMonServer>:9911/sso/login`	`sso/log`
spLoginBinding	The SAML login binding. It specifies the kind of HTTP messages (GET or POST) which can be handled by the service provider. Possible values are `HTTP_POST` for POST messages, and `HTTP_REDIRECT` for GET messages.	`HTTP_RE`
spLogoutPath	The path to the AppMon server's logout servlet. The fully qualified URL looks like this: `https://<AppMonServer>:9911/sso/logout`.	`sso/log`
spLogoutBinding	The SAML logout binding. It specifies the kind of HTTP messages (GET or POST) which can be handled by the service provider. Possible values are `HTTP_POST` for POST messages, and `HTTP_REDIRECT` for GET messages. The default is `HTTP_REDIRECT`.	`HTTP_RE`
spCertificateFingerprint	The fingerprint (or "thumb print") of the service provider certificate. Currently, this certificate has to be located in the AppMon Server keystore. The fingerprint should look like this: `3a:43:a3:db:b6:e9:e2:af:b7:4c:5f:29:b7` Both colons (:) and spaces are allowed as separators. The *SHA-256* algorithm is used for creating that fingerprint. Without fingerprint, the service provider does not sign or encrypt its messages.	
spMetadataValidityPeriodInDays	The default validity of service provider metadata in days.	180

SAML General Attributes

Attribute	Description	Default value
ssoEnabled	Determines whether the SAML/SSO feature is turned on. Possible values are `true` and `false`.	`false`
allowWeakAuthnResponses	The SAML responses sent to the AppMon Server are verified against the SAML specification. If the response does not comply the check fails and authentication fails on the AppMon Server. Some additional checks are made which should prevent some injection attacks. If set to `true` this attribute disables those additional checks.	`false`
ssoRequestCacheTimeout	AppMon Server side timeout for the authentication and logout request sent to the identity provider in seconds.	`300`

Downloads

- `sso.config.xml`
- `idp.xml`

Security and compliance whitepaper

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Understanding privacy regulation

Payment card industry data security standard

AppMon Application Monitoring implementation best practices

User experience monitoring (UEM) security and privacy

Cross-site scripting

User experience monitoring (UEM) and security

Privacy conclusions

If you need a hard-copy, please print/save to PDF/HTML.

Understanding privacy regulation

Application Performance Management (APM) solutions are designed to capture and retain transaction data that typically contains personal information, such as Social Security numbers and credit card information. APM solutions handles this data carefully and does not inhibit your company's compliance with industry regulations. For this reason, you should consider APM products and practices within the context of your organization's security and risk management practices.

The table below provides brief summaries of key data-protection laws and how they influence APM processes.

Table 1. Important Privacy Regulations

Regulation	Description	Impact on APM

PCI DSS	The *Payment Card Industry Data Security Standard* is a proprietary information security standard for organizations that handle cardholder information (for example, credit card numbers). The standard was created to reduce credit card fraud.	All vendors that conduct financial transactions over the internet must comply with PCI standards. APM solutions must not leak credit card information.
SOX	The *Sarbanes-Oxley Act* of 2002 is a United States federal law that sets standards for the accuracy of financial information for all U.S. public company boards, management, and public accounting firms.	SOX is focused on the prevention of financial data manipulation. Leaking of financial records through APM can lead to insider stock manipulation and other SOX violations. Therefore access to APM information must be controlled and all APM data must be encrypted.
HIPAA	The *Health Insurance Portability and Accountability Act* of 1996 is a United States federal law that defines standards for electronic health care transactions.	HIPAA addresses the security and privacy of patient data and social security numbers. Data transmission must be encrypted and access to health data must be protected.
CSB 1386	Introduced in 2003, *California Senate Bill 1386* is a California law that regulates the privacy of personal information. It requires anyone working with computerized personal information to disclose any security breaches related to the release of unencrypted personal data.	In most cases, “losing” personal data means a laptop was stolen or lost with personal data on its hard drive. If the disk is not encrypted, organizations must inform agencies and all affected Californian customers about the loss. Affected customers must also be notified if their personal data is published to an insecure location. Hard disk encryption should be considered in regulated industries. The relevance of this law to APM is low, because the risk of losing a server hard disk is low.
201 CMR 17.00	Concerned about identity theft, Massachusetts introduced the *201 CMR 17.00* law in 2010. It requires anyone who stores or uses personal information about a Massachusetts resident to plan and demonstrate technical, physical, and administrative protection of the information.	An APM solution needs to be considered in the context of overall personal information protection. All data transmission must be encrypted, and access to personal data must be restricted.
EU Directive 2002-58 and 95/46/EC	*EU Directive 2002-58* and *95/46/EC* are European Union directives that regulate the processing of personal data within the EU.	These EU directives go beyond the protection of personal data (which requires encrypted transmission and access restrictions). They also require privacy considerations for user tracking (for example, cookies).
PIPEDA	The *Personal Information Protection and Electronic Documents Act* of 2000 is a Canadian law that promotes data privacy in electronic commerce. The act is intended to reassure the European Union that Canadian privacy laws adequately protect the personal information of European citizens.	PIPEDA requires that APM solutions carefully manage personal data, including credit card data.

Payment card industry data security standard

A good starting point for a security discussion is the Payment Card Industry Data Security Standard. Its principles can easily be applied to other security regulations. PCI DSS is an information security standard for organizations that handle cardholder information for major debit, credit, prepaid, e-purse, ATM, and POS cards. Defined by the Payment Card Industry Security Standards Council, the standard increases controls around cardholder data.

Requirements overview and applicability to APM

The following table provides a high-level overview of PCI DSS requirements:

Table 2. PCI DSS Requirements

Requirement	Details	Applicability to APM
Build and maintain a secure network.	Install and maintain a firewall to protect cardholder data.	n/a[1]
	Do not use vendor-supplied defaults for system passwords and other security parameters.	applicable
Protect cardholder data	Protect stored cardholder data.	applicable
	Encrypt transmission of cardholder data across open, public networks.	applicable
Maintain a Vulnerability Management Program	Use and regularly update anti-virus software or programs,	n/a[1]
	Develop and maintain secure systems and applications.	n/a[1]
Implement Strong Access Control Measures	Restrict access to cardholder data by business need to know.	applicable
	Assign a unique ID to each person with computer access.	applicable
	Restrict physical access to cardholder data.	applicable
Regularly Monitor and Test Networks	Track and monitor all access to network resources and cardholder data.	applicable

	Regularly test security systems and processes.	Include APM systems and processes in the tests.
Maintain an Information Security Policy	Maintain a policy that addresses information security for all personnel.	Include APM standards in the policy.

[1] While these requirements are not directly applicable to APM, any APM solution must operate using PCI DSS compliant infrastructure, firewalls, and anti-virus solutions.

PCI compliance relates to business management processes and environments, not to individual products and tools used in these environments. In this way, PCI is like CMM or ISO — these are process certifications, not product certifications.

Some PCI DSS requirements are not applicable to APM; for example, operating a virus scanner, maintaining a firewall, and managing usage and access control for IT systems. While APM vendors have no influence over these processes, they should provide tools and means that allow for proper implementation of these processes where applicable.

Other PCI DSS requirements, including those listed below, are directly relevant to core APM practices. For these requirements, it's important for organizations to verify that the products they rely on to improve and monitor application performance are not unwittingly compromising data privacy or PCI DSS compliance.

Organizations should ask the following five questions of any APM vendor. Each question is followed by a response that is specific to AppMon Application Monitoring:

- **Can the solution operate efficiently within network firewalls?** AppMon software is designed to operate within secure firewalled networks in all security zones. By bundling the Agent traffic, AppMon Collectors simplify configuration. You only need to configure the firewalls for the Collectors and not for the individual Agents.
- **Does the solution allow for customization of vendor-supplied system passwords and parameters?** AppMon tools allow for customization of system accounts, passwords, and other security related parameters.
- **Will the solution work with anti-virus software, file integrity checking software, and similar utilities?** You are encouraged to install any software required to achieve compliance with your organization's security regulations. Note that when you contact Customer Support, it may be necessary to temporarily disable such third-party software if it interferes with AppMon operations.

- **Is a mechanism available for maintaining applications and applying patches to keep systems secure?** When software's complexity increases, the chance of bugs quickly increases and some bugs may be exploited. Therefore, it is important that security patches be applied to applications in a timely manner. AppMon Application Monitoring provides fixes, especially security patches, as quickly as possible to its customers with active maintenance contracts.
- **Does the product support audit best practices through console access and a record of configuration changes?**
  AppMon Application Monitoring products record configuration changes to an audit log. However, auditing is a process that needs to be performed by the customer.

Security scanning as part of product development and quality assurance

Before it releases a product, AppMon scans it for security issues proactively. AppMon Application Monitoring software is also checked for PCI readiness. As part of the Quality Assurance development cycle, the Nessus security scanning tool scans check ciphers, SSL, certificates, and non-HTTPs content. Scanning tools like Nessus additionally focus on cross-site scripting and parameter injection vulnerabilities for web applications (not including AppMon Application Monitoring). Many corporations use these tools as part of their PCI DSS compliance-testing process.

In the event of an audit request or if you need more information, contact your AppMon Account Manager or Customer Support.

AppMon Application Monitoring implementation best practices

AppMon Application Monitoring is part of a new generation of lightweight, always-on application monitoring tools that allow organizations to move from reactive crisis mode to proactive management. AppMon Application Monitoring delivers 24/7 visibility into transactions and infrastructure with zero sampling, zero configuration, and less than 2% overhead.

AppMon, with its patented PurePath and PureStack technologies, is a key capability that powers the AppMon product family. To capture performance information, AppMon injects Agents into system components. This functionality includes the ability to record confidential information and other security-relevant data.

It is important that you understand the security concepts integrated into AppMon so that your organization can set up AppMon securely and adhere to regulatory compliance standards.

Before delving further into security details, take a look at a typical web shop implementation of AppMon Application Monitoring, along with its security features and options.

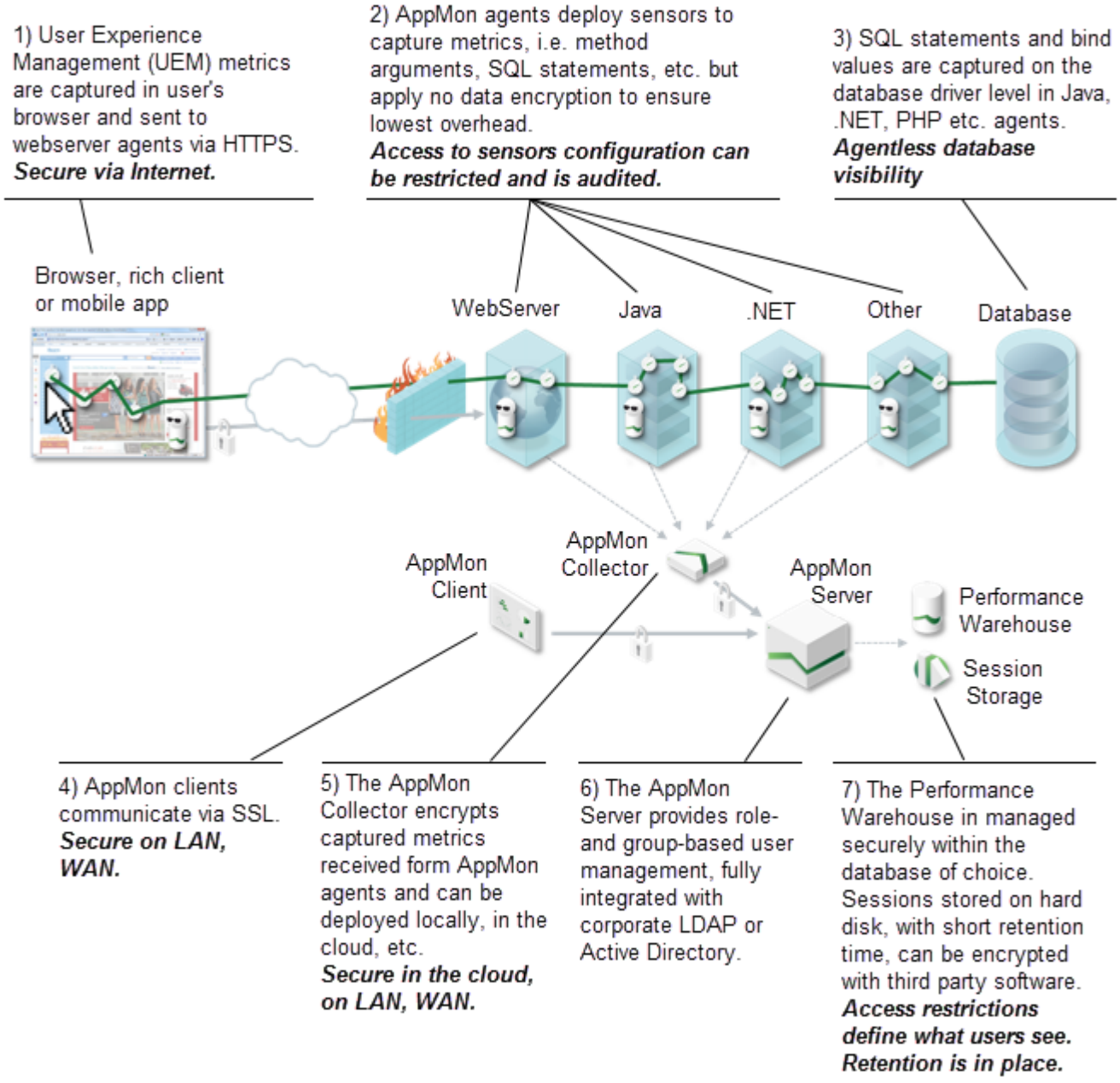


Figure 1. System Overview of Web Application with AppMon

(1)–(3): In a typical distributed environment, user transactions — for example, a purchase in a web shop or an e-banking transaction with a mobile app — are triggered on the client side (in the browser or in the app) and are processed by a number of servers, from front-end through back-end to databases. Most AppMon agents (2) are deployed within the application and its servers, and typically behind a firewall. Only a few (JavaScript and Browser Agents, and Mobile App Agents (1)) run on the client side. All those Agents capture performance metrics; they hook into processes and virtual machines for this purpose. The metrics, which include method arguments, are usually sent to the *Collector* (5), a server in the data center. The communication between Agents and Collector can be encrypted by using SSL. By default it is not encrypted for performance reasons: it is traffic inside the data center, and behind a firewall. The only exception is the Client Agent (1), which does not run inside the data center, but runs on client computers or mobile devices. Client Agents always use encryption to transmit their data securely to their recipients in the data center.

(4)–(7): A Collector (5) encrypts all received information and sends it to the AppMon Server (6), using SSL. The cipher selection during the SSL handshake is dependent on the Java virtual machines on both ends; Oracle and IBM Java implementations are slightly different in their SSL implementations. However, the AppMon Server discards unsecure or broken ciphers; the server's cipher blacklist can be extended by configuration. The AppMon Server stores all transactional data first in the Session Store (7). This Session Store is used for tracking detailed information, including cardholder information, for example for customer complaint resolution or for problem tracking. Typically, the Session Store has a low retention time: its size is configurable and can be adapted to your needs. Performance metrics are displayed to the user by the AppMon Client (4). The communication between Server and Client can be encrypted via SSL.

Measures are recorded in the Performance Warehouse for a long period of time, allowing comparison of current performance with the years before. However, neither cardholder information nor any personally identifiable information is stored in the Performance Warehouse. The only exception to this is if a credit card number is used as a measure splitting ID; this practice needs to be avoided except for diagnosis in extreme cases. This unusual and rare usage scenario requires manual configuration, and by default does not occur.

Access to the AppMon Server is restricted to defined user accounts. Typically a corporate LDAP or Active Directory is used to perform user authentication and to provide access control rights.

How AppMon meets PCI DSS requirements

You can configure AppMon to capture cardholder data to provide richer performance insights (for example, with method arguments, SQL statements, bind values, exception arguments, etc.). When required, you can configure AppMon to prevent the capture of personally identifiable information, such as cardholder information.

AppMon implements strong security controls to meet the following PCI requirements:

- Control of the capture and protection of confidential data.
- Assurance that security measures are in place.

Best practices for governing confidential data

PCI DSS requires strong protection of credit card data. As soon as a credit card number (Primary Account Number, or PAN) is stored, processed, or transmitted, PCI DSS requirements must be fulfilled. Other credit card data such as cardholder name, expiration date, and service code must also be protected. PCI DSS also demands protection of sensitive authentication data. This includes magnetic strip data (which is not applicable for AppMon), card verification codes and values, passwords, and PINs. Sensitive authentication data must not be stored, except for support issues where there is a strong business justification for it.

Prevent the capture of confidential data (PCI DSS Requirement #3)

The easiest way to comply with the PCI DSS requirements for protecting cardholder data is to prevent its capture in the first place. Authentication information should not be recorded at all.



Use the sensor configuration capabilities of AppMon to prevent the capture of cardholder data. Specifically, prepare the checkout process and instrument-critical methods that contain cardholder information. For example, don’t capture method parameters.

Data can also be collected via memory dumps. AppMon collects memory dumps for diagnostic purposes when a Java virtual machine or .NET instance runs out of memory. Such memory dumps can contain confidential strings.



You can turn off the automatic generation of memory dumps. Additionally, you can use permissions to restrict the manual generation of memory dumps. Permissions for analyzing memory dumps can also be restricted.

Note that it is not always possible to prevent the capture of cardholder data. Issue tracking is one exception, for example. Cardholder data must be protected in such cases, as discussed in the following sections.

Secure data storage (PCI DSS Requirement #3)

The AppMon Server, which in production environments is always located behind a firewall, stores and processes cyclic measures and PurePath metrics data in a Session Store. Session Stores contain a number of bulk files that are kept in plain format for performance reasons. *Plain* means that AppMon does not encrypt the data. However, Session Store information has a short retention time based on the available storage and configuration.



Operating system permissions deployed on the Session Store, combined with physical access controls, are the minimum requirements for protecting cardholder information from direct access. Additionally, you can use external tools to encrypt the Session Store. Full disk encryption is recommended because it normally does not interfere with performance. You can use dm-crypt on Linux servers or BitLocker on Windows servers for the encryption of data partitions. Self-encrypting hard drives are also an option. You should configure the size of your Session Store reasonably so that it does not keep data longer than is necessary.

It is not easy to retrieve cardholder information directly from a Session Store. Session Store files are highly compressed and the various fields for cardholder information cannot be easily linked together by an attacker because they are strewn amidst millions of data points. The risk that an attacker will be able to retrieve complete credit card data is fairly low. An attacker would need to either copy huge data files or use an analysis tool (for example, the AppMon Viewer) to distill credit card data from the large amount of metrics data. Both cases normally can be prevented, or at least detected, with secure system design.

The Performance Warehouse, which stores performance metrics over a longer period of time in a relational database, is less likely to hold confidential data. PurePath information that contains confidential strings is usually stripped out. Access controls and encryption may also be applied to the Performance Warehouse database system.



Business transactions can be defined in AppMon to capture a variety of data and parameters, including parameters that hold confidential values. For example, you can intentionally generate a business transaction that stores cardholder data. Measures to avoid such situations include:

- Implementing permissions that restrict the rights for creation of business transactions.
- Regular monitoring of all business transactions and deletion of business transactions that can contain cardholder information.
- Hiding confidential strings.

Secure data transmission (PCI DSS Requirement #4)

Most AppMon Agents are located inside the application infrastructure, either on the web or on application servers. For performance reasons, the communication between the Agent and the Collector is in plain format. For the same performance reasons, the Collector should be located in the same network and behind a firewall.



Because data transmission between Agents and Collectors is not encrypted by default, you must either restrict physical access on the network or put a Collector on the same machine as the Agent. If the AppMon Server is located outside the application infrastructure, place Collectors in the local network and enforce SSL encryption between Collectors and the AppMon Server. You can also use encrypted SSL connection between Agents and Collectors.

AppMon Agents that are located on the client side always encrypt their communication to either their Web Server Agent counterpart or to the next Collector.

Best practices for ensuring security measures are in place

PCI DSS requires reliable identification of users, working with permissions and access controls to authorize users and track their activities. This is necessary to implement the policies that restrict access to cardholder data.

Restrict access on a “need-to-know” basis (PCI DSS Requirement #7)

Operators can use the AppMon Client to inspect recorded metrics and thus, potentially access confidential information such as cardholder data.

Best practice is to restrict the access to confidential information. Configure AppMon with access permissions, so only people who need to know are able to inspect confidential strings (method arguments, SQL statements). All other users see this information blanked out.

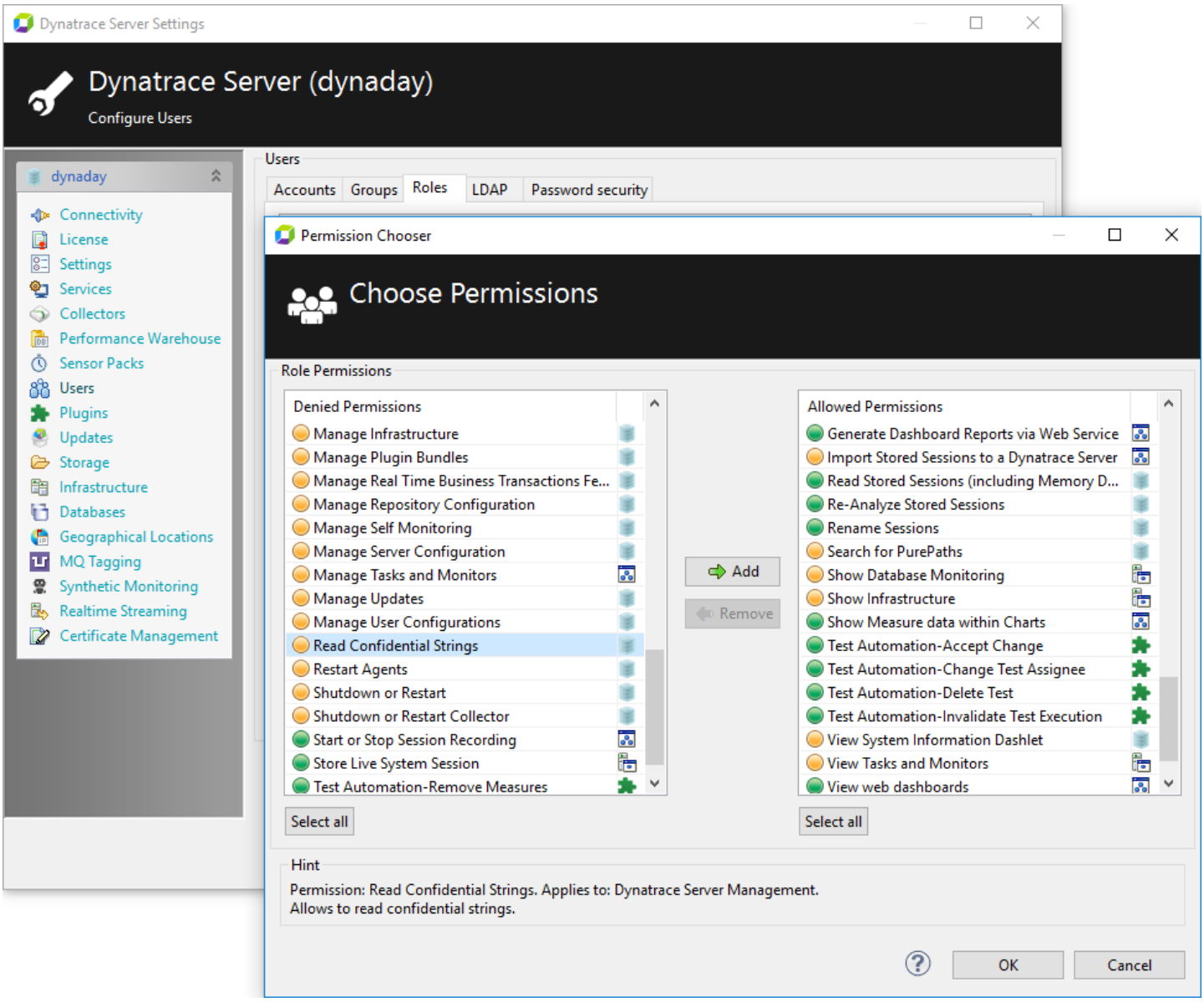


Figure 2. AppMon Server Settings -Permission to Read Confidential Strings

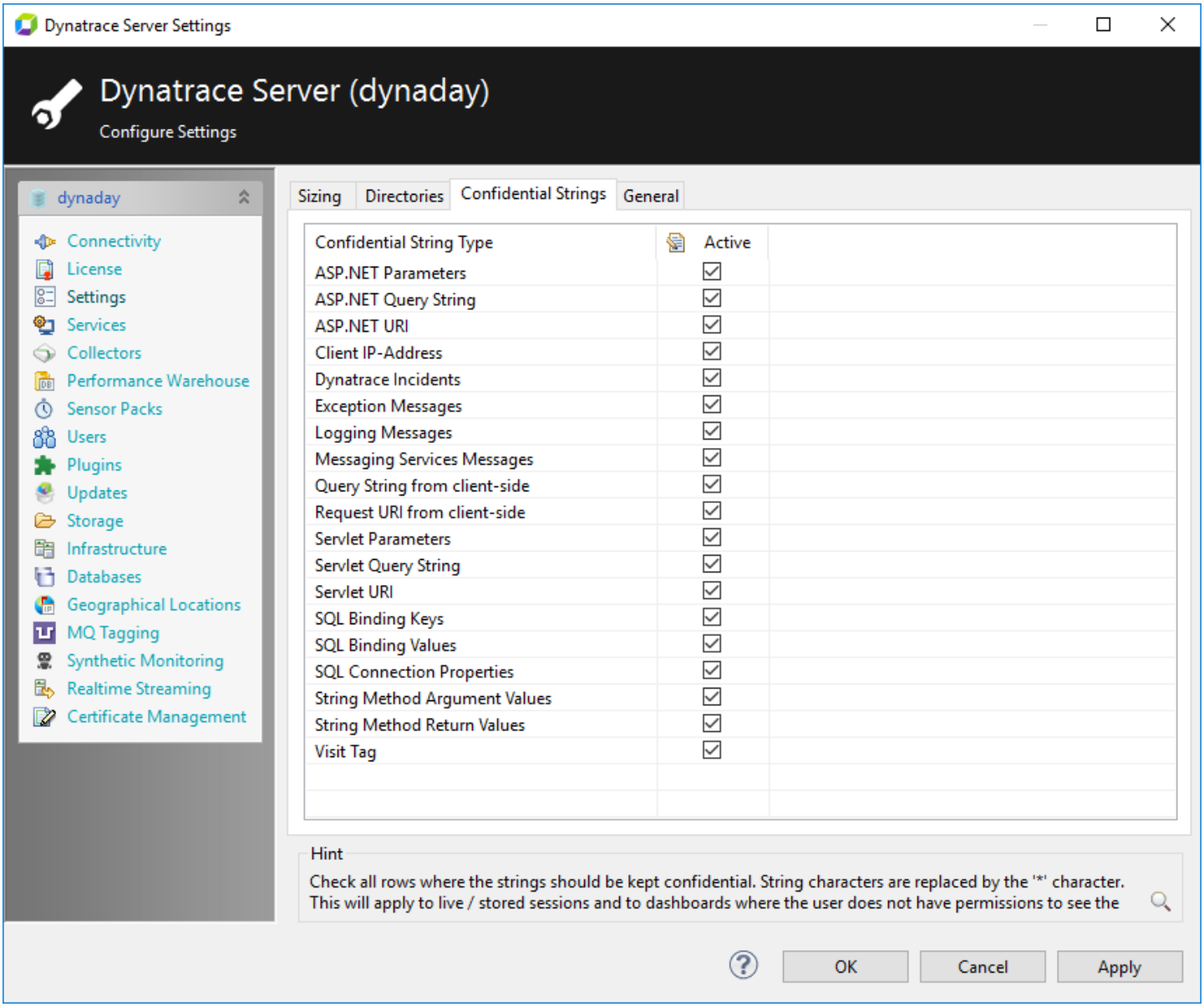


Figure 3. AppMon Server Settings - Confidential Strings: Selecting What Gets Hidden

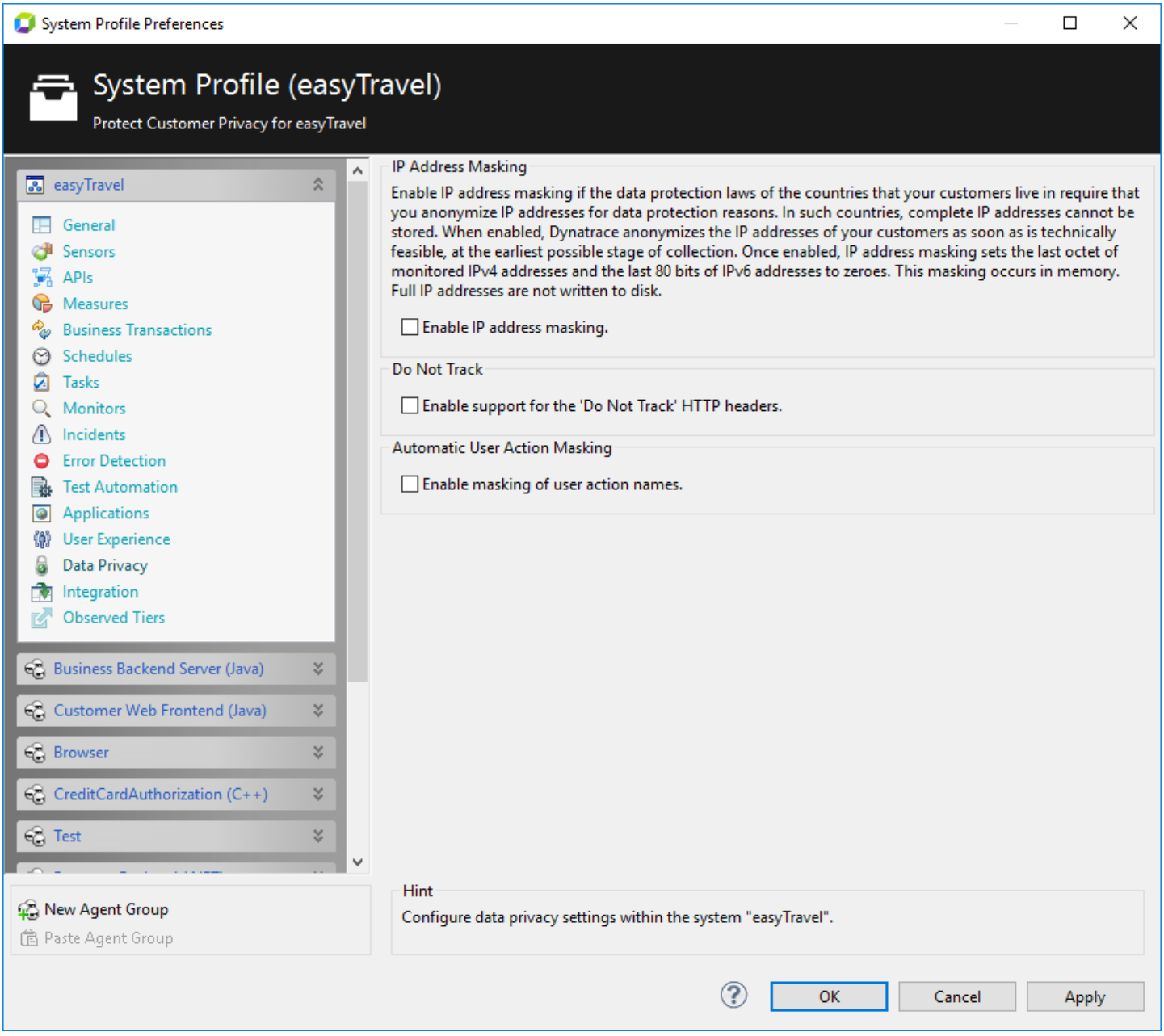


Figure 4. System Profile Preferences - Data Privacy Settings

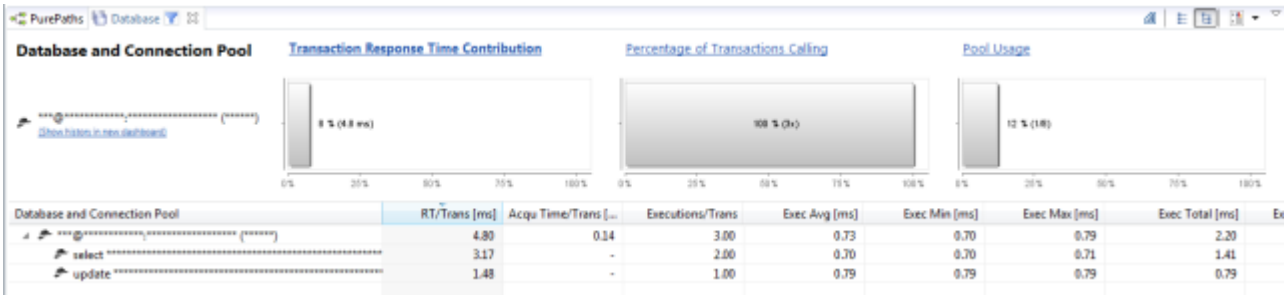


Figure 5. Confidential Strings in SQL Statements

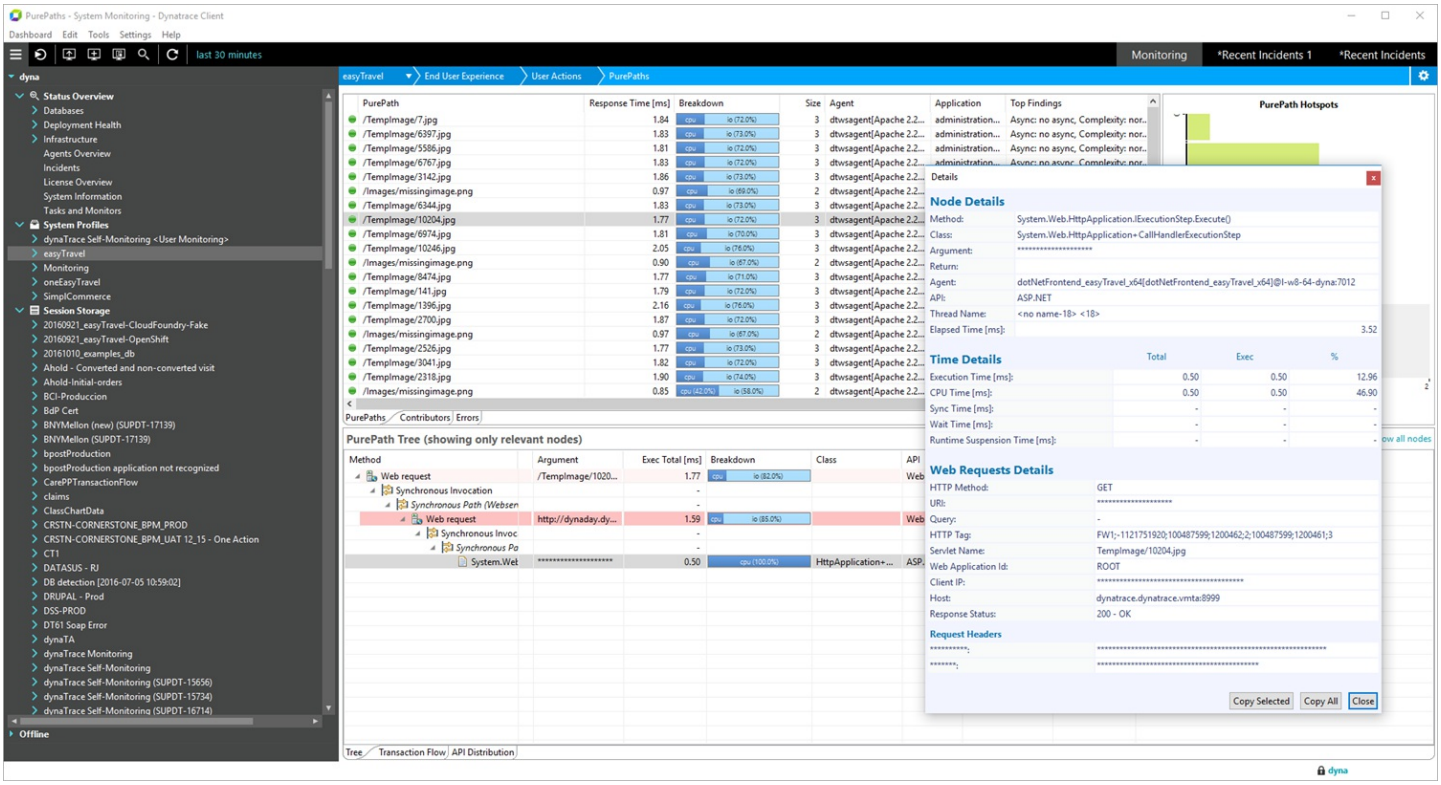


Figure 6. Confidential Strings in a PurePath



Restrict export control. When you export a session (for example, to allow a developer to review an erroneous session), blank out all parameters. If it is necessary to see the method arguments, you can use a screen sharing or web conference tool to give screen access and record the session for auditing purposes. This recommendation meets PCI DSS requirement #10: track and monitor all access to network resources and cardholder data.

Restrict permissions (PCI DSS Requirement #7)

In addition to access controls, constitutes PCI DSS requirement #7 concerns permissions that control who is authorized to define which users are able to access cardholder data and also who can change configuration settings. Remember that the most effective way to restrict access to credit card data is not to capture it at all. If you don’t capture sensitive data, then the combination of access controls and permissions can prevent ad-hoc changes to data capture policy.

Enforce User identification (PCI DSS Requirement #8).

To comply with PCI DSS requirements for reliable user identification, a best practice is to reuse an existing LDAP or Active Directory for AppMon user identification.



Integrate AppMon with an LDAP or Active Directory. Specify an LDAP group to which AppMon users must belong. Activate SSL encryption for LDAP authentication.

Enable Log Audits (PCI DSS Requirement #10)

You can log changes to AppMon System Profiles for auditing. Therefore it is impossible for users to change configuration or permission settings without leaving traces of the changes.

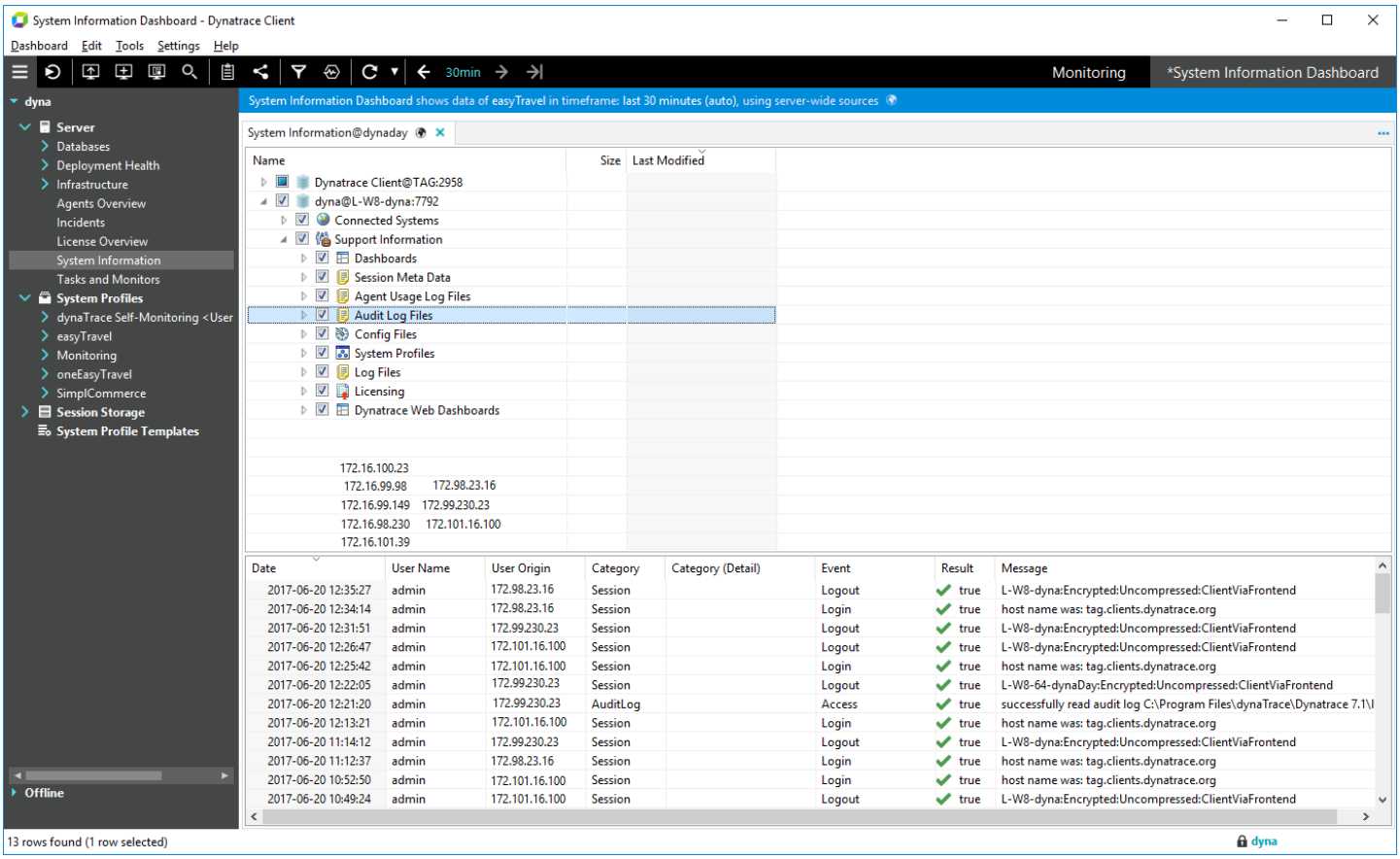


Figure 7. Auditing Logs

Note that it is not possible to log which users had access to sensitive cardholder data.



If cardholder data capturing is required, we recommend that you use permissions to strictly limit access to method parameters and SQL statements.

PCI DSS conclusion

When you use proper configuration and follow all these recommendations carefully, AppMon Application Monitoring and User Experience complies with the PCI DSS requirements and you can still manage the performance of your business-critical applications.

User experience monitoring (UEM) security and privacy

Real User Monitoring (RUM), also known as *User Experience Management (UEM)*, measures how fast an application performs from the perspective of the end user. The metrics for these measurements must be taken on the user's side in the browser or in the app, because the user experience data must include the browser's rendering time, the time it takes to load objects from Content Delivery Networks (CDN), and other variables outside of direct application control.

Page actions are client-side actions performed by the user that trigger one or more web requests, such as AJAX requests. A page action starts with a user interaction such as clicking a link or entering a URL into the browser's address bar, and ends when all the web requests and render operations caused by the interaction are complete. To measure page actions, AppMon injects a JavaScript Agent into the HTML code using a simple script tag that is no longer than a single line of code. The purpose of this small JavaScript is to collect information about the application and actions within the user's browser, and then to report the information. By using the XML HTTP request XHR, AppMon is the only solution in the APM market that returns metrics securely to its originating server, not to a third-party server in the Cloud (for example, Google). Gathered metrics are used by the UEM correlation engine to identify page actions.

When AppMon measures performance for the end user, it needs to collect data on the client side and report it back to the server side. This means that some of the PurePath metrics are captured outside the server infrastructure, outside the data center. Is this approach secure? Can this approach be used to inject something fraudulent into the server? What about cookies and privacy? Answers to these questions are discussed below.

Cross-site scripting

One possible attack scenario is to send fake information to Web Server Agents in the hope that the information triggers a failure and thereby allows for unauthorized code injection.

The chance that such an attack could be successful is low, because a Web Server Agent checks and validates all incoming data. If incoming data does not match reference data (for example, time stamps, request IDs, and cookie information), the data is immediately discarded.

Even if an attacker overcomes this obstacle, the risk of unauthorized code injection still remains extremely low:

- An attacker probably would be able to manipulate the APM data. This could result in a smaller or higher load, but the manipulation would likely be detected and would not do any serious harm. Such an attack would seem to offer little benefit.
- A SQL injection is not possible. The Web Server Agents don't access databases. Instead, they transmit received data via an AppMon Collector to an AppMon Server instance. The Server itself does not write the received information to a database. It writes the data to the Session Store, into a consecutive file.
- Code injection is unlikely. The Web Server Agent is designed for performance and security, and therefore it is very lean. It only checks the parameters and then hands them over to the Collector instance. Additionally, AppMon does automatic unit tests to verify that such a situation does not happen.

Another potential attack scenario is through JavaScript. However, although AppMon does inject JavaScript, such an approach cannot be used for a cross-site scripting attack. For instance, an attacker who wanted to replace the JavaScript in the HTML code would need to attack the web server directly: It makes no difference if AppMon is installed.

User experience monitoring (UEM) and security

Cookies

AppMon UEM requires cookies to identify sessions. Therefore it is a technical necessity that a web server agent set a temporary session cookie; otherwise UEM cannot measure a user's performance experience. This cookie is sent on the first web request and expires when the browser is closed. It is non-persistent and is not used for ad tracking. AppMon sets these first-party cookies for the sole purpose of application performance management, not for tracking a user's purchasing bias or click behavior.

Regulations like EU Directive 2002-58, which is about the processing of user data and protection of privacy, allow for first-party cookies. *First-party cookies* are defined as cookies that are directly used for communication between a website and a user. For example, if you purchase goods in a web shop, the shop uses first-party cookies to identify you and keep track of your shopping cart; you have a direct business relationship with the web shop. The vendor needs to know who you are, otherwise you can't pay and the vendor can't deliver your purchase to you.

In contrast, third-party cookies are cookies that communicate with other websites, for example advertising services or analytics engines. If a website asked to share your address data with another company, you might not agree to the transaction. Regulations therefore rely on the opt-in concept, in which a user has to explicitly authorize the sharing of their data with a third-party.

Unlike other APM solutions, AppMon is the only compliant vendor that uses first-party cookies. Real-user metrics are returned securely to their originating URL (the web server behind the URL) and not to a different server in the Cloud that is not under the vendor's control. The session identification via the first-party cookie is standard technical and business practice. Other APM vendors, which send metrics to a server in the Cloud, create third-party cookies that must require user opt-in to enforce EU directives.

User tracking

Modern browsers support the "do not track" feature, which is a technology that enables end users to opt out of tracking by websites. Although accepting a user's tracking opt-out setting is not activated by default, you can configure AppMon to accept the opt-out. In such instances, AppMon UEM respects the user's privacy, does not set a cookie, and does not measure that user's performance experience.



The "do not track" feature can be activated through **System Profile > Data Privacy**.

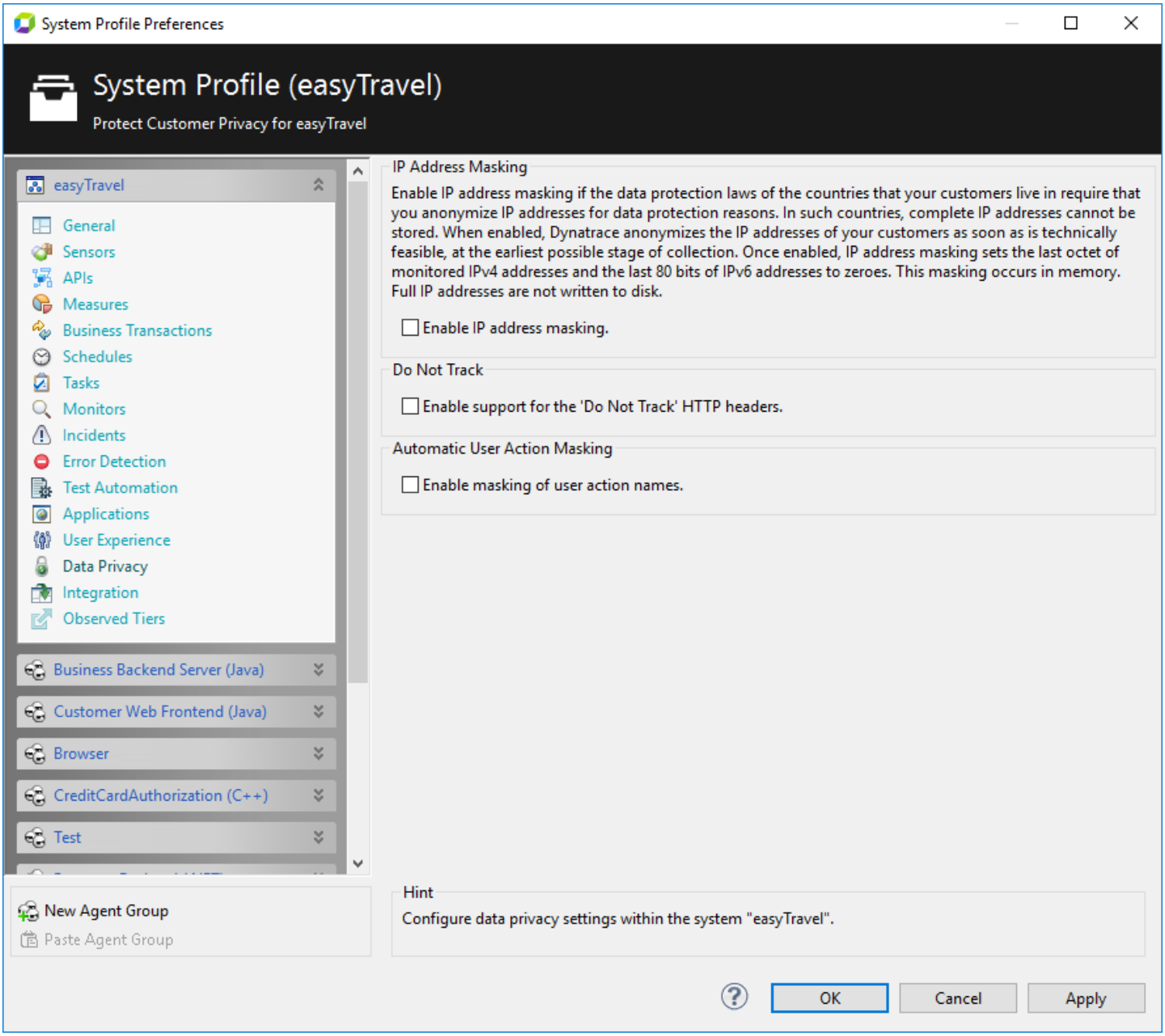


Figure 8. Do Not Track Privacy Setting

Identifying users

Although cookies are used to identify all web requests of a user, as required for the UEM correlation engine, the user remains anonymous by default. Out of the box, AppMon captures the user anonymously, based on the user's IP information and device type.

To capture additional information, you must configure AppMon using the visit tagging features. For example, if a user logs on to your web shop, you can identify the user by account data and then see what the user does on your web page. This function is helpful for user complaint resolution: If the user has a problem and contacts your support desk, Customer Support can identify the user's session and find out exactly what happened.

You can configure visit tagging through **System Profile > User Experience > <application name> > General**. In the **Tag visits with** field, specify the measure that allows the tagging.

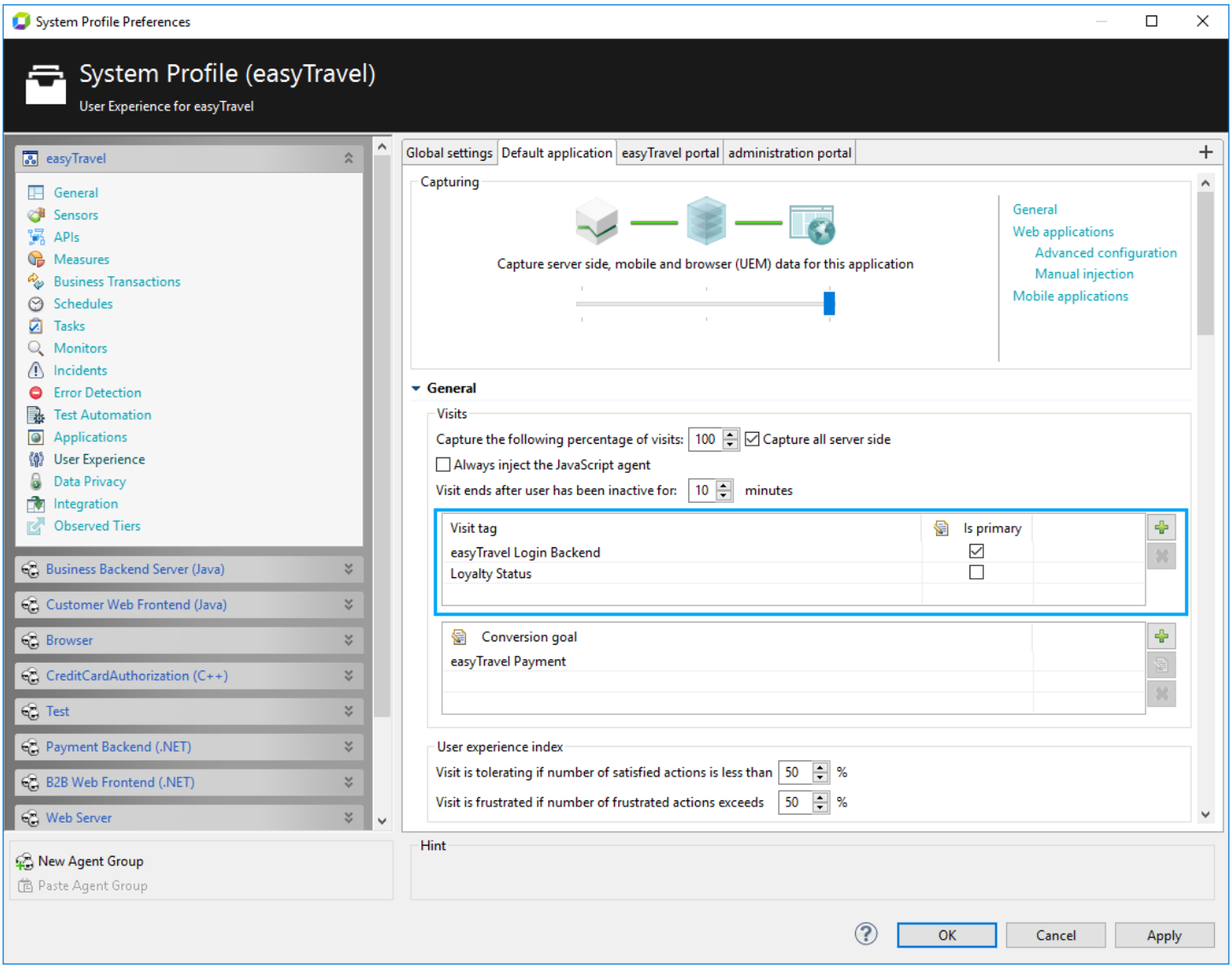


Figure 9. Visit Tagging

Specify how to identify a user, for example the login method and the parameter that contains the user ID.

To obey privacy requests, do not instrument visit tagging or select a parameter that does not allow direct user identification: for example, the department or location instead of the account ID.

Although this feature offers a powerful means of providing high service levels to external visitors, it violates privacy regulations if you configure it for use on your Intranet applications. In European countries, it is illegal to track employees.



To comply with privacy regulations, do not identify internal users directly. Use aggregation information instead; for example, track the department or site that the user comes from.

Privacy conclusions

AppMon UEM's high business value is derived from its ability to measure end users' experience of a website. It provides insight into an individual user's session to see how quickly the web pages render and measures overall performance on the client side. The built-in features of AppMon allow you to configure and operate APM in accordance with privacy regulations.

JMX management

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Configuration of the JMX interface

Server control MBean

System MBeans

Measure MBeans

Use the Java Management Extension (JMX) to manage and monitor Java based applications. AppMon supports JMX, which provides an interface for integration with third-party applications. Through this interface, you can:

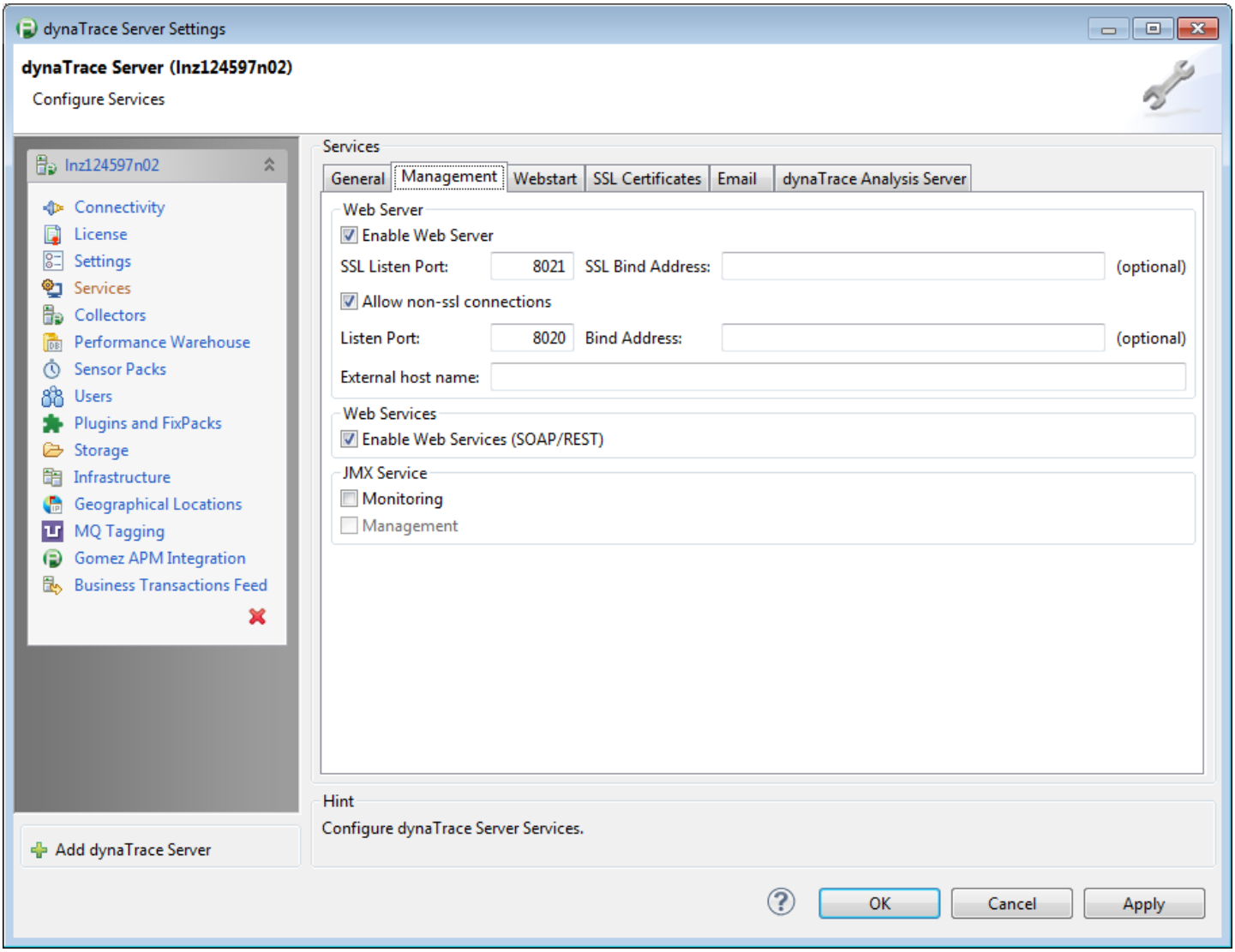
- Retrieve server statistics such as events, PurePaths, and defined profiles.
- Execute base server operations such as starting and stopping the server, dumping memory sessions, and performing Hot Sensor Placement. See Command Line Tool for further information.
- Retrieve data about JVM health: class loading, garbage collection, memory, and other data.
- View the collection of defined realtime metrics, measures, and measurements.
- Query application information such as the latest PurePath, collection of PurePaths, and PurePath nodes.

Configuration of the JMX interface

The JMX interface of AppMon provides a *Monitoring* and a *Management* service. Use monitoring to query measurement data from monitored applications and health data for the AppMon system. Use the Management service to manage the AppMon system via JMX. You can stop or restart the server, clear or store specific sessions, start or stop session recording, and alter the active configuration.

To enable JMX Monitoring and Management:

1. In the AppMon Client, select **Settings > Dynatrace Server** to open the **Dynatrace Server Settings** dialog box, then select **Services**.
2. Select the **Management** tab.
3. In the **JMX Service** area, select **Monitoring** for the Monitoring service, or select both **Monitoring** and **Management** for the Management service. You cannot select **Management** when **Monitoring** is disabled.



AppMon Server Management Settings

> **Warning**
>
> Authentication is **not** required to perform Management operations via JMX. Any user that is enabled to reach the AppMon Server via the JMX service URL can stop or restart the service.

Local access to the JMX interface of the AppMon Server is available whether the JMX Monitoring service is enabled or not, but the AppMon-specific MBeans are only available if **Monitoring** is enabled. Remote access via the JMX service URL is only available if **Monitoring** is enabled:

```
service:jmx:rmi://<dynaTrace server host name>:1099/jndi/rmi://<dynaTrace server host name>:1099/com.dynatrace.diagnostics.management.30
```

Server control MBean

The Server Control MBean represents an AppMon Server and provides current statistical data about the Server and operations. Its `ObjectName` is:

```
com.dynatrace.diagnostics.server:type=ServerControl
```

Attributes

- **CounterSkippedNodesDueToTruncation**: The number of PurePath nodes that were skipped because the PurePath was truncated.
- **CounterSkippedExitEventsDueToMissingEntryEvent**: The number of exit events that were skipped because their entry points were missing.
- **EventInformationBufferSize**: The current buffer size, in bytes, for storing event information objects.
- **EventInformationBufferMaxSize**: The maximum buffer size, in bytes, for storing event information objects.
- **IncompleteNodeCount**: The number of unpaired events that led to incomplete nodes.
- **PurePathBufferMaxSize**: The maximum buffer size, in bytes, for storing PurePaths.
- **PurePathBufferSize**: The current buffer size, in bytes, for storing PurePaths.
- **PurePathNodesCount**: The number of PurePath nodes.
- **TotalNumberOfEventsReceived**: The number of events the server received from the Agents.

Operations

- **shutdownServer**: Shut down the server.
- **restartServer**: Perform a server shutdown followed by a restart.

System MBeans

Each System Profile is represented by a System MBean. The `ObjectName` of those MBeans is:

```
com.dynatrace.diagnostics.system:type=system, systemName=<Name of the System Profile>
```

Attributes

- **ActiveProfileName**: The name of the currently active configuration profile.
- **LatestPath**: Data about the latest completed PurePath.
- **Notifications**: Information about occurred incidents and notifications.
- **PurePaths**: Information about all currently stored PurePaths.
- **ConfigurationNames**: The names of the currently defined configuration profiles.

Operations

- **clear**: Clear the data of the selected session by clearing its PurePaths.
- **store**: Store the data of the selected system to a file.
- **hotSensorPlacement**: Reinstrument the load classes of Agents of the selected system that support hot update.
- **getPurePath**: List the nodes of a PurePath specified by the PurePath's trace tag. For example, `1@http-0.0.0.0-9090-Processor24-GoSpaceFrontend#2`.
- **startSessionRecording**: Start data recording at the selected system. The default name is used for the recorded file.
- **stopSessionRecording**: Stop data recording at the selected system.
- **activateConfiguration**: Activate the provided configuration profile at the selected system.

Measure MBeans

Measures for Realtime Analytics and Charting are specified through the Client's measurement configuration editor. Retrieve these metrics and their values via JMX.

Each measure subscription of each System Profile is represented by a measure MBean that provides current configuration data and measurements of the measure. The `ObjectName` of those MBeans is:

```
com.dynatrace.diagnostics.system.measures:type=<Name of the Metric>, metricGroupId=<Name
```

Configuration attributes

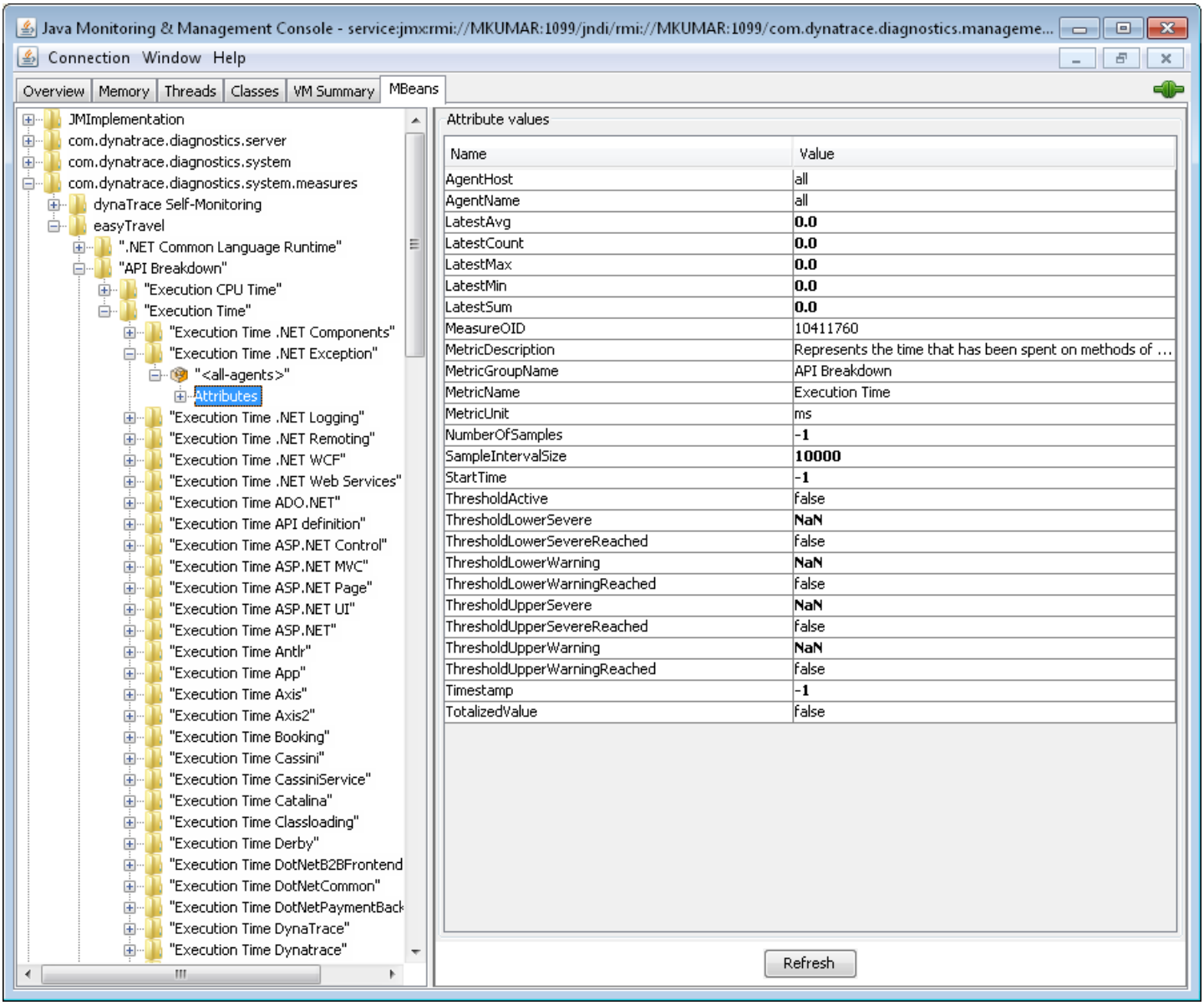
- **ThresholdUpperWarning**: The configured upper **Warning** threshold.
- **ThresholdUpperSevere**: The configured upper **Severe** threshold.
- **ThresholdLowerWarning**: The configured lower **Warning** threshold.
- **ThresholdLowerSevere**: The configured lower **Severe** threshold.
- **ThresholdActive**: Indicates whether at least one of the thresholds is configured.
- **MetricUnit**: The unit of the measure that is displayed in charts.
- **MetricName**: The name of the metric on which the measure is based.
- **MetricGroupName**: The name of the metric group of the metric on which the measure is based.
- **MetricDescription**: A detailed description of what the metric measures.
- **TotalizedValue**: Indicates whether the measure provides cumulative values.
- **SampleIntervalSize**: The default measurement sample interval in milliseconds.

Measurement attributes

- **Timestamp**: A timestamp of the last measurement in milliseconds.
- **ThresholdUpperWarningReached**: Indicates if the upper **Warning** threshold is reached or exceeded.
- **ThresholdUpperSevereReached**: Indicates if the upper **Severe** threshold is reached or exceeded.
- **ThresholdLowerWarningReached** Indicates if the lower **Warning** threshold is reached or exceeded.
- **ThresholdLowerSevereReached**: Indicates if the lower **Severe** threshold is reached or exceeded.
- **StartTime**: The time of the first measurement in milliseconds.
- **NumberOfSamples**: The number of measurements performed up to now.
- **LatestSum**: The summed measurement values of the last sample interval.
- **LatestMin** The lowest measurement value of the last sample interval.
- **LatestMax**: The highest measurement value of the last sample interval.
- **LatestCount**: The number of measurements received during the last sample interval.
- **LatestAvg**: The average measurement value of the last sample interval.
- **AgentName**: The name of the Agent that provided the measurement data.
- **AgentHost**: The host of the Agent that provided the measurement data.

The following figure shows measures in the **Java Management & Monitoring Console**:

```
service:jmx:rmi://<Dynatrace server host name>:1099/jndi/rmi://<Dynatrace server host nam
```



Java Management & Monitoring Console

Command line tool

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Set up communication connections

AppMon offers an easy to use command line interface for basic server operations. You can execute basic operations without an installed client. It's a simpler alternative to JMX Management. The command line tool operates via web services that you enable in the AppMon Server. See Set up Communication Connections for more information.

You must specify the host name and integrated web server port to connect to the AppMon Server. If they are not specified, AppMon uses `localhost:8020`, by default, or port `8021` for a SSL connection. For AppMon 6.3 and later, the default connection type is HTTPS. Use the *nossl* option to connect via HTTP. See General Options for more information.

For AppMon 7.0 and later, you can download the command line tool from the Command Line Tool for Dynatrace AppMon page.

Learn how to use the command line tool.

How to start and stop AppMon services or daemons

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Start and stop instructions

Check if an AppMon Service is running

Start and stop instructions

On all operating systems

Use `<hostname>:8020` in a browser > **Automate & Manage > RESTful Services – for Servers, Collectors, Agents**. For example, http://localhost:8020.

To stop the Server service on this machine, click **Shutdown** in http://localhost:8020/rest/html/management/server/prepareshutdown/. Replace `localhost` with the name of the remote Server to shut down.

Windows

In the AppMon installation directory, run:
`dtserver.exe -service start` or
`dtserver.exe -service stop`.

Alternatively, you can use:

`sc start "dynaTrace Server 7.0"` and
`sc stop "dynaTrace Server 7.0"`

The Dynatrace folder in the Windows start menu has shortcuts for all services installed and running locally. For example, “Stop Dynatrace Server Service.”

In the Windows Services Control Panel, start / stop / restart / change Startup Type for `dynaTrace <service> <version>`.

*NIX

`cd /opt/dynatrace-<major.minorVersion>/init.d` `cd /opt/dynatrace-7.0/init.d` and
`./dynaTraceServer start` as a daemon in the background or
`./dynaTraceServer stop`

> **Tip**
>
> The `dynaTraceServer` script on *NIX handles both the AppMon Server and the AppMon Frontend Server. However, on Windows they are separate services.

Check if an AppMon Service is running

Windows

In the Services Control Panel.

*NIX

Use `ps ax | grep dtserver` (or `dtcollector`, `dtfrontendserver`) to check if an AppMon component is running.

Environment incidents

- Reason: Hypervisor swapping is detected for the AppMon Server's VM.
- How to resolve:
  1. Provide the machine the VMware hypervisor is running on with more memory.
  2. Migrate the VM the AppMon Server is running on to another hypervisor with more memory.
  3. Move other VMs to another hypervisor.
- See the section *Hypervisor Swapping* in http://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/perf-vsphere-memory_management.pdf For further information.

VMware - hypervisor swapping

- Reason: Memory ballooning is detected for the AppMon Server's VM.
- How to resolve:
  1. Provide the machine the VMware hypervisor is running on with more memory.
  2. Migrate the VM the AppMon Server is running on to another hypervisor with more memory.
  3. Move other VMs to another hypervisor.
- For further information, see section *Ballooning* in http://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/perf-vsphere-memory_management.pdf.

VMware - memory ballooning

- Reason: The configured memory limit for the VM the AppMon Server is running is too low.
- How to resolve:
  1. Remove the memory limit for the VM.
  2. Increase the memory limit so it is sufficient.

VMware - memory limit too low

- Reason: There are not enough memory reserved for the AppMon Server's VM.
- How to resolve:
  - Reserve the required memory for the AppMon Server in the vSphere Client.

VMware - reserved memory insufficient

- Reason: No memory is reserved for the for the AppMon Server's VM.

- How to resolve:
  - Reserve the required memory for the AppMon Server in the vSphere Client.

VMware - no memory reserved

- Reason: The configured CPU MHz limit for the VM the AppMon Server is running is too low.
- How to resolve:
  1. Remove the CPU MHz limit for the VM.
  2. Increase the CPU MHz limit so it is sufficient.

VMware - CPU MHz limit too low

- Reason: There are not enough CPU MHz reserved for the AppMon Server's VM.
- How to resolve:
  - Reserve the required CPU MHz for the AppMon Server in the vSphere Client.

VMware - reserved CPU MHz insufficient

- Reason: No CPU MHz are reserved for the AppMon Server's VM.
- How to resolve:
  - Reserve the required CPU MHz for the AppMon Server in the vSphere Client.

VMware - no CPU MHz reserved

VMware specific incidents

- Reason: The power settings of your OS are not configured to provide the maximum CPU performance.

- How to resolve:
  - Windows:
    - Go to *Power Options* and select the *High performance* power plan.
    - Alternatively, go to the advanced settings of your custom power plan and make sure that both the *Minimum processor state* and the *Maximum processor state* are set to 100%.
  - Linux:
    - Set the scaling governors of all your CPU cores to *performance* by using the command: `cpupower frequency-set -g performance`.
      See https://wiki.archlinux.org/index.php/CPU_frequency_scaling#Scaling_governors and https freq/governors.txt for more information.
    - If setting the governor using the *cpupower* command does not work, consult the scaling governor documentation of your Linux distribution.
- Your BIOS and VM hypervisor settings are not considered by the incident. Please make sure that the energy saving mode is disabled for them as well.

Energy saving mode active

- Reason: The CPU frequency of your machine does not meet the requirements of your configured sizing, based on deployment sizing).
- How to resolve:
  1. Provide the AppMon Server with a CPU providing the required frequency.
  2. Switch to a smaller sizing.
  3. If your AppMon Server runs on a VM such as the VMware vSphere Client, configure the VM appropriately.

CPU frequency insufficient

- Reason: The CPU core number of your machine does not meet the requirements of the configured sizing, based on deployment sizing).
- How to resolve:
  1. Provide the AppMon Server with a CPU providing the required core number.
  2. Switch to a smaller sizing.
  3. If your AppMon Server runs on a VM such as the VMware vSphere Client, configure the VM appropriately.

CPU core number insufficient

- Reason: The physical memory of your machine does not meet the requirements of the configured sizing, based on deployment sizing).
- How to resolve:

How to resolve:

1. Provide the AppMon Server with the required physical memory.
2. Switch to a smaller sizing.
3. If your AppMon Server runs on a VM such as the VMware vSphere Client, configure the VM appropriately.

Physical memory insufficient

- Reason: The configured sizing of the frontend and backend server does not match.

- How to resolve:
  - Configure a sizing appropriate to your machine specification in the Sizing Settings dialog box and restart your AppMon Server.

Different sizing on backend and frontend

- Reason: Sizing *unsupported* is configured for the AppMon Server.
- How to resolve:
  - Configure a sizing appropriate to your machine specification in the Sizing Settings dialog box.
- Only use the unsupported sizing under Dynatrace support advisement.

Sizing not supported

General incidents

The environment incidents are raised if we detect a possible issue with the configuration of your AppMon Server or the underlying system.

Application Monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Agents

APIs

Application Monitoring walkthroughs

Baseline and smart alerting explained

Dashboards

Error analysis

How to instrument an application

Incidents and alerting

Monitor specific applications

Problem pattern detection

Reporting

Runtime diagnostics

Sensors

Tasks

Tips and tricks

Business Transactions

Using sessions

Watch and learn

Analyzing non Web Applications

Synthetic and real user monitoring

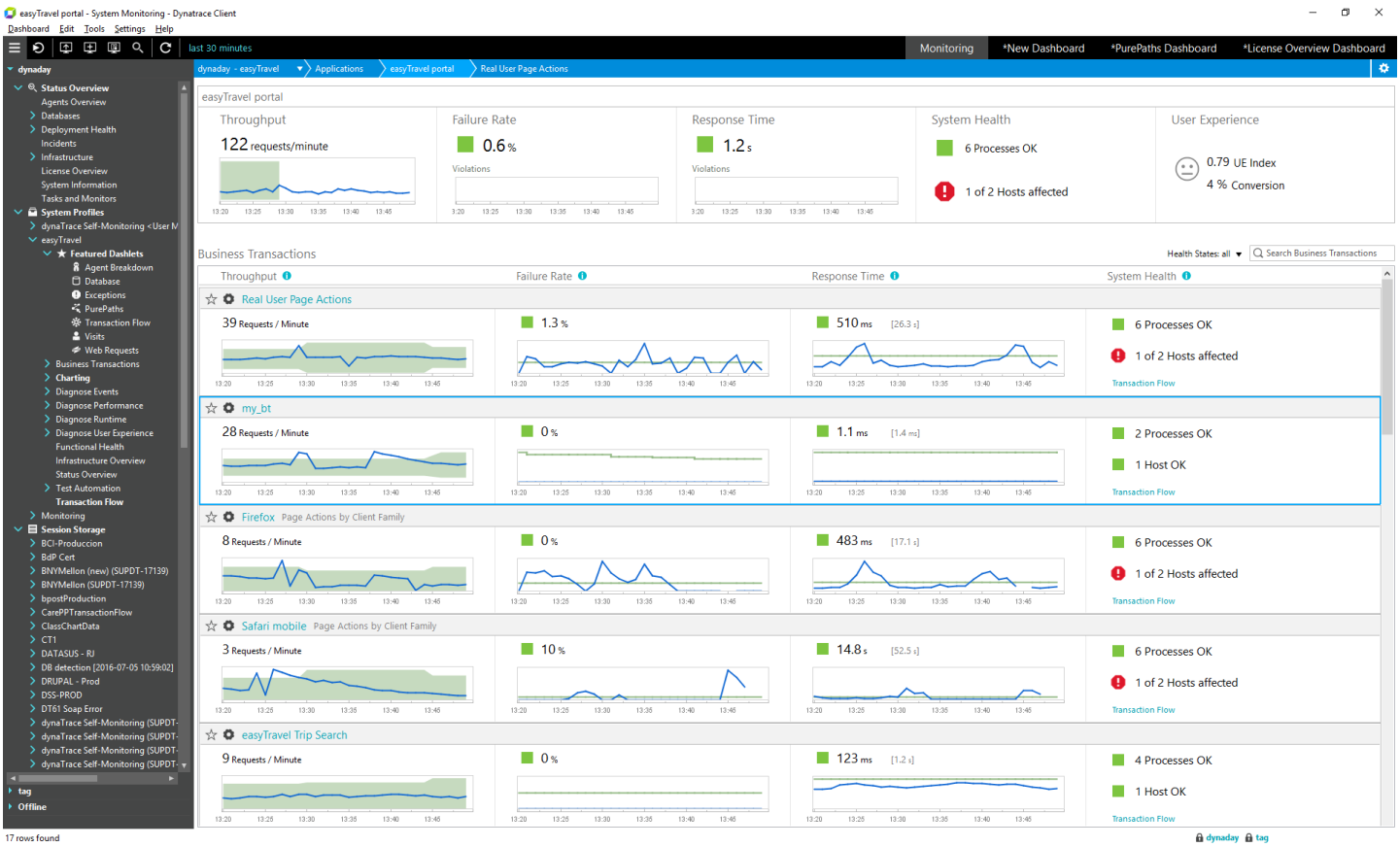
Tomcat performance analysis

This section provides descriptions and usage instructions for the important AppMon concepts and features required to proactively and automatically analyze performance, failure rate, and infrastructure health information for all of your applications and critical business applications, including:

Smart Business Transactions

AppMon extends its Business Transaction capabilities through *Smart Splitting*.

AppMon is able to identify those business transactions that are critical to your end users and business by looking at which transaction patterns are called frequently. Combined with Smart Baselining (see below), this makes it easy to identify the problematic and business-critical business transactions of each application.



Business Transactions

Smart baselining and alerting

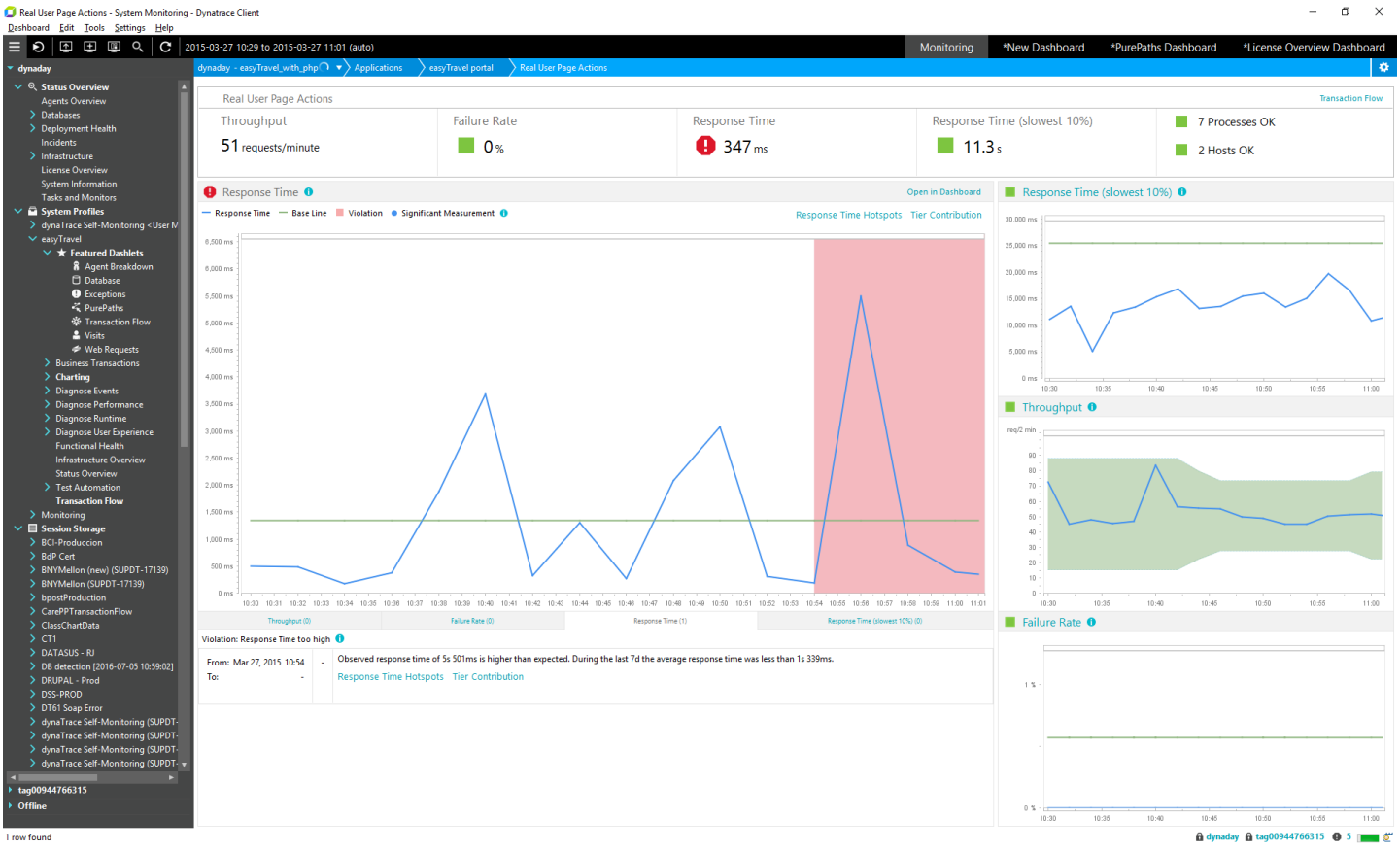
AppMon delivers a baselining approach that automatically detects anomalies and issues alerts reliably.

Typical baselining approaches rely on averages and standard deviations (implicitly assuming standard distributions where there are none) for alerting. Often this results in false alerts.

AppMon uses multiple advanced statistical models that are specific to different types of measurements. For example, the failure rate or the throughput of an application is distributed differently than the response time.

Even in highly dynamic environments, these models result in 90% fewer alerts while still detecting significant anomalies in less than a minute. Smart Baselining reduces “alert spamming” and manual tuning of alerting rules.

The following figure illustrates AppMon analysis derived from Smart Baselining. A message explains the reason for the alert: “Observed response time of 320 ms is higher than expected. During the last 7 days, the average response time was less than 105 ms.”



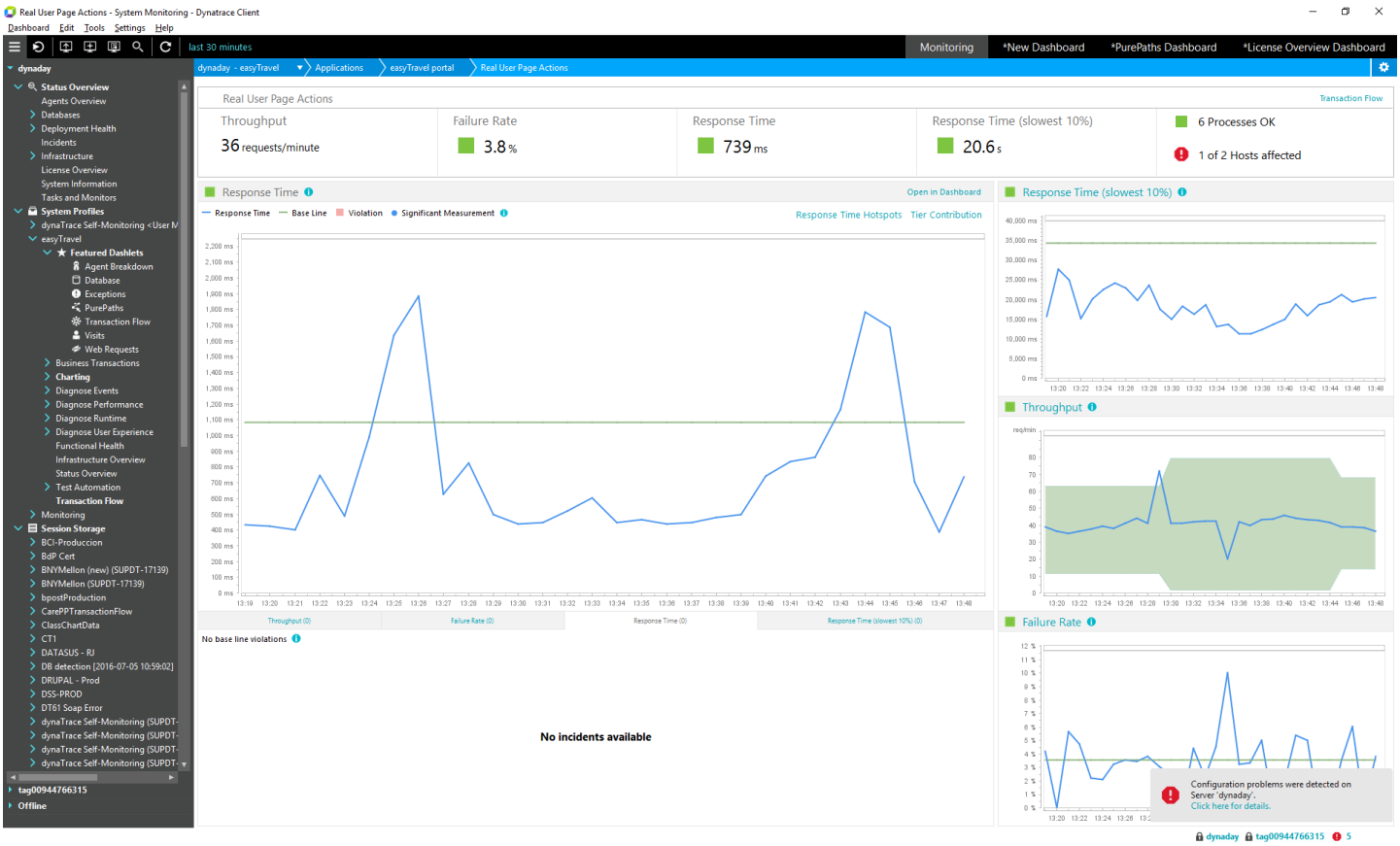
Example Alert

For more information, see Baseline and Smart Alerting Explained.

Integrating host and system health monitoring

AppMon integrates system monitoring with the transactional PurePath-based information, delivering a solution that automatically detects and maps contributing infrastructure components in real time for every user request. This integrated approach allows AppMon to correlate system-level problems with business transactions, and ultimately with the users that are affected. This correlation enables you to pinpoint transactional problems immediately to a specific host or process-level problem, and to see the immediate impact of a host problem on your business-critical functions and end users.

Application Monitoring helps you focus on the right things first when things go wrong.



Integrated Monitoring

Agents

Related pages

Sensors

Sensor Configuration

Sensor properties

Agent Groups

An Agent is a software that is either running on the host or injected into the application process. It gathers performance data and sends them to the AppMon server.

Each Agent specifically designed for a particular technology and/or application layer. Agents aid is to gather data about your application performance, and send it to the AppMon Server for further analysis. AppMon has the following Agents:

- The Java Agent.
- The .NET Agent.
- The Node.js Agent.
- The Apache Agent.
- The IIS Agent.
- The NGINX Agent
- The PHP Agent.
- The JavaScript Agent.
- The zOS Agent.
- The Host Monitoring Agent.
- The Database Agent.
- The OneAgent.
- Custom Agents created using the Agent Development Kit (ADK).

Java Agent

On this page

Sensors

OneAgent

Related pages

Java Agent configuration

Sensors

Sensor Configuration

Sensor properties

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

The Java Agent monitors various Java Runtime Environments. It is loaded by a JVM at startup if configured to do so according the configuration set using a JVM command line option.

Please see Java Agent configuration to learn how to configure the Agent.

Sensors

The Java Agent uses the following sensors:

Category	Sensor	Classic Agent	OneAgent
Client	AWT	✓	✗
Components	EJB Invocation	✓	✗
Database	JDBC	✓	✓
Exceptions	Java Exceptions	✓	✓
Frameworks	Hibernate	✓	✗
	Struts	✓	✗
Logging	Java Logging	✓	✓
Mainframe	CICS/IMS Gateways	✓	✓
Management	JMX MBean Server	✓	✗
Messaging	AMQP	✓	✗
	AMQP Receive Entry Point	✓	✗
	JBoss ESB Tagging	✓	✗
	JMS	✓	✗
	JMS Receive Entry Point	✓	✗

	MQSeries	✓	✗
	MQSeries Receive Entry Point	✓	✗
	Mule Tagging	✓	✗
	Tibco Messaging	✓	✗
Naming services	JNDI	✓	✗
NoSQL	Cassandra	✓	✗
	MongoDB	✓	✗
Threads	Executor Tagging	✓	✓
	Thread Start Tagging	✓	✓
Remoting	Remote Method Invocation (RMI)	✓	✗
	Thrift Tagging	✓	✗
Test Frameworks	Java Tests	✓	✗
Transactions	JTA	✓	✗
	XA	✓	✗
Web Requests	Java Web requests	✓	✗
	Servlets	✓	✓

	User Experience	✓	✗
Web services	Java Web Services	✓	✗

OneAgent

OneAgent released as BETA functionality for AppMon 2017 May. Please be aware that:

- Not all features/sensors available in Classic Agent are already available in the OneAgent for AppMon
- Features/Sensors known from Classic Agent can behave differently

To use OneAgent in AppMon 2017 May you need to install Classic Agents first, and then roll them out to the OneAgent. No additional configuration needed. Before you switch to OneAgent, double check your major requirements with the available feature set.

> **Warning**
>
> You must **not** use the OneAgent in your production environment. The aim of BETA is to test it in pre-production environment, and see how the new Agent affects your custom dashboards/configurations you use to monitor your production environment.

Features

Feature	Classic Agent	OneAgent
Hot Sensor Placement	✓	✗
JMX Measures	✓	✗
Class Browser	✓	✗

Decompilation		
Memory Dump		

AMQP sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

AMQP Receive Entry Point sensor

External links

RabbitMq tutorials

Note

This sensor does not support OneAgent.

AMQP (Advanced Message Queuing Protocol) is a standardized protocol. This link gets an good overview of AMQP. The following AMQP implementations are currently supported by AppMon:

- RabbitMq (Java/.NET)
- Spring-AMQP (with Spring-Rabbit as implementation)

The AMQP Sensor captures data for sending and receiving messages, providing the following information:

Captured Data	Overhead impact	Description
Destination name	low	In case a exchange is used, the destination name has the following format: `Exchange: [exchange name] / RoutingKey: [Routing Key or Que` If no exchange is used, just the Routing Key/Queue Name is set.
Message type	low	Always `AMQP Message`

Transmission type	low	Message sent or received
Message size	low	The size of message in bytes

AMQP Receive Entry Point sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

AMQP sensor

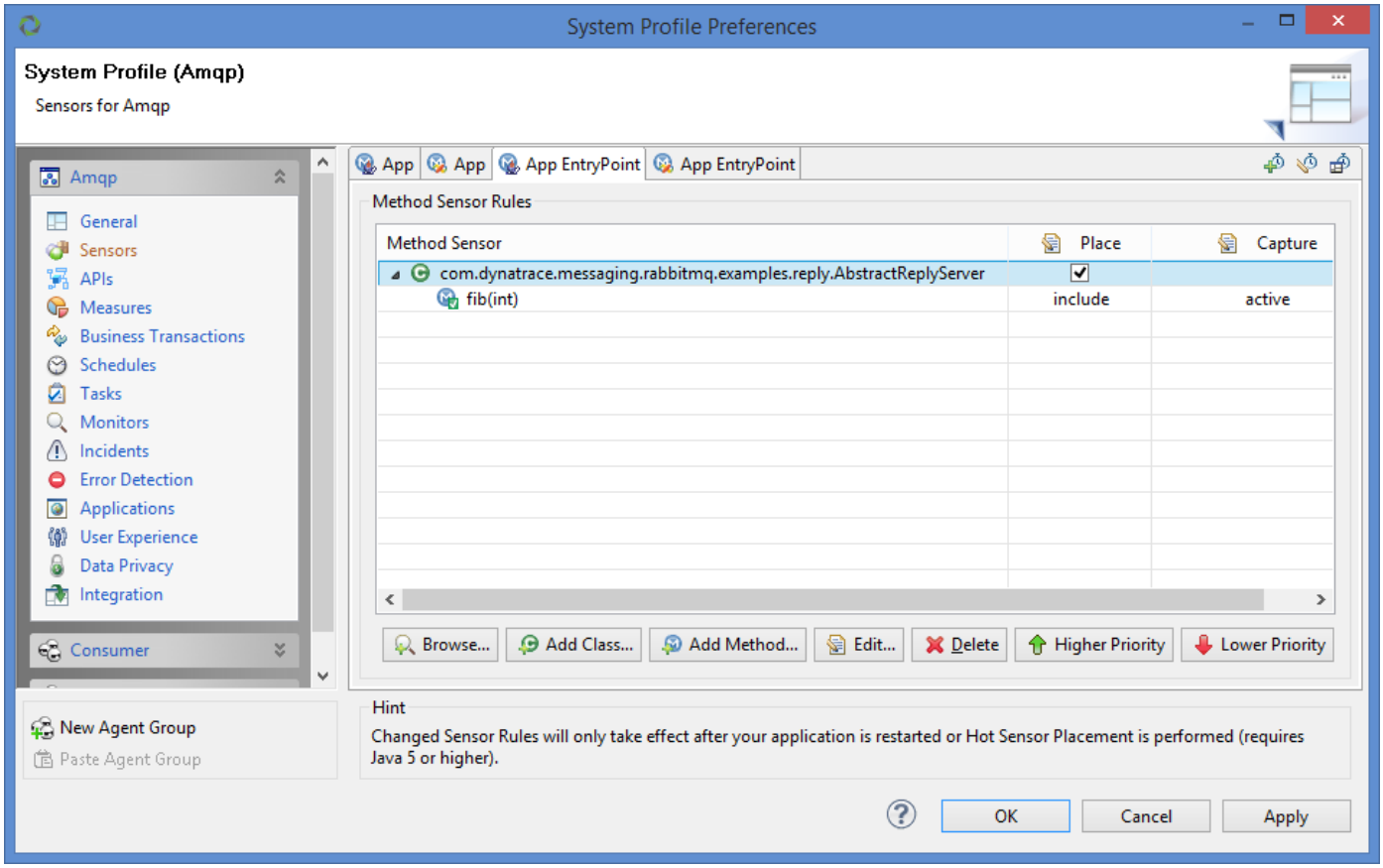
Note

This sensor does not support OneAgent.

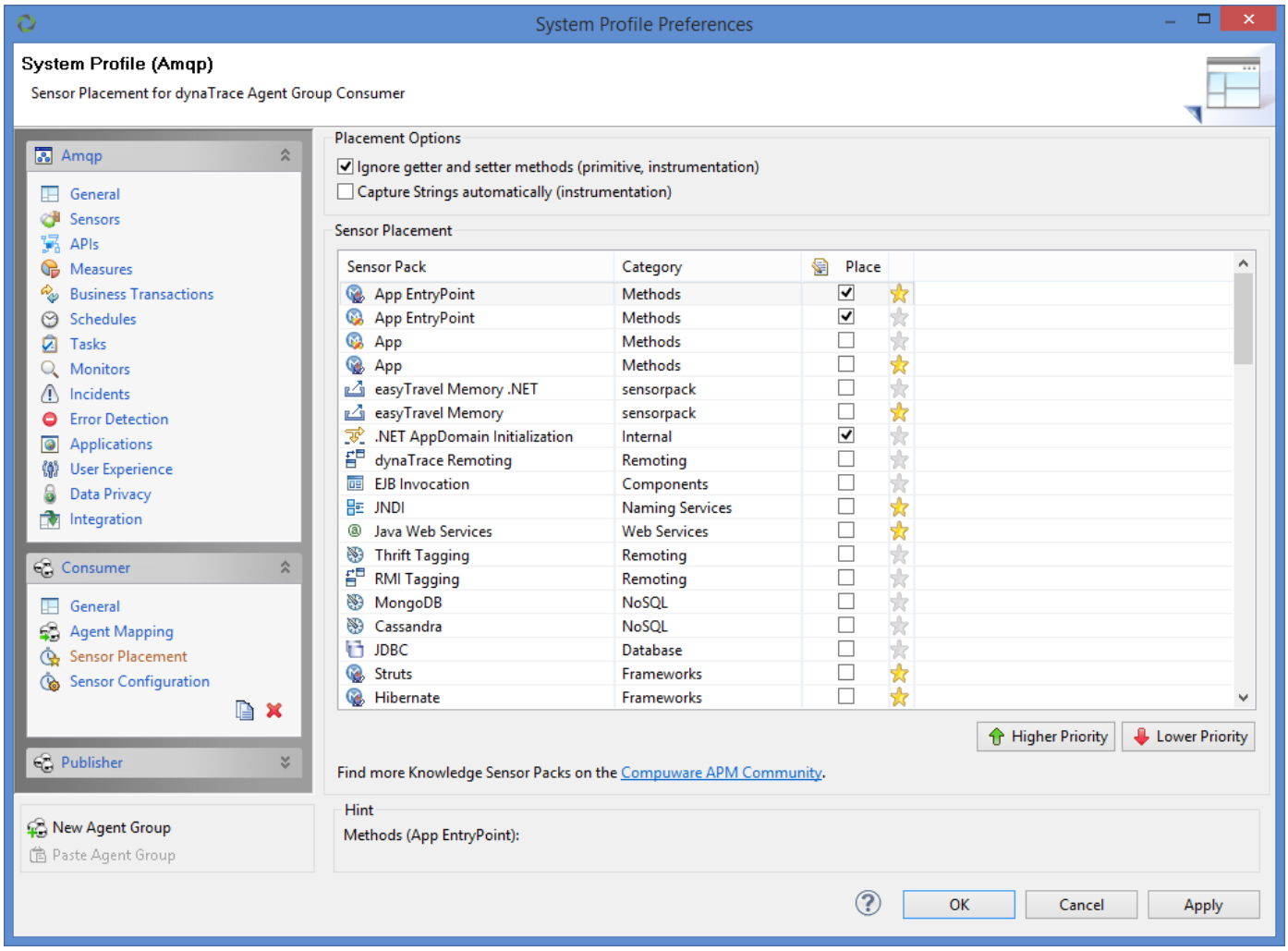
The AMQP Receive Entry Point Sensor enables you to define methods that are responsible for processing AMQP messages after they have been received. These processing methods cannot be recognized automatically, so they have to be specified.

Define a custom method as receive entry point

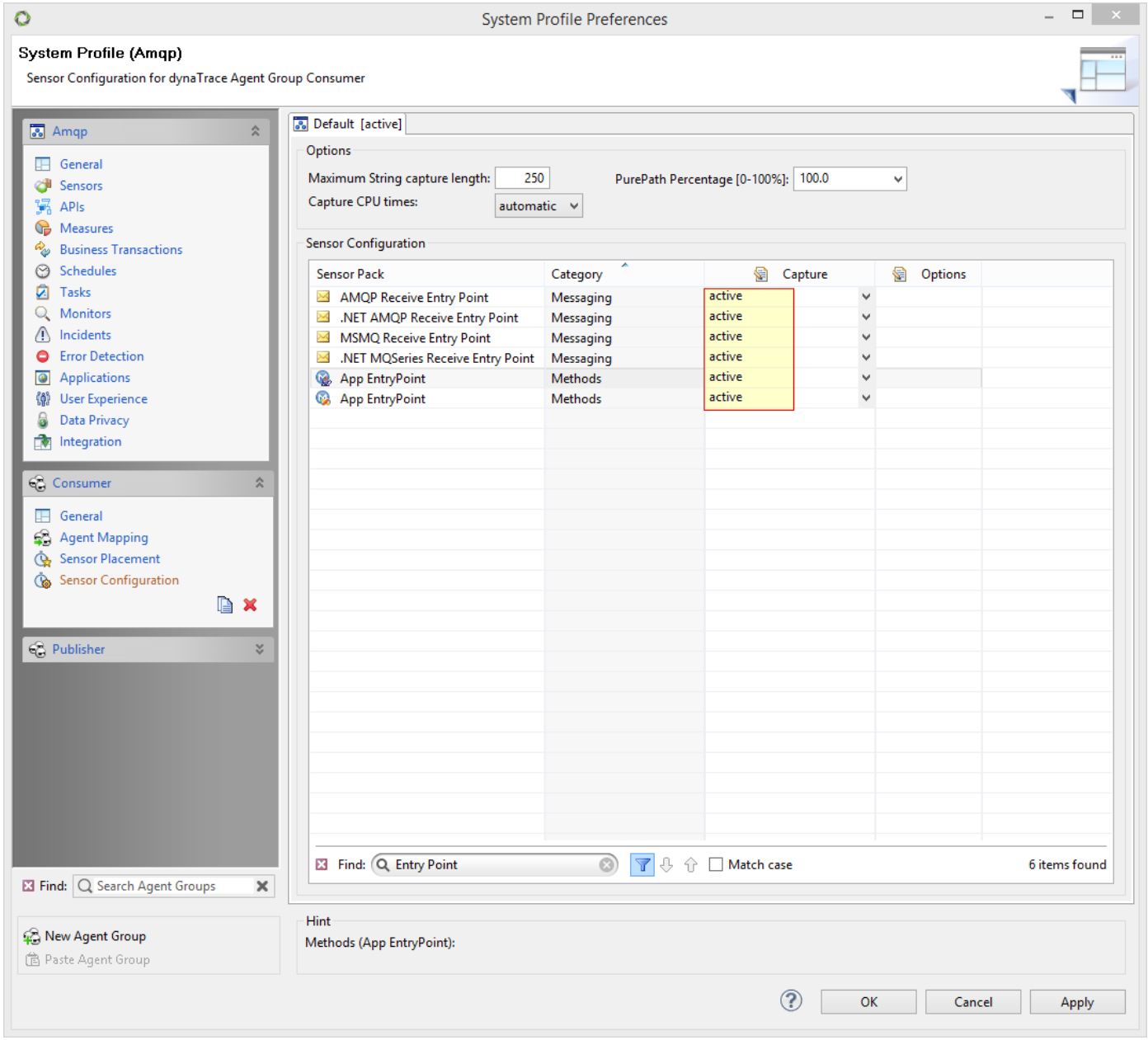
1. Configure a custom sensor rule for the custom method.
   This allows the custom method to appears in the path if a Path is active.



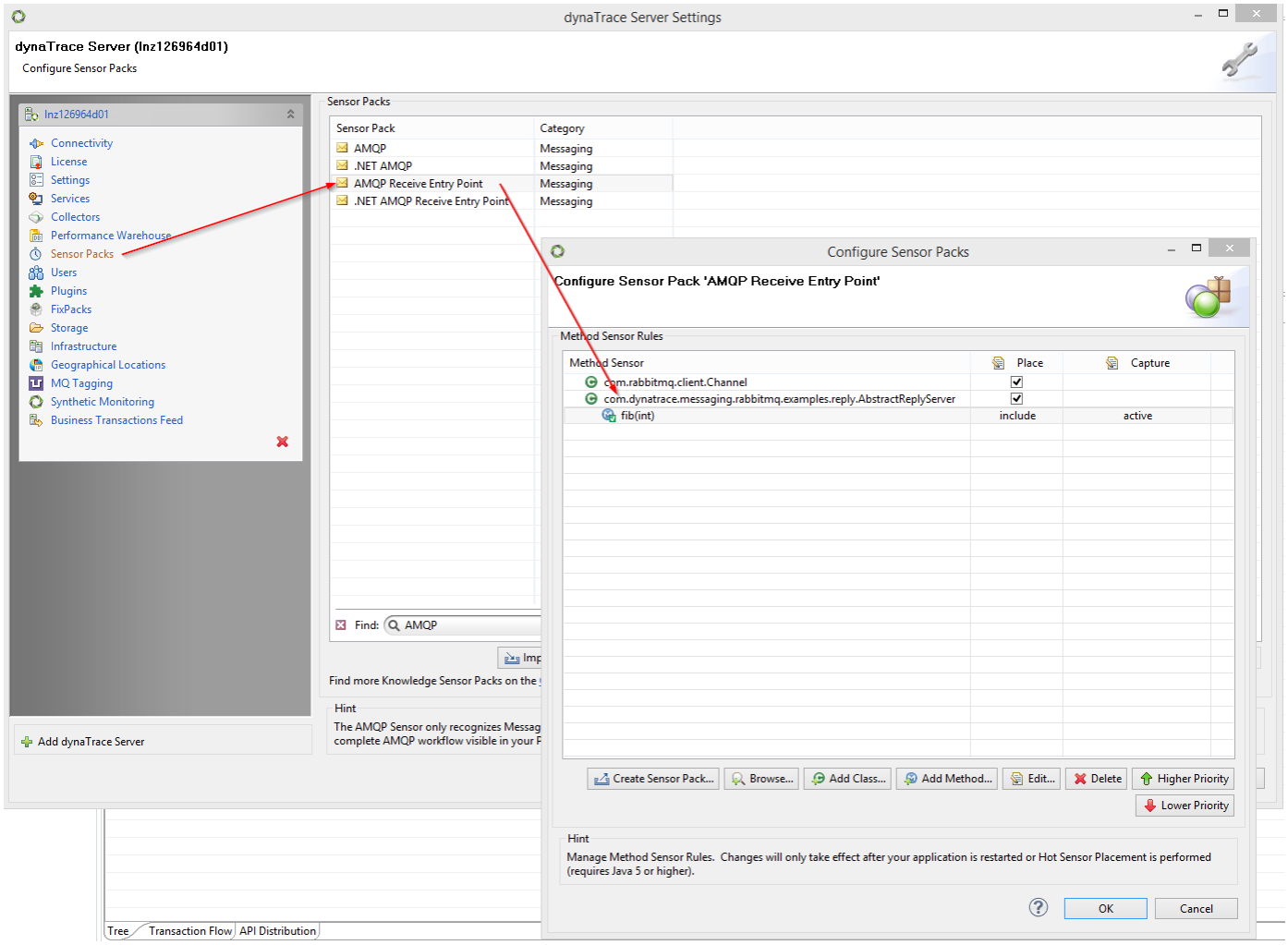
2. Activate the Sensor Group in the correct Agent Group where the method is actually executed.



3. Check that the Sensor Group is activated.



4. Configure the custom method also in the Receive Entry Point Sensor.
   This links the method into the JMS Path.



AWT sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The AWT Sensor Pack traces Abstract Window Toolkit actions. This is especially important for diagnosing Java Applets.

> **Note**
>
> This sensor does not support OneAgent.

Cassandra sensor

Captured Data	Client Overhead	Server Overhead
Connection Pool		
Database host	low	low
Database name	low	low
Database pool size	medium	-
Database user (default)	medium	medium
Database Details		
Row count	-	low
Cassandra-specific		
Consistency level	medium	low
Query string	medium	low

The captured data is retrieved via reflection and strings are cached, so expected overhead is low to medium.

Cassandra CQL binary protocol

Captured Data	Overhead impact
`columnFamily`	low

`consistencyLevel`	low
`count`	low

Reflection is not used by this Sensor and strings are cached, so expected overhead is low.

Thrift Protocol

Captured information

Statement execution data is captured for all CQL3 statements implementing `org.apache.cassandra.cql3.CQLStatement`.

- `STARTUP`
- `CREDENTIALS`
- `OPTIONS`
- `QUERY`
- `PREPARE`
- `EXECUTE`
- `REGISTER`

The CQL binary protocol implementation currently supports all request messages (see CQL binary protocol v1/CQL3 binary protocol v2), which include:

CQL binary protocol

> **Note**
>
> This sensor does not support OneAgent.

EJB Invocation sensor

> **Note**
>
> This may be used to reduce overhead in the SUD.

- **Ignore getter methods**: Suppress placing sensor code in methods starting with *get*.
- **Ignore setter methods**: Suppress placing sensor code in methods starting with *set*.

Sensor properties

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with Java technology > Sensor Configuration** and then select the **EJB Invocation** item in the **Sensor Configuration** list.

Sensor configuration

EJB invocation sensor captures the type and execution time of Enterprise Java Beans.

> **Note**
>
> This sensor does not support OneAgent.

CICS/IMS Gateways sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The CICS/IMS Gateways sensor captures tagging for CICS Transaction Gateway (CTG) or the IMS SOAP Gateway requests.

Executor Tagging sensor

The Executor Tagging Sensor adds the TraceTag to a local field of the job object that is queued within the thread pool. This means that a concurrency problem occurs if the same job object is scheduled twice concurrently.
The Executor Tagging Sensor continues tracing when a job is scheduled for a thread pool. Currently supported thread pools are the pools implementing the Java `Executor` interface, the backport of this interface to java14 (`edu.emory.mathcs.backport.java.util.concurrent.Executor`), and some special thread pool implementations used by WebSphere and WebLogic.

Hibernate sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The Hibernate sensor collects information regarding Hibernate Object Relational Mapping framework data.

> **Note**
>
> This sensor does not support OneAgent.

Java Exception sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Sensor packs

The Java Exception sensor retrieves information about the exception's `Throwable` class, its messages and optionally its stack trace elements. It also retrieves information about exceptions that do not get "thrown" but are created via `new`. The transferred exception event is handled as a single event with no duration.

Sensor Configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with Java technology > Sensor Configuration** and then select the **Exceptions** item in the **Sensor Configuration** list.

Aggregation

Exception aggregation is aimed to reduce the PurePath node count when many exceptions are thrown. The aggregation is done per instrumented node level. The first three exceptions are captured normally, beginning with the 4th exception, aggregation will take effect and only count occurrences of distinct class+message combinations.

Configuration properties

You can define which classes to instrument. There are two possible scenarios:

- Instrument listed classes only.
- Instrument all classes, except for the listed ones.

Select the appropriate option button and list classes to be included/excluded from the instrumentation. You can do this manually or use the Class Browser.

Note that this configuration only applies to `Exception` subclasses, not `Error` subclasses.
For example, a `java.lang.AssertionError` will always show up, even if a rule such as `java.` is active. The `java.lang.NullPointerException` will not show up in the same example, because it will be excluded by the same rule.

Overhead properties

The Java Exception Sensor captures data of the `Throwable` class without using reflection. A delegation suppression mechanism is used for suppressing nested `super()` calls.

In the **Overhead Properties** section in the **Exception Sensor Properties** dialog box, you can enable or disable capture of stack traces for exceptions and fine-tune the amount of data that is captured and transferred from the Agent to the Server.

Overhead Property	Description	Overhead impact	Default value
Capture Stacktrace	Enables capturing of stack traces. Capturing the stack and transferring the data to the server causes high CPU overhead on the instrumented application and consumes network bandwidth.	high	enabled
Maximum Captured Stacktrace depth	Limits the number of elements per stacktrace. Only the topmost elements are transferred. The allowed range is 0—256.	medium	32
Maximum Captured Stacktraces per second	Limits the number of stacktraces captured per second. The allowed range is 0—1024.	high	128

Java Logging sensor

Select the appropriate option button and list log events to be included/excluded from the capture. Select classes manually or using the Class Browser.

- Capture listed log events only.
- Capture all log events, except for the listed ones.

You can define which log events should be captured. There are two possible scenarios:

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with Java technology > Sensor Configuration** and then select the **Java Logging** item in the **Sensor Configuration** list.

Sensor Configuration

The Java Logging Sensor captures data from the logging records. Captured log messages are truncated to the maximum capturing string length. The transferred log event is handled as a single event with no duration.

Java Tests sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The Java Tests sensor captures information from the Java Test Frameworks JUnit and TestNG.

> **Note**
>
> This sensor does not support OneAgent.

Java Web Requests sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

System Requirements

Note

This sensor does not support OneAgent.

The Java Web Requests Sensor allows tracing of PurePaths across thread and process boundaries for HTTP. The Sensor continues tracing when HTTP requests, including redirects, are made. It also traces relevant information about these connections. The sensor captures the URI, query, method, and response code of the HTTP request/response. It causes low overhead impact.

Sensor Configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with Java technology > Sensor Configuration** and then select the **Java Web Requests** item in the **Sensor Configuration** list.

AppMon truncates captured web requests to the specified length. Specify the required length in the edit box. All exceeding characters will be cut.

URI filter

You can also filter captured queries by URI. There are two possible scenarios:

- Capture listed URIs only.

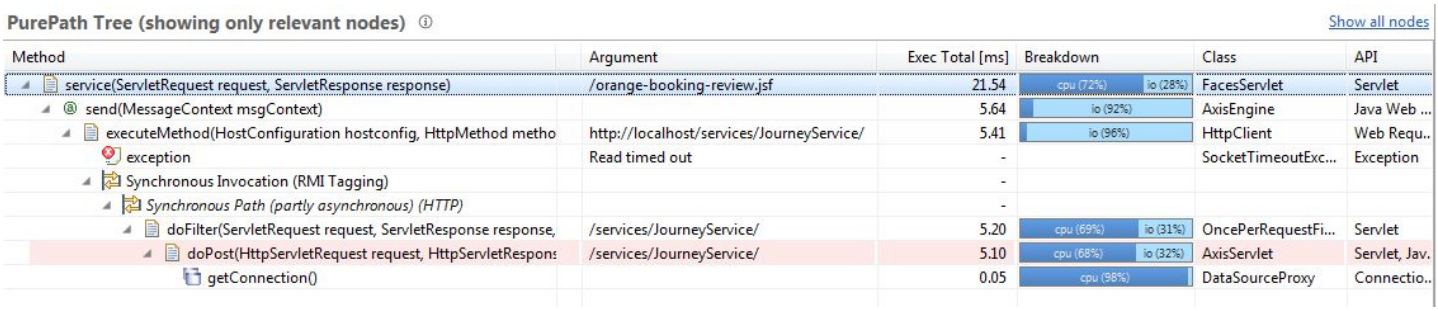
- Capture all URIs, except for the listed ones.

Select the appropriate option button and list URIs to be included/excluded from the capture.

Depending of the size of the configured rules list, the impact of the sensor on Agent overhead may increase.

Example

The following screenshot shows an active Java Web Request, Web Service, and Servlet Sensor capturing an Apache HTTP Client 3.*x* request.



Java Web Service sensor

Note

Properties marked with

are especially relevant for sensor overhead on the system under diagnosis (SUD). Use these properties with caution and check the impact on the SUD after setting these properties.

- **Measure Request/Response Sizes** (

): Capture the sizes of request and response in bytes. You can use this info in charts or other dashboards. Here you can enable or disable measure of the request/respond sizes.

Sensor properties

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with Java technology > Sensor Configuration** and then select the **Java Web Services** item in the **Sensor Configuration** list.

Sensor Configuration

Note

Asynchronous web services are not supported.
The Dispatch API of JAX-WS is not supported.

Captured Data (Client/Server)	Capture place	Overhead
Endpoint/Class	Method entry	low

Namespace/Method	Method entry	low
Bytes sent/received	Method entry and exit	low

The Java Web Service Sensor traces relevant information that is transferred when web services are called. The following table lists the services from which it captures data, and the information it provides about web service calls depending on the side of the call.

Note

This sensor does not support OneAgent.

JBoss ESB Tagging sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The JBoss ESB Tagging sensor traces PurePaths across thread and process boundaries for JBoss Enterprise Service Bus Messages.

> **Note**
>
> This sensor does not support OneAgent.

JDBC sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Sensor configuration

The JDBC Sensor captures information about typical JDBC calls. It provides the following data:

- The SQL string (by default, up to 4096 characters).
- The number of rows returned during the execution of the SQL statement.
- Optionally, the bind values for the SQL statements. If statement aggregation is enabled, capturing of bind values is automatically disabled.

The sensor is not eligible for Hot Sensor Placement.

Sensor configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with Java technology > Sensor Configuration** and then select the **JDBC** item in the **Sensor Configuration** list.

Here you can enable or disable aggregation and bind value capturing. You can also set the limit for SQL command length.

- **No aggregation or bind value capturing**: Each query will be displayed separately. Bind values are not captured.
- **Enable aggregation**: If queries are equal, only *one* DB query statement displays per PurePath parent node, along with the number of executions. This makes PurePaths trees much more readable, but capturing of bind values is disabled. DB connection acquisitions are not aggregated.
- **Enable bind value capturing** (
  
  ): Enables capture of SQL statement's bind values.

- **Maximum size of captured SQL command strings**: Limits the captured SQL statements to the specified length. The default is 512. All symbols, exceeding the limit, are cut.

> **Note**
>
> Properties marked with
> 
> are especially relevant for sensor overhead on the system under diagnosis (SUD). Use these properties with caution and check the impact on the SUD after setting these properties.

The settings are:

Option name	Default	Description
optionJDBCAggregationThresholdJava	5000	Limit of distinct sql statements that will be counted individually (on an instrumented node level). After reaching the threshold, only the count of executions of the overflowing statements is counted on a “[thresholded executions]” summary counter.
optionJDBCAggregationPoolThresholdJava	1000	Same limit, but on a pool level. So, if there are more distinct sqls seen for one distinct connection pool, the overflowing statements are summarized in a “[pool-thresholded executions]” summary counter.
optionJDBCAggregationCacheEnabledJava	true	There is an LRU cache that remembers the last N seen statements, which reduces CPU overhead a lot. However, it can be turned off, for the lowest possible memory footprint of database aggregation.
optionJDBCAggregationCacheMaxEntriesJava	512	The size of the aforementioned LRU cache. Being LRU, it is guaranteed to never exceed this limit.

JMS sensor

Network overhead

Captured message content transfers from the Agent to the Collector and the AppMon Server.
Keep network traffic overhead in mind, especially in production environments.

Here you can enable or disable capturing of the messages content. Please keep in mind, that capturing of the message content can cause much overhead impact, especially for the large messages.

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with Java technology > Sensor Configuration** and then select the **JMS** item in the **Sensor Configuration** list.

Sensor configuration

The sensor transfers the trace tag of the execution path that sends the JMS message to its child execution paths that process this message.

Tagging

See JMS Receive Entry Point Sensor for more on data that can be gained beyond the placement of the JMS Sensor.

Captured Data	Overhead	Description
Destination name	low	
Transmission type (sent or received)	low	
Message type (`TextMessage`, `ByteMessage`, etc.)	low	

Message size (only available for `TextMessage` and `ByteMessage`)	low	
Message contents	high	Network overhead for large messages

The JMS Sensor captures message object as a parameter and also captures the data from there. It provides the following information about the message:

> **Note**
>
> This sensor does not support OneAgent.

JMX MBean Server sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The JMX MBean Server sensor captures JMX based measurements for supported Application Servers.

> **Note**
>
> This sensor does not support OneAgent.

JMS Receive Entry Point sensor

Note

This sensor does not support OneAgent.

The sensor doesn’t have any user configurable properties.

Sensor configuration

The JMS Receive Entry Point Sensor enables you to define methods that are responsible for processing JMS messages.

JNDI sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The JNDI Sensor captures method of Java and RMI Naming Services name lookup. It doesn’t capture additional information and therefore does not introduce any significant overhead.

> **Note**
>
> This sensor does not support OneAgent.

JTA sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The JTA sensor collects information regarding start, commit and rollback of Java Transaction API transactions.

> **Note**
>
> This sensor does not support OneAgent.

MongoDB sensor

In either case, you can view the details of a node in the **Details** popup windows of the **PurePaths** and **Database** dashlets. To display the details in the **Database** dashlet, you have to enable bind values grouping in the appropriate Sensor. The captured data may also be called **Bind Values**.

- **Enable extended capturing**: Enables capturing of detailed information about JSON query strings or document IDs.
- **Capture only document id(s)**: Capture document IDs where available.
- **Capture full JSON objects(s) (queries and documents)**: Capture all JSON objects as serialized strings.
- **Maximum string length**: Limits the maximum length of the captured JSON strings. Exceeding symbols are cut.

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **MongoDB** item in the **Sensor Configuration** list.

Sensor configuration

The MongoDB Sensor captures information about MongoDB calls. See MongoDB for more information about usage of captured data.

> **Note**
>
> This sensor does not support OneAgent.

MQSeries Receive Entry Point sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

WebSphere MQ (MQSeries) sensor

Note

This sensor does not support OneAgent.

For extended functionality beyond the WebSphere MQ (MQSeries) Sensor the MQSeries Receive Entry Point Sensor specifies methods that process the message that has been received by a `get(...)` call. The Sensor identifies the methods that are responsible for processing MQ messages after they have been read by `com.ibm.mq.MQQueue: get(...)` or `com.ibm.mq.MQDestination: get(...)`. It is not possible to automatically recognize the methods responsible for message processing, so these methods have to be specified.

The MQ API offers the `get(...)` calls to receive messages and passes the message using the message object specified in these calls. For this API, there is no wrapping method that does (or at least triggers) all the things that happen on arrival of a message. The message could be processed in the method that has called `receive`, but this calling method again may also return the message, so it is hard to determine where the actual processing of the message happens. The MQSeries Receive Entry Point Sensor provides that information.

Mule Tagging sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The Mule Tagging sensor traces PurePaths across thread and process boundaries for Mule Messages.

> **Note**
>
> This sensor does not support OneAgent.

Remote Method Invocation (RMI) sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The Remote Method Invocation (RMI) sensor collects the time, bytes and objects involved in a Remote Method Invocation (RMI for RMI/JRMP, RMI/t3, RMI/IIOP, ORMI, JBoss Remoting, RMI/P4).

> **Note**
>
> This sensor does not support OneAgent.

Servlet sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Executor Tagging sensor

Sensor properties

External links

Servlet 3.0 Specification

The Servlet Sensor traces relevant information which is transferred over a Java HTTP servlet. It also traces web application name, and the name of the servlet executed for this request.

The Servlet Sensor captures the following info:

Captured Data	Overhead
Request URI	low
Query String	low
Application ID	low
Servlet Name	low
Servlet Attributes	medium
Session Attributes	medium

Request Size in bytes	high
Request Headers	high
Request Parameters	high
Response Size in Bytes	high

Sensor configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with Java technology > Sensor Configuration** and then select the **Servlets** item in the **Sensor Configuration** list.
Here you can enable or disable measure of the request/respond sizes. You can use this info in charts or other dashboards.

Servlet properties

- **Measure Request/Response Sizes** (

): Capture the sizes of request and response in bytes.
- **Always add Dynatrace header to responses**: Force adding the AppMon header to responses, even if it was not available in the request (useful for Load-Test scenarios). You cannot change this value if a DC-RUM appliance is configured. See Integration for details.
- **Capture Details in all Filters and Servlets** (

): If disabled, details like session attributes or request headers are only captured for the first filter/servlet call on a path. If enabled, the details are captured for each filter/servlet call. You can add additional attributes to the initial set of attributes, especially in cascading filter calls. To capture these additional attributes you must enable this property.

URI filter

You can filter captured data by URI. There are two possible scenarios:

- Capture listed URIs only.
- Capture all URIs, except for the listed ones.

Select the appropriate option button and list URIs or URI patterns to be included/excluded from the capture.

Parameters and attributes

Capture values of certain parameters and attributes (i.e. request headers, session attributes, request attributes, request parameters) as strings. To specify a captured attribute explicitly, provide its name in the **Attribute** column. To capture an attribute that contains a certain name, provide a partial name in the **Attribute** column. To capture all available attributes, leave the **Attribute** column empty or type '*'. Keep in mind that may cause a high overhead, Dynatrace does not recommend usage of wildcards in production environments.

> **Note**
>
> Properties marked with
> ⚠
> are especially relevant for sensor overhead on the system under diagnosis (SUD). Use these properties with caution and check the impact on the SUD after setting these properties.

Asynchronous servlet execution support

The Sensor supports asynchronous servlet execution as specified in the Servlet 3.0 specification. This feature requires that the Executor Tagging Sensor be placed and active. Asynchronous execution time — the time between ServletRequest.startAsync(…) and AsyncContext.complete() — is available as **Web Request Response Time** in the **PurePaths** dashlet and the **Web Requests** dashlet. If multiple asynchronous executions are started on a Pure Path, the **Web Request Response Time** is calculated from the first call of `ServletRequest.startAsync(...)` to the last call of `AsyncContext.complete()`.

PurePaths and web requests containing asynchronous execution elements are visualized with the status symbol


Play Framework support

The Servlet sensor also handles Play Framework support of the play framework. However, as Play does not implement the Servlet API, the sensor handles only Play Java Actions. Static content or asynchronous requests are not supported. The same properties as for Servlets are captured.

Struts sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The Struts sensor collects the response, execution and CPU time of calls to methods of the Struts framework.

> **Note**
>
> This sensor does not support OneAgent.

Thread Start Tagging sensor

The Thread Start Tagging Sensor adds the `TraceTag` to a local field of the `Runnable` object that is used to start the thread. A concurrency problem occurs if the same `Runnable` object is started twice concurrently.

The Thread Start Tagging Sensor is also responsible for Oracle Service Bus Tagging.

> **Note**
>
> A thread that is set to run as a daemon thread is not instrumented by this Sensor, because it is expected to be a long-running background thread that is not intended to be part of the PurePath.
>
> If a daemon thread starts another thread, the newly started thread inherits the daemon state. By default, these threads are not tagged. To use thread tagging, manually disable the daemon state of the newly started thread with `thread.setDaemon(false)`.

The Thread Start Tagging Sensor continues tracing when a new thread is created by the `Thread.start()` method.

Thrift Tagging sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The Thrift Tagging sensor traces PurePaths from the application to Cassandra Instances via the Thrift protocol.

> **Note**
>
> This sensor does not support OneAgent.

Tibco Messaging sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

JMS sensor

Note

This sensor does not support OneAgent.

The Tibco Messaging Sensor applies to Tibco Rendezvous and Tibco Business Works Service Engine. The Tibco Sensor captures data from message objects (typically using reflection calls) to provide the following information about the message:

- Destination.
- Transmission type: Whether the message was *sent* or *received*.
- Optionally, message contents

The sensor causes low overhead impact.

Tagging

Tibco Randezvous

The sensor transfers the trace tag of the execution path that sends the Tibco message to its child execution path(s) that process the message.

Tibco Business Works service engine

The sensor transfers the trace tag of the execution path that schedules a Tibco job to its job dispatcher.

Sensor configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **Tibco Messaging** item in the **Sensor Configuration** list.

Here you can enable or disable capturing of the messages content. Please keep in mind, that capturing of the message content can cause much overhead impact, especially for the large messages.

Network overhead

Captured message content transfers from the Agent to the Collector and the AppMon Server.
Keep network traffic overhead in mind, especially in production environments.

Java User Experience sensor

- **Send compressed JavaScript Agent to browser**: To keep the network overhead low, the Agent sends the file compressed. Clear the **Send compressed JavaScript Agent to browser** check box if your web server or any front tier, such as a load balancer, compresses the outgoing content. This prevents it from being compressed twice and lowers the CPU usage on that tier.

> **Notes**
>
> For rules other than *before specific HTML* and *after specific HTML* the contents of the **HTML pattern** column is ignored. The rule list is evaluated top-down, so that if a page matches multiple URI-specific rules, the topmost rule is applied.
>
> The *after specific HTML* rule provides a slightly better performance than the *before specific HTML* rule.

- **Inject before outermost HttpServletResponse**: Injection works in most cases, but it may not work if one of the servlet filters compresses the HTML response body. In this case, the Java Agent wraps the `HttpServletResponse` object passed to the outermost servlet filter.
- **Inject before innermost HttpServletResponse**: Injection works in most cases, but it may not work if one of the servlet filters parses the HTML and removes the script tag previously injected by the Java Agent. In this case, the Java Agent wraps the `HttpServletResponse` object passed to the outermost servlet filter and wraps again whenever some other servlet filter wraps our own object.
- **Inject in a specific servlet or filter**: Use this option as a fallback when none of the automatic options work for all requests. In this case, the Java Agent only wraps the response in the specified filter or servlet.

- **URI-specific injection behavior**: If the auto-injection rules place the JavaScript agent tag in a place unsuitable for your application, you can fine-tune it by setting some URI-specific injection behavior rules as exceptions to the auto-injection rules.
  - **Condition** and **URI pattern** specifies any conditions or URI patterns to use for filtering. For example, you specify that the rule should only apply to pages starting with */community/display*. Wildcards like * or ? are not supported, but for matching any URI you could use *contains /*.
  - **Injection Rule** and **HTML pattern** specify the desired injection behavior:
    - **automatic**: The default behavior. The JavaScript Agent injects automatically as first of script tag in `<head>` at delivery of the web page
    - **do not inject on this agent**: Equivalent setting the Injection Point to off. Agents matching other agent groups may take over and perform an injection themselves.
    - **before specific HTML**: Inject before a specific piece of HTML source code known to occur in the page, such as `<link rel="icon" type="image/png" href="/s/620/3/_/images/common-icons/favicon.png">`. Specify the code in the **HTML pattern** field. You can just paste any piece of your HTML source code.
    - **after specific HTML**: Inject after a specific piece of HTML source code known to occur in the page, such as `<BASE HREF="[https://www.example.com/approot/](https://www.example.com/approot/)">` Specify the code in the **HTML pattern** field. You can just paste any piece of your HTML source code.
    - **do not inject on this and subsequent agents**: Same as **do not inject on this agent**, plus this rule prevents sensor properties set on other (outermore) agents from overriding this choice and performing the injection there.

When you use this sensor on a Java application server, you can optimize the servlet filter chain injection point using the following options:

```
...
<head>
...
<script
   type="text/javascript"
   src="/dtagent55_jnps_5191.js"
   data dtconfig="rid=RID_-1381208364|
   rpid=15233675|sffs|domain=mydomain.com">
</script>
<script type="text/javascript">
    var contextPath = '/community';
</script>
```

JS Tag Example

- **Inject manually**: JavaScript Agent tag is not injected into HTML responses, AppMon Monitor beacon is not handled. Use this if you put the JavaScript Agent tag manually in your HTML code.
- **Automatically inject JavaScript agent into HTML**: JavaScript Agent tag is automatically injected into HTML responses. See JavaScript Agent for auto-injection rules.

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **User Experience** item in the **Sensor Configuration** list.

Sensor configuration

The User Experience sensor allows you to see how your applications are doing with the real users. It injects JavaScript Agent to your web pages, and collects timing and action information about user visit.

> **Note**
>
> This sensor does not support OneAgent.

WebSphere MQ (MQSeries) sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

WebSphere MQ mapping

WebSphere MQ tagging

Note

This sensor does not support OneAgent.

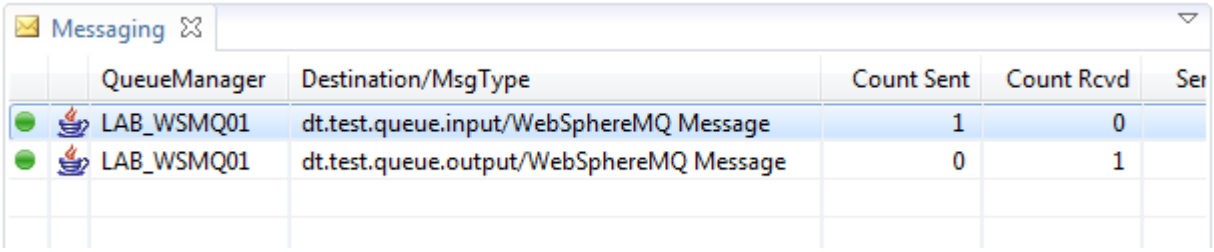
This sensor captures PUT and GET queue operations when accessing WebSphere MQ using the WebSphere MQ-Classes for Java. Additional data is captured to give more insight into the message.

Captured information

- Destination: the name of the queue where the message is sent to or received from.
- queueManager: the name of the Queue-Manager.
- transmissionType: whether the message was *sent* or *received*.
- messageSize: the size of the transferred message.

Visualization

The captured messages are aggregated and can be displayed in the **Messaging** dashlet.



For extended functionality see the MQSeries Receive Entry Point Sensor.

WebSphere MQ tagging

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

WebSphere Message Broker

CICS Agent installation and configuration

WebSphere MQ mapping

AppMon provides the capability to follow transactions through WebSphere MQ queue managers. As the page title implies you don't need to instrument the queue manager with a separate Sensor or Agent – as is the case with other messaging. You just have to make sure the application that PUTs and the application that GETs are both instrumented by an Agent that supports MQ tagging. The following agents provide Sensors for MQ tagging:

- Java Agent: MQSeries Sensor to tag outgoing messages, MQ Receive Entry Point Sensor to start new PurePaths for incoming messages.
- .NET Agent: .NET MQSeries Sensor to tag outgoing messages, .NET MQ Receive Entry Point Sensor to start new PurePaths for incoming messages.
- WebSphere Message Broker Agent – No extra sensor needs to be placed.
- CICS Agent: The CICS Sensor is automatically configured to start new PurePaths for any incoming messages.

Configuration

Note

This is not a Sensor in a System Profile, but an AppMon Server setting.

The **Settings > Dynatrace Server > MQ Tagging** settings specify what information should be used to identify WebSphere MQ messages. This information is used to link the subpaths from the client to the server side for applications that use MQ. The tagging always uses the MQ message ID. The correlation ID, the queue manager name, and the queue name can be added, as shown in the figure.

The default setting uses the message ID and the correlation ID for tagging. This setting does not require any additional configuration.

If the message ID of the MQ messages is unique within your environment, it is not necessary to add any additional information for tagging.

If the message ID is not unique within your environment (for example, if an application reads a message, processes it, and puts its contents to another queue with the same message ID), additional information is required for tagging. Otherwise, paths might be linked together incorrectly. The AppMon Server log file contains a message that indicates the problem.

The following log messages indicate that the custom tags are not unique.

```
WARNING [ExecPath] Invalid subPath count on synchronous link node: 2, link added by sensc
```

```
WARNING [com.dynatrace.diagnostics.server.datacenter.i] Cyclic path references detected:
WARNING [com.dynatrace.diagnostics.server.datacenter.i] Failed to link root paths: custom
```

If your environment requires using the queue or queue manager names to uniquely identify messages, you might need to set up WebSphere MQ Mapping.

WebSphere MQ mapping

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Using the MQ mappings tool

Related pages

WebSphere MQ (MQSeries) sensor

The WebSphere MQ mapping is not a sensor, but closely related to the WebSphere MQ (MQSeries) Sensor. *1MQ mapping is required under the belowconditions* to make sure that the AppMon Server can identify which PUTs and GETs belong together.

- Your WebSphere MQ setup has alias, remote, or cluster queues.
- The queue name and queue manager name are included in the MQ tagging configuration.

The MQ mapping contains information about the infrastructure. For example, if there is a queue QUEUE1 and an alias queue ALIASTOQUEUE1, the MQ mapping tells the server that messages put into ALIASTOQUEUE1 can be read from QUEUE1.

You can generate MQ mapping by using a tool that obtains the required information from the queue managers and sends it to the AppMon Server. This tool is provided in the `tools/createmqmappings` directory of the AppMon Server installation. To use this tool, copy it to the machine that has the MQ client library installation and can query information from the queue manager. JRE 1.6 is required to run the tool.

To determine whether MQ mapping is necessary:

1. Instrument all processes of your application and run your tests.
2. Open the **Messaging** dashlet that displays all queues that are used by your application.

If any queues are missing, you did not successfully instrument all layers of your application. In this situation, MQ mapping won't help.

You can safely ignore queues that have the same PUT and GET counts. No mapping is needed for them.

Some queues might appear to have only PUTs or only GETs:

- If the number of PUTs and GETs is not symmetric, that's another indicator that you did not instrument all layers of your application. In this situation, MQ mapping won't help.
- If you see a queue with GETs and a corresponding queue with the same number of PUTs, select these two queues and drill down into the **PurePath** dashlet. If you verify that these Paths should be linked together, set up MQ mapping.

Using the MQ mappings tool

To use the createmqmappings tool, execute the `createmqmappings.cmd` (Windows) or `createmqmappings.sh` (UNIX) script. The script code is provided below.

```
Usage: createmqmappings (-m <mq-qmgrname> | -h <mq-hostname> [-p <mq-port>] [-c <mq-chanr
                         -s <dt-hostname> [-ssl] [-port <dt-port>] [-user <dt-user>] [-pas

Specify <mq-qmgrname> to connect to a local queue manager.
Specify <mq-hostname> (and optionally <mq-port>, <mq-channelname>, <mq-qmgrname>) to conr
Optionally an <mq-user> and <mq-pass> can be specified for connecting to MQ
If a <mq-user> but no <mq-pass> is specified via command line the password will be prompt
If a <dt-user> but no <dt-pass> is specified via command line the password will be prompt
<dt-user> only has to have the "Web Service Interface Access" permission

Default values:
  <mq-port>         1414
  <mq-channelname>  SYSTEM.ADMIN.SVRCONN
  <dt-port>         8020 (or 8021 if "-ssl" is specified)
  <dt-user>         admin
  <dt-pass>         admin
```

The `createmqmappings` tool uses the IBM PCFAgent to send the commands listed in the following table.

Command	Description
`MQCMD_INQUIRE_Q_MGR`	Get the queue manager name and ID.
`MQCMD_INQUIRE_Q`	Get queue attributes.
`MQCMD_INQUIRE_CLUSTER_Q_MGR`	Get names and IDs of other queue managers in the cluster.

These commands only read from the queue manager and don't change any configuration. They are sent to the default command input queue of the queue manager, which is usually `SYSTEM.ADMIN.COMMAND.QUEUE`. By default, it also uses `SYSTEM.DEFAULT.MODEL.QUEUE` to receive replies. Those are the only two queues used.

Since the `createmqmappings` tool only inquires data from the queue manager, you can create a SVRCONN channel for a user that is restricted to only the minimum set of permissions.

Perform the following steps to grant the required permissions:

1. Create a new user that is **not** a member of the **mqm** group.
2. Define a server connection channel for this user by using the `runmqsc` command.

```
DEFINE CHANNEL(DYNATRACE.SVRCONN) CHLTYPE(SVRCONN) MCAUSER('<username>')
```

3. Grant the user the `connect`, `inquire`, and `display` permissions for the queue manager.

```
setmqaut -m <queuemanager-name> -t qmgr -p <username> +connect +inq +dsp
```

4. Grant the `put` permission for the `SYSTEM.ADMIN.COMMAND.QUEUE` queue. This queue is used to send the `inquire` commands to the queue manager.

```
setmqaut -m <queuemanager-name> -t q -n SYSTEM.ADMIN.COMMAND.QUEUE -p <username> +put
```

5. Grant the `display` and `get` permissions for the `SYSTEM.DEFAULT.MODEL.QUEUE` queue. This queue is used to receive the replies from the queue manager.

```
setmqaut -m <queuemanager-name> -t q -n SYSTEM.DEFAULT.MODEL.QUEUE -p <username> +dsp
```

6. Grant the `display` permissions for the `**` queue wildcard. This permission allows the `createmqmappings` tool to read the queue definitions.

```
setmqaut -m <queuemanager-name> -t q -n '**' -p <username> +dsp
```

XA sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The XA sensor collects state information of XA transactions.

> **Note**
>
> This sensor does not support OneAgent.

.NET Agent

On this page

Sensors

OneAgent

Related pages

.NET Agent configuration

Sensors

Sensor Configuration

Sensor properties

Java Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

The .NET Agent monitors Microsoft Common Language Runtimes. It is loaded by a CLR and configured through global environment variables and enabled for individual processes using the .NET Agent configuration tool. So, this agent is always loaded but only active when enabled for a specific process.

Please see .NET Agent configuration to learn how to configure the Agent.

Sensors

Category	Sensor	Classic Agent	OneAgent
Components	.NET Components	✓	✗
Database	ADO.NET	✓	✓
Exceptions	.NET Exception	✓	✓
GUI controls	.NET Windows Forms	✓	✗
	.NET WPF	✓	✗
Logging	.NET Logging	✓	✗
Memory	.NET Memory	✓	✗
Messaging	.NET AMQP	✓	✗
	.NET AMQP Receive Entry Point	✓	✗
	MSMQ	✓	✗
	MSMQ Receive Entry Point	✓	✗
Remoting	.NET WCF	✓	✓
	.NET Remoting	✓	✗

	.NET Service Fabric	✓	✓
Test framework	.NET Test	✓	✗
Threads	.NET Thread Tagging	✓	✓
Web requests	ASP.NET	✓	✓
	ASP.NET Events	✓	✗
	.NET Web Request	✓	✓
Web Service	.NET Web Service Tagging	✓	✗

OneAgent

OneAgent released as BETA functionality for AppMon 2017 May. Please be aware that:

- Not all features/sensors available in Classic Agent are already available in the OneAgent for AppMon
- Features/Sensors known from Classic Agent can behave differently

To use OneAgent in AppMon 2017 May you need to install Classic Agents first, and then roll them out to the OneAgent. No additional configuration needed. Before you switch to OneAgent, double check your major requirements with the available feature set.

> **Warning**
>
> You must **not** use the OneAgent in your production environment. The aim of BETA is to test it in pre-production environment, and see how the new Agent affects your custom dashboards/configurations you use to monitor your production environment.

> **Note**
>
> .NET Core requires OneAgent.

It is recommended to use OneAgent for following scenarios:

- To monitor Microsoft Azure
- To improve Collector scalability
- To resolve Collector connectivity problems
- To reduce AutoSensor overhead.

Features

Feature	Classic Agent	OneAgent
Argument capturing	✓	✓
AutoSensor	✓	✓
Class browsing	✓	✗
CPU Sampling	✓	✓
Custom Perfcounter metrics	✓	✗
Custom sensors	✓	✓
Decompilation	✓	✗
Deep object access	✓	✗
Delegation suppression	✓	✓

External API	✓	✗
GC/Memory metrics	✓	✓
Memory dumps	✓	✗
Thread dumps	✓	✗

ADO.NET sensor

Note

Properties marked with

are especially relevant for sensor overhead on the system under diagnosis (SUD). Use these properties with caution and check the impact on the SUD after setting these properties.

- **No aggregation or bind value capturing**: Each query will be displayed separately. Bind values are not captured.
- **Enable aggregation**: If queries are equal, only *one* DB query statement displays per PurePath parent node, along with the number of executions. This makes PurePaths trees much more readable, but capturing of bind values is disabled. DB connection acquisitions are not aggregated.
- **Enable bind value capturing** (

): Enables capture of SQL statement's bind values.
- **Maximum size of captured SQL command strings**: Limits the captured SQL statements to the specified length. The default is 512. All symbols, exceeding the limit, are cut.

Here you can enable or disable aggregation and bind value capturing. You can also set the limit for SQL command length.

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with .NET technology > Sensor Configuration** and then select the **ADO.NET** item in the **Sensor Configuration** list.

Sensor configuration

Bind value capturing, however, might increase overhead significantly.

Capturing base information does not add significant overhead. No reflection is used to gain the data, because it is provided by available properties and fields.

- The SQL string (by default, up to 4096 characters)

- The SQL string (by default, up to 4096 characters).
- The number of rows returned during the execution of the SQL statement.
- Optionally, the bind values for the SQL statements.
- The following connection pool data:
  - `poolName`
  - `dbName`
  - `dbHost`
  - `url`
  - `dbType`
  - `username`
  - `poolSize`

The ADO.NET sensor captures the following data:
The ADO.NET Sensor traces the preparation and execution performance information of SQL statements. It can also capture `ConnectionPool` data.

ASP.NET sensor

Capture values of certain parameters and attributes (i.e. request headers, session attributes, request attributes, request parameters) as strings. To specify a captured attribute explicitly, provide its name in the **Attribute** column. To capture an attribute that contains a certain name, provide a partial name in the **Attribute** column. To capture all available attributes, leave the **Attribute** column empty or type '*'. Keep in mind that may cause a high overhead, Dynatrace does not recommend usage of wildcards in production environments.

Parameters and attributes

Select the appropriate option button and list URIs or URI patterns to be included/excluded from the capture.

- Capture listed URIs only.
- Capture all URIs, except for the listed ones.

You can filter captured data by URI. There are two possible scenarios:

URI filter

Always add Dynatrace header to responses: Force adding the AppMon header to responses, even if it was not available in the request (useful for Load-Test scenarios). You cannot change this value if a DC-RUM appliance is configured. See Integration for details.

Sensor properties

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with .NET technology > Sensor Configuration** and then select the **ASP.NET** item in the **Sensor Configuration** list.

Sensor configuration

ASP.NET sensor also supports OWIN/Katana. This works with OneAgent only.

Supported technologies

Internally, ASP.NET sets the response code to 200 (OK) by default. It might occur that only one thread sees a change in the response status, leading to different response codes in the details of each ASP.NET node.
The ASP.NET Sensor traces incoming requests in the ASP.NET pipeline. The sensor places an additional HTTP tag `X-dynatrace`. In certain scenarios, the ASP.NET pipeline might split the processing of a request.

MSMQ sensor

- **No tagging***: Disables message tagging.
- **Use the 'Extension' property for tagging**: Adds tracing information to the **Extension** property of the message.
- **Use the 'Label' property for tagging**: Adds tracing information to the **Label** property of the message.

To be able to create a continuous PurePaths from the message sender to the receiver, the sensor has to attach some tracing information to the message. Select the message property that can be used for tagging:

Sensor configuration

The MSMQ gives you insight into .NET messages. It captures destination information, sent and received bytes, and the execution time.

MSMQ Receive Entry Point sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The MSMQ Receive Entry Point sensor specifies a method that processes a message after it is received via MessageQueue::Receive.

> **Note**
>
> This sensor does not support OneAgent.

.NET AMQP sensor

The sensor doesn't have any user configurable properties.

Sensor configuration

The .NET AMQP sensor collects destination information, sent/received bytes, and the execution time for Advanced Message Queuing Protocol (AMQP).

> **Note**
>
> This sensor does not support OneAgent.

.NET AMQP Receive Entry Point sensor

The sensor doesn't have any user configurable properties.

Sensor configuration

The .NET AMQP Sensor only recognizes Messages being sent and received. The methods processing these Messages need to be defined as Entry Points in this Sensor, in order to have the complete AMQP workflow visible in your PurePaths.

> **Note**
>
> This sensor does not support OneAgent.

.NET Components sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The .NET Component Sensor Pack collects the time, bytes and objects of Serviced Components.

> **Note**
>
> This sensor does not support OneAgent.

.NET Exception sensor

Overhead Property	Description	Overhead impact	Default value
Capture Stacktrace	Enables capturing of stack traces. Capturing the stack and transferring the data to the server causes high CPU overhead on the instrumented application and consumes network bandwidth.	high	enabled
Maximum Captured Stacktrace depth	Limits the number of elements per stacktrace. Only the topmost elements are transferred. The allowed range is 0—256.	medium	32
Maximum Captured Stacktraces per second	Limits the number of stacktraces captured per second. The allowed range is 0—1024.	high	128

In the **Overhead Properties** section in the **Exception Sensor Properties** dialog box, you can enable or disable capture of stack traces for exceptions and fine-tune the amount of data that is captured and transferred from the Agent to the Server.

Overhead properties

Select the appropriate option button and list classes to be included/excluded from the instrumentation. You can do this manually or use the Class Browser.

- Instrument listed classes only.
- Instrument all classes, except for the listed ones.

You can define which classes to instrument. There are two possible scenarios:

Configuration properties

Exception aggregation is aimed to reduce the PurePath node count when many exceptions are thrown. The aggregation is done per instrumented node level. The first three exceptions are captured normally, beginning with the 4th exception, aggregation will take effect and only count occurrences of distinct class+message combinations.

Aggregation

To edit the System Profile specific settings, right click you System Profile, and select **Edit System**

Profile > Agent Group/tier with .NET technology > Sensor Configuration and then select the **.NET Exception** item in the **Sensor Configuration** list.

Sensor Configuration

The .NET Exception Sensor instruments the exception's constructor to retrieve information about the exception's class, its message, and optionally its stack trace elements. It also retrieves information about exceptions that do not get "thrown" but are created via `new`. The transferred exception event is handled as a single event with no duration.

.NET Logging sensor

Select the appropriate option button and list log events to be included/excluded from the capture. Select classes manually or using the Class Browser.

- Capture listed log events only.
- Capture all log events, except for the listed ones.

You can define which log events should be captured. There are two possible scenarios:

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with .NET technology > Sensor Configuration** and then select the **.NET Logging** item in the **Sensor Configuration** list.

Sensor configuration

The .NET Logging Sensor captures data from the logging records. Captured log messages are truncated to the maximum capturing string length. The transferred log event is handled as a single event with no duration.

> **Note**
>
> This sensor does not support OneAgent.

ASP.NET Events sensor

The sensor doesn't have any user configurable properties.

Sensor configuration

The ASP.NET Events sensor extends the ASP.NET Sensor Pack to include lifecycle and control events on ASP.NET Pages, Custom Controls and Web Parts

> **Note**
>
> This sensor does not support OneAgent.

.NET Remoting sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The .NET Remoting Sensor Pack collects the time, bytes and objects involved in a .NET Remoting Call.

> **Note**
>
> This sensor does not support OneAgent.

.NET Service Fabric sensor

> **Note**
>
> This sensor is only available with OneAgent.

Service Fabric Reliable Actors are not supported at the moment.

Known limitations

For further information on Service Fabric service remoting please refer to the official Microsoft documentation.
The .NET Service Fabric sensor traces incoming and outgoing service remoting calls, supporting both stateful and stateless services.

.NET Test sensor

The sensor doesn’t have any user configurable properties.

Sensor configuration

The .NET Tests sensor captures related information for the Microsoft Testing Framework and NUnit.

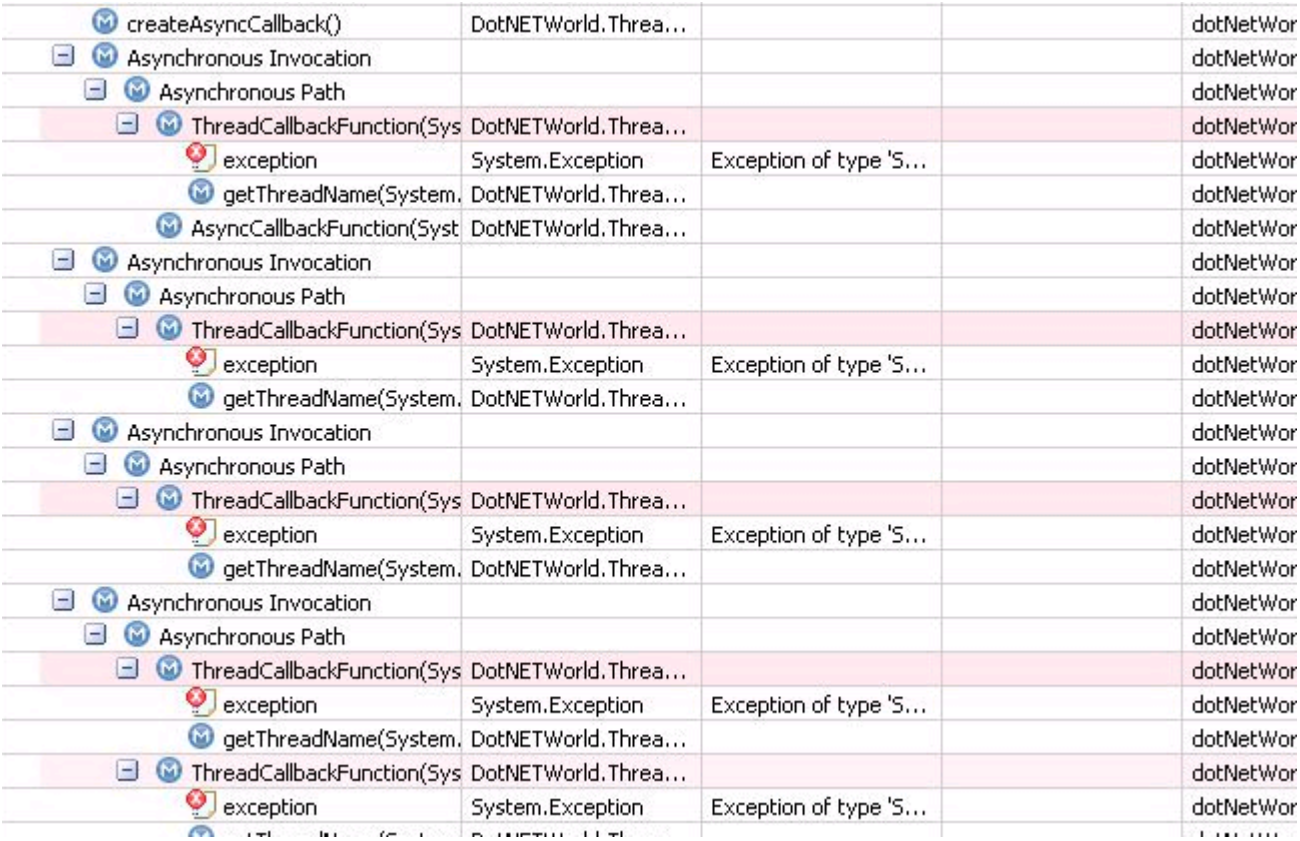
> **Note**
>
> This sensor does not support OneAgent.

.NET Thread Tagging sensor

The sensor doesn't have any user configurable properties.

Sensor configuration

The overhead of this Sensor is very low. There are only two instrumented points on each Thread Start.



Thread transitions

Thread transitions are noted as **Asynchronous Invocation** in the PurePath, as shown.

- Starting threads by using normal Thread-Start.
- Starting threads by using parameterized Thread-Start.
- Executing delegates in the background.
- Executing delegates with children in the background.

The .NET Thread Tagging Sensor traces calls across thread boundaries. The following .NET

techniques are supported:

.NET Web Request sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Sensor properties

The .NET Web Requests Sensor traces HTTP requests, including redirects, to see the lifecycle of a request from its initial client call to the server and further across thread and process boundaries.

The sensor tags the request with an additional HTTP header `X-dynatrace`. The header is processed for each tier that has an active agent and is passed by the request during its lifetime, which creates negligible overhead.

The .NET Web Requests Sensor captures the URI, query strings, response code, and type of an HTTP method (POST, GET, etc.).

Sensor Configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier with .NET technology > Sensor Configuration** and then select the **.NET Web Requests** item in the **Sensor Configuration** list.

AppMon truncates captured web requests to the specified length. Specify the required length in the edit box. All exceeding characters will be cut.

URI filter

You can also filter captured queries by URI. There are two possible scenarios:

- Capture listed URIs only.
- Capture all URIs, except for the listed ones.

Select the appropriate option button and list URIs to be included/excluded from the capture.

Depending of the size of the configured rules list, the impact of the sensor on Agent overhead may increase.

.NET WCF sensor

The value (string) always starts with `FW1;`

- SOAP header level; header name `dynaTrace` and NS `http://ns.dynatrace.com/wcf`
- HTTP header level; header name `X-dynaTrace`

WCF Sensor tagging occurs on the
The .NET WCF Sensor tags WCF calls from client to server through SOAP or HTTP.

.NET Windows Forms sensor

The sensor doesn't have any user configurable properties.

Sensor configuration

The .NET Windows Forms Sensor Pack collects information about GUI-related .NET Windows Forms calls.

> **Note**
>
> This sensor does not support OneAgent.

.NET WPF sensor

The sensor doesn't have any user configurable properties.

Sensor configuration

The .NET WPF Sensor Pack collects information about .NET Windows Presentation Foundation calls.

> **Note**
>
> This sensor does not support OneAgent.

Node.js Agent

Related pages

Node.js Agent configuration

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

The Node.js Agent tracks all web requests to Node.js web applications. Unlike other Agents, the Nodejs Agent does not deliver host monitoring data. To accomplish this you must use the Host Monitoring Agent. Communication between the Agent and Collector, and between the Collector and Server uses HTTP(S) on ports 8042 (8043) and 8040 (8041) respectively. When configuring the Agent to communicate directly with the Server, it utilizes http(s) on ports 8040(8041).

Node.js Agent sensors are not user-configurable.

Please see Node.js Agent configuration to learn how to install and configure the Agent.

OneAgent

Node.js is a true OneAgent, and doesn’t work on the Classic Agent platform.

Apache Agent

On this page

Classic Agent

OneAgent

Sensors

FAQ

Related pages

Web Server Master Agent configuration

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

IIS Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

Child pages

Apache Modules sensor

Apache User Experience sensor

Web Server sensor for Apache

The Apache Web Server Agent monitors Apache Webservers. Depending on the platform you use — Classic Agent or OneAgent, its architecture and functionality may differ. Find the detailed information below.

See Configure Apache Agent to learn how to configure the Agent.

Classic Agent

The Agent for Web Servers consists of two parts:

- The *Web Server Module* (slave) is the module that is loaded by the web server. It gathers the information that is used for analysis on the AppMon Server.
- The *Web Server Agent* (master) is a separate process that collects data of all worker processes (slaves). It is responsible for handling communication with the Collector, retrieving configuration data etc. See Web Server Agent configuration to learn how to configure this master agent.

Communication

The Web Server Modules communicate with the master Agent through these channels:

- A shared memory segment propagates configuration data, received from the Collector by the master Agent, to the worker processes and to hold other information that must be shared across all worker processes. The shared memory segment is backed by a file in `<DT_HOME>/agent/conf` on all platforms where the worker processes and the Agent process must have read/write access.
- The Web Server Modules (slaves) send collected data to the master Agent via a stateless UDP connection. The master Agent collects the data, transforms it, and forwards it to the Collector (standalone or Server-embedded).

OneAgent

The Agent for Web Servers consists only of a *Web Server Module* that is loaded by the web server. ollects data of all worker processes (slaves). It gathers the information that is used for analysis on the AppMon Server. The OneAgent is responsible for handling communication with the Collector. Find more information about OneAgent here.

To use OneAgent you need to install Classic Agents first, and then roll them out to the OneAgent.

Sensors

Sensor	Classic Agent	OneAgent
Webserver sensor	✓	✓
Apache modules	✗	✓
User Experience	✓	✓

FAQ

How do I find out which modules are instrumented?

Check the Agent log file in the AppMon Client:

1. Open the Agents Overview dashlet.
2. Right-click the required Agent and select **Show Agent Log File**.
3. In the System Information dashlet, select the required log file.

How can I make sure that my module is monitored?

You can add the essential modules to the **Always capture timings of the following modules** list from the Apache Modules sensor properties. These modules are always instrumented. Keep in mind, that the sensor only supports OneAgent.

Web Server sensor for Apache

Related pages

Supported web servers and platforms

The Web Server Sensos traces PurePath for all HTTP requests including redirects, served by the web server. It also traces relevant information about these connections, including URI, query, method response code of the HTTP request/response. It can also be configured to capture request/response headers and request parameter.

Sensor Configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **Web Server** item in the **Sensor Configuration** list.

Web Server Sensor properties

- **Measure Request/Response Sizes**: Captures the sizes of request and response in bytes.
- **Always add Dynatrace header to responses**: Forces adding the AppMon header to responses, even if it was not available in the request (useful for Load-Test scenarios). You cannot change this value if a DC-RUM appliance is configured. See Integration for details.

URI filter

You can filter captured data by URI. There are two possible scenarios:

- Capture listed URIs only.
- Capture all URIs, except for the listed ones.

Select the appropriate option button and list URIs or URI patterns to be included/excluded from the capture.

Headers and parameters

Capture values of certain headers and parameters and attributes (i.e. request headers, response headers, request parameters) as strings. To specify a captured parameter explicitly, provide its name in the Attribute column. To capture a parameter that contains a certain name, provide a partial name in the **HTTP header and parameter** column. To capture all available parameters, leave the column empty or type '*'. Keep in mind that may cause a high overhead, Dynatrace does not recommend usage of wildcards in production environments.

Apache Modules sensor

You can enable automatic instrumentation of all detected Apache modules, by selecting the appropriate checkbox. Up to 50 modules can be instrumented. If you want to exclude some particular modules, specify them in the list below. These modules are never captured.

Here you can define, which modules should always be instrumented. Specify their names in the top list. To simplify the view of the captured info, enable aggregation, by selecting **Aggregate Apache modules** checkbox. In this case, all modules processed in under 100 microseconds are aggregated to a single method node.

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **Apache Modules** item in the **Sensor Configuration** list.

Sensor configuration

The Apache Modules sensor collects timings from user-defined Apache modules. You can choose which modules to instrument, and which to exclude.

Apache User Experience sensor

Related pages

User experience management

The User Experience sensor allows you to see how your applications are doing with the real users. It injects JavaScript Agent to your web pages, and collects timing and action information about user visit.

Sensor configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **User Experience** item in the **Sensor Configuration** list.

- **Inject manually**: JavaScript Agent tag is not injected into HTML responses, AppMon Monitor beacon is not handled. Use this if you put the JavaScript Agent tag manually in your HTML code.
- **Automatically inject JavaScript agent into HTML**: JavaScript Agent tag is automatically injected into HTML responses. See JavaScript Agent for auto-injection rules.

JS Tag Example

```
...
<head>
...
<script
   type="text/javascript"
   src="/dtagent55_jnps_5191.js"
   data dtconfig="rid=RID_-1381208364|
   rpid=15233675|sffs|domain=mydomain.com">
</script>
<script type="text/javascript">
    var contextPath = '/community';
</script>
```

When you use this sensor on a Java application server, you can optimize the servlet filter chain injection point using the following options:

- **Inject before outermost HttpServletResponse**: Injection works in most cases, but it may not work if one of the servlet filters compresses the HTML response body. In this case, the Java Agent wraps the `HttpServletResponse` object passed to the outermost servlet filter.
- **Inject before innermost HttpServletResponse**: Injection works in most cases, but it may not work if one of the servlet filters parses the HTML and removes the script tag previously injected by the Java Agent. In this case, the Java Agent wraps the `HttpServletResponse` object passed to the outermost servlet filter and wraps again whenever some other servlet filter wraps our own object.
- **Inject in a specific servlet or filter**: Use this option as a fallback when none of the automatic options work for all requests. In this case, the Java Agent only wraps the response in the specified filter or servlet.
- **URI-specific injection behavior**: If the auto-injection rules place the JavaScript agent tag in a place unsuitable for your application, you can fine-tune it by setting some URI-specific injection behavior rules as exceptions to the auto-injection rules.
  - **Condition** and **URI pattern** specifies any conditions or URI patterns to use for filtering. For example, you specify that the rule should only apply to pages starting with */community/display*. Wildcards like * or ? are not supported, but for matching any URI you could use *contains /*.
  - **Injection Rule** and **HTML pattern** specify the desired injection behavior:
    - **automatic**: The default behavior. The JavaScript Agent injects automatically as first of script tag in `<head>` at delivery of the web page
    - **do not inject on this agent**: Equivalent setting the Injection Point to off. Agents matching other agent groups may take over and perform an injection themselves.
    - **before specific HTML**: Inject before a specific piece of HTML source code known to occur in the page, such as `<link rel="icon" type="image/png" href="/s/620/3/_/images/common-icons/favicon.png">`. Specify the code in the **HTML pattern** field. You can just paste any piece of your HTML source code.
    - **after specific HTML**: Inject after a specific piece of HTML source code known to occur in the page, such as `<BASE HREF="[https://www.example.com/approot/](https://www.example.com/approot/)">` Specify the code in the **HTML pattern** field. You can just paste any piece of your HTML source code.
    - **do not inject on this and subsequent agents**: Same as **do not inject on this agent**, plus this rule prevents sensor properties set on other (outermore) agents from overriding this choice and performing the injection there.

Notes

For rules other than *before specific HTML* and *after specific HTML* the contents of the **HTML pattern** column is ignored. The rule list is evaluated top-down, so that if a page matches multiple URI-specific rules, the topmost rule is applied.

The *after specific HTML* rule provides a slightly better performance than the *before specific HTML* rule.

- **Send compressed JavaScript Agent to browser**: To keep the network overhead low, the Agent sends the file compressed. Clear the **Send compressed JavaScript Agent to browser** check box if your web server or any front tier, such as a load balancer, compresses the outgoing content. This prevents it from being compressed twice and lowers the CPU usage on that tier.

IIS Agent

On this page

Architecture

Classic Agent

OneAgent

Sensors

Related pages

IIS User Experience sensor

Web Server sensor for IIS

Web Server Master Agent configuration

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

The IIS Web Server Agent monitors Microsoft Internet Information Services webservers. Depending on the platform you use — Classic Agent or OneAgent, its architecture and functionality may differ. Find the detailed information below.

See Configure IIS Agent to learn how to configure the Agent.

Architecture

Classic Agent

The Agent for Web Servers consists of two parts:

- The *Web Server Module* (slave) is the module that is loaded by the web server. It gathers the information that is used for analysis on the AppMon Server.
- The *Web Server Agent* (master) is a separate process that collects data of all worker processes (slaves). It is responsible for handling communication with the Collector, retrieving configuration data etc. See Web Server Agent configuration to learn how to configure this master agent.

Communication

The Web Server Modules communicate with the master Agent through these channels:

- A shared memory segment propagates configuration data, received from the Collector by the master Agent, to the worker processes and to hold other information that must be shared across all worker processes. The shared memory segment is backed by a file in `<DT_HOME>/agent/conf` on all platforms where the worker processes and the Agent process must have read/write access.
- The Web Server Modules (slaves) send collected data to the master Agent via a stateless UDP connection. The master Agent collects the data, transforms it, and forwards it to the Collector (standalone or Server-embedded).

OneAgent

The Agent for Web Servers consists only of a *Web Server Module* that is loaded by the web server. ollects data of all worker processes (slaves). It gathers the information that is used for analysis on the AppMon Server. The OneAgent is responsible for handling communication with the Collector. Find more information about OneAgent here.

To use OneAgent you need to install Classic Agents first, and then roll them out to the OneAgent.

Sensors

Sensor	Classic Agent	OneAgent
Webserver sensor	✓	✓
User Experience sensor	✓	✓

Web Server sensor for IIS

Related pages

Supported web servers and platforms

The Web Server Sensos traces PurePath for all HTTP requests including redirects, served by the web server. It also traces relevant information about these connections, including URI, query, method response code of the HTTP request/response. It can also be configured to capture request/response headers and request parameter.

Sensor Configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **Web Server** item in the **Sensor Configuration** list.

Web Server Sensor properties

- **Measure Request/Response Sizes**: Captures the sizes of request and response in bytes.
- **Always add Dynatrace header to responses**: Forces adding the AppMon header to responses, even if it was not available in the request (useful for Load-Test scenarios). You cannot change this value if a DC-RUM appliance is configured. See Integration for details.

URI filter

You can filter captured data by URI. There are two possible scenarios:

- Capture listed URIs only.
- Capture all URIs, except for the listed ones.

Select the appropriate option button and list URIs or URI patterns to be included/excluded from the capture.

Headers and parameters

Capture values of certain headers and parameters and attributes (i.e. request headers, response headers, request parameters) as strings. To specify a captured parameter explicitly, provide its name in the Attribute column. To capture a parameter that contains a certain name, provide a partial name in the **HTTP header and parameter** column. To capture all available parameters, leave the column empty or type '*'. Keep in mind that may cause a high overhead, Dynatrace does not recommend usage of wildcards in production environments.

IIS User Experience sensor

Related pages

User experience management

The User Experience sensor allows you to see how your applications are doing with the real users. It injects JavaScript Agent to your web pages, and collects timing and action information about user visit.

Sensor configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **User Experience** item in the **Sensor Configuration** list.

- **Inject manually**: JavaScript Agent tag is not injected into HTML responses, AppMon Monitor beacon is not handled. Use this if you put the JavaScript Agent tag manually in your HTML code.
- **Automatically inject JavaScript agent into HTML**: JavaScript Agent tag is automatically injected into HTML responses. See JavaScript Agent for auto-injection rules.

JS Tag Example

```
...
<head>
...
<script
   type="text/javascript"
   src="/dtagent55_jnps_5191.js"
   data dtconfig="rid=RID_-1381208364|
   rpid=15233675|sffs|domain=mydomain.com">
</script>
<script type="text/javascript">
    var contextPath = '/community';
</script>
```

When you use this sensor on a Java application server, you can optimize the servlet filter chain injection point using the following options:

- **Inject before outermost HttpServletResponse**: Injection works in most cases, but it may not work if one of the servlet filters compresses the HTML response body. In this case, the Java Agent wraps the `HttpServletResponse` object passed to the outermost servlet filter.
- **Inject before innermost HttpServletResponse**: Injection works in most cases, but it may not work if one of the servlet filters parses the HTML and removes the script tag previously injected by the Java Agent. In this case, the Java Agent wraps the `HttpServletResponse` object passed to the outermost servlet filter and wraps again whenever some other servlet filter wraps our own object.
- **Inject in a specific servlet or filter**: Use this option as a fallback when none of the automatic options work for all requests. In this case, the Java Agent only wraps the response in the specified filter or servlet.
- **URI-specific injection behavior**: If the auto-injection rules place the JavaScript agent tag in a place unsuitable for your application, you can fine-tune it by setting some URI-specific injection behavior rules as exceptions to the auto-injection rules.
  - **Condition** and **URI pattern** specifies any conditions or URI patterns to use for filtering. For example, you specify that the rule should only apply to pages starting with */community/display*. Wildcards like * or ? are not supported, but for matching any URI you could use *contains /*.
  - **Injection Rule** and **HTML pattern** specify the desired injection behavior:
    - **automatic**: The default behavior. The JavaScript Agent injects automatically as first of script tag in `<head>` at delivery of the web page
    - **do not inject on this agent**: Equivalent setting the Injection Point to off. Agents matching other agent groups may take over and perform an injection themselves.
    - **before specific HTML**: Inject before a specific piece of HTML source code known to occur in the page, such as `<link rel="icon" type="image/png" href="/s/620/3/_/images/common-icons/favicon.png">`. Specify the code in the **HTML pattern** field. You can just paste any piece of your HTML source code.
    - **after specific HTML**: Inject after a specific piece of HTML source code known to occur in the page, such as `<BASE HREF="[https://www.example.com/approot/](https://www.example.com/approot/)">` Specify the code in the **HTML pattern** field. You can just paste any piece of your HTML source code.
    - **do not inject on this and subsequent agents**: Same as **do not inject on this agent**, plus this rule prevents sensor properties set on other (outermore) agents from overriding this choice and performing the injection there.

Notes

For rules other than *before specific HTML* and *after specific HTML* the contents of the **HTML pattern** column is ignored. The rule list is evaluated top-down, so that if a page matches multiple URI-specific rules, the topmost rule is applied.

The *after specific HTML* rule provides a slightly better performance than the *before specific HTML* rule.

- **Send compressed JavaScript Agent to browser**: To keep the network overhead low, the Agent sends the file compressed. Clear the **Send compressed JavaScript Agent to browser** check box if your web server or any front tier, such as a load balancer, compresses the outgoing content. This prevents it from being compressed twice and lowers the CPU usage on that tier.

NGINX Agent

On this page

Architecture

Classic Agent

OneAgent

Sensors

FAQ

Related pages

NGINX Modules sensor

NGINX User Experience sensor

Web Server sensor for NGINX

Web Server Master Agent configuration

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

IIS Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

The NGINX Web Server Agent monitors NGINX Webservers. Depending on the platform you use — Classic Agent or OneAgent, its architecture and functionality may differ. Find the detailed information below.

See Configure NGINX Agent to learn how to configure the Agent.

Architecture

Classic Agent

The Agent for Web Servers consists of two parts:

- The *Web Server Module* (slave) is the module that is loaded by the web server. It gathers the information that is used for analysis on the AppMon Server.
- The *Web Server Agent* (master) is a separate process that collects data of all worker processes (slaves). It is responsible for handling communication with the Collector, retrieving configuration data etc. See Web Server Agent configuration to learn how to configure this master agent.

Communication

The Web Server Modules communicate with the master Agent through these channels:

- A shared memory segment propagates configuration data, received from the Collector by the master Agent, to the worker processes and to hold other information that must be shared across all worker processes. The shared memory segment is backed by a file in `<DT_HOME>/agent/conf` on all platforms where the worker processes and the Agent process must have read/write access.
- The Web Server Modules (slaves) send collected data to the master Agent via a stateless UDP connection. The master Agent collects the data, transforms it, and forwards it to the Collector (standalone or Server-embedded).

OneAgent

The Agent for Web Servers consists only of a *Web Server Module* that is loaded by the web server

The Agent for Web Servers consists only of a *Web Server Module* that is loaded by the web server. It collects data of all worker processes (slaves). It gathers the information that is used for analysis on the AppMon Server. The OneAgent is responsible for handling communication with the Collector. Find more information about OneAgent here.

To use OneAgent you need to install Classic Agents first, and then roll them out to the OneAgent.

Sensors

Sensor	Classic Agent	OneAgent
Webserver sensor	✓	✓
NGINX modules	✗	✓
User Experience sensor	✓	✓

FAQ

How do I find out which modules are instrumented?

Check the Agent log file in the AppMon Client:

1. Open the Agents Overview dashlet.
2. Right-click the required Agent and select **Show Agent Log File**.
3. In the System Information dashlet, select the required log file.

Web Server sensor for NGINX

Related pages

Supported web servers and platforms

The Web Server Sensos traces PurePath for all HTTP requests including redirects, served by the web server. It also traces relevant information about these connections, including URI, query, method response code of the HTTP request/response. It can also be configured to capture request/response headers and request parameter.

Sensor Configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **Web Server** item in the **Sensor Configuration** list.

Web Server Sensor properties

- **Measure Request/Response Sizes**: Captures the sizes of request and response in bytes.
- **Always add Dynatrace header to responses**: Forces adding the AppMon header to responses, even if it was not available in the request (useful for Load-Test scenarios). You cannot change this value if a DC-RUM appliance is configured. See Integration for details.

URI filter

You can filter captured data by URI. There are two possible scenarios:

- Capture listed URIs only.
- Capture all URIs, except for the listed ones.

Select the appropriate option button and list URIs or URI patterns to be included/excluded from the capture.

Headers and parameters

Capture values of certain headers and parameters and attributes (i.e. request headers, response headers, request parameters) as strings. To specify a captured parameter explicitly, provide its name in the Attribute column. To capture a parameter that contains a certain name, provide a partial name in the **HTTP header and parameter** column. To capture all available parameters, leave the column empty or type '*'. Keep in mind that may cause a high overhead, Dynatrace does not recommend usage of wildcards in production environments.

NGINX Modules sensor

- **Capture timings of Nginx modules**: Captures timing of NGINX modules.
- **Capture information about proxied requests**: Captures information about the web requests issued by NGINX proxy.
- **Aggregate timings of Nginx modules**: Aggregates all modules processed in under 100 microseconds will be aggregated to a single method node.

Properties

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **NGINX Modules** item in the **Sensor Configuration** list.

Sensor configuration

The NGINX Modules sensor collects timings from user-defined NGINX modules.

NGINX User Experience sensor

Related pages

User experience management

The User Experience sensor allows you to see how your applications are doing with the real users. It injects JavaScript Agent to your web pages, and collects timing and action information about user visit.

Sensor configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **User Experience** item in the **Sensor Configuration** list.

- **Inject manually**: JavaScript Agent tag is not injected into HTML responses, AppMon Monitor beacon is not handled. Use this if you put the JavaScript Agent tag manually in your HTML code.
- **Automatically inject JavaScript agent into HTML**: JavaScript Agent tag is automatically injected into HTML responses. See JavaScript Agent for auto-injection rules.

JS Tag Example

```
...
<head>
...
<script
   type="text/javascript"
   src="/dtagent55_jnps_5191.js"
   data dtconfig="rid=RID_-1381208364|
   rpid=15233675|sffs|domain=mydomain.com">
</script>
<script type="text/javascript">
    var contextPath = '/community';
</script>
```

When you use this sensor on a Java application server, you can optimize the servlet filter chain injection point using the following options:

- **Inject before outermost HttpServletResponse**: Injection works in most cases, but it may not work if one of the servlet filters compresses the HTML response body. In this case, the Java Agent wraps the `HttpServletResponse` object passed to the outermost servlet filter.
- **Inject before innermost HttpServletResponse**: Injection works in most cases, but it may not work if one of the servlet filters parses the HTML and removes the script tag previously injected by the Java Agent. In this case, the Java Agent wraps the `HttpServletResponse` object passed to the outermost servlet filter and wraps again whenever some other servlet filter wraps our own object.
- **Inject in a specific servlet or filter**: Use this option as a fallback when none of the automatic options work for all requests. In this case, the Java Agent only wraps the response in the specified filter or servlet.
- **URI-specific injection behavior**: If the auto-injection rules place the JavaScript agent tag in a place unsuitable for your application, you can fine-tune it by setting some URI-specific injection behavior rules as exceptions to the auto-injection rules.
  - **Condition** and **URI pattern** specifies any conditions or URI patterns to use for filtering. For example, you specify that the rule should only apply to pages starting with */community/display*. Wildcards like * or ? are not supported, but for matching any URI you could use *contains /*.
  - **Injection Rule** and **HTML pattern** specify the desired injection behavior:
    - **automatic**: The default behavior. The JavaScript Agent injects automatically as first of script tag in `<head>` at delivery of the web page
    - **do not inject on this agent**: Equivalent setting the Injection Point to off. Agents matching other agent groups may take over and perform an injection themselves.
    - **before specific HTML**: Inject before a specific piece of HTML source code known to occur in the page, such as `<link rel="icon" type="image/png" href="/s/620/3/_/images/common-icons/favicon.png">`. Specify the code in the **HTML pattern** field. You can just paste any piece of your HTML source code.
    - **after specific HTML**: Inject after a specific piece of HTML source code known to occur in the page, such as `<BASE HREF="[https://www.example.com/approot/](https://www.example.com/approot/)">` Specify the code in the **HTML pattern** field. You can just paste any piece of your HTML source code.
    - **do not inject on this and subsequent agents**: Same as **do not inject on this agent**, plus this rule prevents sensor properties set on other (outermore) agents from overriding this choice and performing the injection there.

Notes

For rules other than *before specific HTML* and *after specific HTML* the contents of the **HTML pattern** column is ignored. The rule list is evaluated top-down, so that if a page matches multiple URI-specific rules, the topmost rule is applied.

The *after specific HTML* rule provides a slightly better performance than the *before specific HTML* rule.

- **Send compressed JavaScript Agent to browser**: To keep the network overhead low, the Agent sends the file compressed. Clear the **Send compressed JavaScript Agent to browser** check box if your web server or any front tier, such as a load balancer, compresses the outgoing content. This prevents it from being compressed twice and lowers the CPU usage on that tier.

PHP Agent

On this page

Sensors

OneAgent

Related pages

PHP Agent configuration

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

The PHP agent, which analyses PHP application performance and behavior, is an extension of PHP interpreter that is supported on the Apache HTTP Server, PHP-FPM or IIS.

Please see PHP Agent configuration to learn how to configure the Agent.

Sensors

PHP Agent uses the following sensors:

Category	Sensor	Classic Agent	OneAgent
Database	PHP MySQL	✓	✓
	PHP Oracle	✓	✓
	PHP PDO	✓	✓
	PHP PostgreSQL	✓	✓
Exceptions	PHP Exceptions	✓	✓
Logging	PHP Logging	✓	✓
Network	PHP Web Requests	✓	✓
PHP	PHP General	✓	✓
	PHP Server	✓	✓
Test Automation	PHP Unit	✓	✗
Server	PHP Server	✓	✓

OneAgent

OneAgent released as BETA functionality for AppMon 2017 May. Please be aware that:

- Not all features/sensors available in Classic Agent are already available in the OneAgent for AppMon

- Features/Sensors known from Classic Agent can behave differently

To use OneAgent in AppMon 2017 May you need to install Classic Agents first, and then roll them out to the OneAgent. No additional configuration needed. Before you switch to OneAgent, double check your major requirements with the available feature set.

> **Warning**
>
> You must **not** use the OneAgent in your production environment. The aim of BETA is to test it in pre-production environment, and see how the new Agent affects your custom dashboards/configurations you use to monitor your production environment.

Features

Feature	Classic Agent	OneAgent
PHP 7 supported	✗	✓
Marking arbitrary methods as external calls	✓	✗

PHP General sensor

- **Show compilation time**: Enables capture of PHP file compilation time.
- **Show include, include_once, require, require_once functions**: Enables capture of mentioned PHP functions.

Properties

1. Start the AppMon Client.
2. Open the **System Profile Preferences** window for your application. Under **System Profiles**, right-click your application and choose **Edit System Profile**.
3. Enable monitoring of PHP scripts. To do this, navigate to **General** settings for your application, and select the **PHP** check box in the **Technology** section.
4. Place the PHP General sensor.
   a. Go to the **Sensor Placement** section for your web server.
   b. In the **Sensor Placement** table, find the PHP General sensor and select the check box in the **Place** column to enable it.
5. Enable capturing of PHP scripts compilation time.
   a. Go to the **Sensor Configuration** section for your web server.
   b. Find the PHP General sensor in the list and access the properties screen using the provided **Properties** link.
   c. Select **Show compilation time** check box.
6. Enable capturing information about files called by include, include_once, require, or require_once functions.
   In the PHP General properties pop-up window, select the **Show include, include_once, require** or **require_once functions** check box.
7. Click **OK** to save the settings.

To place and configure the PHP General sensor, perform the following steps:

Sensor configuration

The PHP General sensor captures compilation time for each loaded PHP file. Additionally, it enables you to see which files were called by **include**, **include_once**, **require**, or **require_once** functions in other PHP files, and how much time it took to load each file.

PHP Database sensors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Sensor configuration

Captured information

The section provides information about following a group of sensors used for database instrumentation:

- PHP MySQL Sensor (it covers MySQL and MySQLi interfaces for both MySQL and MariaDB databases, for both procedural and object oriented calls)
- PHP Oracle Sensor
- PHP PDO Sensor
- PHP PostgreSQL Sensor

Each of these sensors enables you to see how the connection to the database is established and captures SQL queries issued by PHP for different database interfaces.

Sensor configuration

Perform the following steps to enable monitoring of PHP database queries:

1. Start the AppMon Client.
2. Open the **System Profile Preferences** window for your application. Then under **System Profiles**, right-click your application and choose **Edit System Profile**.
3. Enable PHP monitoring. To do this, navigate to **General** settings for your application, and select the PHP check box in the **Technology** section.
4. Click **Apply** to save the configuration.
5. Enable monitoring of database queries.
   a. Go to **Sensor Placement** section for your Agent Group.
   b. In the **Sensor Placement** table, find required PHP Database (MySQL, PostgreSQL, PDO, or Oracle) sensor and select the check box in the **Place** column.

6. Optionally modify the default length of captured SQL commands.
   To do this, right click you System Profile, and select **Edit System Profile > Agent Group/tier with PHP technology > Sensor Configuration** and then select the required PHP database sensor in the **Sensor Configuration** list. Change the maximum number of characters if needed. Click **OK** to save the settings.

Properties

Maximum SQL capture length: Limits the captured SQL statements to the specified length. All symbols, exceeding the limit, are cut. Default value is 512.

Captured information

The PHP agent instruments database actions as follows:

MySQL

All MySQL functions set API to **Database**.

Function	Description
mysql_query	Captures the query.
mysql_unbuffered_query	Captures the query.
mysql_db_query	Captures the query and reports database name as an argument.
mysql_connect	Reports all arguments. When a password is used as an argument it is always masked with * (asterisk) characters.
mysql_close	Processes the query, but does not present it as node within a PurePath.

See PHP documentation for an explanation of MySQL functions.

MySQLi (MySQL improved)

All MySQLi functions set API to **Database**; it applies to both conventions: the object-oriented style and the procedural style.

Function	Description
mysqli::query mysqli_query	Captures the query.
mysqli::multi_query mysqli_multi_query	Captures the query.
mysqli::real_query mysqli_real_query	Captures the query.
mysqli::prepare mysqli_prepare	Captures the query.
mysqli_stmt::prepare mysqli_stmt_prepare	Captures the query.
mysqli::mysqli	Instruments establishing connection to database. Reports all the arguments.
mysqli::real_connect mysqli_real_connect	Instruments establishing connection to database. Reports all the arguments.
mysqli_stmt::bind_param mysqli_stmt_bind_param	Processes the query, but does not present it as a node within a PurePath.
mysqli::close mysqli_close	Processes the query, but does not present it as a node within a PurePath.

See PHP documentation for an explanation of MySQLi functions.

PDO

All PDO functions set API to **Database**.

Functions	Description
PDO::exec	Captures the query.
PDO::prepare	Captures the query.

PDO::query	Captures the query.
PDO::__construct	Instruments establishing connection to database. Reports all the arguments.
PDO::bindParam	Processes the query, but does not present it as a node within a PurePath.
PDO::bindValue	Processes the query, but does not present it as a node within a PurePath.

See PHP documentation for an explanation of PDO functions.

PostgreSQL

All processed PostgreSQL functions set API to **Database**.

Functions	Description
pg_connect	Instruments establishing connection to database
pg_query	Captures the query.
pg_query_params	Captures the query.
pg_prepare	Captures the query
pg_execute	Captures the query
pg_connection_reset	Instruments re-establishing connection to database.
pg_close	Instruments closing connection to database, but does not present it as a node within a PurePath.

See PHP documentation for an explanation of PostgreSQL functions.

Oracle

All processed Oracle functions set API to **Database**.

Functions	Description
oci_connect	Instruments establishing connection to database
oci_new_connect	Instruments establishing connection to database
oci_parse	Captures the query.
oci_bind_by_name	Processes the query, but does not present it as a node within a PurePath.
oci_execute	Captures the query
oci_free_statement	Instruments statement's release
pg_connection_reset	Instruments re-establishing connection to database.
oci_close	Instruments closing connection to database, but does not present it as a node within a PurePath

See PHP documentation for an explanation of Oracle functions.

PHP Exceptions sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Sensor description

Sensor configuration

Sensor description

The PHP Exceptions sensor captures PHP run-time errors, warnings and notices, compile-time errors, user-generated error, message and notice messages. It is also capable of catching stack traces. See PHP documentation for more information about reported PHP errors and exceptions.

Sensor configuration

To start monitoring PHP exceptions.

1. Start the AppMon Client.
2. Open the **System Profile Preferences** window for your application.
   Under **System Profiles**, right-click your application and choose **Edit System Profile**.
3. Enable PHP monitoring.
   To do this, navigate to **General** settings for your application, and select the **PHP** check box in the **Technology** section.
4. Enable monitoring of PHP exceptions.
   a. Go to the **Sensor Placement** section for your agent group.
   b. In the **Sensor Placement** table, find the PHP Exceptions sensor and select the check box in the **Place** column to enable it.

5. Select error and logging types that you want to monitor.
   a. Go to the **Sensor Configuration** section for your agent group.
   b. Find the PHP Exceptions sensor in the list and access the properties screen using the provided **Properties** link.
   c. Select the exceptions that you want to capture.
6. Optionally decide if you want stack trace capture to be active. You can control the depth of the reported stack traces in the **Maximum captured Stacktrace depth** field. The maximum value allowed is 256. Stack trace capturing is enabled by default and set to a safe value of 32.
7. Click **OK** to save the settings.

> **Note**
>
> Capturing stack traces causes high Agent overhead if large number of exceptions occurs in the instrumented application. Use the stack trace capturing with care and consider limiting the depth of grabbed stack traces.

Properties

- **Configuration Properties**: Select the exceptions you want to capture.
- **Overhead properties**: Enable or disable capture of stack traces for exceptions and fine-tune the amount of data that is captured and transferred from the Agent to the Server.
  - **Capture Stacktrace**: Enables capturing of stack traces.
    Capturing the stack and transferring the data to the server causes high CPU overhead on the instrumented application and consumes network bandwidth.
  - **Maximum Captured Stacktrace depth**: Limits the number of elements per stacktrace. Only the topmost elements are transferred.
    The allowed range is 0—256. The default value is 32.

PHP Logging sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Sensor configuration

The PHP Logging sensor captures Zend Framework run-time log messages of the following levels: emergency, alert, critical, error, warning, notice, info, and debug. See Zend Framework Logging documentation for more information about the Zend Framework logging system.

Sensor configuration

Perform the following steps to start monitoring Zend Framework logging functions.

1. Start the AppMon Client.
2. Open the **System Profile Preferences** window for your application. Under **System Profiles**, right-click your application and choose **Edit System Profile**.
3. Enable PHP monitoring. To do this, navigate to the **General** settings for your application, and the select the **PHP** check box in the **Technology** section.
4. Enable monitoring of PHP logging.
   a. Go to the **Sensor Placement** section for your agent group.
   b. In the **Sensor Placement** table, find the PHP Logging sensor and select the check box in the **Place** column to enable it.
5. Select logging level that you want to monitor.
   a. Go to the **Sensor Configuration** section for your agent group.
   b. Find the PHP Logging sensor in the list and access the properties screen using the provided **Properties** link.
   c. Select the logging levels that you want to capture.
6. Click **OK** to save the settings.

Properties

Select the logging levels you want to capture.

PHP Server sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Sensor configuration

Sensor results

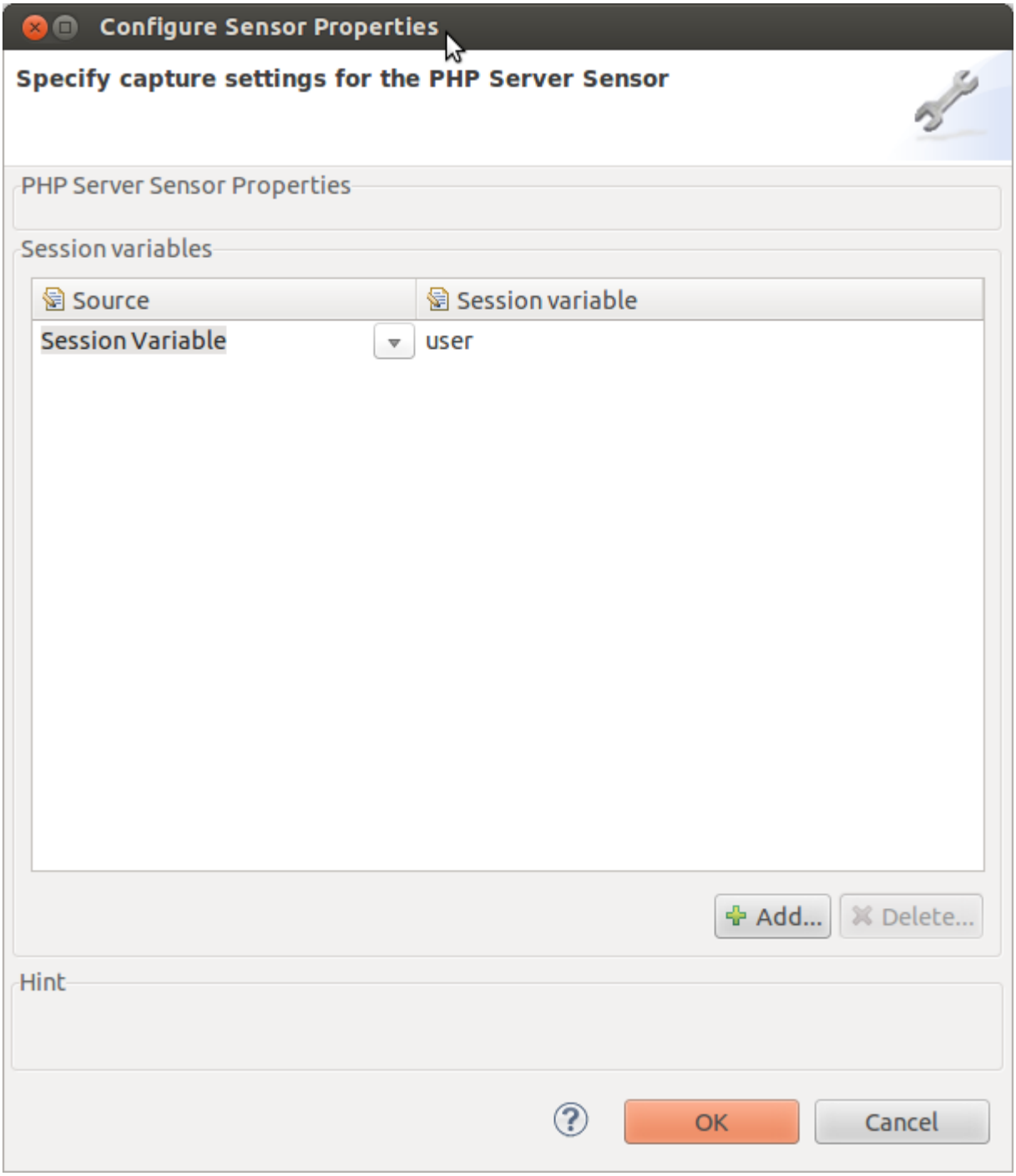
The PHP Server Sensor captures PHP session variable values.

Sensor configuration

Perform the following steps to place and configure the PHP Server sensor:

1. Start the AppMon Client.
2. Open the **System Profile Preferences** window for your application.
   Under **System Profiles**, right-click your application and choose **Edit System Profile**.
3. Enable monitoring of PHP scripts. Tpo do this, navigate to **General** settings for your application, and the select the **PHP** check box in the **Technology** section.
4. Place the PHP Server sensor.
   a. Go to the **Sensor Placement** section for your web server.
   b. In the **Sensor Placement** table, find the PHP Server sensor and select the check box in the **Place** column to enable it.

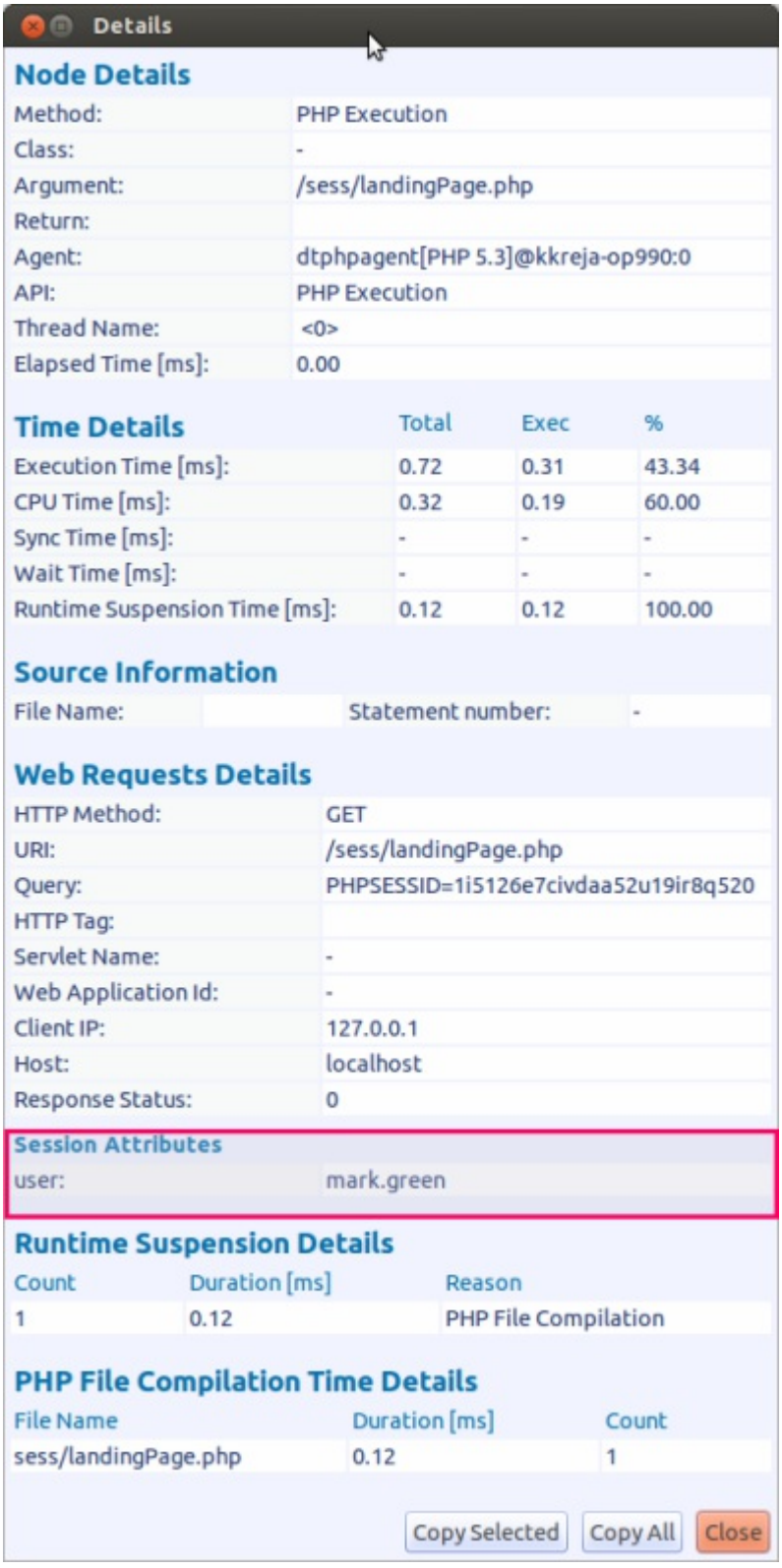
5. Enable capturing of a specific PHP session variable.
   a. Go to the **Sensor Configuration** section for your web server.
   b. Find the PHP Server sensor in the list and access the properties screen using the provided **Properties** link.
   c. Enter the name of the session variable you would like to capture. The following graphic shows a sample setting for capturing the value of the user session variable.



6. Click **OK** to save the settings.

Sensor results

The details of the **PHP Execution** node contain the captured session variable, as shown in the following graphic:



Details window of the PHP execution node shows the captured session variable

PHP Unit sensor

The sensor doesn't have any user configurable properties.

Sensor configuration

The PHP Unit sensor captures information from the PHPUnit test framework.

> **Note**
>
> This sensor does not support OneAgent.

PHP Web Requests sensor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Captured information

Sensor configuration

PurePaths with and without web requests sensor placed

The PHP Web Request (formerly cURL) sensor allows you to track PHP external calls. Use this sensor if your applications depend on calling external servers. With all environment elements instrumented with an AppMon Agent, you should be able to see who is calling whom and why and what is executing. This allows you to see entire application delivery chain as one (as opposed to atomized paths for each web server).

With the Web Requests sensor in place for the modular infrastructure, the communication between each component, the number of calls made, the call rate, protocols used and the impact is shown as one logical unit.

Captured information

- PHP cURL functions
- PHP cURL function arguments
- PHP SOAP functions
- PHP SOAP function arguments
- PHP file_get_contents function
- PHP file_get_contents function arguments
- PHP readfile function
- PHP readfile function arguments
- PHP fopen function
- PHP fopen function arguments
- PHP file function
- PHP file function arguments

- External PHP calls

The full URL is always reported as an argument in PurePath dashlet. The **Transaction Flow** shows all remote calls made to other web servers.
If you have AppMon Agents deployed for each web server that takes part in communication, you can see all file and function executions as a single PurePath.

Sensor configuration

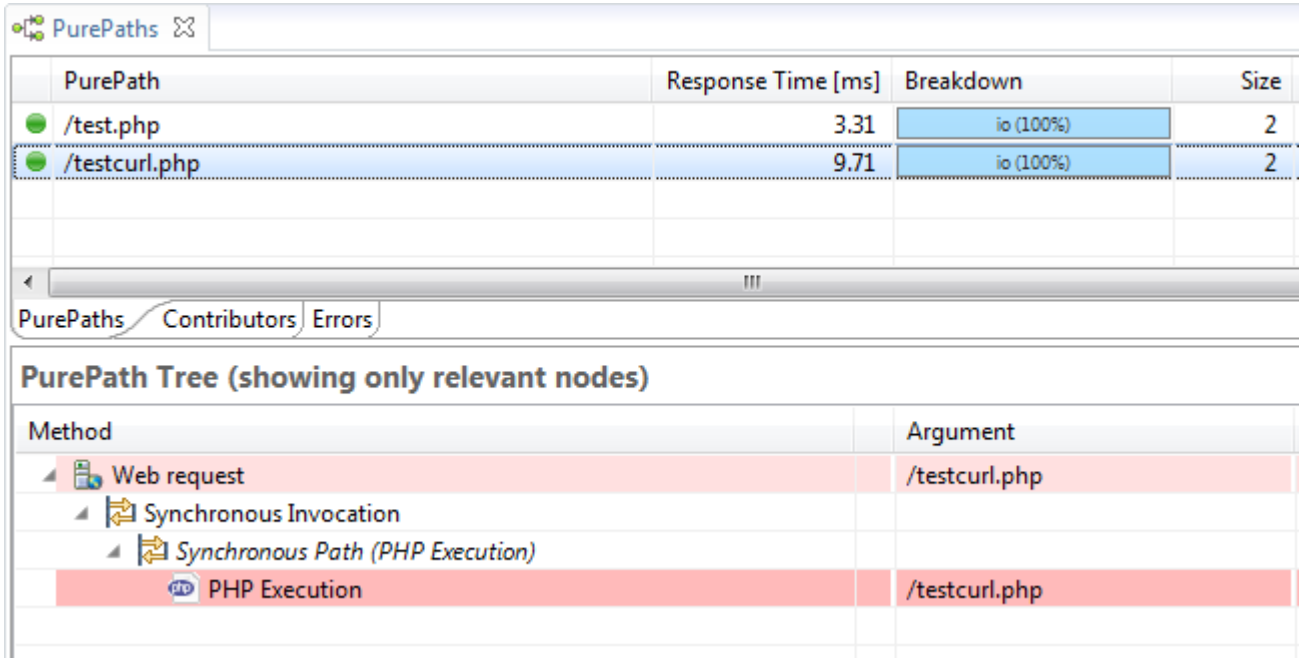
To place and configure the PHP Web Requests sensor:

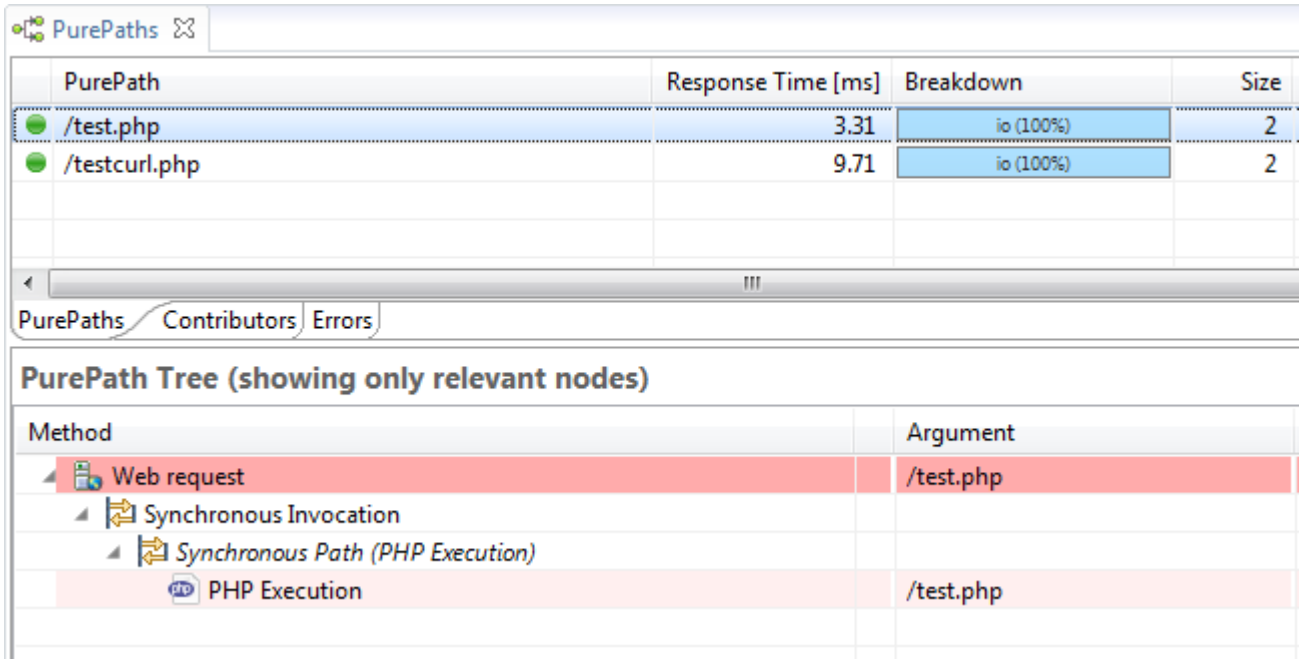
1. Start the AppMon Client.
2. Open the **System Profile Preferences** window for your application. Then under **System Profiles**, right-click your application and choose **Edit System Profile**.
3. Enable PHP scripts monitoring. To di this, navigate to **General** settings for your application, and select the PHP check box in the **Technology** section.
4. Place the PHP Web Requests sensor.
   a. Go to the **Sensor Placement** section for your agent group.
   b. In the **Sensor Placement** table, find the **PHP Web Requests** sensor and select the check box in the **Place** column to enable it.
5. Click **OK** to save the settings.

PurePaths with and without web requests sensor placed

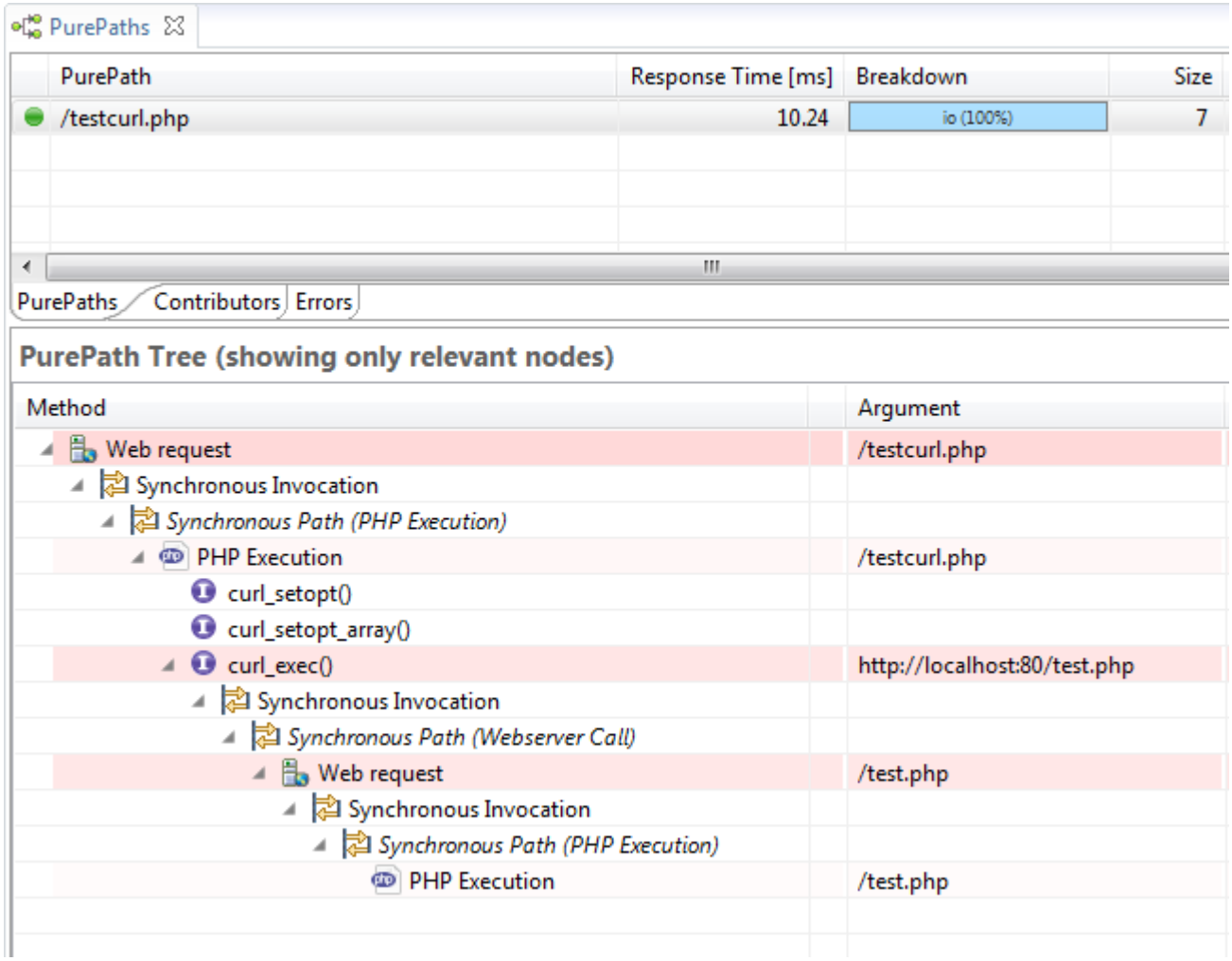
When a PHP code contains a cURL, or any other listed above, execution functions referring a remote file:

- Without Web Requests sensor placement, you see separate PurePaths reported for each PHP file.





- With Web Requests sensor placement, you see a single PurePath with all invocations combined.



JavaScript Agent

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Agent injection

Analysis performed by the JavaScript Agent

Action name detection

Bandwidth detection

Related pages

How does UEM work?

How to instrument a web application

System Profile - User Experience

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

The JavaScript diagnoses Browser environments. It is a script that can be automatically injected into web pages (of mime type `text/html`) via Java or Web Server Agents – or manually, by using the `<script>` tag in the page header, at design time if not otherwise possible.

Agent injection

To instrument a website, two script tags have to be injected: an inline tag and a tag with a `src`-attribute. The inline tag has to be injected as the very first script and consists of essential instrumentation parts of the JavaScript Agent. For best capturing experience it is recommended that also the second tag is injected right after the inline part, but before any other JavaScript.

Asynchronous injection

For AppMon 6.5 and later, the JavaScript Agent injects asynchronously by default using the `defer` script tag attribute (see MDN Documentation for details).

Asynchronously loaded scripts do not block page loads, but there are a few drawbacks. Since the Agent cannot redefine any library variables if they already have been initialized (like jQuery or AngularJS), it is strictly recommended that `defer` is also used for any third party frameworks to be instrumented. The JavaScript Agent must be available before those frameworks start initializing, otherwise it is not possible to capture actions triggered by scripts. You can always disable asynchronous injection by clearing the **Load the JavaScript Agent asynchronously** check box in **<System Profile> > User Experience > Web applications**.

Automatic injection

The following is a sample tag-combination automatically injected by the Agent as first of script tags in `<head>` at delivery of the web page – with settings according to the configuration in **<System Profile> > User Experience**: `<script type="text/javascript">(function(){var a=window;a.dT_?a.console&&a.console.log("Duplicate agent injection detected, turning off redundant initConfig."):window.dT_||(window.dT_={cfg:"tp=0,50,10|lab=1|reportUrl=dynaTraceMonitor|agentUri=/ajax/dtagent_pqtx_0000000.js|auto=1|domain=#DOMAIN#|rid=RID_1640928385|rpid=1646377341|app=#APP#"})})();...</script><script type="text/javascript" src="/dtagent_bdpx_0000000.js"></script>`

The first tag (without `src`-attribute) is the agent's *initialization code*. It must be delivered inline and is executed synchronously. It performs early-wrapping of native browser functions and third party libraries if available to ensure capturing can be done reliably.

The second tag requests the part of the JavaScript Agent that can be loaded asynchronously and therefore won't block a page from being loaded. The filename defines it's version and which parts of the Agent should be delivered:

- `dtagent` is the JavaScript Agent Name set in **<System Profile> > User Experience > Global Settings**.
- `7000000001289` is the AppMon Agent version (major, minor, revision, buildnumber).
  Each part of the version string consists of four characters, trimming leading zeroes, such that it is read like this: `7 0000 0000 1289`, which translates to `7.0.0.1289`.
- `bdpx` is the feature hash, and shows information captured by the Agent, such as bandwidth, dojo, perceived render time and basic XHR detection.

Disabling JavaScript Agent Injection for a Specific Files / Locations

You can disable JavaScript Agent injection for specific files and/or locations.

To disable injection:

1. Right click your system profile and select **Edit System Profile -> your agent group -> Sensor Configuration item -> User Experience**, and click **Properties**.
2. In the **URI-specific injection behavior** area, click **+**.
3. Specify the file or location to be excluded from the injection, in the **Condition** and **URI pattern** fields. See User Experience sensor properties for parameters description.

4. From the **Injection Rule** list, select **do not inject on this agent**, to allow agents from other groups to inject here or **do not inject on this and subsequent agents**, to completely disable agent injection.

Manual injection

If the preferable auto-injection is not possible you need to insert the initialization tag manually at design time as first script tag in `<head>`. Since the initialization tag and it's contained code varies depending on your application settings, it can easily be retrieved by using the Server REST Interfaces API using https://<AppMon server>:8021/api/v1/profiles/<systemprofile>/applications/<application>/javascriptagent/initcode

The initialization code performs operations to load the JavaScript Agent (either with or without the `defer` attribute) and updates it's configuration if required. It is not necessary to update the initialization tag every time you change the UEM configuration, though adjustments are applied faster for new visitors if you do so.

Analysis performed by the JavaScript Agent

- Detection of page loads.
- Detection of user actions: A user action can be a page load or an Ajax call that the user initiates by a mouse click, key press, etc. Cascading calls are correlated into one single user action, for example a page load that performs several Ajax calls during load.
- Browser errors.
- The client's bandwidth.
- User action details such as action duration and page load detail timings for each action step, including browser timings, visually complete, and speed index.

The JavaScript Agent analyzes the above browser and user behaviors on the client side and captures actions if the corresponding event handler is registered.
It sends this information to the instrumented web server using several parameters in the monitor signal / simple XHR / Ajax request. The web server is named `dynaTraceMonitor` by default – resulting in a path relative to the instrumented server. Set the items in **<System Profile> > User Experience**. Set the monitor request name in the **Global Settings** tab. Set the monitor request path in the **<applicationName>**.

By default the signal is sent immediately after a page was successfully loaded or an Ajax action is finished. If a user navigates to another page before the page finished loading, the signal is sent during a beforeunload or unload event.

Action name detection

The JavaScript Agent tries to find a meaningful name for an action. To do this, it checks several properties (such as inner HTML, caption, or hint) of the HTML element (such as button or anchor) that triggers the action. It also tries to get the caption if there is a more complex HTML structure with multiple nested tags.

Set Action Name with Custom Attribute `data-dtname`

If the standard action name detection does not suffice you can set the custom attribute `data-dtname` within the HTML tags and use it as a caption. For example:

```
<label for="txtFirstname">Firstname</label> <input data-
dtname="Firstname Text Input" type="text" value="firstname"
name="firstname" title="Firstname" id="txtFirstname" />
```

The above leads to the following caption:

```
click on "Firstname Text Input"
```

Resolving captions for actions

As mentioned above, there are several mechanisms how the JavaScript Agent decides which name fits an action best.
It starts with the innermost HTML node that's clicked (such as a button, image tag, link…) and checks the following things in order of precedence):

1. The attribute named `data-dtname`.
2. The nodeName (such as image, anchor, input, …), returns if `html`, `body` or `head` tag or the `document` element is found.
3. The innerText / textContent.

If none of these return any reasonable result, the Agent starts a recursive algorithm that checks different things depending on the nodeName of the currently checked HTML node. If nothing is found, the parent node is checked.

Bandwidth detection

To detect a client's bandwidth, the JavaScript Agent requests signals of different sizes from the server. These GET-requests force the Agent to generate images of a specific file size and send them back to the browser. The bandwidth is calculated according to the loadtime of these images. To reduce failures and guarantee the validity of the results, we recommend that you allow GET-requests on `dynaTraceMonitor?bwstate=*.` Do not gzip or compress the requests. This reduces the file size of the generated images and the time it takes the clients to download them and such biases bandwidth to be better than it actually is.

Which features is it needed for?

- Bandwidth measure
- Network contribution time measure, calculated based on bandwidth and on resource size which is measured on the server side
- User experience index for page actions for bandwidths lower than broadband
  The threshold is multiplied by a factor between 1 and 2. Max. factor is 2.

How is it implemented?

- Bandwidth detection triggers 2 seconds after the JavaScript Agent was loaded
- The JavaScript Agent loads a series of fixed size images (0k, 3k, 10k, 30k, 100k, 300k, 1000k) and measure the time
- First (0k) image is used for latency calculation, which are subtracted from all other measurements
- All subsequent images are loaded until the download time without latency exceeds 100ms

What can be configured?

- Bandwidth detection can be enabled per application (default: off)
- Can be enabled separately for mobile browsers (default: off)
- Frequency of bandwidth checks can be configured (default: every 5min)

What is the impact? The bandwidth detection can have an impact on the user experience, because it might be triggered while the page is still loading. Even if there is no impact on the response time, synthetic tools still might consider the timings of the bandwidth check images for their calculation of the response time.

- Typical impact on the response time: **0 - 100ms** (depending on individual page performance)
- Impact on the total resource size of a page: **up to 1443k**

The bandwidth feature requires an instrumented agent to get the generated images, thus it just measures the bandwidth between the client and the Java/Webserver Agent. If the latter is located elsewhere, for example in a UEM only scenario with CORS, the measured bandwidth does not reflect the connection speed between the webserver that delivered the actual page and the browser.

zOS Agent

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

CICS Agent

IMS Agent

zDC (data collection subsystem)

zLocal Agent

zRemote Agent

Java Agent(s)

Related pages

CICS Agent installation and configuration

CICS Transaction Server sensor

Transaction measure

DTAX transaction

zDC

zOS Agent logs

zOS Agent messages

zOS Agent troubleshooting

Enterprise tier setup

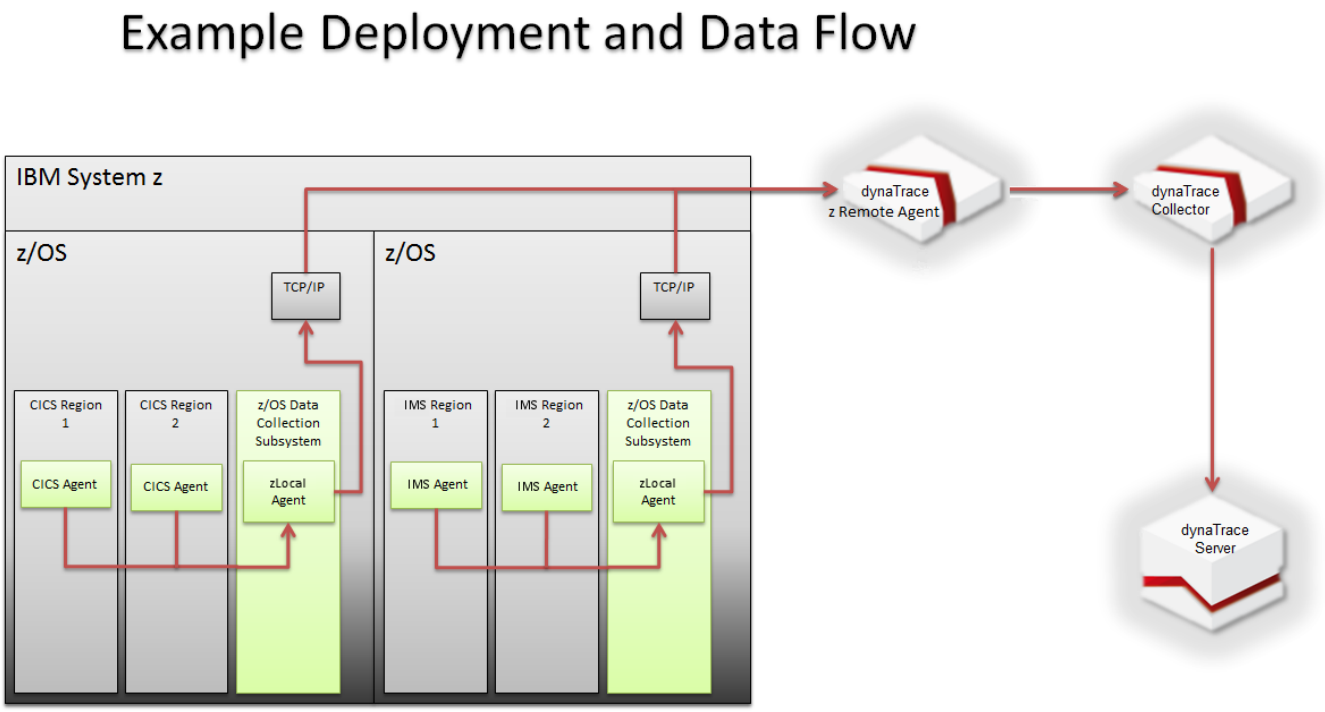
CICS transaction gateway

WebSphere Message Broker

WebSphere MQ tagging

WebSphere MQ mapping

The z/OS Agent extends PurePath transaction-based tracing to a mainframe environment.



While the term *z/OS Agent* sounds like a single entity, it includes the following individual agent components:

CICS Agent

The CICS Agent installs into every CICS region that you want to monitor. It includes a PLT program that initiates at CICS startup, and an Agent program that captures various events and forwards them to the zDC.

The CICS Agent can either start new PurePaths for specific transaction codes or link CICS transaction PurePaths to distributed PurePaths for transactions initiated using WebSphere MQ, CICS Transaction Gateway, or SOAP (over HTTP). The CICS agent supports DB2, DL/I and MQ API calls tracing, as well as any programs invoked via CICS LINK or LE dynamic call. Linked PurePaths are also provided with child transaction details started for DPL LINK or START TRANSACTION requests within a PurePath.

IMS Agent

The IMS Agent must be injected into the Control Region of each IMS system for which you want to capture Purepaths. It consists of programs that capture data on various IMS transaction processing events, and forwards that information to the zDC.

The IMS Agent traces IMS transactions submitted from the IMS MQ Bridge or MQ Trigger Monitor, IMS TM Resource Adapter, the IMS SOAP Gateway, and the IMS Connect API.

zDC (data collection subsystem)

The zDC (z/OS Data Collection subsystem) handles the life cycle of the zLocal Agent, formerly called the zUNIX Agent or USS (Unix System Services) Agent. It controls communication between the zLocal Agentand CICS and IMS Agents. You can enter Operator start, stop, and modify commands from a system console to control the zDC and the lifecycle of the zLocal Agent.

zLocal Agent

The zLocal Agent and zRemote Agent handle the TCP/IP connection between the mainframe and the AppMon Collector. The zLocal Agent installs on the mainframe and passes performance data to the zRemote Agent, reducing CPU consumption on z/OS.

> **Note**
>
> The z/OS Agent can't directly connect to an AppMon Collector without first going through a zRemote Agent. Using the zRemote Agent reduces CPU consumption on z/OS. The zLocal Agent only transfers data to the zRemote Agent, which can run on Windows or Linux (64-bit OS only).

zRemote Agent

The zRemote Agent is an aggregation and data processing component that installs in the distributed environment and extends the zLocal Agent. The zRemote Agent offloads some of the Agent data processing in the zLocal Agent, reducing the MIPS load incurred in instrumenting mainframe applications.

> **Note**
>
> The recommended best practice is to have the zRemoteAgent only service one zLocalAgent (zDC). Refer to the instructions on installing multiple zRemoteAgent launcher service instances for configuring multiple zRemoteAgents.

Java Agent(s)

You must also install AppMon Java Agents to monitor business transactions that use either the CICS Transaction Gateway or the IMS Soap Gateway.

Installing the z/OS Agent

On the target mainframe, the mainframe systems programmer can download and install z/OS SMP/E maintenance

z/OS Agent maintenance

1. A distributed systems installer edits the AppMon System Profile(s) to add the mainframe tier(s).
2. The distributed systems installer installs the zRemote Agent.
3. On the target mainframe, the mainframe systems programmer downloads and installs the z/OS components.
4. Your mainframe security administrator configures security for the zDC.
5. If monitoring one or more CICS regions, your mainframe CICS systems programmer installs the CICS Agent.
6. If monitoring one or more IMS control regions, your mainframe IMS systems programmer installs the IMS Agent.

z/OS Agent installation summary

Depending on your team, a single individual may be able to administer more than just one of the required areas/regions of the mainframe.

- **Distributed Systems Installer** – Installs the AppMon Server and the zRemote Agent.
- **Mainframe Systems Programmer** – Installs the zLocal Agent and the SMP/E installation of the product.
- **Mainframe CICS Systems Programmer** – Installs the CICS Agent and tables in each of the CICS regions to be monitored.
- **Mainframe IMS Systems Programmer** – Injects the IMS Agent into each IMS control region to be monitored.
- **Mainframe Security Administrator** – Sets up security for the zDC.

Efficiently z/OS Agent installation is typically a team effort involving the following:

Before you install the z/OS Agent

Before you install the z/OS Agent

Since the z/OS Agent consists of multiple components in both the distributed and mainframe environments, the installation and configuration process is more complex than for other Agents. However, with a bit of planning and coordination with the varied architecture groups, your z/OS Agent installation can go smoothly.

Download and SMP/E install z/OS Agent components

4. Run the modified SMPINSTL job. If you use the default high level qualifier, the following datasets are created::

```
DT.LZDT700.GLOBAL.CSI
DT.LZDT700.SMPLOG
DT.LZDT700.SMPLTS
DT.LZDT700.SMPMTS
DT.LZDT700.SMPPTS
DT.LZDT700.SMPSCDS
DT.LZDT700.SMPSTS

DT.LZDT700.TZONE.CSI
DT.LZDT700.SZDTAUTH
DT.LZDT700.SZDTIMSA
DT.LZDT700.SZDTLOAD
DT.LZDT700.SZDTSAMP

DT.LZDT700.DZONE.CSI
DT.LZDT700.AZDTLOAD
DT.LZDT700.AZDTSAMP
```

   If the job ends with a RC of 4, the SMP/E portion of the installation is a success. The installation process creates and populates four target libraries. If you use the default high level qualifier, the target dataset names are:

   - `DT.LZDT700.SZDTAUTH`, which contains zDC load modules.
   - `DT.LZDT700.SZDTLOAD`, which contains CICS agent load modules.
   - `DT.LZDT700.SZDTIMSA`, which contains IMS and IMS Connect agent load modules.
   - `DT.LZDT700.SZDTSAMP`, which includes sample JCL and CICS RDO definitions.

3. If you changed the default HLQ in the download job, make the same change to the SMPINSTL job and any subsequent maintenance download or install jobs.

2. Run the Modified download JCL. If you use the default high level qualifier, the following datasets are

created:

```
DT.DTMF.LZDT700.F1
DT.DTMF.LZDT700.F2
DT.DTMF.LZDT700.F3
DT.DTMF.LZDT700.MCS
DT.DTMF.LZDT700.MNT
```

The `DT.DTMF.LZDT700.F2` PDS contains JCL for the SMP/E portion of the install and subsequent maintenance. After the SMP/E APPLY step has been run, copies of these members may be found in the SZDTSAMP dataset. The following members are provided:

Member	Description
`FTPINSTL`	A copy of the FTP download job above
`FTPMAINT`	A job to download cumulative PTF maintenance only. Note that all currently available maintenance is also delivered with the product install above.
`RFNINSTL`	A copy of the RFN download job above
`RFNMAINT`	A job to download cumlative PTF maintenance only through RFN commands. This member needs to be updated with the SET HASH value in the aforementioned RFNINSTL job. As the hash value changes, it will be updated in this documentation. Note that all currently available maintenance is also delivered with the product install above.
`SMPINSTL`	An SMP/E job to complete the product install.
`SMPMAINT`	An SMP/E job to apply cumulative PTF maintenance to the target libararies.

1. Determine the desired high level qualifier for the install dataset names, and set the HLQ variable accordingly in the JCL below. If using the RFN process, set a USS directory for temporary files required by SMP/E and the VOLSER where you want the install datasets created.

```
//FTPINSTL JOB ('ACCTINFO'),'User name or comment',NOTIFY=&SYSUID,
//            MSGLEVEL=(1,1),CLASS=A,MSGCLASS=X
//*
//* This job obtains Dynatrace 7.0 install XMIT files via FTP
//* and issues the TSO RECEIVE command to unpack them.
//*
```

```
//
//* Change the JOB card and the HLQ on the SET statement below to meet
//* site standards. If the default HLQ is changed, a matching change
//* is required when running subsequent install and maintenance jobs.
//*
// EXPORT SYMLIST=(HLQ)
// SET HLQ='DT'             <===== SET HLQ HERE
//*
//* For use with z/OS V1, delete the lines above, remove SYMBOLS= from
//* the SYSIN DD, and issue the edit command: C &HLQ. <newHLQ>
//*
//FTP     EXEC PGM=FTP,DYNAMNBR=20,PARM='(EXIT TIMEOUT 99',TIME=5
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *,SYMBOLS=EXECSYS
us-et-ftp01.dynatrace.com
Dynatrace70
Dynatrace70
binary
locsite recfm=fb lrecl=80 blksize=27920
locsite tracks primary=5 secondary=5
get MCS '&HLQ..DTMF.LZDT700.MCS.XMIT' (REPLACE
locsite tracks primary=1 secondary=1
get F1 '&HLQ..DTMF.LZDT700.F1.XMIT' (REPLACE
locsite tracks primary=30 secondary=15
get F2 '&HLQ..DTMF.LZDT700.F2.XMIT' (REPLACE
locsite tracks primary=45 secondary=15
get F3 '&HLQ..DTMF.LZDT700.F3.XMIT' (REPLACE
locsite tracks primary=75 secondary=15
get MNT '&HLQ..DTMF.LZDT700.MNT.XMIT' (REPLACE
quit
/*
//*
//RECEIVE  EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(0,NE)
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  *,SYMBOLS=EXECSYS
 RECEIVE INDSN('&HLQ..DTMF.LZDT700.MCS.XMIT') NODISPLAY
 DATASET('&HLQ..DTMF.LZDT700.MCS') +
 RELEASE
 RECEIVE INDSN('&HLQ..DTMF.LZDT700.F1.XMIT') NODISPLAY
 DATASET('&HLQ..DTMF.LZDT700.F1') +
```

```
 RELEASE
 RECEIVE INDSN('&HLQ..DTMF.LZDT700.F2.XMIT') NODISPLAY
 DATASET('&HLQ..DTMF.LZDT700.F2') +
 RELEASE
 RECEIVE INDSN('&HLQ..DTMF.LZDT700.F3.XMIT') NODISPLAY
 DATASET('&HLQ..DTMF.LZDT700.F3') +
 RELEASE
 RECEIVE INDSN('&HLQ..DTMF.LZDT700.MNT.XMIT') NODISPLAY
 DATASET('&HLQ..DTMF.LZDT700.MNT') +
 RELEASE
/*
//XMITDEL  EXEC PGM=IDCAMS,REGION=1024K,COND=(0,NE)
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *,SYMBOLS=EXECSYS
 DEL &HLQ..DTMF.LZDT700.MCS.XMIT PURGE
 DEL &HLQ..DTMF.LZDT700.F1.XMIT PURGE
 DEL &HLQ..DTMF.LZDT700.F2.XMIT PURGE
 DEL &HLQ..DTMF.LZDT700.F3.XMIT PURGE
 DEL &HLQ..DTMF.LZDT700.MNT.XMIT PURGE
/*
//
```

```
//RFNINSTL JOB ('ACCTINFO'),'User name or comment',NOTIFY=&SYSUID,
//             MSGLEVEL=(1,1),CLASS=A,MSGCLASS=X
//*
//* This job obtains Dynatrace 7.0 install pax files via RFN. Step 1
//* executes GIMGTPGK to receive the pax files from the FTP server
//* and store them in a USS directory. Step 2 executes GIMUNZIP to
//* to unpack the pax files.
//*
//* Change the JOB card and the SET statements below to meet
//* site standards. If the default HLQ is changed, a matching change
//* is required when running subsequent install and maintenance jobs.
//*
//* The hash value below should be correct for the current product
//* cumulative maintenance level, which changes monthly.
//*
// EXPORT SYMLIST=(HLQ,USSDIR,VOLUME,HASH)
// SET HLQ='DT'                  <===== SET HLQ HERE
// SET USSDIR='/u/myuserid'      <===== SET USS home/temp directory
```

```
// SET VOLUME='VOLSER'           <===== SET VOLSER
// SET HASH='1066717A6CC9326A51BD15E473CABE823CD6034D'
//*
//* For use with z/OS V1, delete the lines above, remove SYMBOLS= from
//* the SYSIN DD, and issue the edit command: C &HLQ. <newHLQ>
//*
//* Run GIMTGTPKG to obtain the install package from the FTP site
//*
//GIMGTPKG EXEC PGM=GIMGTPKG,REGION=0M
//SMPOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SMPNTS   DD PATH='&USSDIR',PATHDISP=KEEP
//SMPCPATH DD PATH='/usr/lpp/smp/classes/',PATHDISP=KEEP
//SMPJHOME DD PATH='/usr/lpp/java/J7.0/',PATHDISP=KEEP
//SMPSRVR  DD *,SYMBOLS=EXECSYS
<SERVER host="us-et-ftp01.dynatrace.com"
        user="Dynatrace70"
        pw="Dynatrace70">
  <PACKAGE file="/GIMZIP/GIMPAF.XML"
           hash="&HASH"
           id="DT70">
  </PACKAGE>
</SERVER>
/*
//*
//* Run GIMUNZIP to unpack the pax files
//*
//INSUNZIP EXEC PGM=GIMUNZIP,REGION=0M,COND=(0,NE)
//*
//SYSUT3   DD UNIT=SYSALLDA,SPACE=(CYL,(50,10))
//SYSUT4   DD UNIT=SYSALLDA,SPACE=(CYL,(25,5))
//* Message output datasets:
//SMPOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SMPDIR   DD PATH='&USSDIR/DT70',PATHDISP=KEEP
//SMPCPATH DD PATH='/usr/lpp/smp/classes/',PATHDISP=KEEP
//SMPJHOME DD PATH='/usr/lpp/java/J7.0/',PATHDISP=KEEP
//SYSIN    DD *,SYMBOLS=EXECSYS
<GIMUNZIP>
  <ARCHDEF
    name="SMPPTFIN/S0001.DT.D070000.MEDIA.PROD.RTME.LZDT700.MNT.pax.Z"
```

```
    name="SMPPTFIN/S0001.DT.R070000.MEDIA.PROD.DTMF.LZDT700.MNT.pax.Z"
    volume="&VOLUME"
    newname="&HLQ..DTMF.LZDT700.MNT">
   </ARCHDEF>
   <ARCHDEF
    name="SMPPTFIN/S0002.DT.R070000.MEDIA.PROD.DTMF.LZDT700.MCS.pax.Z"
    volume="&VOLUME"
    newname="&HLQ..DTMF.LZDT700.MCS">
   </ARCHDEF>
   <ARCHDEF
    name="SMPRELF/DT.R070000.MEDIA.PROD.DTMF.LZDT700.F1.pax.Z"
    volume="&VOLUME"
    newname="&HLQ..DTMF.LZDT700.F1">
   </ARCHDEF>
   <ARCHDEF
    name="SMPRELF/DT.R070000.MEDIA.PROD.DTMF.LZDT700.F2.pax.Z"
    volume="&VOLUME"
    newname="&HLQ..DTMF.LZDT700.F2">
   </ARCHDEF>
   <ARCHDEF
    name="SMPRELF/DT.R070000.MEDIA.PROD.DTMF.LZDT700.F3.pax.Z"
    volume="&VOLUME"
    newname="&HLQ..DTMF.LZDT700.F3">
   </ARCHDEF>
</GIMUNZIP>
/*
//
```

Note

This JCL takes advantage of inline symbols that first appeared in z/OS R 2.1

You can directly download the installation files for all mainframe z/OS Agent components to your mainframe from the Dynatrace FTP server. Use the sample JCL below to do this. There are two retrieval methods. The first method (JOB FTPINSTL) uses standard FTP procedures. The second method (JOB RFNINSTL) accesses the same FTP server but uses Receive From Network commands which requires an SMP/E computed hash value. The current hash value is included with the RFN JCL listed below. This hash value is also used for the RFNMAINT member located in the SZDTSAMP dataset. Refer to the SZDTSAMP member list further down the page.

zDC

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installing the zDC

zDC reference information

Running multiple zDC systems on a single z/OS image

zDC messages

Multiple TCP/IP stacks

Related pages

zOS Agent

zLocal Agent

zOS Agent messages

The zDC (z/OS Data Collection subsystem) handles the life cycle of the zLocal Agent, formerly called the zUNIX Agent or USS (Unix System Services) Agent. It controls communication between the zLocal Agentand CICS and IMS Agents. You can enter Operator start, stop, and modify commands from a system console to control the zDC and the lifecycle of the zLocal Agent.

Installing the zDC

The AppMon zDC runs as an authorized z/OS process (typically as a system task). This means the programs must reside in an authorized library. It is intended to start automatically as a started task (STC) at a system IPL. This gives the zDC continuously availability to collect PurePath data from CICS and IMS Agents and forward it to the AppMon Server.

The zDC can also be run as a batch job. The service class of the zDC must be high enough so that it is always available for keep-alive messages from the zRemoteAgent. The zDC should have a priority equal to or higher than the CICS application regions and the IMS dependent regions.

1. Download the z/OS components to the mainframe. Make a note of the high-level qualifier used in the download procedure. The sample high-level qualifier of `DT.DYNA` is used in this procedure. Substitute your value as needed.

2. Create the RACF user ID for the zDC process.
   This ID must have a z/OS Unix segment. The recommended home directory name is `/u/dt`. If a directory other than `/u/dt` is used, update the DTAGTCMD argument in the zDC parameter file (the SYSIN DD of the proc) to specify the correct path to the `dtzagent` binary, which defaults to `/u/dt/agent/lib64/dtzagent`. Only the `/u/dt/` part can be changed, because the path to the Agent binary must remain `agent/lib64/dtzagent`, regardless of the high-level qualifier used for this directory path. The `/u/dt` directory or the high-level qualifier that replaces it must be writeable by the zDC process. If the zDC is run as a started task, the ID must be capable of representing the zDC started task. Give the ID the same name as the zDC started task and add it to the access list of the RACF class STARTED with at least READ authority.

3. Change the dataset high-level qualifier represented by *hlq* in member `COPYAGNT` of `SZDTSAMP` to the value you used in the Download and SMP/E install of the z/OS Agent Components procedure.

4. Run the COPYAGNT job to create the required z/OS Unix subdirectories under the home directory of the zDC started task, copy the z/OS zLocal Agent binary file from `SZDTSAMP`, and set it to be executable. If using the default home directory of `/u/dt/`, the resulting file is `/u/dt/agent/lib64/dtzagent`. If the home directory has been changed, the paths in this job must be changed to correspond. File and directory names for z/OS zLocal are case-sensitive. The path names below the home directory cannot be changed.

   The `COPYAGNT` job should run under the user ID that was created for the zDC process, so that user ID owns the `dtzagent` file. If this is inconvenient, you can also use the `chown` and `chgrp` commands to reset the owning user and group for the `dtzagent` file after running `COPYAGNT`.

5. Authorize *hlq*.`LZDT700.SZDTAUTH.Substitute` *hlq* with the high-level qualifier value you used in the Download and SMP/E install of the z/OS Agent Components procedure. If you are also planning on using the IMS Agent, you must also authorize the `SZDTIMSA` library.

   For example, create a member named `SYS1.PARMLIB(PROGDT)` containing:

```
APF FORMAT(DYNAMIC)
APF ADD DSNAME(<hlq>.LZDT630.SZDTAUTH) VOLUME(XXXXXX)
```

Then issue the console command:

```
SET PROG=DT
```

6. Copy the ZDCMEPC sample zDC started task procedure from SZDTSAMP to a system proclib used for started tasks.
   The default name for the procedure is MEPC. Customize it as necessary for local standards. If a different subsystem name is chosen, change the SUBSYSTEM_ID() in the zDC SYSIN parameters to match. The proc and the two sysin members ZDCSYSIN and ZDCMEPCA that it uses contain dataset names that must be edited to replace HLQ. with the appropriate high-level qualifiers.

7. Update the zDC startup parameters that are supplied through the SYSIN DD of the zDC proc, which defaults to the `ZDCSYSIN` member of `SZDTSAMP`.

   - `DTAGTCMD(/u/dt/agent/lib64/dtzagent` – Update with the correct path to this program if the default was not used.
   - nobootstrap=false - Leave the default value of false if you want to use the bootstrap Agent. If you do not want to use a bootstrap Agent set the value to true. If this parameter is not specified the default is to use the bootstrap Agent.
   - `name=<agentname>` – This name does not require an agent mapping in a system profile. It is simply used for the name of the log files. This name should be reflective of the fact that this is a zlocalagent, and it could include the smfid of the lpar that it services. For example, “name=zlocalagent_<lpar>”. This name should be different than the corresponding zremoteagent, so that the log files can be easily distinguished. This agent does not connect to the AppMon Server directly.
   - zremoteagent=<host>[:port] - Update with the IP address of the zRemote Agent. The zRemote Agent is a required parameter. The port parameter is optional and defaults to 8898.

   See the NON-BOOTSTRAP Agent section for additional information on using the NON-BOOTSTRAP Agent.

   See zRemote Agent for additional information on how to set up the zRemote Agent.

8. Optional but recommended: Add a command to your system startup to automatically start the zDC subsystem at IPL.

zDC reference information

The following sections describe the basic operation and configuration parameters for the zDC.

dtzagent

The zDC manages the lifecycle of the z/OS Local Agent named `dtzagent` that runs in the z/OS Unix environment:

- It starts the z/OS Local Agent when the zDC initializes at system startup or is manually started.
- It terminates the Agent when the zDC is stopped.

The `COPYAGNT` job creates the z/OS Unix file containing this program.

libdtzagent.so - the NON-BOOTSTRAP Agent

If you want to run the `dtzagent` NON-BOOTSTRAP, it must be downloaded from the AppMon Server. Complete the instructions to start the zDC and connect it to the AppMon Server using the standard `dtzagent` binary. The first time you connect to the AppMon Server, do not use the `nobootstrap=true` parameter. If you do use this parameter the first time, the zDC will not start successfully. You can comment it out for this first execution.

Assuming that you have installed the product to `/u/dt` (if you have not, then replace this folder with your installation folder name), when the zDC successfully connects to the AppMon Server, the agent `libdtzagent.so` is copied to a folder `/u/dt/agent/downloads/rr.rr.rr.bbbb /native/zos-s390-64` (where `rr.rr.rr` is the release number and `bbbb` is the build number). In the `SZDTSAMP` dataset, the member `OCOPYAGT` is a job that copies the `libdtzagent.so` to the `/u/dt/agent/lib64` folder. When you uncomment or add the `nobootstrap=true` parameter In the `ZDCSYSIN`, the non-bootstrap agent that was copied into the `lib64` folder is used.

The following are examples of log messages in the zDC job output. The first example is with a bootstrap Agent. The second example is from a zDC using a non-bootstrap Agent.

When using the bootstrapped `dtzagent`:

```
2014-11-13 19:04:31 [009cb288] info    [native] Server/Collector requests us to
use Agent rr.rr.rr.bbbb with a hash of 67c137cc48e340da68e1062f58996e95
2014-11-13 19:04:32 [009cb288] info    [native] Loading Agent /u/dt/agent/downloads/rr.rr
```

Using `nobootstrap=true`, `libdtzagent.so`:

```
2014-11-14 23:40:18 [009cb288] info    [native] Configured to not download module - loadi
2014-11-14 23:40:18 [009cb288] info    [native] Loading Agent /u/dt/agent/lib64/libdtzage
```

System security

The `dtzagent` or `libdtzagent.so` creates a log directory and log files in its working directory, which defaults to the home directory specified in the user ID's RACF z/OS Unix segment, so write permissions are required for this location. Permissions assigned to files created by `dtzagent` or `libdtzagent.so` can be controlled by setting DT_UMASK. The default is umask(022).

No attempt is made by the zDC to limit which user IDs can enter operator commands.

The zDC uses TCP/IP sockets communication, so it may be necessary to update your security system's access rules if they are configured to deny TCP/IP access by default.

General SYSIN dataset format

Positions 1 through 71 of each statement can contain parameter data. Positions 72 through 80 are ignored and can contain a sequence number, but are not required to do so.

Each parameter takes the form KEYWORD(value). Each keyword and the requirements for the values associated with it are documented in the ZDCSYSIN member of the sample dataset SZDTSAMP.

zDC SYSIN parameters

All values that tailor the execution of the zDC reside in the dataset pointed to by DDNAME SYSIN.

The sample started task procedure (ZDCMEPC) refers to the ZDCSYSIN member of SZDTSAMP as shown below:

```
//SYSIN DD DISP=SHR,DSN=<hlq>.LZDT700.SZDTSAMP(ZDCSYSIN)
```

The dataset must contain 80 character records, blocked to a multiple of 80 bytes.

When a change is made to the SYSIN dataset, the ZDC must be stopped and restarted for the change to take effect.

Commonly used parameters

- `DTAGTCMD(/u/dt/agent/lib64/dtzagent)`
  - `nobootstrap=true` - The NON-BOOTSTRAP agent `libdtzagent.so`. If the nobootstrap=true parameter is not used, the bootstrap version of the z/OS Local Agent is loaded and attached by the zDC. Automatic updates of this binary are performed. This "bootstrap" Agent queries the AppMon Server to see if it includes a more current version of the z/OS Local Agent. If it does, that Agent is downloaded from the Server and used. This check is performed each time the zDC is started.
    `name=<Agent name>` - This name does not require an agent mapping in a system profile. It is simply used for the name of the log files. This name should be reflective of the fact that this is a zlocalagent, and it could include the SMF id of the LPAR that it services. For example, "name=zlocalagent_<lpar>". This name should be different than the corresponding zremoteagent, so that the log files can be easily distinguished. This agent does not connect to the Server.
  - `transactiontimeout=60`
  - `[zremoteagent=ipaddress:port])` – The port parameter is optional and defaults to 8898. See zRemote Agent for information on how to register the Windows service or start the Linux/Unix daemon with the right zdclistenerport, name, and server parameters.
  - loglevel=info - the loglevel used by the z/OS Local Agent
- `SUBSYSTEM_ID(MEPC)` – The name of the zDC subsystem. Specify the desired subsystem name here if MEPC is not appropriate.
- `DTLOGLEVEL(3)` – The logging level produced by the zDC. Set this parameter to `4` to suppress informational messages. It should not be set to less than `3` unless requested by Customer Support. Values less than `3` are used for diagnostic debugging.
- `DTMSG_SMOSIZE(10)` – A number that defines the maximum amount of storage (in Mbytes) for messages that can be queued in the zDC shared memory object pending their writing to the AppMon Agent. The default is 10 MB. This default value may need to be increased if very high volumes of transactions are traced.
- `DTCHDIR(/u/dt)` – Used to change the z/OS Unix current directory where `dtzagent` creates temporary files for `stdin`, `stdout`, and `stderr`. It defaults to the home directory of the associated user ID.

- `ZIIP_ENABLE(YES)` requests the zLocal agent to offload Agent message data processing and transmission to a zIIP CPU if available. ZIIP_ENABLE(NO) is the default. ZIIP_ENABLE(YES) will make virtually all of the CPU time that is consumed by the zDC to process Agent data eligible to run on a zIIP assist processor. The relatively small amount of background activity that occurs when the zDC is idle is not affected by this option and continues to run on a general CPU.
- `DTMSG_TRANBUFSIZE(n,m)` overrides the default number and size of Transaction buffers. Transaction buffers offer better performance for CICS and IMS agents by placing event messages related to each PurePath into dedicated buffers. Rather than send individual event messages for each PurePath as they occur, they are blocked into one or more buffers and sent together. Additionally, the buffer of related messages is processed more efficiently by the zremoteagent . The n parameter is the number of buffers in thousands. For example, 2 = 2000 buffers. A zero value disables Transaction buffers. The `m` parameter must be either `2` or `4` to specify 2K or 4K buffer size. The 4K buffer size is recommended unless storage consumption is a significant concern. The minimum non-disabled value is `1,2`. The maximum is `126,4` or `248,2` corresponding to total size of 512 Megabytes. An upper bound for the number of buffers that may be required is one for each IMS message region Agent and MAXTASK times the number of CICS Agents, but actual requirements are likely to be considerably smaller.

Continuation lines

If a parameter value must span multiple lines, specify the value through position 71 and continue the value on the next statement starting in position 1

Comments

A comment is specified by placing an asterisk in position 1 of a statement. The entire statement is considered part of the comment.

A comment can be enclosed between a slash-asterisk and an asterisk-slash:

```
/* This is a comment. */
```

Multiple lines can be contained within comments of this type.

Running multiple zDC systems on a single z/OS image

A zDC connects one or more CICS or IMS Agents running on the same z/OS system to a single AppMon Server, so it may be necessary to configure multiple zDC-started tasks on a single z/OS system when different application groups are being traced by different AppMon Servers. At times, it may be useful to configure an alternate zDC for testing purposes as well.

Only one zDC on a particular z/OS system can specify DEFAULT(YES) in its SYSIN parameters. Additional zDCs will fail to initialize unless they specify DEFAULT(NO).

See Adding SIT override INITPARM for Non-Default ZDC sub-systems to configure a CICS Agent to connect to a non-default zDC.

Check maintenance level

Browse Install Library: `<*hlq*>.LZDT700.SZDTAUTH` where <*hlq* > is the High Level Qualifier of the SMPE target zone.

Browse the beginning of the ZDCDTAGT load module. You should see the header: *ZDCDTAGT 00BASE YYYYMMDD HH.MM VER 06.03.00 COPYRIGHT (C)…* In the example above, *00BASE* indicates this is the release `7.0.0` GA version of the module without any maintenance. When service is applied, the field contains a problem number, and release numbers and dates will vary over time.

ZDCDTAGT

The manager for the dynaTrace z/OS zLocal Agent. ZDCDTAGT can produce diagnostic messages with this format:
`ZDC99<n><i> messages where n=loglevel and i=severity`

zDC commands supported by the z/OS modify command

For example: `Modify <JobName or StcProcName>,DT1 <cmd>`

<cmd>	Description
DISPLAY	Display z/OS Unix Agent subtask processor stats.
DTM%V	Force display (ZDC952) of utilization of the zLocal Agent message queue.

LOG=?	Query loglevel.
LOG=n	Set loglevel to n (0-8) where lower numbers increase the level of detail. Default is 3.
START	Start zLocal Agent.
STDO	Append previously unreported lines from zLocal agent stdo.log to SYSPRINT.
STOP	Stop zLocal Agent.
ZLALOGLEVEL=Xxx	Set zLocal agent loglevel to Xxx (FINEST,FINER,FINE,CONFIG,INFO,WARNING,SEVERE,DEBUG,NONE). No default. Available in 6.5 and above.

zDC sample output from MODIFY MEPx,DT1 DISPLAY

The display skips most rows with zero counters if the log level is at or above the default level of 3. The loglevel can be changed for diagnostic purposes to greatly increase logging, including displaying the full set of counters `using MODIFY MEPx,DT1 LOG=1`):

```
14:13:33.854126 ZDC027I MODIFY COMMAND RECEIVED
14:13:33.857174 ZDC028I DT1 DISPLAY
14:13:33-977432 ZDC994I ShoS Counters
                  35:Msgs sent from MAgent to zDC via unnamed pipe
                   1:QUEUE_MAX_EVENTS(EQH)High watermark elements used
                 128:EQH queue elements defined
                 128:EQH elements currently free
                   0:Internal Performance:XmPosts avoided
                   0:Internal Performance:XmPost Skip DupSrbs
               1,758:zDC Elapsed time (approx sec)
                   0: Bytes read from DTM Msgs
                  49:SMO Msgs written
                  34:Internal:Redundant Post bypassed
                  15:Internal:Attempt Rd=Wr cursor for vStorage Perf.
                  15:z/OS Unix has no ready Msgs, waiting for work
                  48:z/OS Unix Msgs read
               3,808: Bytes read from SMO Msgs
               2,267:TBC Total Transactional Buffers
```

```
                2,267:TBC total transactional buffers
                2,267:TBC number in Free queue
                   25:TBC Msgs written
                   25:TBC Msgs read
                   10:TBC TBCs written
                   10:TBC TBCs read
                   10:TBC Post bypass
                   41:TBC No Ready Q
                1,905:TBC Bytes written
                1,905:TBC Bytes read
                    2:TBC Age<=PingDelay*.5
                    2:TBC Age<=PingDelay*.75
                    2:TBC Age<=PingDelay*1
                    2:TBC Age<=PingDelay*1.25
                    2:TBC Age>=PingDelay*1.25
           10,485,760:DTMSG_SMOSIZE Bytes allocated
                9,248:DTMSG_SMOSIZE Bytes used
14:13:36-712702 ZDC995W  DispTcb:     009D1438               0.900sec
14:13:36-712792 ZDC995W  DispTcb:     009BBAD0               0.577sec
14:13:36-712832 ZDC995W  DispTcb:     009BBCF0               0.013sec
14:13:36-712882 ZDC995W  DispTcb:     009BBE88               0.012sec
14:13:36-714022 ZDC995W  DispTcb:     009BE170               0.924sec
14:13:36-714062 ZDC995W  DispTcb:     009BE390               0.590sec
14:13:36-714112 ZDC995W  DispTcb:     009BE528               0.251sec
```

Rows that end with an exclamation mark should have zero counts.

Emergency zDC termination

zDC makes every effort possible to release system resources (most especially common storage) when it terminates. This is true if the termination is normal due to a shutdown command or abnormal due to a cancel command or a program abend.

Recovery (ESTAEX) routines are always in effect to trap abnormal terminations and to free all system resources.

However, there may be a situation where zDC cannot be cancelled due to a z/OS anomaly or failure, which precludes the functioning of the cancel command.

If you are unable to communicate with zDC and want to terminate it, try the commands below in the specified order until the zDC terminates. Go on to step 2 only if step 1 is unsuccessful, go on to step 3 only if step 2 is unsuccessful, and so on. To avoid the need to manually clean up system resources using the ZDCDELET process below, wait a short while between commands to permit dumps to be taken and system resources to be freed.

1. Issue the `STOP jobname` command.
2. Issue the `MODIFY jobname,SHUTDOWN IMMED` command.
3. Issue the `CANCEL jobname` command.
4. Issue the `FORCE jobname` command.

Attempt to restart the zDC. If the restart is successful you can continue to run this job. If however the restart is not successful and zDC indicates that it cannot continue, you can execute the emergency zDC cleanup program named ZDCDELET.

```
ZDCDELET
ZDCDELET is a stand-alone job executed via JCL, which will attempt to terminate a specifi
JCL to execute ZDCDELET is shown below:
//*
//* PLACE YOUR JOB STATEMENT HERE
//*
//DELETE EXEC PGM=ZDCDELET,PARM=xxxx
//STEPLIB DD DISP=SHR,DSN=<hlq>.LZDT630.SZDTAUTH
The PARM= on the EXEC statement must specify the 4-character subsystem ID of the ZDC inst
User ABEND 100 and 101 may occur and messages describing the abends will be written to th
User ABEND 100 indicates that the PARM= is not exactly 4 bytes long.
User ABEND 101 indicates that the subsystem ID specified on the PARM= operand cannot be f
```

Messages ZDC400 through ZDC403 may be written to the joblog as well. These can be found in the ZDC Messages section below.

zDC messages

Refer to the zOS Agent Messages page for a list of all zDC messages.

Multiple TCP/IP stacks

A z/OS system with multiple TCP/IP stacks presents an additional complication. In this situation, an additional step is needed in the server job:

```
//MEPC     PROC
...
//STEP0 EXEC PGM=BPXTCAFF,PARM='TCPIP_stack_name'
//*
//ZDCMAIN EXEC PGM=ZDCMAIN,REGION=0M,PARM='LANGUAGE=EN'
//STEPLIB   DD DISP=SHR,DSN=hlq.LZDT630.SZDTAUTH
//* Notes:
//* SYSPRINT is required in all cases.
//* SYSPRINT must be allocated to spool, not disk, on JES2 systems.
//* SYSPRIN3 is required on JES3 systems only.
//SYSPRINT  DD SYSOUT=*
//SYSPRIN3  DD SYSOUT=*
//SYSIN     DD DISP=SHR,DSN=hlq.LZDT630.SZDTSAMP(ZDCSYSIN)
```

The BPXTCAFF program is used to associate a specific TCP/IP stack with the current address space. The PARM value is the job name used to start the desired TCP/IP stack.

CICS Agent installation and configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Adding the AppMon CICS programs to the CICS DFHRPL

Installing the AppMon CICS programs and transaction

Adding the AppMon CICS Agent to the CICS startup program list

Adding the AppMon CICS Agent shutdown to the shutdown program list table

Preparing your System Profile for CICS Agents

Adding SIT override INITPARM for non-default zDC subsystems or FINE logging

SOAP support

CICS PurePath

Related pages

Related topics

zOS Agent

CICS Transaction Server sensor

Transaction measure

DTAX transaction

zDC

zOS Agent logs

zOS Agent messages

zOS Agent troubleshooting

The CICS Agent installs into every CICS region that you want to monitor. It includes a PLT program that initiates at CICS startup, and an Agent program that captures various events and forwards them to the zDC.

The CICS Agent can either start new PurePaths for specific transaction codes or link CICS transaction PurePaths to distributed PurePaths for transactions initiated using WebSphere MQ, CICS Transaction Gateway, or SOAP (over HTTP). The CICS agent supports DB2, DL/I and MQ API calls tracing, as well as any programs invoked via CICS LINK or LE dynamic call. Linked PurePaths are also provided with child transaction details started for DPL LINK or START TRANSACTION requests within a PurePath.

Adding the AppMon CICS programs to the CICS DFHRPL

Add the following PDS (or its contents) to the DFHRPL concatenation, substituting <hlq> with the high-level qualifier you set while Download and SMP/E install of the z/OS Agent Components.

```
// DD DISP=SHR,DSN=<hlq>.LZDT630.SZDTLOAD
```

Installing the AppMon CICS programs and transaction

The sample CICS definitions below install programs and a transaction that are required by the CICS Agent. A copy of these definitions can be found in `SZDTSAMP` member `CICRDO`, which is provided for use with the batch RDO utility DFHCSDUP.

These transaction and group names are recommended by AppMon, but you can change them in accordance with your installation policies. Coordinate the group name and group list name with your CICS administrator. Replace `XYZLIST` with the name of your group list (`GRPLIST`).

```
CEDA DEFINE TRANSACTION(DTAX) DESCRIPTION(DYNATRACE AGENT UTILITY TRAN)
     GROUP(DTA1) PROGRAM(ZDTPLT) TASKDATAKEY(CICS) TASKDATALOC(ANY)

CEDA DEFINE PROGRAM(ZDTPLT) DESCRIPTION(DYNATRACE AGENT UTILITY PROG)
     GROUP(DTA1) DATALOCATION(ANY) EXECKEY(CICS)
```

```
CEDA DEFINE PROGRAM(ZDTPLTSD) DESCRIPTION(DYNATRACE AGENT SHUTDOWN PROG)
     GROUP(DTA1) DATALOCATION(ANY) EXECKEY(CICS)

CEDA DEFINE PROGRAM(ZDTAGT66) DESCRIPTION(DYNATRACE CTS 4.1 AGENT)
     GROUP(DTA1) CEDF(NO) CONCURRENCY(THREADSAFE) DATALOCATION(ANY)
     LANGUAGE(ASSEMBLER)

CEDA DEFINE PROGRAM(ZDTAGT67) DESCRIPTION(DYNATRACE CTS 4.2 AGENT)
     GROUP(DTA1) CEDF(NO) CONCURRENCY(THREADSAFE) DATALOCATION(ANY)
     LANGUAGE(ASSEMBLER)

CEDA DEFINE PROGRAM(ZDTAGT68) DESCRIPTION(DYNATRACE CTS 5.1 AGENT)
     GROUP(DTA1) CEDF(NO) CONCURRENCY(THREADSAFE) DATALOCATION(ANY)
     LANGUAGE(ASSEMBLER)

CEDA DEFINE PROGRAM(ZDTAGT69) DESCRIPTION(DYNATRACE CTS 5.2 AGENT)
     GROUP(DTA1) CEDF(NO) CONCURRENCY(THREADSAFE) DATALOCATION(ANY)
     LANGUAGE(ASSEMBLER)

CEDA DEFINE PROGRAM(ZDTAGT70) DESCRIPTION(DYNATRACE CTS 5.3 AGENT)
     GROUP(DTA1) CEDF(NO) CONCURRENCY(THREADSAFE) DATALOCATION(ANY)
     LANGUAGE(ASSEMBLER)

CEDA DEFINE PROGRAM(ZDTAGT71) DESCRIPTION(DYNATRACE CTS 5.4 ETP AGENT)
     GROUP(DTA1) CEDF(NO) CONCURRENCY(THREADSAFE) DATALOCATION(ANY)
     LANGUAGE(ASSEMBLER)

CEDA DEFINE PROGRAM(ZDTSOAPH) DESCRIPTION(DYNATRACE SOAP HEADER PROG)
     GROUP(DTA1) EXECKEY(USER) CONCURRENCY(THREADSAFE)
     DATALOCATION(ANY) LANGUAGE(ASSEMBLER)

CEDA DEFINE PROGRAM(ZDTDC2) DESCRIPTION(COBOL2 PRIMER PROGRAM)
     GROUP(DTA1) EXECKEY(CICS) CONCURRENCY(THREADSAFE)
     DATALOCATION(ANY) LANGUAGE(COBOL)

CEDA DEFINE PROGRAM(ZDTDC2A) DESCRIPTION(COBOL2 PRIMER PROGRAM)
     GROUP(DTA1) EXECKEY(CICS) CONCURRENCY(THREADSAFE)
     DATALOCATION(ANY) LANGUAGE(COBOL)

 CEDA INSTALL GROUP(DTA1)
```

```
CEDA ADD GROUP(DTA1) LIST(XYZLIST)
```

Adding the AppMon CICS Agent to the CICS startup program list

Add the AppMon CICS PLTPI program `ZDTPLT` after the `DFHDELIM` entry in your PLTPI source code and assemble the table. This step is optional for test installations because the DTAX transaction may be used instead to enable the agent after CICS initialization. It is recommended that the `ZDTPLT` entry immediately precede the `TYPE=FINAL` specification.

The JCL procedure `DFHAUPLE` in `CICSHLQ.SDFHINST(DFHAUPLE)` can be used to build the PLTPI table. This is an example PLTPI table that contains the AppMon CICS Agent.

```
*
* PLT USED TO SUPPORT DYNATRACE AGENT INITIALIZATION
*
         DFHPLT TYPE=INITIAL,SUFFIX=SI
         DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
* Other PLT startup programs here...
         DFHPLT TYPE=ENTRY,PROGRAM=ZDTPLT
         DFHPLT TYPE=FINAL
         END
```

Important note for Compuware Xpediter Global storage protection users

The PLT startup program `ZDTPLT` initializes the CICS Agent's exit workarea, which CICS obtains on its behalf. Products such as Compuware Xpediter/CICS may be configured to enforce strict storage access controls and may abend `ZDTPLT` and prevent the AppMon CICS Agent from starting unless it is excluded from these controls. If you use the Xpediter/CICS global storage protection feature, add a "monitor exceptions" entry to the XDDBPINP DD in the CICS region JCL to exclude `ZDTPLT*`. For example:

```
DBPA 9.4.1 TRAN=*,PROGRAM=ZDTPLT*,CSECT=*
```

Adding the AppMon CICS Agent shutdown to the shutdown program list table

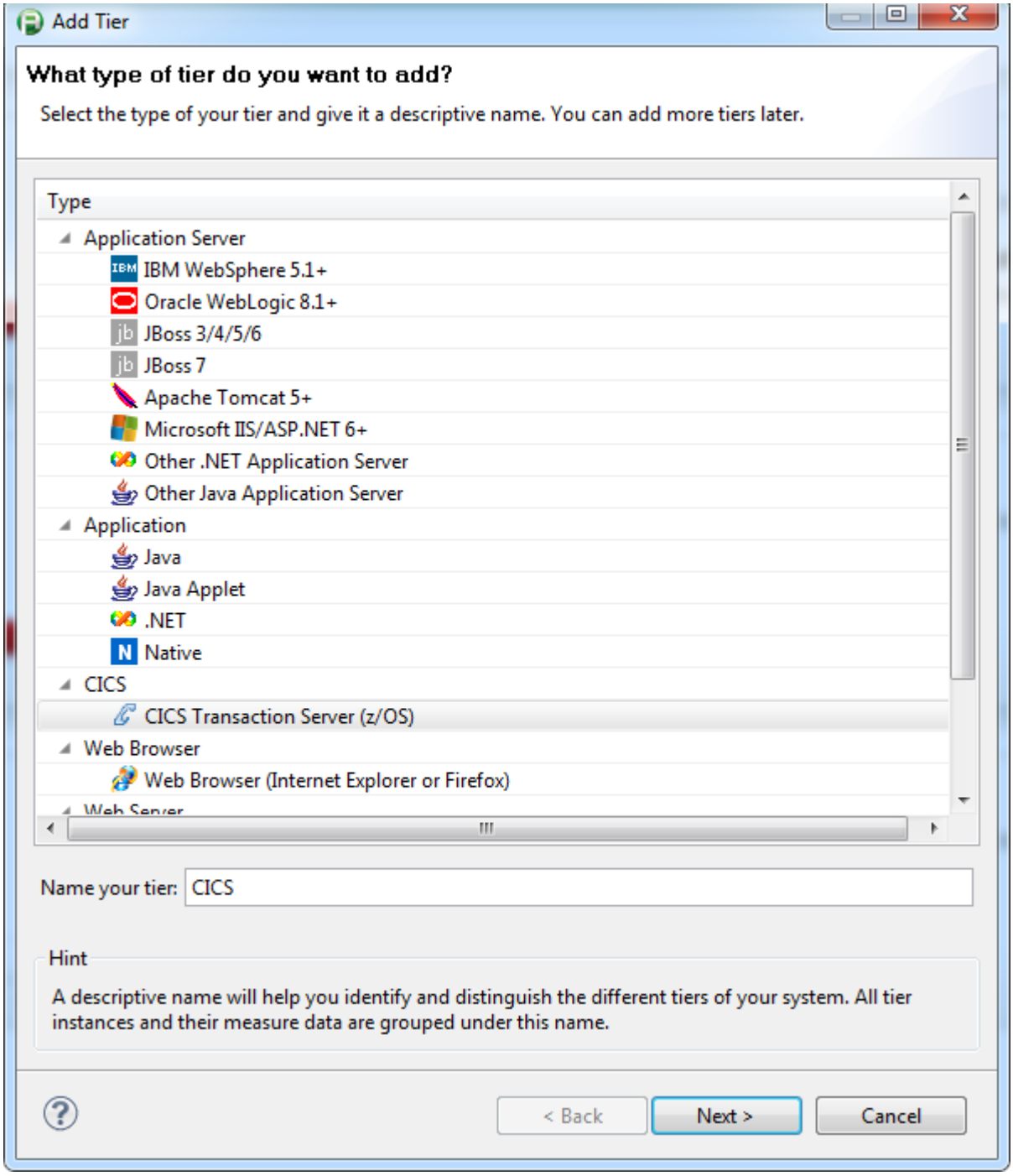
Add the CICS Agent shutdown program `ZDTPLTSD` before the `DFHDELIM` entry in your PLTSD source code and assemble the table. It is recommended that you place the `ZDTPLTSD` entry immediately after the `TYPE=INITIAL` specification.

The JCL procedure `DFHAUPLE` in `CICSHLQ.SDFHINST(DFHAUPLE)` can be used to build the PLTSD table. This is an example PLTSD table that contains the CICS Agent shutdown program.

```
*
* PLT USED TO SUPPORT DYNATRACE AGENT SHUTDOWN
*
         DFHPLT TYPE=INITIAL,SUFFIX=SD
         DFHPLT TYPE=ENTRY,PROGRAM=ZDTPLTSD
* Other PLT shutdown programs here...
         DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
         DFHPLT TYPE=FINAL
         END
```

Preparing your System Profile for CICS Agents

Add a CICS tier to the system profile that was created in the installation of the zDC. The wizard will present a number of tiers that are available. Select the CICS Agent.



Select the Tier Type

Adding SIT override INITPARM for non-default zDC subsystems or FINE logging

The PLT startup program ZDTPLT automatically connects to the default zDC at CICS region initialization. If multiple zDCs are running, the zDC that specifies DEFAULT(YES) is chosen unless an INITPARM parameter in the CICS system initialization parameters specifies that it must connect to a zDC with a particular name:

```
INITPARM=(ZDTPLT='MEPC[,FINE]'),
```

Fine logging can be set for diagnostic purposes by supplying the optional second parameter.

For linking CICS Agents to non-default zDCs, see zDC > Running multiple zDC systems on a single z/OS image.

SOAP support

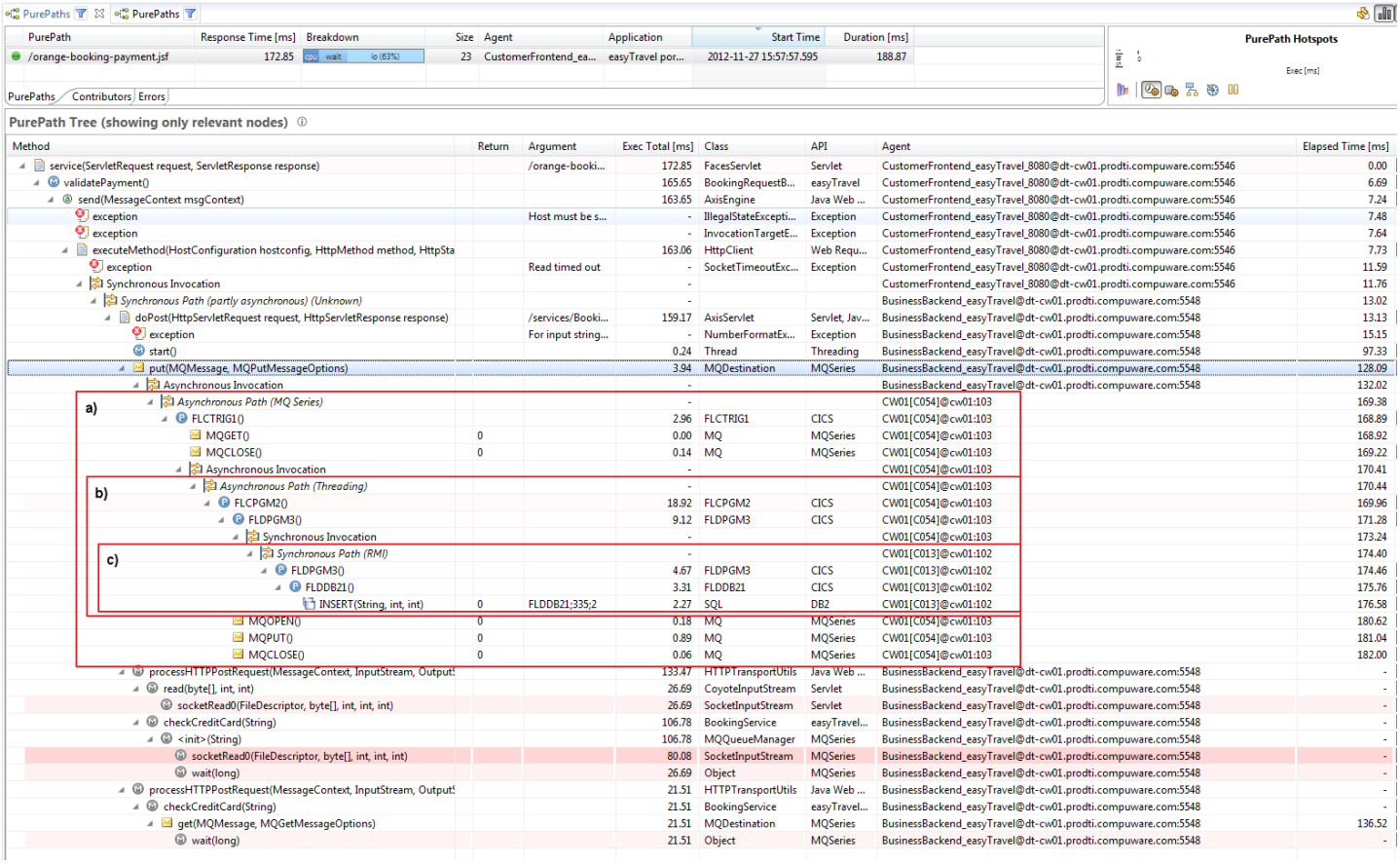
To trace SOAP service provider programs that are invoked by DFHPITP from CICS SOAP pipelines, update the service provider pipeline definitions for the services that should be traced. Add a `<headerprogram>` stanza that invokes the supplied ZDTSOAPH header program. For example, add the definitions below to the `cics_soap_1.1_handler` section after any existing header program sections:

```
<headerprogram>
    <program_name>ZDTSOAPH</program_name>
    <namespace>\*</namespace>
    <localname>\*</localname>
    <mandatory>true</mandatory>
</headerprogram>
```

To trace outbound SOAP requests that originate within CICS transactions that are being traced by any CICS agent sensor, add the same `<headerprogram>` stanza above to the service requester pipeline definitions for those SOAP services that should be traced. Outbound SOAP requests that occur within CICS transactions that are not being traced are ignored, but tracing is not limited to requests from SOAP programs acting as CICS SOAP service providers.

CICS PurePath

The following figure shows a CICS PurePath that was started by an MQGet call. The message was initially put into the queue by a Java application, so the CICS PurePath has been linked to the Java PurePath by MQ tagging.



CICS PurePath

The PurePath contains three CICS subpaths. The first path **a)** in CICS region C054 was started by a call to the MQ API. The first node FLCTRIG1 represents the initial program of the FLC1 transaction and contains details about the transaction. FLCTRIG1 started another transaction **b)** (EXEC CICS START) in the same region. This transaction did a DPL call to synchronously run another transaction **c)** in CICS region C013.

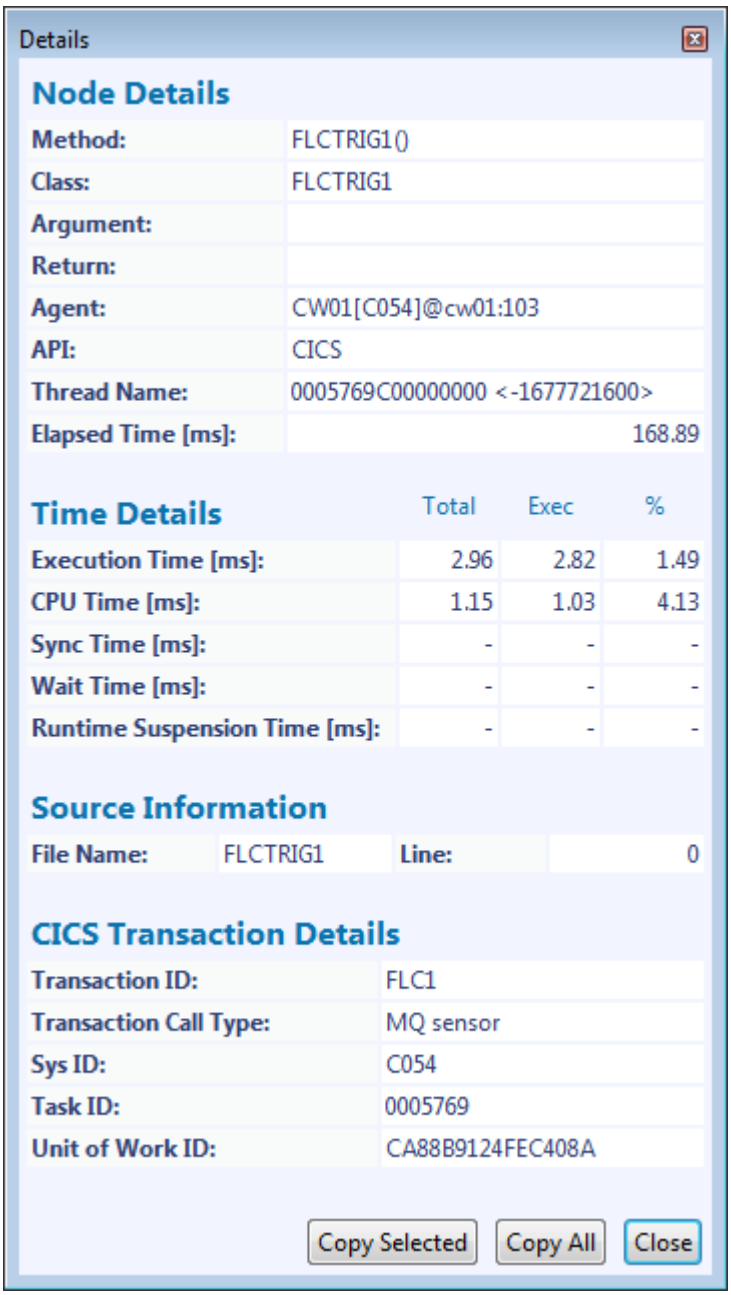
The following summarizes the different nodes in the PurePath:

- 

  Program - Program nodes are created for all programs that are called by EXEC CICS LINK calls. The initial program of a transaction (first node of CICS subpaths) also contains details about the transaction.
- **Method** - CICS program name.

- **Argument** - The transaction ID is captured as an argument on the initial program.
- **Class** - Load module name.
- **API** - CICS.
- **Return** - Abend code if Abend occurred.

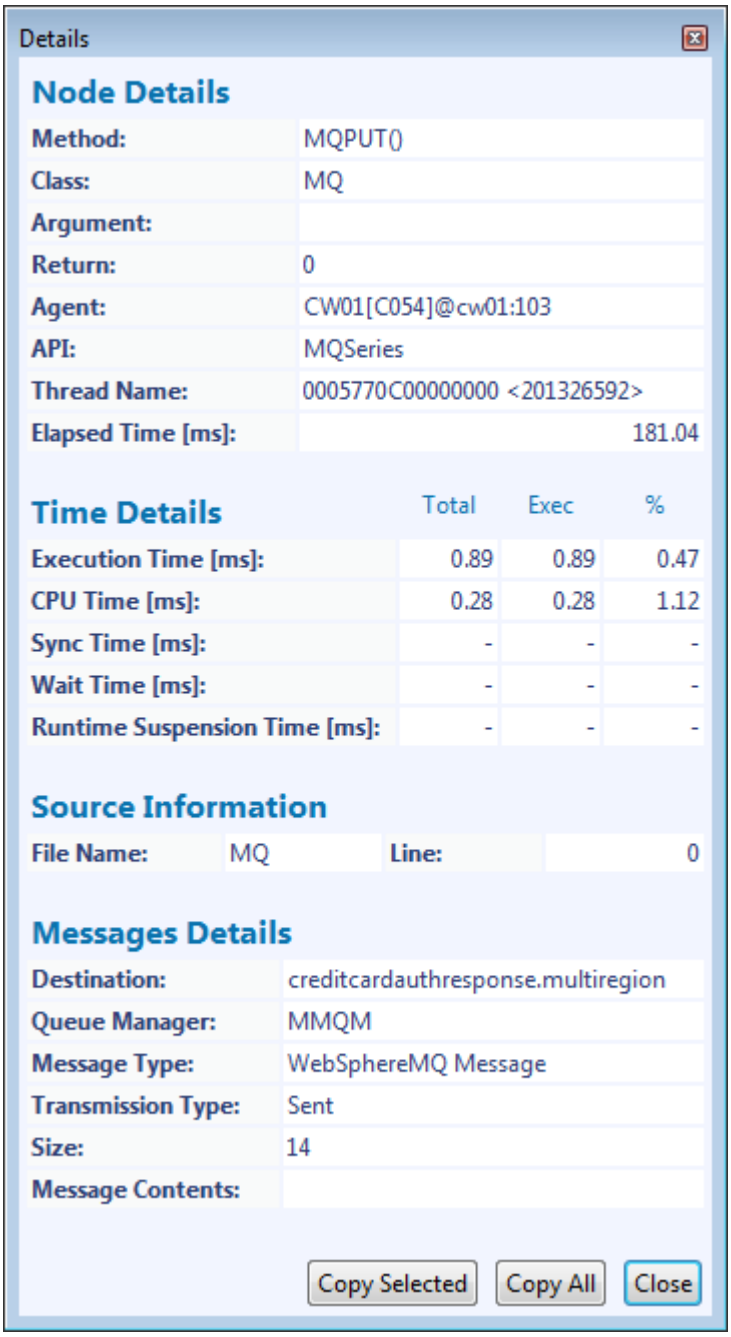
The following image shows the details displayed when you hover the cursor over this Program node.



- 
  Messaging - Messaging nodes are created for all MQ API calls.
- **Method** - MQ API call, e.g. MQPUT
- **Argument** - (not applicable)

- **Class** - MQ
- **API** - MQSeries
- **Return** - MQ reason code

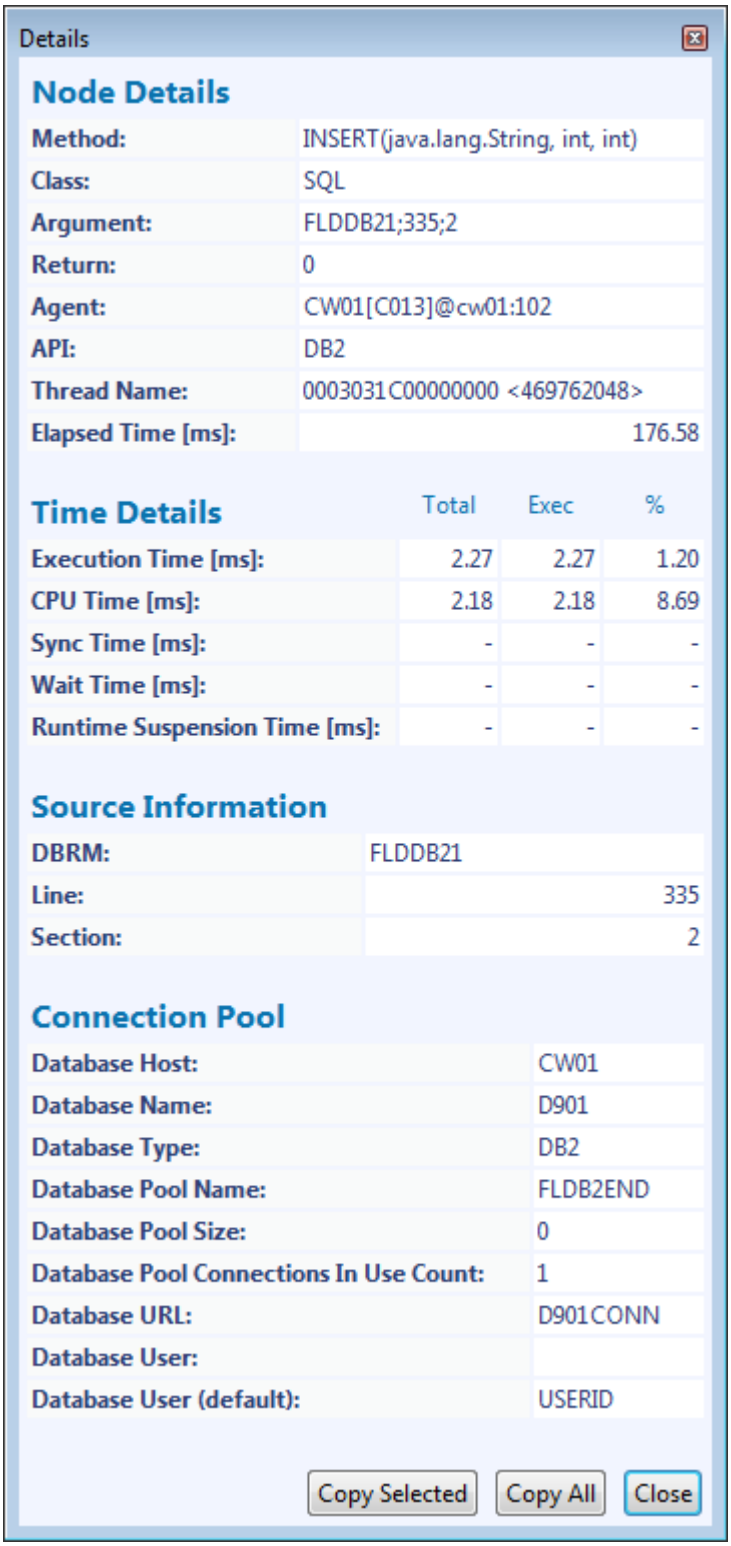
The following image show the details displayed when you hover the cursor over this Messaging node.



- 
  Database - Database nodes are created for all DB2 API calls.
- **Method** - DB2 API call, e.g. SELECT
- **Argument** - DBRM Name, Source Line, Section Number

- **Class** - SQL
- **API** - DB2
- **Return** - SQL code

The following image show the details displayed when you hover the cursor over this Database node.



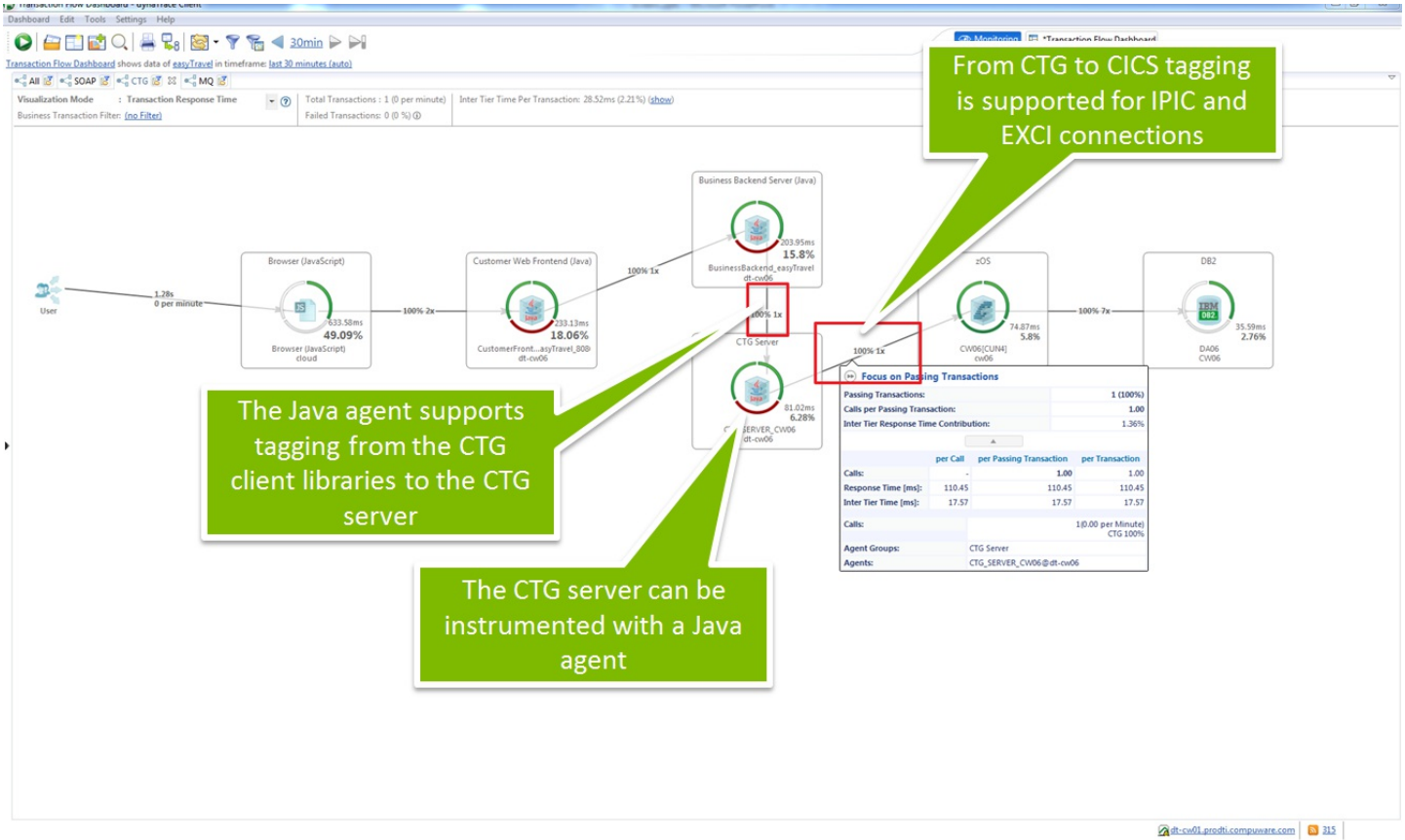
The connection pool attachment is filled with data from the DB2ENTRY definition. The URL is mapped to the name of the CICS DB2CONN entry. The pool size is mapped to the THREADLIMIT field in the DB2ENTRY.

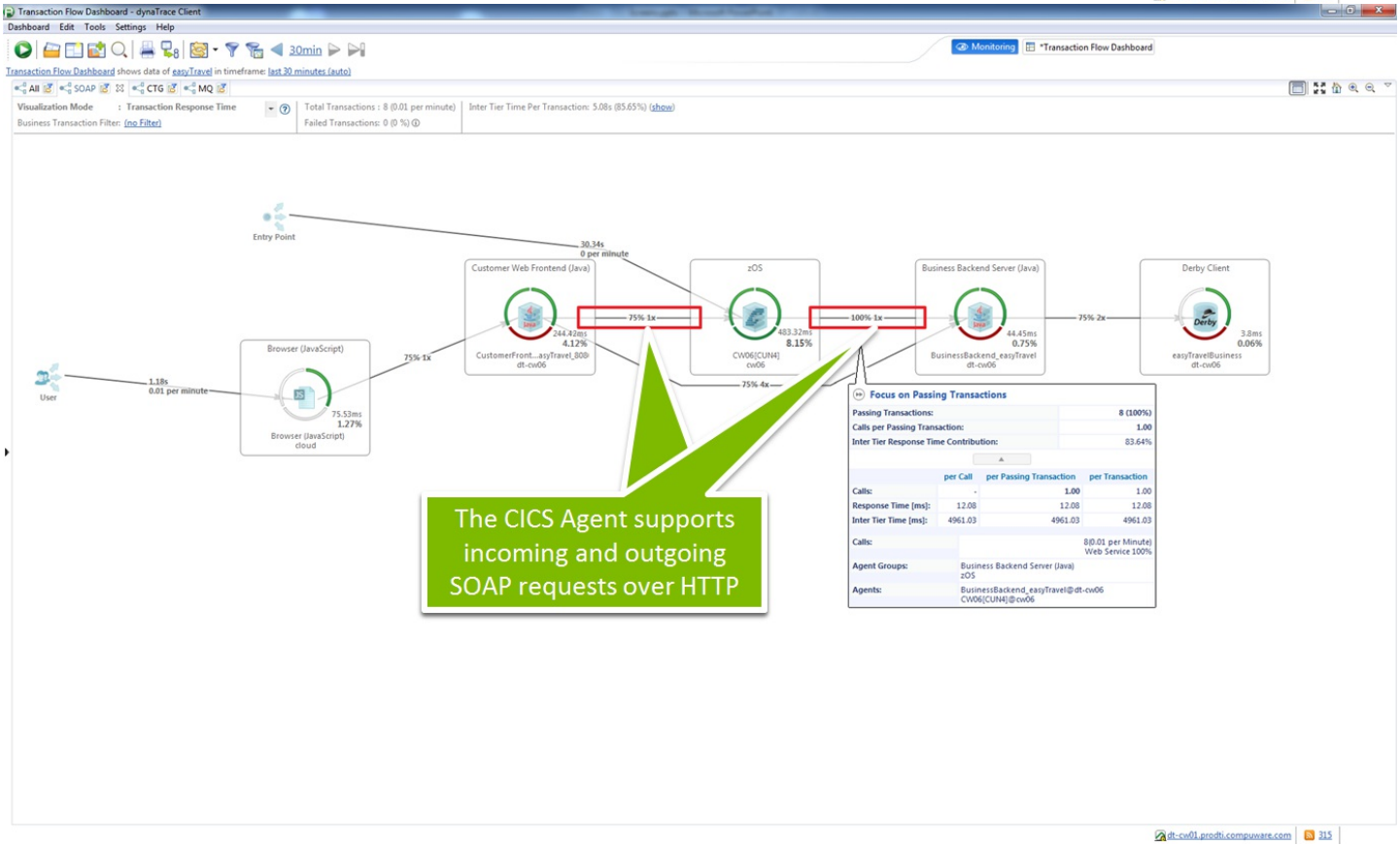
Note

Transaction tracing started in another CICS region as a result of a CICS START TRANSACTION or DPL LINK request depends on the association data that accompanies these transactions when they flow over IPIC connections. This association data is also provided for transactions started over MRO connections for CTS version 4.2 and above. VTAM ISC connections do not support association data and transactions flowing over ISC are not traced.

Example Transaction Flows

The following screenshots show what the transaction flow for the different protocols may look like.


From CTG to CICS tagging is supported for IPIC and EXCI connections
The Java agent supports tagging from the CTG client libraries to the CTG server
The CTG server can be instrumented with a Java agent
Focus on Passing Transactions


The CICS Agent supports incoming and outgoing SOAP requests over HTTP
Focus on Passing Transactions

CICS Transaction Server sensor

Related pages

zOS Agent

CICS Agent installation and configuration

Transaction measure

DTAX transaction

zDC

zOS Agent logs

zOS Agent messages

zOS Agent troubleshooting

The CICS Transaction Server sensor captures MQ, DB2 and CICS API calls.

Sensor configuration

To edit the System Profile specific settings, right click you System Profile, and select **Edit System Profile > Agent Group/tier > Sensor Configuration** and then select the **CICS Transaction Server** item in the **Sensor Configuration** list.

Sensor properties

- **Enable DB2 sensor**: Enables/disables DB2 API details. If disabled, no DB2 API call information reports in the traces produced by any of the enabled sensors.
  - **Enable aggregation**: Produces a summarized report of DB2 activity. For example if a CICS transaction issues 50 INSERT API calls, one node is produced to show all 50 calls instead of 50 separate nodes per each insert. Aggregation reduces the number of events processed by the AppMon Server, and thus improves overall performance on the AppMon Server. Return values are not available in aggregation mode.
  - **Capture DB2 fetches**: When enabled each DB2 fetch causes data to be sent to the AppMon Server. In CICS systems that have transactions which issue an excessive number of FETCH calls, performance can be greatly improved by disabling the capture of DB2 fetches.
- **Enable DL/I sensor**: Enables/disables CICS DLI API details. If disabled, no DL/I API call information will be reported in the traces produced by any of the enabled sensors.
  - **Enable aggregation**: Produces a summarized report of DLI activity. For example if a CICS transaction issues 50 ISRT API calls, one node is produced to show the 50 calls 50 separate nodes per each call. Aggregation reduces the number of events processed by the AppMon Server, and thus improves overall performance on the AppMon Server. Return values (DL/I status codes) are not available in aggregation mode.
- **Enable SOAP sensor**: Enables/disables the SOAP Sensor, which traces all outgoing SOAP requests originating from CICS transactions that are being traced by any enabled sensor; and all incoming SOAP requests that have an HTTP header with an AppMon tag. Note that no inbound SOAP request tracing occurs unless the request flows through a CICS SOAP pipeline that has been reconfigured to include the ZDTSOAPH header program.
- **Enable CTG sensor**: Enables/disables the CTG Sensor. You will need to install the Java Agent in the CTG. For more information, see CICS Transaction Gateway.
- **CICS subagent name format**: Adapts the CICS subagent name to append either the **SubSystemID**, or the **Jobname**, or the **SNA ID**. This property *must* be set in the agent mapping of the zRemoteAgent(s). The CICS Transaction Sensor must be placed first. The change takes effect upon the next CICS subagent registration event.

- **Enable Tracing LE Programs**: Enables/disables the LE Sensor, which traces COBOL and PLI dynamic calls through the LE runtime.
  - **Program include filters**: A list of program name patterns to be traced by the LE sensor when they are invoked through LE runtime dynamic calls. Each entry in the list may be a program name or a program name prefix with an asterisk at the end. LE dynamic calls to programs whose names do not match an entry in the list will be ignored by the LE sensor. By default, all program names are enabled.
  - **Program exclude filters**: A list of program name patterns to be excluded from tracing by the LE sensor. Each entry in the list may be a program name or a program name prefix with an asterisk at the end. Programs whose names match an entry in the list will be ignored by the LE sensor. If a particular program name matches both the include and exclude lists, the exclude list takes priority and the program is ignored.
    Note that any PL/I program that is FETCHed to be used as table instead of being executed MUST be excluded, or the associated application may not function properly
- **Enable MQ sensor**: Enables/disables the MQ Sensor.
  - **Queue include filters**: A list of queue name patterns to be traced. Each entry in the list may be a queue name or a queue name prefix with an asterisk at the end. Messages from queues whose names do not match an entry in the list will be ignored by the MQ sensor. By default, all queue names are enabled.
  - **Queue exclude filters**: A list of queue name patterns to be excluded from tracing. Each entry in the list may be a queue name or a queue name prefix with an asterisk at the end. Messages from queues whose names match an entry in the list will be ignored by the MQ sensor. If a particular queue name matches both the include and exclude lists, the exclude list takes priority and transactions that process this queue are ignored.
- **Enable transaction start sensor**: Enables/disables the transaction start sensor, which traces any transaction that matches the transaction filters without setting a tag to link it to a previous path. Set filters carefully as this will interfere with other sensors such as MQ and SOAP that need to set tags after the start of a CICS transaction. It does not affect the CTG sensor or tracing of transactions that are linked to a parent CICS transaction that issued a START or DPL LINK request. This sensor can be used to start a path to trace a CICS transaction that cannot be linked to previous activity outside CICS. Note that the CICS and IMS Tagging ADK may be used to support tagging of transactions that use unsupported protocols as an alternative to the transaction sensor.
  - **Transaction filter**: A list of transaction code patterns to be traced. Each entry in the list may be a transaction code or a transaction code prefix with an asterisk at the end. Transactions whose names do not match an entry in the list will be ignored by the TX sensor. An entry with a single asterisk “*” is ignored because monitoring CICS internal transactions is not recommended.
  - **Enable Node Limit**: Sets a limit on the number of nodes in a single transaction within CICS. The default limit is 500.

Transaction measure

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Charting CPU usage for all CICS transactions

Related pages

Related topics

zOS Agent

CICS Agent installation and configuration

CICS Transaction Server sensor

DTAX transaction

zDC

zOS Agent logs

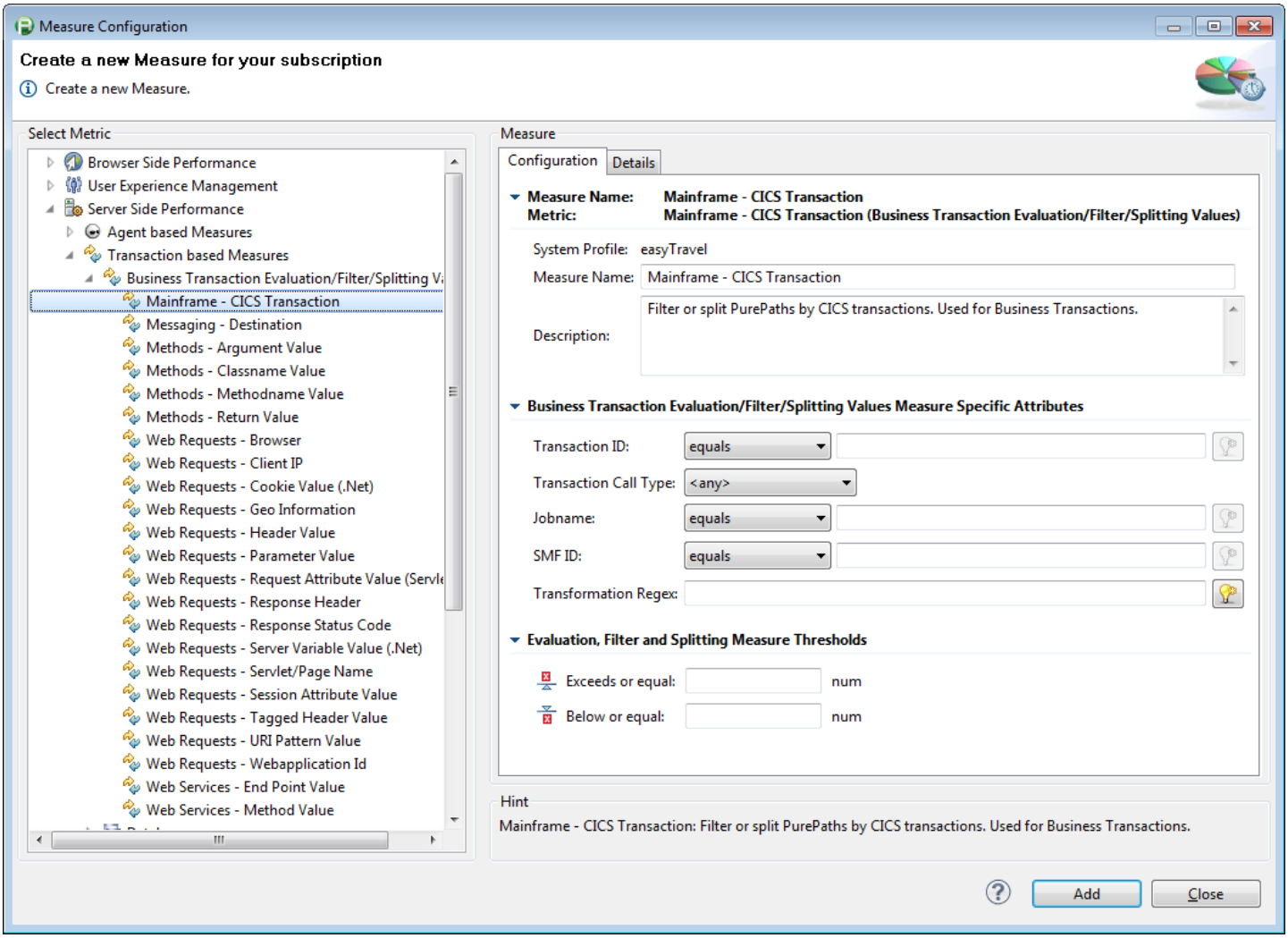
zOS Agent messages

zOS Agent troubleshooting

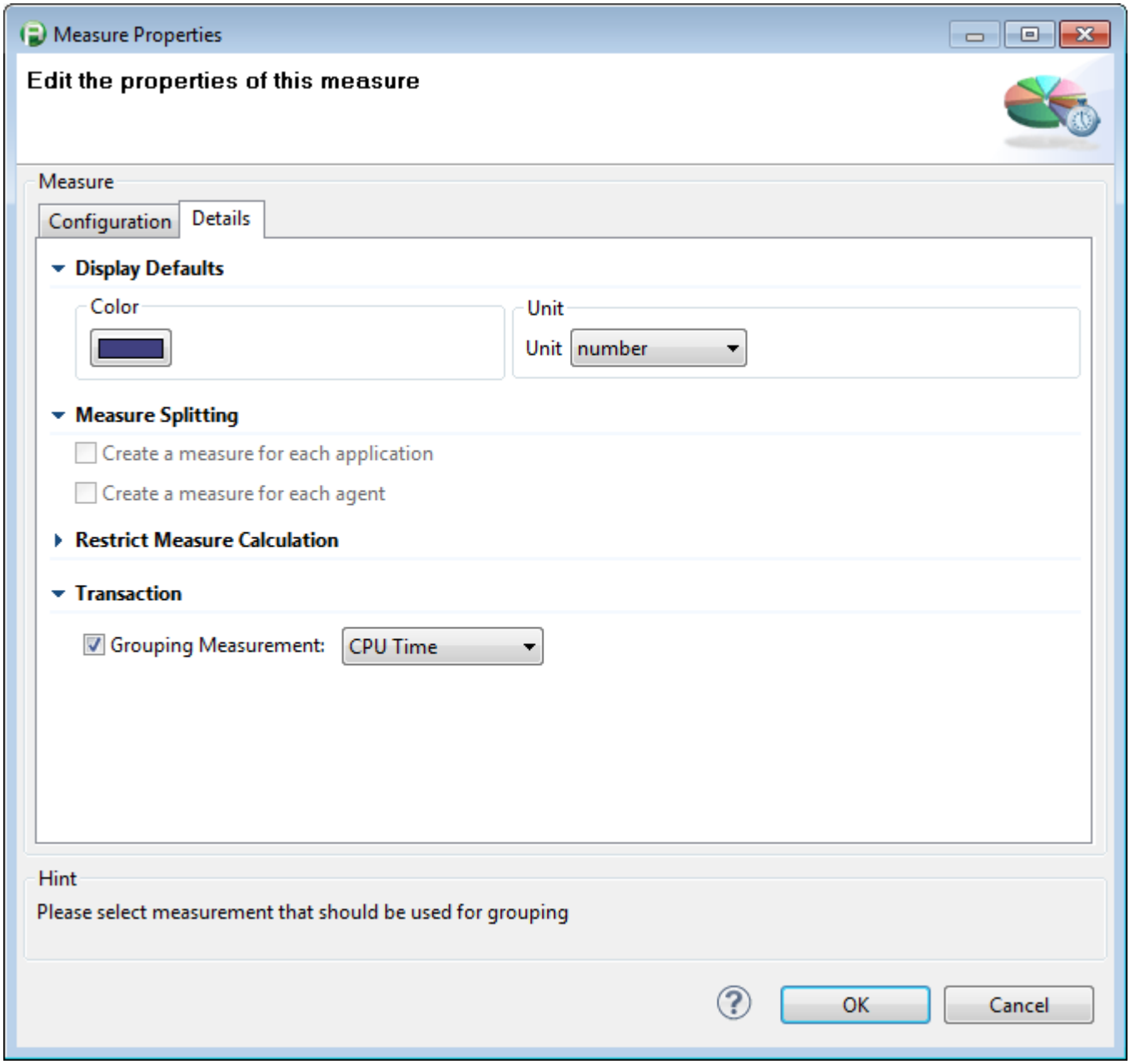
Overview

AppMon provides a measure to split or filter Business Transactions by CICS transaction. If used as a splitting measure, the value is created as `<SMF ID>_<Jobname>_<Transaction ID>` (for example, CW01_H01AC013_DTI1). You can also use the measure to evaluate the CPU time or the response time of CICS transactions. The screenshot below shows the configuration dialog boxes.

AppMon provides a measure that you can use to split or filter Business Transactions by CICS transaction. If used as a splitting measure, the value will be created as `<SMF ID>_<Jobname>_<Transaction ID>` (for example, CW01_H01AC013_DTI1). You can also use the measure to evaluate the CPU time or the response time of CICS transactions. The screenshot below shows the configuration dialog boxes.



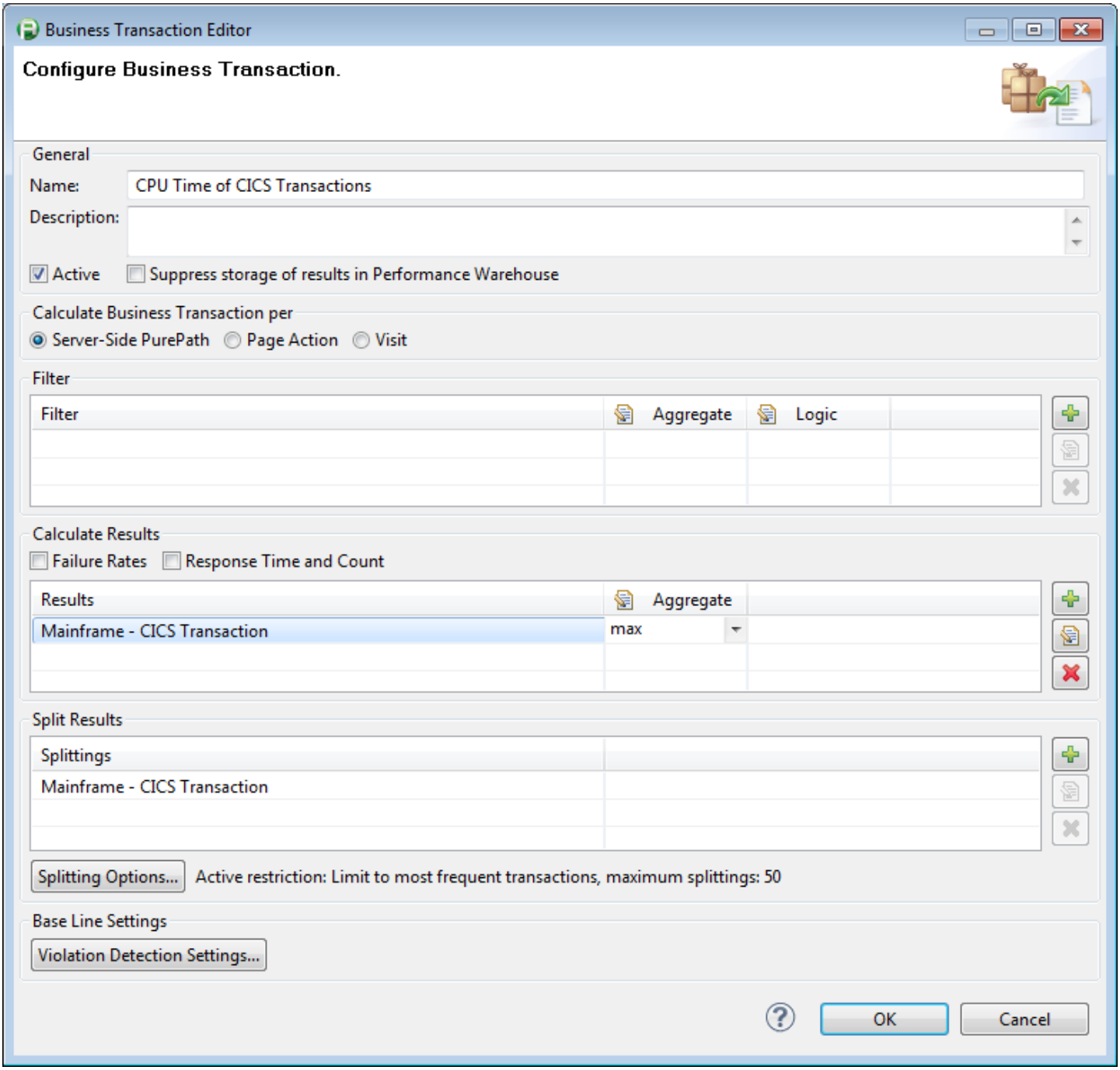
Measure Configuration dialog box



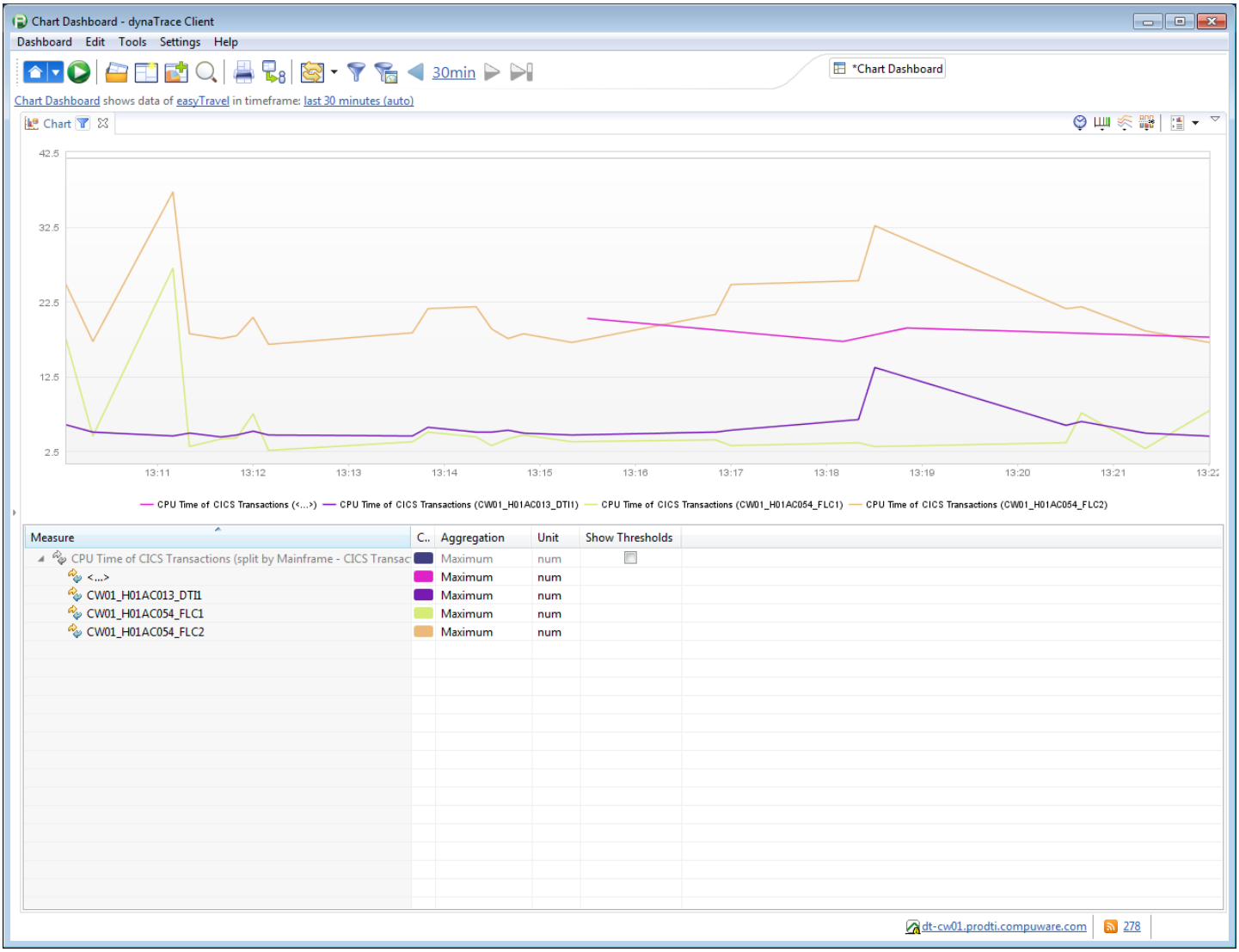
Measure Properties dialog box

Charting CPU usage for all CICS transactions

To configure a Business Transaction that monitors the CPU usage of all CICS transactions, create a new CICS Transaction measure and add it as a Splitting and Evaluation measure. The screenshots below show the result.



Business Transaction editor



Measure that monitors the CPU usage of all CICS transactions

IMS Agent injection and configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Injecting the AppMon IMS Agent

IMS SOAP gateway instrumentation

Preparing your System Profile for IMS Agents

AppMon System Profile - IMS Agent Groups

IMS sensor properties

The IMS Agent must be injected into the Control Region of each IMS system for which you want to capture Purepaths. It consists of programs that capture data on various IMS transaction processing events, and forwards that information to the zDC.

The IMS Agent traces IMS transactions submitted from the IMS MQ Bridge or MQ Trigger Monitor, IMS TM Resource Adapter, the IMS SOAP Gateway, and the IMS Connect API.

> **Note**
>
> It is only necessary to inject the IMS Agent into the Control Region. This lets it be used by all message processing regions associated with the Control Region. The job to inject the IMS Agent must be re-executed after each IMS restart.

The following are the requirements for generating a PurePath for different application types.

IMS MQ Bridge application requirements

- An AppMon instrumented front-end application.

- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Control Region.
  - Has the IMS Control Region sensor placed.
  - Has the Enable MQ Sensor box checked in the Sensor Properties.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Application Region(s).
  - Has the IMS Application Region sensor placed.
- The AppMon IMS Agent installed in the target IMS system.

IMS SOAP application requirements

- An AppMon instrumented front-end application.
- An IMS SOAP Gateway instrumented with the AppMon Java Agent.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS SOAP Gateway.
  - Has the CICS/IMS Gateways sensor placed.
  - Has the Java Web Services sensor placed.
- Either an AppMon exit added to IMS Connect or the SOAP gateway(s) configured to insert a tracking ID or both.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Control Region.
  - Has the IMS Control Region sensor placed.
  - Has the Enable SOAP Sensor box checked in the Sensor Properties.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Application Region(s).
  - Has the IMS Application Region sensor placed.
- The AppMon IMS Agent installed in the target IMS system.

IMS TM resource adapter application requirements

- An AppMon instrumented front-end application with a Java application server tier that uses the IMS TM Resource Adapter protocol.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Control Region.
  - Has the IMS Control Region sensor placed.
  - Has the Enable IMS TM Resource Adapter Sensor box checked in the Sensor Properties.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Application Region(s).
  - Has the IMS Application Region sensor placed.
- The AppMon IMS Agent installed in the target IMS system.

IMS connect API application requirements

- An AppMon instrumented front-end application with a System Profile that contains an Agent Group that has the IMS Connect API sensor placed.
- An AppMon exit added to IMS Connect.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Control Region.
  - Has the IMS Control Region sensor placed.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Application Region(s).
  - Has the IMS Application Region sensor placed.
- The AppMon IMS Agent installed in the target IMS system.

IMS MQ Trigger monitor application requirements

- An AppMon instrumented front-end application.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Control Region.
  - Has the IMS Control Region sensor placed.
- A System Profile that contains an Agent Group that does the following:
  - Maps to the IMS Application Region(s).
  - Has the IMS Application Region sensor placed.
  - Has the Enable MQ sensor box checked in the Sensor Properties.
- The AppMon IMS Agent installed in the target IMS system.

The following sections provide additional details about the instrumentation process. Detail information on the parameters can also be found in the ZDTINST PDS member.

Injecting the AppMon IMS Agent

To inject the AppMon IMS Agent, authorize *hlq.* LZDT610.SZDTIMSA by adding it to the APF list and then run the ZDTIINST job from the <HLQ>.SZDTSAMP PDS.

Sample JCL:

```
//S1       EXEC PGM=ZDTIINST,PARM='<IMS_Id>,<zDC_Id>'
//STEPLIB  DD DISP=SHR,DSN=<HLQ>.SZDTIMSA
```

Below is the complete list and description of the parameters to the IMS Agent injection program (ZDTIINST):

```
// ...    PARM='<IMS_Id>,<zDC_Id>,<E|D|G|M>|<PathTableSize>,
               <WaitZdcMin>,<Y|N>'
```

There are five positional parameters.

Symbol	Position	Required	Description
`<IMS_Id>`	1	Y	The four-character IMSID of the control region. This parameter is always required except when using the garbage collection function to reclaim ECSA storage for replaced, no longer used agent programs. For that function the IMSID parameter is not used.
`<zDC_Id>`	2	Y	The 4 character zDCID used in the SUBSYSTEM_ID() parameter of the target zDC. This parameter is always required except when the requested target zDC is the DEFAULT zDC. In that case supply an asterisk or omit the parameter. The default zDC, if one exists, will be selected.

`<E|D|G|M>| <PathTableSize>`	3	A one-character function parameter. A function parameter is optional and specifies an action to be taken on a previously injected (existing) IMS Agent. A function parameter is not applicable to the initial injection of an IMS Agent, and is mutually exclusive with the other form of the third positional parameter, `<PathTableSize>`. If a valid action code is specified, the injection program assumes that the action is to be taken on the currently injected IMS Agent. No attempt will be made to inject the agent or to update the current agent's load modules (implement service). Function values are: • **E** - Enable (switch on) an injected IMS Agent that was previously disabled. • **D** - Disable (switch off) an injected IMS Agent. The IMS Agent will no longer be invoked. Use the `E` function to re-enable it. • **G** - Garbage collection. If new IMS Agent service updates (PTFs) are installed, ECSA storage for old (unused) IMS Agent load modules is reclaimed with this function. See the section below titled Garbage Collection Function Notes for details on using this function. • **M** - Modify the recovery option of an injected IMS Agent. This value is used in conjunction with Parameter #5. • No third parameter - Inject the IMS Agent. This also is used to dynamically install new IMS Agent service updates (PTFs). • **<PathTableSize>** - Up to a 9 digit number that specifies the minimum number of pathid table entries to be allocated. The actual number of entries may be higher since the storage is allocated above the bar in megabytes. The number of entries will be adjusted upwards to use the amount of storage available. This parameter is optional and specifies the maximum number of 'in-flight' IMS transactions that the IMS Agent can track at one time. The default value is 4095. This value may only be specified when injecting the IMS Agent. It is mutually exclusive with the other form of the third positional parameter, `<E|D|G|M>`.
`<WaitZdcMin>`	4	Specifies a two char wait time for the target zDC to initialize in minutes. This parameter is optional. The default value is 30 minutes. The target zDC must have been initialized at least once since the last system IPL before the IMS Agent can complete initialization.

<Y|N>	5	A one-character value that indicates whether the IMS Agent should capture an SVC dump when ABEND recovery is driven. This parameter is optional and can be specified when the IMS Agent is initially injected, or specified in conjunction with the Modify or Enable function parameters to toggle dump capture during recovery on or off for a previously injected IMS Agent. The default value is N - dump capture is disabled.

Example 1: Inject the IMS Agent into IMS IB01 for zDC ZDC1. Accept the default PathTableSize and wait for up to 20 minutes for ZDC1 to initialize. Note that the third positional parameter was not specified.

```
//S1       EXEC PGM=ZDTIINST,PARM='IB01,ZDC1,,20'
//STEPLIB  DD DISP=SHR,DSN=<HLQ>.SZDTIMSA
```

Example 2: Disable the IMS Agent previously injected into IMSB for zDC ZDC1.

```
//S1       EXEC PGM=ZDTIINST,PARM='IMSB,ZDC1,D'
//STEPLIB  DD DISP=SHR,DSN=<HLQ>.SZDTIMSA
```

Example 3: Enable the previously injected, but disabled IMS Agent in IMSB for zDC ZDC1.

```
//S1       EXEC PGM=ZDTIINST,PARM='IMSB,ZDC1,E'
//STEPLIB  DD DISP=SHR,DSN=<HLQ>.SZDTIMSA
```

Example 4: Inject the IMS Agent into IMS IMSA for zDC ZDC1 and accept the defaults for Parameter#3 and Parameter#4 but specify that agent ABEND recovery should also capture an SVC dump.

```
//S1       EXEC PGM=ZDTIINST,PARM='IMSA,ZDC1,,,Y'
//STEPLIB  DD DISP=SHR,DSN=<HLQ>.SZDTIMSA
```

Example 5: Set agent ABEND recovery dump capture off for a previously injected IMS Agent.

```
//S1       EXEC PGM=ZDTIINST,PARM='IMSA,ZDC1,M,,N'
//STEPLIB  DD DISP=SHR,DSN=<HLQ>.SZDTIMSA
```

Injection notes

Start the zDC at least once after the system IPL before executing the IMS Agent injection job.

If the target zDC has not been up since the last IPL, the injection job will wait for it to come up. The wait time defaults to 30 minutes but can be set by the <WaitZdcMin> parameter. If the zDC is still not up after the wait time limit expires then injection cannot proceed and the job terminates with user ABEND code 100.

If the zDC has been up at least once since the last system IPL, the injection job can complete even when the zDC is not currently up. In this case warning messages are issued indicating that the zDC is not started and data collection is not available, and the job completes with RC=4.

Start and fully initialize the IMS system before you execute the IMS Agent injection job. If you stop and start the IMS system, the IMS Agent injection job must be re-run. If you start a new IMS dependent region, or stop and restart one, no action is required.

Using an automation facility like IMS Time-Controlled Operations (TCO) is recommended to execute the IMS Agent injection job.

All IMS Agent injection messages are written to the job log. You can control the level of messages by including specific DDNAMEs in the injection job JCL. The default if no DDNAMEs are included is level 4 for informational messages. The specific DDNAMEs and message levels are describe below:

//DTMSGLV0 DD SYSOUT=* For finest level 0 DEBUG messages and above

//DTMSGLV2 DD SYSOUT=* For fine level 2 DEBUG messages and above

//DTMSGLV4 DD SYSOUT=* For informational level 4 messages and above

//DTMSGLV5 DD SYSOUT=* For warning level 5 messages and above

Injection job failures typically result in a user ABEND code 100. In most cases the cause of failure can be determined by prior diagnostic messages that have been written to the job log. A symptom dump may also be written to the job log, depending on your zOS system parameters, but a dump will not be captured unless one of the following DDNAMEs is included in the injection job JCL:

```
//SYSABEND DD SYSOUT=*
//SYSMDUMP DD DSN=DT.IMSAGENT.INJECT.SDUMP,DISP=(OLD,KEEP,KEEP)
```

Garbage collection function notes

If new IMS Agent service updates (PTFs) are installed, ECSA storage for old (unused) IMS Agent load modules can be reclaimed with this function. After the injection job is run using the libraries containing the service updates, IMS message region activity is necessary to update internal control blocks to point to the new modules. After that occurs, you can rerun the injection program using the 'G' function to mark those old unused modules as candidates for deletion. Then after at least 3 hours has elapsed, run the injection program a second time using the 'G' function to release the ECSA storage containing those marked modules. You can run the injection program using the 'G' function repeatedly, there is no negative side effect to doing so. But in order to have the garbage collection process reclaim unused module storage, the process must be invoked at least twice after an IPL, with at least 3 hours elapsing between the first and second invocation.

IMS SOAP gateway instrumentation

The IMS SOAP Gateway runs on z/OS, z/Linux and Windows. Modify your IMS SOAP Gateway startup script (iogstart.sh or iogstart.bat) to instrument the gateway with the AppMon Java Agent. Refer to Java Agent Configuration for more details.

Windows example iogstart.bat:

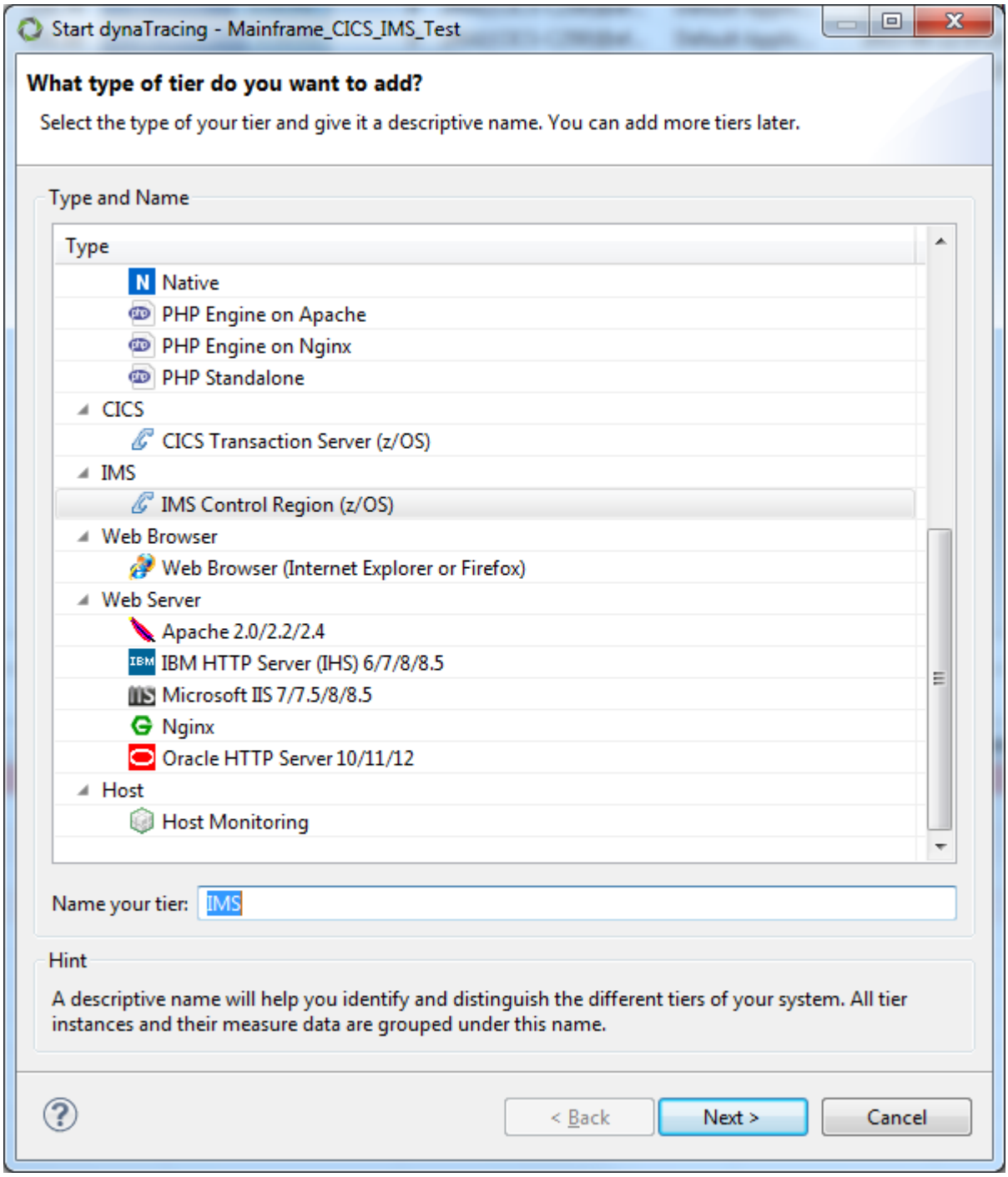
```
Set zDt=-agentpath:"C:\Program Files\dynaTrace\dynaTrace <dTMajor>.<dTMinorVersion>.0\age
lib\dtagent.dll"=name=IMSGw,server=<dTServerName>:9998,transformationmaxavgwait=150"
start "IMS SOAP Gateway" "!\_RUNJAVA!" !JAVA_OPTS! !zDT! -classpath !CLASSPATH!
org.apache.catalina.startup.Bootstrap %* start
```

IMS Connect Job Modification

Add the authorized AppMon IMS Agent dataset <HLQ>.SZDTIMSA to the IMS Connect job STEPLIB to permit IMS Connect to load the AppMon exit HWSTECL0. If IMS Connect Extensions is in use, concatenate SZDTIMSA after the IMS Connect Extensions library, If a locally developed HWSTECL0 exit is in use, concatenate SZDTIMSA ahead of the dataset that contains the local exit. The IMS Connect exit is required to support the IMS Connect API protocol and to support the IMS SOAP Gateway in cases where the SOAP Gateway has not been configured to insert a tracking ID using the iogmgmt -tracking command.

Preparing your System Profile for IMS Agents

Add an IMS tier to the system profile that was created in the installation of the zDC. The wizard presents a number of tiers that are available. Select the IMS Agent. You may want to add a tier for the Control Region(s) and a separate tier for the Application Region(s). You should name your tier(s) appropriately.



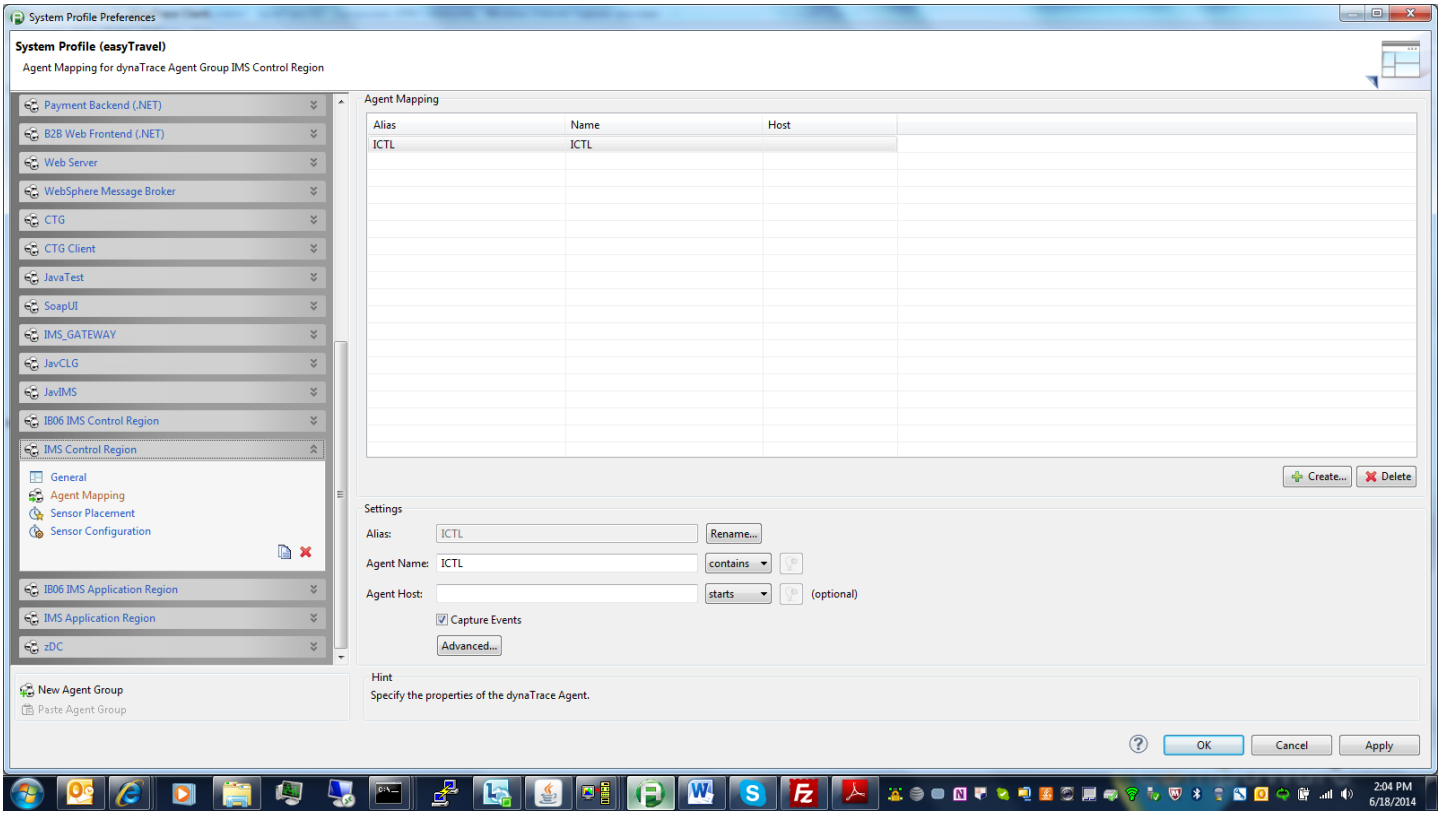
Select the Tier Type

AppMon System Profile - IMS Agent Groups

In the AppMon Client, edit the System Profile for the IMS Agent Groups.

Add or edit the IMS Agent Groups for the IMS Agent. You may choose to include all IMS Agents within a single Agent Group, or assign some IMS Agents to separate Agent Groups.

Within each IMS Agent Group, under Agent Mapping, you must specify an Agent Name. This assigns the IMS Agent(s) to the desired Agent Group. An Agent is assigned to the first Agent Group in the list for which the Agent Name satisfies the Agent Mapping criteria. The following is an example of a name specification that you can use to map all IMS Control Regions to the Agent Group.

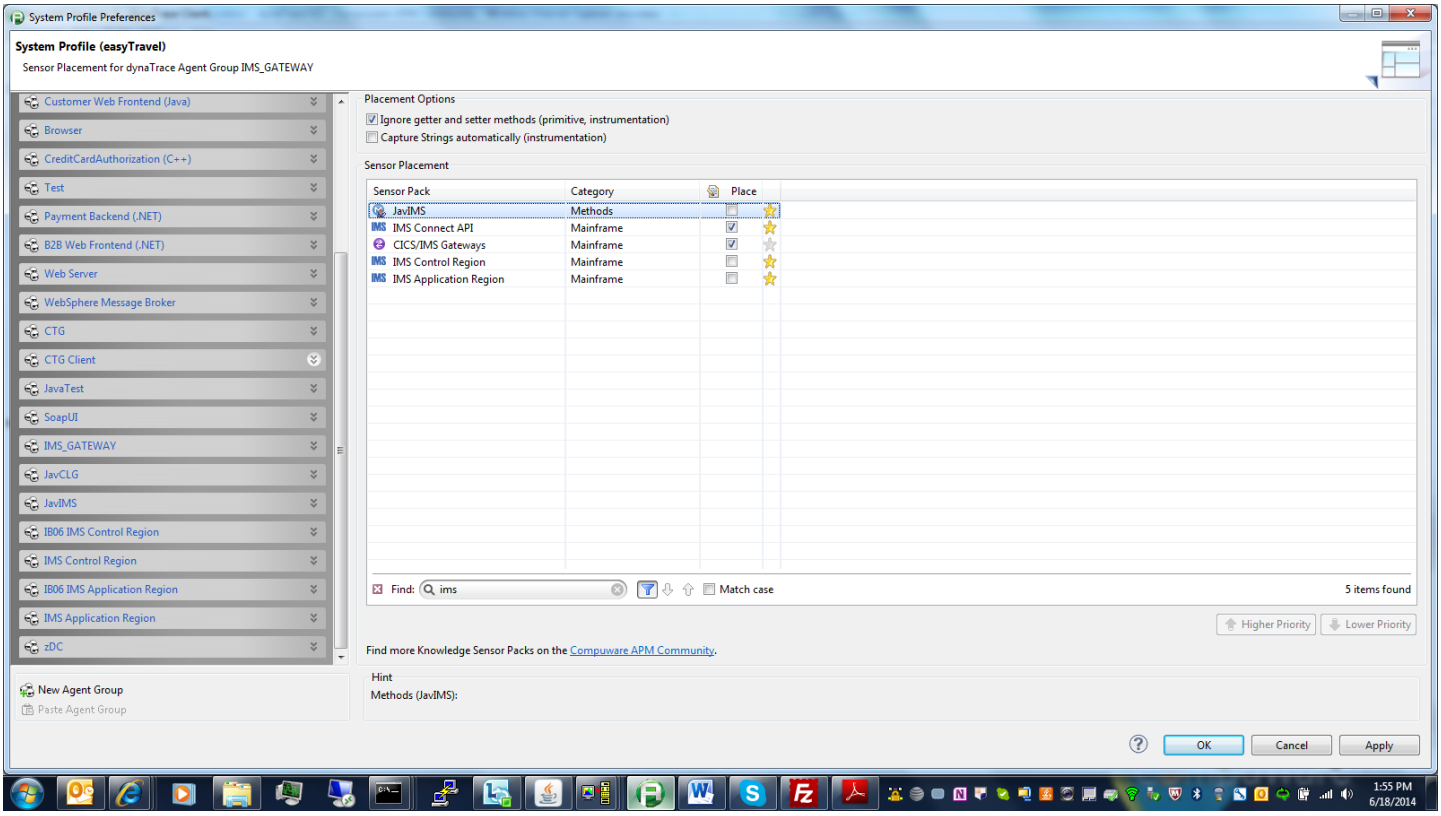


Control Region Agent Mapping

Within each IMS Agent Group, under Sensor Placement, select the IMS sensors to place.

Note

Click in the sensor list and click **Ctrl-F [Find] IMS** to view the IMS related sensor packs. You can use this functionality anywhere in the product

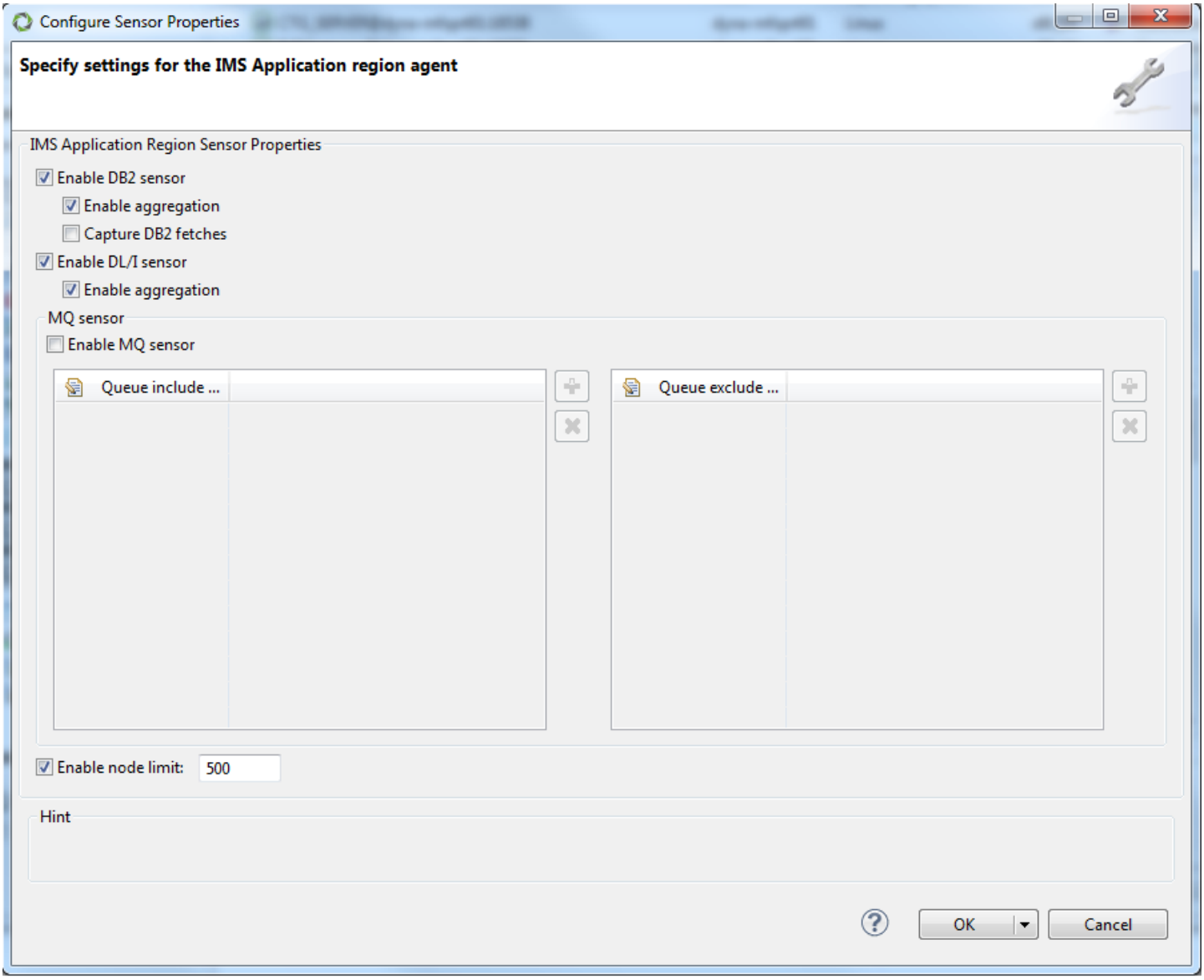


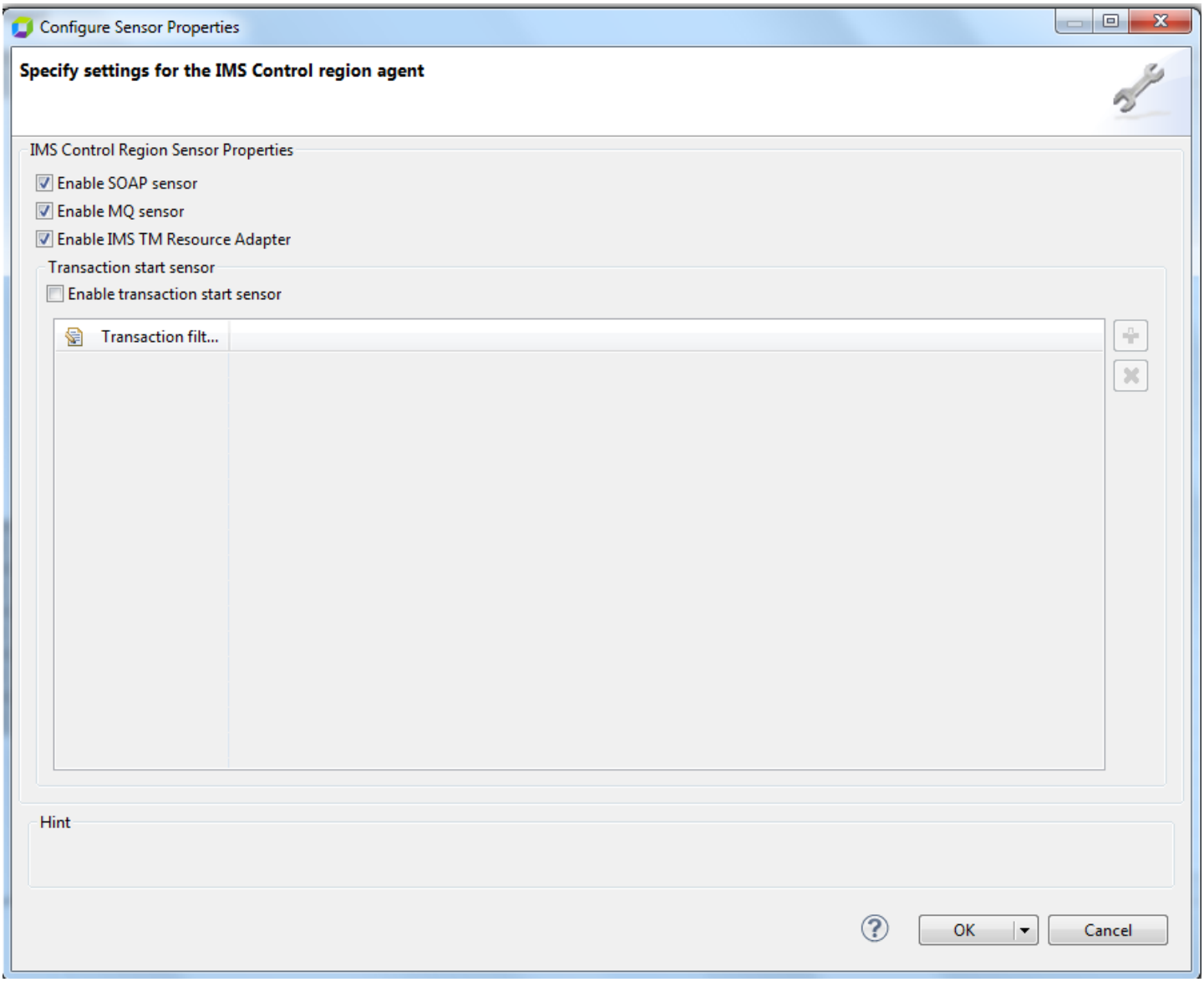
Agent Group Sensor Placement

The IMS sensors that you can place are:

- IMS Connect API
- CICS/IMS Gateways
- IMS Control Region
- IMS Application Region

Within each IMS Agent Group, under Sensor Configuration, select properties for the IMS sensors. The Java sensors for IMS Connect API and IMS SOAP Gateway have no configuration options. You can configure sensor properties for the IMS Control Region and IMS Application Region sensors.


Configure Sensor Properties
Specify settings for the IMS Application region agent
IMS Application Region Sensor Properties
Enable DB2 sensor
Enable aggregation
Capture DB2 fetches
Enable DL/I sensor
Enable aggregation
MQ sensor
Enable MQ sensor
Queue include ...
Queue exclude ...
Enable node limit:
500
Hint
OK
Cancel



IMS sensor properties

Use the following settings to configure the IMS Control Region Agent:

Setting	Default Value	Remarks
Enable SOAP sensor	enabled	Enables/disables the SOAP Sensor. When disabled, AppMon does not trace transactions that start through the IMS SOAP Gateway. When enabled, all transactions that start through the IMS SOAP Gateway are traced.
Enable MQ sensor	disabled	Enables/disables the MQ Sensor. When disabled, AppMon does not trace transactions that start through the IMS MQ Bridge. When enabled, AppMon traces all transactions that start through the IMS MQ Bridge.
Enable IMS TM Resource Adapter sensor	enabled	Enables/disables the IMS TM Resource Adapter Sensor. When disabled, AppMon does not trace transactions that start through the IMS TM Resource Adapter. When enabled, all transactions that start through the IMS TM Resource Adapter are traced.

Enable transaction start sensor	disabled	Enables/disables the transaction start sensor. When disabled, AppMon ignores the Transaction filters, and only traces the transactions that start through SOAP (when SOAP sensor is enabled), MQ (when MQ sensor is enabled), ITRA (when IMS TM Resource Adapter sensor is enabled), or the IMS Connect API. This setting does not affect the transaction codes that other sensors trace. This setting does not have to be on to trace transactions that start through IMS program switches, if the originating transaction is traced. When enabled, Transaction filters determine if AppMon should trace a transaction, if it does not start through SOAP, MQ, ITRA, or IMS Connect API. This sensor can start a path to trace an IMS transaction that cannot link to previous activity outside of IMS.
Transaction filters	empty	A list of transaction code patterns to be traced. Each entry in the list may be a transaction code or a transaction code prefix with an asterisk at the end. The TX sensor ignores transactions with names that don't match an entry in the list. If an entry has a single asterisk *, all IMS transactions are traced. We do not recommend this.

Use the following settings to configure the IMS Application Region Agent:

Setting	Default value	Remarks
Enable DB2 sensor	enabled	Enables/disables the DB2 Sensor. When disabled, AppMon does not capture DB2 calls from the application region. When enabled, AppMon captures all DB2 calls (see Capture DB2 fetches).
Enable aggregation	enabled	Enables/disables the aggregation of DB2 calls by call type in the UI. When disabled, each DB2 call is a separate node in the PurePath. When enabled, AppMon summarizes each recurring DB2 call as a single node in the PurePath.
Capture DB2 fetches	disabled	Enables/disables the capture of DB2 fetches. When disabled, DB2 fetches from the application region are not included in the captured DB2 calls. When enabled, DB2 fetches from the application region are included.
Enable DL/I sensor	enabled	Enables/disables the DL/I Sensor. When disabled, AppMon does not capture DL/I calls from the application region. When enabled, it captures all DL/I calls.
Enable aggregation	enabled	Enables/disables the aggregation of DL/I calls by call type in the UI. When disabled, each DL/I call is a separate node in the PurePath. When enabled, AppMon summarizes each recurring DL/I call as a single node in the PurePath.
Enable MQ sensor	disabled	Enables/disables the MQ Sensor. When disabled, AppMon does not capture MQ API calls from the application region and cannot start traces in the application region when an MQGET occurs, such as transactions that start through the IMS MQ Trigger Monitor. When enabled, AppMon captures all MQ API calls and can start traces. A trace in the application region is only started when the transaction issuing the MQGET is not already being traced by another sensor and the queue name passes the MQ queue name include/exclude filtering.
MQ queue include filters	empty	A list of queue name patterns to be included for potentially starting a trace when an MQGET occurs. Each entry in the list may be a queue name or a queue name prefix with an asterisk at the end. The MQ sensor will never start a trace on an MQGET from queues whose names do not match an entry in the list. **By default, all queue names are enabled for tracing.** This list is has no effect on the normal capture of MQ calls from the application region when another sensor is tracing the transaction.

MQ queue exclude filters	empty	A list of queue name patterns to be excluded from potentially starting a trace when an MQGET occurs. Each entry in the list may be a queue name or a queue name prefix with an asterisk at the end. The MQ sensor will never start a trace on an MQGET from queues whose names match an entry in the list. If a particular queue name matches both the include and exclude lists, the exclude list takes priority. This list is has no effect on the normal capture of MQ calls from the application region when another sensor is tracing the transaction.
Enable node limit	disabled	Enables/disables node limit truncation. When disabled, AppMon ignores the associated specified value and does not restrict the number of nodes in the subpath. When enabled, AppMon truncates the subpath when it reaches the specified number of nodes.

IMS Agent sensor notes

Agent initialization

Program ZDTIINST injects the IMS Agent into an IMS system, but the Control Region agent is not fully initialized until the IMS system receives an input message (it need not be for a transaction that will be tracked). Until this occurs, the application (dependent) region ADK and MQ sensors will not function. Similarly, an IMS application region is not fully initialized until some IMS work is scheduled in it.

Until the Control Region sensor configuration is obtained from the AppMon Server, no Control Region paths will be started. Until the dependent region sensor configuration is obtained, all sensors are assumed to be disabled.

Therefore, it is expected that until the agent is fully initialized and the sensor configuration is known, there may be some missed or incomplete PurePaths.

MSC considerations

The IMS agent can generate a PurePath that spans IMS Control Region hops across MSC connections on the same or different LPARs, provided that the IMS agent is injected into all participating IMS systems, and the applicable IMS Control region sensor is enabled for all participating IMS Control Regions. For example, if a transaction is submitted to IMSA from the MQ Bridge and that sensor is enabled for the IMSA Control Region, a path will be started. If that transaction, or any switched to transaction, is MSC routed to IMSB, the MQ sensor must be enabled for the IMSB Control Region as well in order to continue the path.

ADK or MQI get (Trigger Monitor) started transactions cannot be traced across MSC connections. Any transactions that are started due to program switches after the path is started will be included in the path, provided they execute on the local IMS system.

zLocal Agent

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

zRemote Agent

zDC

The zLocal Agent and zRemote Agent handle the TCP/IP connection between the mainframe and the AppMon Collector. The zLocal Agent installs on the mainframe and passes performance data to the zRemote Agent, reducing CPU consumption on z/OS.

Note

The z/OS Agent can't directly connect to an AppMon Collector without first going through a zRemote Agent. Using the zRemote Agent reduces CPU consumption on z/OS. The zLocal Agent only transfers data to the zRemote Agent, which can run on Windows or Linux (64-bit OS only).

zRemote Agent

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Installation and configuration

Optional parameters to register the Windows service / start the *NIX daemon

Windows service

*NIX daemon

The zRemote Agent is an aggregation and data processing component that installs in the distributed environment and extends the zLocal Agent. The zRemote Agent offloads some of the Agent data processing in the zLocal Agent, reducing the MIPS load incurred in instrumenting mainframe applications.

Note

The recommended best practice is to have the zRemoteAgent only service one zLocalAgent (zDC). Refer to the instructions on installing multiple zRemoteAgent launcher service instances for configuring multiple zRemoteAgents.

Installation and configuration

The parameter for `DTAGTCMD` is part of the zDC SYSIN. See zDC SYSIN Parameters for more information.

The port in `zremoteagent=ipaddress:port` defaults to 8898. That `ipaddress` and `port` should correspond to a distributed machine where the actual zRemoteAgent executes. The port parameter is optional and defaults to 8898.

The zRemote Agent is supported on the following operating systems:

- Windows x64 2008 R2 / 2012 R2
- Linux x64

Optional parameters to register the Windows service / start the *NIX daemon

- `zdclistenerport` is the port that the zRemoteAgent listens to for zDC connections. This corresponds to the port in the `zremoteagent` SYSIN DTAGTCMD parameter described above. This setting is optional, and if not present defaults to 8898.
- `name` is the standard AppMon Agent name mapped in the System Profile. This name should reflect that this is a zremoteagent. If it services a particular LPAR, it could include the SMF ID of that LPAR, for example, `"name=zremoteagent_<lpar>"`. When you click **Add Tier > CICS** in the default Monitoring System Profile, the mapping defaults to CICS_Monitoring. This value is also optional, but is highly recommended.
- `server` is the name of AppMon Collector with which to connect. This setting is optional, and if not present defaults to `localhost`.

Windows service

For a default AppMon full installation on 64-bit Windows, `<DT_HOME>` is `C:\Program Files\dynaTrace\dynaTrace 7.0`, for an Agent-only installation, go to `C:\Program Files (x86)\dynaTrace\dynaTrace Agent 7.0`. The zRemote Agent`dtzagentlauncher.exe` is in `<DT_HOME>\agent\lib64`.

The associated `dtzagentlauncher.ini` must be modified similar to the following:

```
`-exec=dtzagent.exe
      -Deof=eof
      -name=CICS_Monitoring
      -server=localhost
      -zdclistenerport=8898`
```

Installing a 2nd (or 3rd) zRemoteAgent launcher service instance

Create a new `.ini` file containing the different parameters (usually `zdclistenerport=`, and `name=`). To install the new service instance, issue the following command. In this example, the new port =8899, so the new `.ini` file is named as such.

```
dtzagentlauncher -service install "-ini=dtzagentauncher8899.ini" -
servicename=<your service name>
```

*NIX daemon

Install a `.jar` with `java -jar <pathToJar>/dynatrace-agent-7.0.0.####-unix.jar`.
When you issue a `cd/opt` command and perform a default 64-bit AppMon full or Agent-only installation on a 64-bit *NIX, then `<DT_HOME>` is `/opt/dynatrace-7.0.`
The executable is `dtzagent` in `<DT_HOME>/agent/lib64`.

You may have to use `chmod` or `chown` to get the right permissions.

For testing, you can use the example shell script to interactively start the zRemote Agent in the background with `<DT_HOME>/init.d/dynatracezremoteagent start`.
To permanently add the zRemote Agent to your boot sequence and start it as a daemon, either copy `[](/support/doc/appmon/images/application-monitoring/agents/zos-agent/zremote-agent/dynatracezremoteagent)` to, or reference it in, `/etc/init.d` and activate it with `chkconfig --add dynaTracezRemoteAgent`.

You can also start the zRemote Agent with `<DT_HOME>/agent/lib64/dtzagent zdclistenerport=<port> name=<AgentName>[server=<CollectorName>[:port]]`.
The `server` (Collector) argument defaults to `[localhost:9998](http://localhost:9998)`.
The `zdclistenerport` argument defaults to 8898. See above for a description of the `name` parameter.

You can also use the new dtzagentlauncher by modifying the `dtzagentlauncher.ini` file

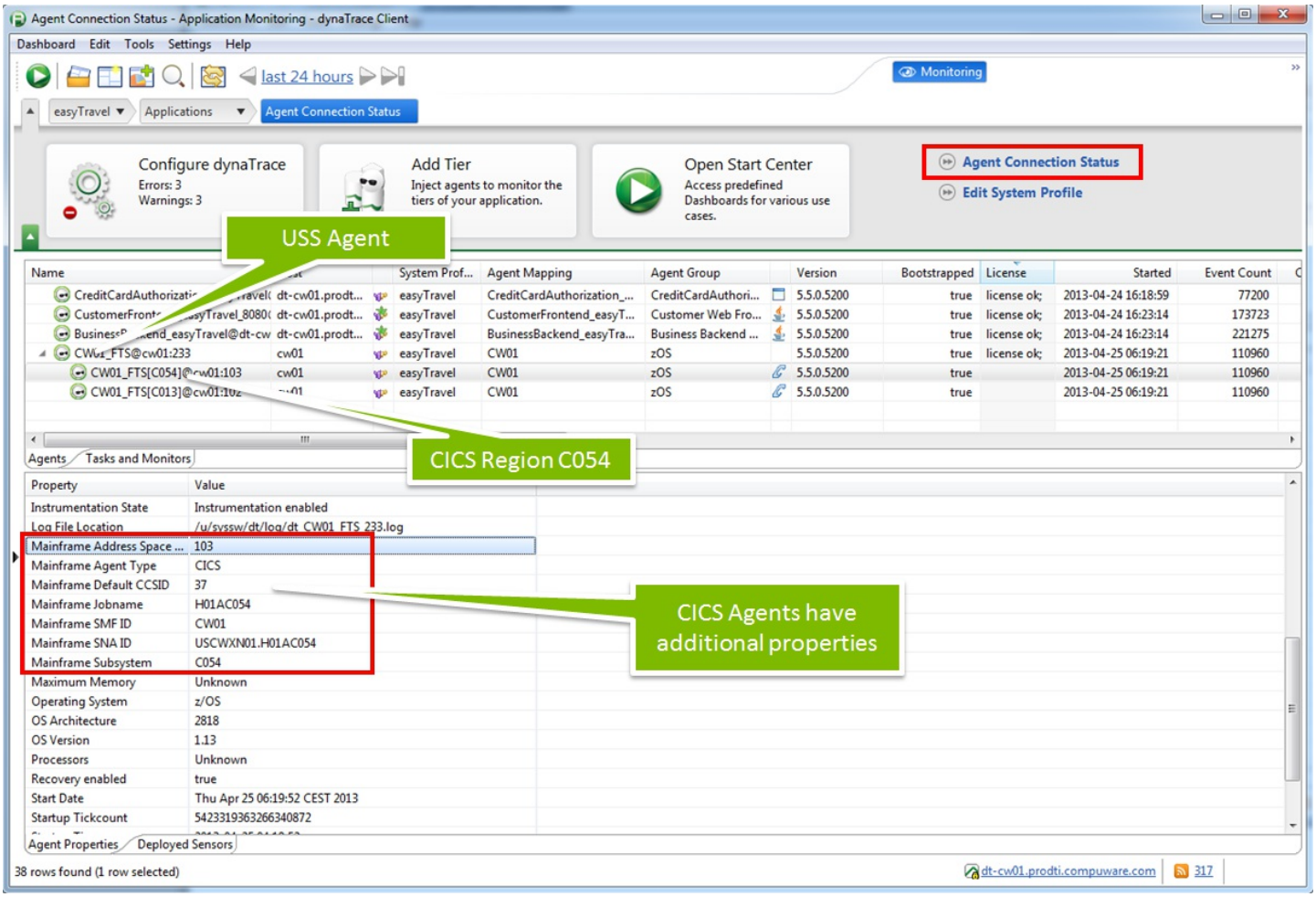
```
`-Deof=eof
      -name=CICS_Monitoring
      -server=localhost
      -zdclistenerport=8898`
```

Installing a 2nd (or 3rd) zRemoteAgent launcher instance

Create a new `.ini` file that contains the different parameters (usually `zdclistenerport=` and the `name=`). To start the new instance, specify the `-ini` parameter. For example:

```
./dtzagentlauncher -ini=dtzagentauncher8899.ini
```

Testing the installation



Agent Connection Status details

- Whether the license is OK.
- Whether the Agent was mapped into the right System Profile.
- Whether the Agent is currently sending events.

The **Agent Connection Status** view also indicates:

- If the zRemote Agent started properly, it should appear in the list of agents.

- The zDC and its zLocal Agent do not appear in the agents overview. If no CICS or IMS Agents appear, check the zRemote Agent and zLocalAgent log files to verify that they are running and communicating properly.
- Each CICS or IMS Agent that connects to the zDC appears in the list as a subagent of the zRemote Agent. Each Agent sets several properties that contain details about the CICS or IMS region. If no Agents show up in the list, check the Agent configuration. To verify the connectivity between the CICS Agent and the zRemote Agent, you can use the DTAX transaction PING command to test connectivity, as described in the DTAX documentation. The IMS agents do not appear immediately when they are injected into the IMS subsystem. The next transaction that is processed by the control region causes the IMS control region agent to appear, and then the next transaction processed by each application region causes that IMS application region agent to appear.

In the AppMon Client, you can verify the CICS and/or IMS Agent installation by viewing the **Agent Connection Status**, which is available through the Configuration Panel in the **Application Overview** and in the **Agents Overview** dashlet.

DTAX transaction

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Change the log level

Send a ping message

Enable or disable the AppMon data capture

DTAX messages

Related topics

zOS Agent

CICS Agent installation and configuration

CICS Transaction Server sensor

Transaction measure

zDC

zOS Agent logs

zOS Agent messages

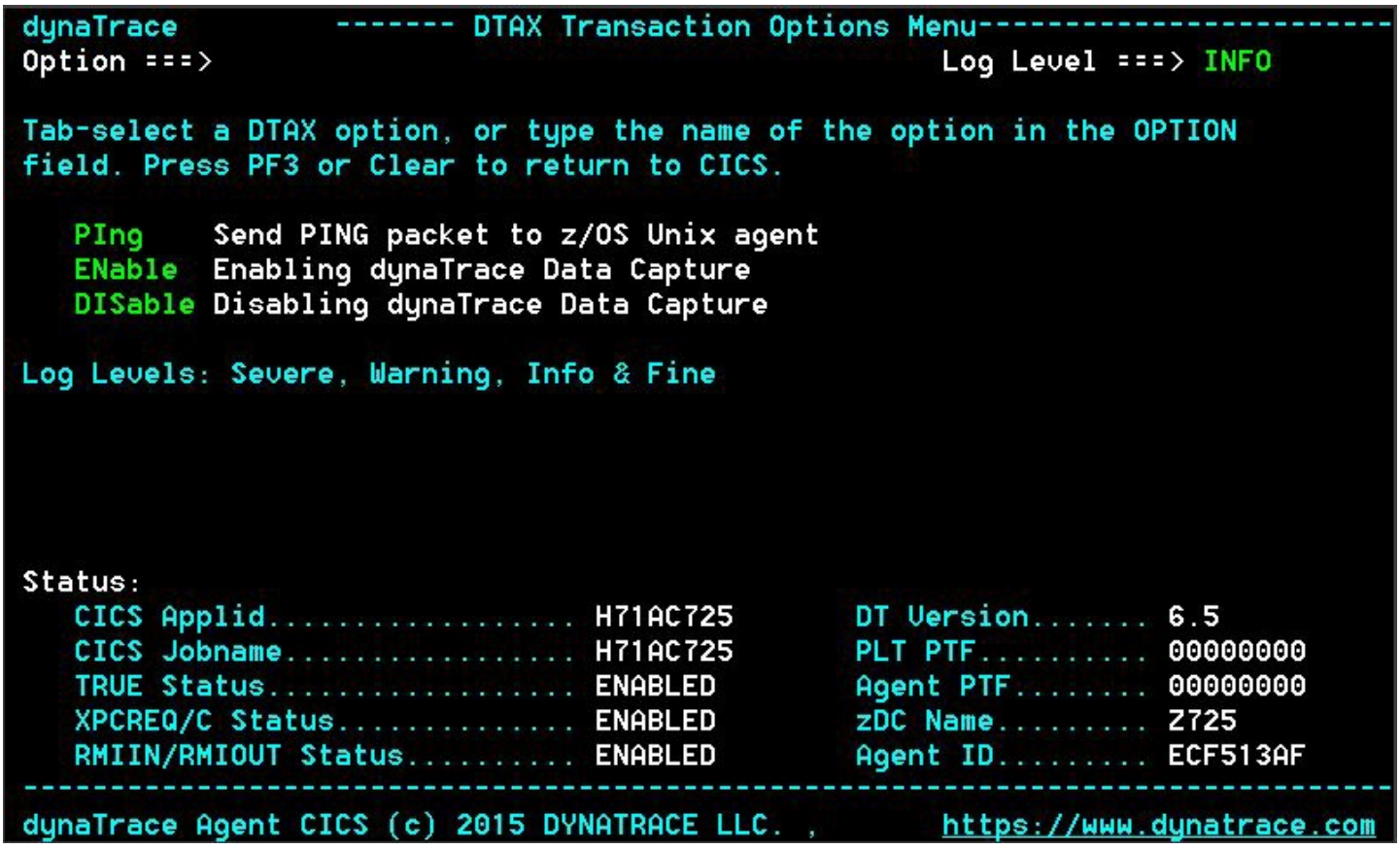
zOS Agent troubleshooting

Note

These functions are primarily intended to be used for support purposes. None of them is required for normal operation, but you may wish to automate agent enablement by submitting DTAX ENABLE and DTAX DISABLE transactions through an operations automation system.

The DTAX transaction runs automatically every 5 minutes. It initializes the Agent connection to a newly started or restarted zDC at that time. Because data is only collected while the zDC is active, it should be running prior to CICS or IMS region initialization. In addition, data may be lost for up to 5 minutes after the zDC is restarted or when the zDC is initially started after the CICS region initializes. Optionally, the reconnection process can be completed immediately by manually entering the DTAX transaction and performing the DISABLE function followed by the ENABLE function after reinitializing the zDC. Use the ENABLE function alone if the CICS region had not been connected to a zDC and thus shows a DISABLED status.

The DTAX transaction also provides a UI to access basic functionality of the CICS Agent. Logon to CICS and issue the DTAX transaction without parameters for a brief description of its functions and current Agent status.



Change the log level

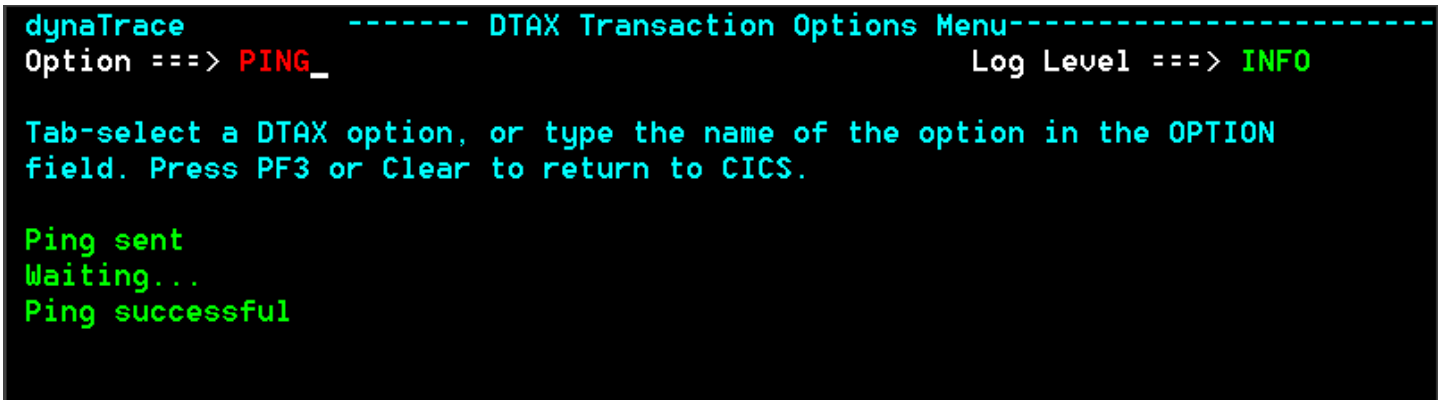
The CICS Agent logs messages to the zRemote Agent's log file, which and can be viewed using the System Information dashlet. The log level of the CICS Agent is set to INFO and, unlike the other Agents, it cannot be changed within the AppMon Client. It can, however, be changed using the DTAX transaction. Set the cursor to the upper right corner and type one of the following log levels: **Severe**, **Warning**, **Info** or **Fine**.

Send a ping message

You can use the ping message functionality to verify the connectivity between the z/OS CICS Agent and the zRemote Agent.

```
DTAX ping
```

The ping message generates the following DTAX output



On success, this also generates a log entry in the log output of the zRemote Agent, such as:

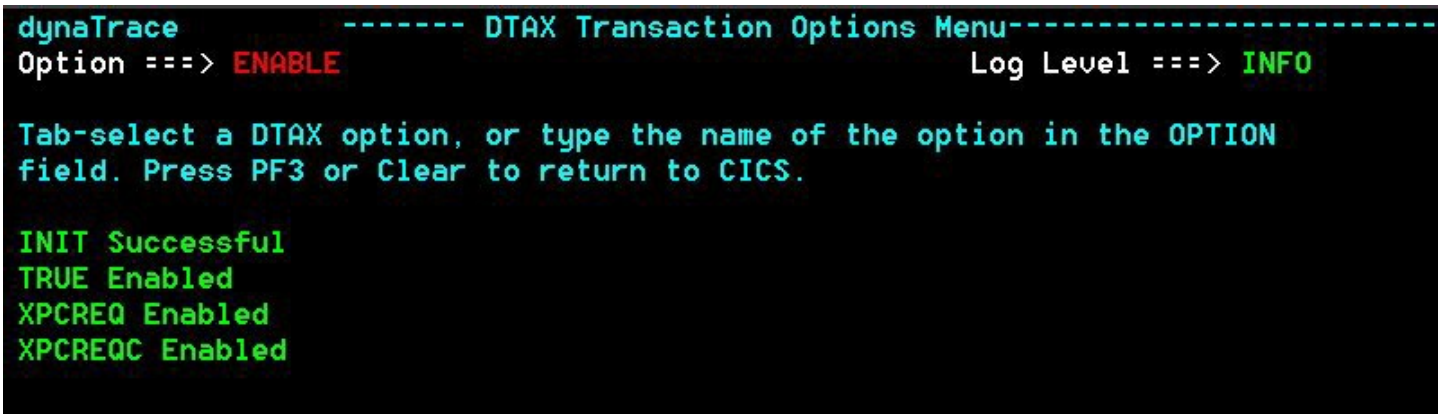
```
info    [native] ASID[53], smfID[S0W1], sysid[C449], jobName[H71AC449] Ping data=asid=53,
```

Enable or disable the AppMon data capture

These functions enable or disable all the exits used by the zLocal Agent. The **Enable** command also starts the DTAX five (5) minute restart cycle if one was not in progress. Likewise, the **Disable** command terminates any active DTAX five (5) minute restart cycle.

Enable CICS Agent

Enable should generate the following DTAX output:

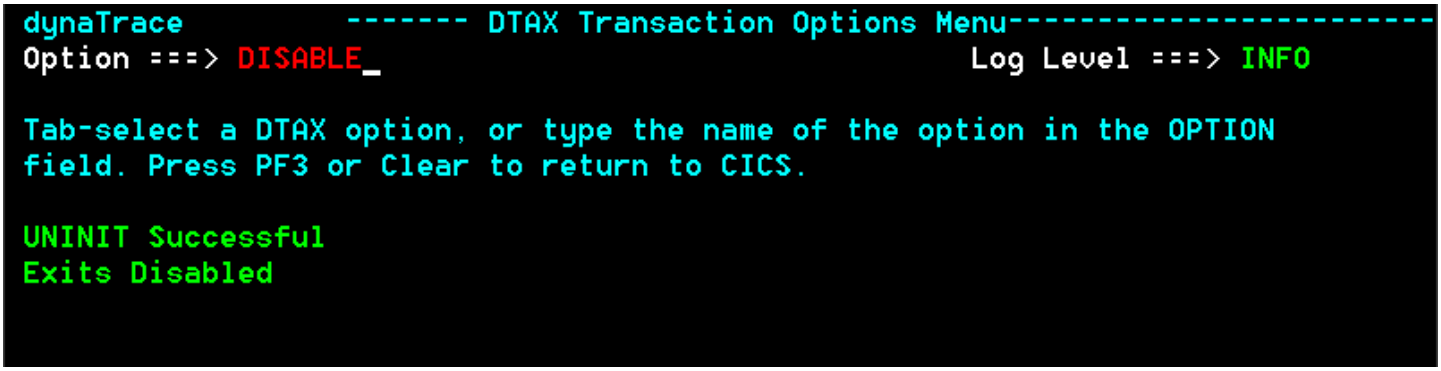


Enable should also register this CICS address space with the AppMon Collector. From the zRemote Agent logs:

```
info    [native] ASID[53], smfID[S0W1], sysid[C449], jobName[H71AC449], subagentid[f526fc
info    [native] New subAgent registered successfully: f526fc70
info    [native] Subagent license is updated, id=[0xf526fc70] = license ok;
```

Disable CICS Agent

Disable should generate the following DTAX output:



Disable should also un-register this address space from the AppMon Collector. From the zRemote Agent logs:

```
info    [native] Unregistering subAgent f526fc70
```

DTAX messages

All DTAX messages are described on the zOS Agent Messages page.

zOS Agent messages

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

zDC system messages

DTAX system messages

IMS Agent system messages

Related pages

zDC

DTAX transaction

The AppMon zOS Agent components can issue the following types of system message:

zDC system messages

ZDC000I INITIALIZATION @@ FOR zDC @@ VER @@.@@.@@

EXPLANATION:	This informational message indicates that initialization of zDC has started or ended. The subsystem ID and the version of the product are displayed.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC001E AN ERROR HAS OCCURRED DURING LOAD FOR MODULE @@

EXPLANATION:	At zDC system startup an attempt is made to load all the processing modules into storage. This load was unsuccessful for the module named in this message.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Ensure that the //STEPLIB or //JOBLIB DD statements in the JCL executing zDC point to the correct data sets and that these data sets contain the correct release of zDC.

ZDC002E MODULE=@@ R15=@@ R1=@@

EXPLANATION:	This message follows ZDC001 and gives the module name as well as the contents of R15 and R1 at the completion of the LPAD operation.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Save the information in this message if you have to contact your systems programmer or contact support.

ZDC003E COMMON AREA STORAGE NOT AVAILABLE FOR MODULE @@

EXPLANATION:	This message indicates that the attempt to load the module named in the message into extended common storage has failed because there is not enough available storage to hold the module.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Increase the amount of common storage, specifically subpool 241, and start zDC again.

ZDC004E AN ERROR HAS OCCURRED DURING BLDL FOR MODULE @@

EXPLANATION:	At zDC system startup an attempt is made to load all the processing modules into storage. The BLDL for this load was unsuccessful for the module named in the message.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Ensure that the //STEPLIB or //JOBLIB DD statements in the JCL executing zDC point to the correct data sets and that these data sets contain the correct release of zDC.

ZDC005E MODULE=@@ R15=@@ R1=@@

EXPLANATION:	This message follows ZDC004 and gives the module name, and the contents of R15 and R1 at the completion of the BLDL operation.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Save the information in this message if you have to contact your systems programmer or contact support.

ZDC006E SUBSYSTEM @@ ALREADY EXISTS AND IS ACTIVE

EXPLANATION:	This message indicates that the 4-character subsystem ID you specified in the //SYSIN DD parameter data set already exists on your system and it is an active zDC subsystem.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Ensure that the subsystem ID you have specified is unique on your system then restart zDC or stop the instance of zDC that holds this name.

ZDC007E ECSA STORAGE NOT AVAILABLE FOR SUBSYSTEM @@, R15=@@

EXPLANATION:	This message indicates that the attempt to obtain storage for the subsystem control block has failed. There is not enough storage above the line in extended CSA subpool 241 for these blocks.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Increase the amount of extended CSA subpool 241 storage and restart zDC.

ZDC008I MODULE P/T DATE TIME VERSION ADDRESS LENGTH

EXPLANATION:	This message is displayed only at zDC startup. It names the columns shown below in multiple message numbers ZDC009. The module name, highest problem tracking number, the date and time the module was last assembled, the version of the module and the main storage address where it is loaded are shown.
SYSTEM ACTION:	Processing continues.

USER RESPONSE:	None.

ZDC009I @@ @@ @@ @@ @@ @@ @@

EXPLANATION:	This message lists one module loaded by zDC. See the ZDC008 message for a description of the columns.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC010E SDUMPX TAKEN SUCCESSFULLY FOR zDC @@ ABEND

EXPLANATION:	This message is displayed on the system console by the zDC dump routines. It names the abend code.
SYSTEM ACTION:	Execution terminates.
USER RESPONSE:	Prepare the dump for examination by your systems staff or for transmission to support. Reexecute zDC.

ZDC011E SDUMPX FAILED FOR zDC @@ ABEND. RETCODE=@@/@@

EXPLANATION:	This message is displayed on the system console by the zDC dump routines and it indicates that the attempt to obtain a system dump has failed. The return and reason code for the failure is shown in the message.
SYSTEM ACTION:	Execution terminates.
USER RESPONSE:	Examine the return and reason codes or supply this message to support. Reexecute zDC.

ZDC012E @@ ABEND DUMP IN DYNATRACE zDC @@

EXPLANATION:	This message is the title for all SDUMPX macros issued by zDC. This title indicates the abend code that caused the dump to be taken.

SYSTEM ACTION:	Execution terminates.
USER RESPONSE:	Ensure that the dump is preserved and can be given to your systems programming staff, or sent to support. Reexecute zDC.

ZDC014E ZDCLOGER VSAM @@ ERROR

EXPLANATION:	This message indicates that a non-zero return code has resulted from a VSAM zDC log request. The name of the request is contained in the message.
SYSTEM ACTION:	The log task terminates but execution of zDC continues.
USER RESPONSE:	Examine the following messages, which contain detailed information about the nature of the error. Keep this information for your systems programming staff or for support. Try to and re-attach and re-open the log(MVS command:Modify ZdcStcName,LOG OPEN).

ZDC015E ZDCLOGER VSAM RPL FEEDBACK WORD @@

EXPLANATION:	This message contains the value of the VSAM RPL feedback word following a VSAM PUT to the log data set.
SYSTEM ACTION:	The log task terminates but execution of zDC continues.
USER RESPONSE:	Examine the following messages, which contain detailed information about the nature of the error. Keep this information for your systems programming staff or for support. Try to and re-attach and re-open the log(MVS command:Modify ZdcStcName,LOG OPEN).

ZDC016E ZDCLOGER VSAM ACB ERROR FLAG @@

EXPLANATION:	This message contains the value of the VSAM ACB error flag following a VSAM OPEN or CLOSE of the log data set.
SYSTEM ACTION:	The log task terminates but execution of zDC continues.
USER RESPONSE:	Examine the following messages, which contain detailed information about the nature of the error. Keep this information for your systems programming staff or for support. Enter the ISPF zDC control system and re-attach and re-open the log.

ZDC017E SRB ABEND @@ AT DISPLACEMENT @@ INTO ZDCSRBRT

EXPLANATION:	This message indicates that an abend has occurred at the displacement shown into the SRB routine named ZDCSRBRT.
SYSTEM ACTION:	Execution terminates.
USER RESPONSE:	If a dump is produced, save it for your systems staff or for transmission to support. Reexecute zDC.

ZDC018E EXTENDED CSA STORAGE NOT AVAILABLE FOR SRB ROUTINE

EXPLANATION:	This message indicates that a STORAGE OBTAIN has failed reporting that extended CSA storage from subpool 241 is not available.
SYSTEM ACTION:	Execution terminates.
USER RESPONSE:	Report this condition to your systems programmer who may want to increase the amount of extended CSA storage in the system.

ZDC019E zDC INITIALIZATION FAILED. RETCODE=@@/@@

EXPLANATION:	This message indicates that the initialization routines have failed and the return and reason codes describing this failure are shown
SYSTEM ACTION:	zDC execution terminates.
USER RESPONSE:	Examine the SYSPRINT data set for any other messages that may be relevant to the failure. Report this message to support.

ZDC020I A LIST OF @@ EXIT ROUTINES FOLLOW:

EXPLANATION:	This message is the header for the ZDC021 messages that follow. All of the exit routines that you have active plus the one for zDC should be listed. Note that additional exit routines may be added after zDC starts and, if so, their names will not be in this list.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	Verify that the exits you expect to be active are active.

ZDC021I EXIT ROUTINE MODULE NAME=@@ STATUS=@@

EXPLANATION:	The module name for each exit routine is named in this message.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC023E ZDCQUERT VSAM @@ ERROR R15 RETURN CODE @@

EXPLANATION:	This message indicates that a non-zero return code has resulted from a VSAM zDC queue request. The name of the request is contained in the message.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	Examine the following messages, which contain detailed information about the nature of the error. Keep this information for your systems programming staff or for support.

ZDC024E ZDCQUERT VSAM RPL FEEDBACK WORD @@

EXPLANATION:	This message contains the value of the VSAM RPL feedback word following a VSAM PUT to the queue data set.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	Examine the following messages, which contain detailed information about the nature of the error. Keep this information for your systems programming staff or for support.

ZDC025E ZDCQUERT VSAM ACB ERROR FLAG @@

EXPLANATION:	This message contains the value of the VSAM ACB error flag following a VSAM OPEN or CLOSE of the queue data set.
SYSTEM ACTION:	Processing continues.

USER RESPONSE:	Examine the following messages, which contain detailed information about the nature of the error. Keep this information for your systems programming staff or for support.

ZDC026I STOP COMMAND ACCEPTED

EXPLANATION:	This message confirms that the STOP command entered by an operator has been accepted by zDC and end of job processing has commenced.
SYSTEM ACTION:	Execution terminates.
USER RESPONSE:	None.

ZDC027I MODIFY COMMAND RECEIVED

EXPLANATION:	This message confirms that the MODIFY command entered by an operator has been received by zDC and processing of it has commenced. If there are any errors encountered during parsing or execution of the command they will be indicated later.
SYSTEM ACTION:	Execution continues.
USER RESPONSE:	None.

ZDC028I @@

EXPLANATION:	This message contains a copy of the entire command entered by the operator via the console. Only the first 120 bytes of the command are displayed.
SYSTEM ACTION:	Execution continues.
USER RESPONSE:	None.

ZDC033E INVALID COMMAND. PLEASE REENTER

EXPLANATION:	The command you just entered via the operator console was invalid and will not be processed.

SYSTEM ACTION:	The command is ignored and processing continues.
USER RESPONSE:	Reenter a valid command if possible. If the command was valid, refer this problem to support.

ZDC035I @@ ARMED FOR AUTOMATIC RESTART

EXPLANATION:	This message indicates that you have selected ARM processing and this is the first time the element named in this message has been armed. That is, this execution is NOT due to an automatic restart.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC036I @@ RESTARTED DUE TO PREVIOUS ABEND. RSNCD=@@

EXPLANATION:	This message indicates that this zDC has been automatically restarted by the system. The reason code indicates whether the JCL or START command has been changed (RSNCD=108) or has not been changed (RSNCD=104)
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC037E @@ IXCARM REGISTER FAILURE. RETCD=@@. RSNCD=@@

EXPLANATION:	An attempt to issue IXCARM macro has resulted in a return code greater than 4. This indicates that the request did not complete. The actual return and reason codes are shown in this message.
SYSTEM ACTION:	Processing continues but without ARM support.
USER RESPONSE:	See the “z/OS MVS Sysplex Services Reference” manual for a complete list of the possible return and reason codes. Correct any environmental errors or report this problem to support.

ZDC038E @@ IXCARM READY FAILURE. RETCD=@@. RSNCD=@@

EXPLANATION:	An attempt to issue IXCARM macro has resulted in a return code greater than 4. This indicates that the request did not complete. The actual return and reason codes are shown in this message.
SYSTEM ACTION:	Processing continues but without ARM support.
USER RESPONSE:	See the “z/OS MVS Sysplex Services Reference” manual for a complete list of the possible return and reason codes. Correct any environmental errors or report this problem to support.

ZDC039I @@ AUTOMATIC RESTART NOT SUPORTED ON THIS SYSTEM

EXPLANATION:	An attempt to issue IXCARM macro has resulted in a return code indicating that ARM support is not available on this system, or it may be available but stopped.
SYSTEM ACTION:	Processing continues but without ARM support.
USER RESPONSE:	See the “z/OS MVS Sysplex Services Reference” manual for a complete list of the possible return and reason codes. You may need to issue the operator command SETXCF START,POLICY,TYPE=ARM in order to activate ARM processing.

ZDC040I @@ DEREGISTERED FOR AUTOMATIC RESTART

EXPLANATION:	This message indicates that an IXCARM has been issued in order for the server to go to a normal end of job.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC041E LICENSE MANAGER FAILURE RETCD/RSNCD = @@/@@

EXPLANATION:	This message indicates that the call to the license management system (LMS) has resulted in return and reason codes shown in the message text. These codes indicate that LMS was unable to process the request to obtain permission to executed. These codes do NOT indicate simply that the zDC is not licensed on this LPAR, but they do indicate that a failure within LMS itself has occurred.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Report this message to support.

ZDC042E DYNATRACE LICENSE MANAGER RETCD/RSNCD = @@/@@

EXPLANATION:	This message indicates that the call to the license management system (LMS) has resulted in return and reason codes shown in the message text. This message is written in place of the standard license manager error messages because storage in which to display these messages was not available. This message is NOT an indication that there is a problem with the license manager. This is a storage shortage issue only.
SYSTEM ACTION:	If the RETCD is 04 execution of zDC continues. If the RETCD is 08 execution ends.
USER RESPONSE:	Report this message to support.

ZDC043I @@

EXPLANATION:	This message comes from the License Management System (LMS) and is one of a list of messages indicating the license status of zDC on this LPAR. These messages are explained completely in the License Management System Messages and Codes manual.
SYSTEM ACTION:	If the messages indicate that zDC is not licensed to run, then execution ends.
USER RESPONSE:	Report this message to support.

ZDC044I Dynatrace LLC. License Management Verification

EXPLANATION:	Informational message that License checking is starting.
SYSTEM ACTION:	If the messages indicate that zDC is not licensed to run, then execution ends.
USER RESPONSE:	Report this message to support.

ZDC045I Monitoring will be enabled for PurePath agents with licenses

EXPLANATION:	Informational message that License checking is starting.
SYSTEM ACTION:	If the messages indicate that zDC is not licensed to run, then execution ends.

USER RESPONSE: Report this message to support.

ZDC046I Each type of agent supported by the ZDC will be checked

EXPLANATION: Informational message that License checking is starting.

SYSTEM ACTION: If the messages indicate that zDC is not licensed to run, then execution ends.

USER RESPONSE: Report this message to support.

ZDC047I Valid license for @@ agent found

EXPLANATION: Informational message that License checking is starting.

SYSTEM ACTION: If the messages indicate that zDC is not licensed to run, then execution ends.

USER RESPONSE: Report this message to support.

ZDC048I THE FOLLOWING MODULES ARE LOADED INTO ECSA

EXPLANATION: This message is printed in the SYSPRINT data set and indicates that all modules named next via ZDC009I messages were loaded into ECSA.

SYSTEM ACTION: Processing continues.

USER RESPONSE: None.

ZDC049I THE FOLLOWING MODULES ARE LOADED INTO PRIVATE STORAGE

EXPLANATION: This message is printed in the SYSPRINT data set and indicates that all modules named next via ZDC009I messages were loaded into the private area of zDC.

SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC052I zDC IS RUNNING ON @@ RELEASE @@

EXPLANATION:	This informational message indicates the name and release level on which this zDC is running.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC053I LPAR NAME @@ CVTSNAME @@

EXPLANATION:	This informational message indicates the name of the LPAR on which this zDC is running and the name of the operating system on which this zDC is running.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC054E zDC ACTIVE SUBSYSTEM @@ ALREADY MARKED AS DEFAULT

EXPLANATION:	This message indicates that during the process of establishing a new zDC subsystem on this LPAR, a previous subsystem named in the message is already active and is already marked as the default subsystem.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Only one zDC subsystem at a time may be marked as the default. Change the parameter specification of this zDC to indicate DEFAULT(NO) and reexecute zDC. Or you can stop the zDC job with the named subsystem, which will allow another subsystem to specify it as the default one. Note that if the zDC instance with the subsystem named is unable to be cancelled, or if it has been cancelled and its subsystem remains active, then you can specify DEFAULT(YES,FORCE). The existing subsystem will be marked as not the default and the new one will be marked as the default.

ZDC059I zDC NOW EXECUTING UNDER USERID @@

EXPLANATION:	This message is written to SYSPRINT indicating that the user ID you specified in the TCPIP_USERID is now in control of the main task of zDC.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC060I LOG @@ @@ SUCCESSFULLY

EXPLANATION:	This message is written to SYSPRINT indicating that the action you requested in a “LOG” command has been successfully executed.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC061E INSUFFICIENT ACCESS TO LOG DATA SET. SEE JOBLOG

EXPLANATION:	This message is written to SYSPRINT indicating that the USERID under which zDC is running does not have the necessary security access to the log data set. This access must be ACCESS(ALTER) in order that the log data set can be created and/or extended and be available.
SYSTEM ACTION:	Processing continues but without the log.
USER RESPONSE:	Either grant the user ID under which zDC is running ACCESS(ALTER) to the log data set, or supply a user ID that has this access level. Note that the log data set must be defined as a “generic” entity in order to pass the RACROUTE used in zDC.

ZDC062E LOG TASK INITIALIZATION FAILURE RC=@@/@@

EXPLANATION:	This message is written to SYSPRINT indicating that the log subtask has failed to initialize.
SYSTEM ACTION:	zDC processing continues but without the log.
USER RESPONSE:	Scan the system console and the JES joblog for messages relating to this problem. Look carefully for any system access (RACF, ACF/2, TOPSECRET) messages relating to the log data set.

ZDC064E QUEUE TASK INITIALIZATION FAILURE

EXPLANATION:	This message is written to SYSPRINT indicating that the queue subtask has failed to initialize.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Report this message to support.

ZDC065I zDC INITIALIZATION SUCCESSFUL. RETCODE=00/00 @@

EXPLANATION:	This message is written to SYSPRINT indicating that initialization of zDC has been successful.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC066E QUEUE TASK FAILURE RETCODE=@@/@@

EXPLANATION:	This message is written to SYSPRINT indicating that the queue subtask has failed. Return and reason codes are contained in this message.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	If `"QUEUE"=DtAgt`, in the message, review the following: • The zDC task tries to connect to the zRemoteAgent. One of the first things it does is bootstrap the zLocalAgent from the server via the zRemoteAgent. If the bootstrapping fails, meaning that the zLocalAgent can't be loaded, the zDC task will try 3 times then issue a user abend. The fallback is that if this happens, the zDC will try to load the most recently bootstrapped zLocalAgent. If it is the first time the zDC has been started in a new work area on the OMVS filesystem, and there is no previously downloaded zLocalAgent, then this user abend will be issued. • Check that the zRemoteAgent parameter in the zDC //SYSIN is correct. This needs to point to the zRemoteAgent instance, which needs to be up and running, and connected to a server, so that the zLocalAgent bootstrapping can happen. • Review the zDC spool output for bootstrap messages. If `"QUEUE"=QUEUE`, in the message, report this message to support.

ZDC067E LOG TASK FAILURE RETCODE=@@/@@

EXPLANATION:	This message is written to SYSPRINT indicating that the log subtask has failed. Return and reason codes are contained in this message.
SYSTEM ACTION:	Processing continues but without the log.
USER RESPONSE:	Scan the system console and the JES joblog for messages relating to this problem. Look carefully for any system access (RACF, ACF/2, TOPSECRET) messages relating to the log data set.

ZDC069W zDC QUEUE IS AT @@% OF CAPACITY

EXPLANATION:	This message indicates that the queue is filling and that you should be aware that if it reaches 100% new internal events will not be accepted. This message will first be displayed when the queue reaches 75% and it will be displayed again each at each 5% interval. When queue drops below 75% this message will no longer be displayed. Note that this message appears on the system console as well as in the SYSPRINT data set.
SYSTEM ACTION:	Execution continues.
USER RESPONSE:	Report this message to support.

ZDC071W UNABLE TO CONSTRUCT SDUMPX SYMREC RETCODE=@@/@@

EXPLANATION:	This message indicates that the ZDCSDUMP module could not create a symptom record to be included in the SDUMP that it was about to write.
SYSTEM ACTION:	The dump is created but without a symptom record.
USER RESPONSE:	None.

ZDC072W WLM ENCLAVE CREATION FAILED. @@ @@/@@

EXPLANATION:	This message indicates that the process to create an WLM enclave has not been successful. The function name is shown in the message along with the return and reason codes from execution of this function. This is a warning message only because execution of zDC continues. See below.
SYSTEM ACTION:	zDC execution continues but all processing is performed in task mode and the enclave SRB is not dispatched. Hence there is no execution on any zIIP.

USER RESPONSE: Report this message to your systems programmer who may need to call support.

ZDC073I WLM ENCLAVE @@ CREATED SUCCESSFULLY

EXPLANATION: This message indicates that the process to create an WLM enclave has been successful and zDC is a member of the enclave named in the message.

SYSTEM ACTION: Processing continues.

USER RESPONSE: None.

ZDC074E WLM ENCLAVE DELETION FAILED. @@ @@/@@

EXPLANATION: This message indicates that the process to delete an WLM enclave has not been successful. The function name is shown in the message along with the return and reason codes from execution of this function.

SYSTEM ACTION: zDC execution ends.

USER RESPONSE: Report this message to your systems programmer who may need to call support.

ZDC076E WLM ENCLAVE SRB EXECUTION FAILED. RETCODE=@@

EXPLANATION: This message indicates that the enclave SRB has failed to execute correctly. If the RETCODE in this message is 1C, then message ZDC077 will be written next with the details concerning the failure.

SYSTEM ACTION: zDC execution ends.

USER RESPONSE: Report this message to your systems programmer who may need to call support.

ZDC077E WLM ENCLAVE SRB REASON CODES=@@ @@ @@

EXPLANATION: The three codes in this message correspond to the IEAMSCHD SYNCHCOMPADDR, SYNCHCODEADDR and SYNCHRSNADDR values.

SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Report this message to your systems programmer who may need to call support.

ZDC083E SUBSYSTEM @@ ALREADY EXISTS BUT IS NOT A zDC SUBSYSTEM

EXPLANATION:	This message indicates that the 4-character subsystem ID you specified in the //SYSIN DD parameter data set already exists on your system but is used by some other program that is not AppMon's zDC.
SYSTEM ACTION:	zDC terminates.
USER RESPONSE:	Ensure that the subsystem ID you have specified is unique on your system then restart zDC.

ZDC100I SYSIN PARAMETER LISTING

EXPLANATION:	This message indicates that the //SYSIN parameters specified by the customer are listed next.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC101I END OF SYSIN PARAMETER LISTING

EXPLANATION:	This message indicates that the //SYSIN parameters specified by the customer have all been listed above.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC102E SYSTEM VARIABLE SUBSTITUTION FAILED. @@

EXPLANATION:	During the parsing of the //SYSIN DD data, the z/OS system variable substitution routine is called for each parameter. A non-zero return code resulted from one of these calls.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the parameter in error and reexecute zDC.

ZDC103E UNABLE TO OPEN //SYSIN DATA SET

EXPLANATION:	The attempt to open the SYSIN data set failed. There is no return code associated with this error but a message may be displayed in the joblog that better explains the error.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Ensure that the //SYSIN DD statement exists in the JCL used to invoke zDC and ensure that it points to a data set with DCB characteristics of LRECL=80,

ZDC105E ILLOGICAL SEQUENCE OF COMMENT CONTROL CHARACTERS

EXPLANATION:	Comment control characters are "/*" to begin a comment and "/" to end a comment. These must be paired; one of each. All statements between this pair will be treated as a comment. This message indicates that the pairing of these control sequences is not correct.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the comment pairing error and reexecute zDC.

ZDC106E UNKNOWN SYSIN KEYWORD @@

EXPLANATION:	The SYSIN keyword shown in the message text is not one of the valid keywords.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the KEYWORD in the SYSIN data set and reexecute zDC.

ZDC107E SUBSYSTEM_ID PARAMETER MISSING

EXPLANATION:	The SUBSYSTEM_ID parameter names a unique 4-character name for this zDC instance. This name must not be duplicated on this LPAR, and must NOT be defined in the SYS1.PARMLIB IEFSSNxx member.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Add the SUBSYSTEM_ID parameter to the //SYSIN data set and reexecute zDC. This 4-character name must be unique across all subsystems running on this LPAR but may be the same on different LPARs. That is, you may name all your zDC subsystems with the same name on each LPAR; But on any one LPAR this name must be unique. And DO NOT DEFINE THIS NAME IN THE IEFSSNxx SYS1.PARMLIB member.

ZDC110E NAME PARAMETER MISSING

EXPLANATION:	The NAME parameter is required and must specify a name for this zDC that is meaningful to personnel in your enterprise who may see messages from this zDC. The value you specify in the NAME parameter is not used for any purpose except to uniquely identify this instance of zDC.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Add the NAME parameter and reexecute zDC.

ZDC116E AUTOMATIC RESTART MANAGEMENT NAME MISSING

EXPLANATION:	You have specified ARM(YES) but have not specified the name to be used for Automatic Restart Management. This name must be from 1 to 16 bytes long. zDC will register with the z/OS automatic restart management and will then be eligible for this facility. This name can consist of uppercase letters, numbers, and the following special characters: underscore, dollar sign pound sign or At sign. The first character cannot be numeric.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Please see “z/OS V1Rn.0 MVS Setting Up a Sysplex” for a complete discussion of how an installation invokes this facility. When all of the steps outlined in this manual have been followed, then add the ARM_NAME parameter and reexecute zDC. In the meantime, specify ARM(NO) or omit the parameter.

ZDC121E SUBSYSTEM_ID PARAMETER INVALID

EXPLANATION:	The SUBSYSTEM_ID parameter has been specified, but is not valid. The value for this parameter must be 4 bytes long consisting only of uppercase letters and numbers and the international character $ # or @. The first character must be a letter.
SYSTEM ACTION:	zDC execution ends.

USER RESPONSE:	Correct the SUBSYSTEM_ID parameter and reexecute zDC.

ZDC122E DISPLAY_NAME PARAMETER INVALID

EXPLANATION:	The DISPLAY_NAME parameter has been specified, but is not valid. The value for this parameter must be from 1 to 40 bytes long. Any displayable characters are valid.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the DISPLAY_NAME parameter and reexecute zDC.

ZDC123E DEFAULT PARAMETER INVALID

EXPLANATION:	The DEFAULT parameter has been specified, but is not valid. Only two values are allowed: YES or NO. Only one zDC instance in a given LPAR may specify that it is the default; all others must specify NO.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the DEFAULT parameter and reexecute zDC.

ZDC125E EVENT_WAIT PARAMETER INVALID

EXPLANATION:	The EVENT_WAIT parameter can only specify “YES” or “NO”. If this parameter is omitted “YES” is assumed. If EVENT_WAIT is specified as “YES” or allowed to default to “YES”,it indicates that when events are processed, programs will be kept from continuing their execution until the event has been successfully processed by the zDC address space. In most cases this is the desirable option. If it is determined that too much time is lost while zDC is processing the event, then specify “NO” to this parameter. The liability is that events may be lost if zDC terminates abnormally.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the EVENT_WAIT parameter and reexecute zDC.

ZDC126E TCPIP_USERID PARAMETER INVALID

EXPLANATION:	The TCPIP_USERID parameter must specify a 1- to 8- character valid user ID. You may omit this parameter in which case the user ID under which zDC was submitted is used.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Either correct the user ID parameter or omit it entirely. Reexecute zDC.

ZDC127E LOG_INITIAL PARAMETER INVALID

EXPLANATION:	The LOG_INITIAL parameter must specify either (OPEN) (CLOSED) or (). Any other values are invalid. If you omit this parameter or specify () the default is LOG_INITIAL(CLOSED). Specify all the other parameters for logging that are applicable to your SMS environment, and also specify LOG_INITIAL(CLOSED). You can dynamically start logging using the specified parameters if the need arises. With the log in a CLOSED state no system resources are consumed.
SYSTEM ACTION:	zDC execution ends
USER RESPONSE:	Correct the LOG_INITIAL parameter and reexecute zDC.

ZDC128E LOG_DSNAME PARAMETER IS INVALID

EXPLANATION:	The LOG_DSNAME parameter must specify a 1- to 35- character data set name for the optional logging. Be sure you haven”t specified more than 35 characters because IDCAMS appends a .DATA and a .INDEX to this name when the cluster is defined.
SYSTEM ACTION:	zDC execution ends
USER RESPONSE:	Correct the LOG_DSNAME and reexecute zDC.

ZDC130E LOG_VOLSER PARAMETER IS INVALID

EXPLANATION:	The LOG_VOLSER parameter must specify a 6-character volume serial number for the volume on which the log is to be defined. You must specify this parameter only if your SMS environment dictates it, else you may omit this parameter and take your system defaults.
SYSTEM ACTION:	zDC execution ends
USER RESPONSE:	Correct the LOG_VOLSER and reexecute zDC.

ZDC131E LOG_TRKS PARAMETER IS INVALID

EXPLANATION:	The LOG_TRKS parameter must specify a 1- to 8-digit number representing the number of tracks you want defined in the primary and secondary extents of the log data set. The maximum value for this parameter is 65535 and the minimum value is 1.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the LOG_TRKS and reexecute zDC.

ZDC132E LOG_CYLS PARAMETER IS INVALID

EXPLANATION:	The LOG_CYLS parameter must specify a 1- to 8-digit number representing the number of cylinders you want defined in the primary and secondary extents of the log data set. The maximum value for this parameter is 65535 and the minimum value is 1.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the LOG_CYLS and reexecute zDC.

ZDC133E LOG_MGMTCLASS PARAMETER IS INVALID

EXPLANATION:	The LOG_MGMTCLASS parameter must specify a 1- to 8- character name for the management class SMS parameter that is defined in your installation for the log data set.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the LOG_MGMTCLASS parameter and reexecute zDC.

ZDC134E LOG_DATACLASS PARAMETER IS INVALID

EXPLANATION:	The LOG_DATACLASS parameter must specify a 1- to 8- character name for the data class SMS parameter that is defined in your installation for the log data set.

SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the LOG_DATACLASS parameter and reexecute zDC.

ZDC135E LOG_STORCLASS PARAMETER IS INVALID

EXPLANATION:	The LOG_STORCLASS parameter must specify a 1- to 8- character name for the storage class SMS parameter that is defined in your installation for the log data set.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the LOG_STORCLASS parameter and reexecute zDC.

ZDC155E ARM PARAMETER INVALID

EXPLANATION:	The ARM parameter must specify either ARM(YES), if you want Automatic Restart Management to restart this zDC in case of abnormal termination, or ARM(NO), if you do not want the zDC restarted. Initially you should specify ARM(NO) until you have assured yourself that the zDC is working correctly under normal circumstances. When this is the case you can request ARM support.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the ARM parameter and reexecute zDC

ZDC156E ARM_ELEMENT_NAME PARAMETER INVALID

EXPLANATION:	The ARM_ELEMENT_NAME parameter specifies a 1- to 16- character name to be used by z/OS in identifying this program to the Automatic Restart Management system. Only letters and numbers and the special characters “_”, “$”, “#” and “@” are valid. In addition, the first byte must not be numeric.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the ARM_ELEMENT_NAME parameter and reexecute zDC.

ZDC185E PROTECT PARAMETER INVALID

EXPLANATION:	The PROTECT parameter must specify PROTECT(YES) if pages in extended common storage are to be protected from inadvertent alteration by other programs running in the LPAR or PROTECT(NO) if this protection is not to occur. If this parameter is omitted then no protection will exist. Any program running in key 0 can alter this storage.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the PROTECT parameter and reexecute zDC.

ZDC187E DEFAULT PARAMETER MISSING

EXPLANATION:	The DEFAULT parameter is required as either DEFAULT(YES) or DEFAULT(NO). Only one zDC subsystem in the LPAR can specify DEFAULT(YES). All others must specify DEFAULT(NO).
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Add the DEFAULT parameter and reexecute zDC.

ZDC189E OPERATING_SYSTEM PARAMETER INVALID

EXPLANATION:	The OPERATING_SYSTEM parameter should never be specified at a customer parameter. Its use is only for emulating releases of z/OS earlier than the current one.
SYSTEM ACTION:	zDC ends.
USER RESPONSE:	Remove this parameter from the SYSIN data set.

ZDC194E zIIP_ENABLE PARAMETER INVALID

EXPLANATION:	The zIIP_ENABLE parameter can specify YES or NO. If zIIP_ENABLE(YES) is specified, then the SRB under which all XML creation and TCP/IP SOCKET(SEND) routines run will be zIIP enabled. If you specify NO to zIIP_ENABLE this SRB will not run on your zIIP.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the zIIP_ENABLE parameter and resubmit zDC.

ZDC196E DTMSG_QSIZE PARAMETER IS INVALID

EXPLANATION:	The DTMSG_QSIZE parameter specifies a Data Space size in Kiloby The default is 1024 (1024K). This Data Space used to queue messages from the CICS Agent to the z/OS Unix agent. The minimum value is 8 and the maximum is 262144K (256Megabytes).
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Specify a value in the range given.

ZDC197E DTMSG_SMOSIZE PARAMETER IS INVALID

EXPLANATION:	The DTMSG_SMOSIZE parameter specifies a 64-Bit SMO size in Mb”s The default is 1 Mb. This 64-Bit SMO is used to queue messages from the CICS Agent to the z/OS Unix agent. The minimum value is 1 and the maximum is 256 Mb. (256Megabytes).
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Choose a correct value for SMOSIZE and then resubmit zDC.

ZDC198E DT_UMASK PARAMETER IS INVALID

EXPLANATION:	The DT_UMASK parameter specifies user file-creation mask. The default is 0022(octal) and corresponds to rwxr-xr-x permission for new files. Error is generated if the parameter is over 3 characters, or contain non-octal characters (o:7).
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Choose a correct value for DT_UMASK and then resubmit zDC.

ZDC199E DTMSG_TBCSIZE PARAMETER IS INVALID

EXPLANATION:	The DTMSG_TBCSIZE parameter specifies a 64-Bit Transaction buffer total size in Mb”s and either a 2K or 4K buffer size. The default is (1,2) - 1Mb and 2K buffer. This 64-Bit SMO is used to buffer messages related to a transaction before being sent to the z/OS Unix agent. The minimum value is 1 and the maximum is 256 Mb (256Megabytes).
SYSTEM ACTION:	zDC execution ends.

USER RESPONSE:	Choose a correct value for TBCSIZE and then resubmit zDC.

ZDC200I DTMSG_QSIZE PARAMETER IS DEPRICATED

EXPLANATION:	The DTMSG_QSIZE parameter specified DataSpace size in Kilobytes It is replace by DTMSG_SMOSIZE(Megabytes) The parameter has been converted to Megabytes and used to set the SMO size.
SYSTEM ACTION:	zDC execution continues.
USER RESPONSE:	Use DTMSG_SMOSIZE in the future.

ZDC201I ZDCLOGBUFFERSIZE PARAMETER IS INVALID

EXPLANATION:	ZDCLOGBUFFERSIZE specifies a log buffer in kilobytes. It is a SMO memory are areas to buffer the ZDC .log file. The ZDC .log file in the same directory as the ZLocal agent .log files. The default is (64) - 64K. The minimum is 16K, the maximum is 10240K =10M.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct value for ZDCLOGBUFFERSIZE and resubmit.

ZDC202I ZDCMAXLOGFILESIZE PARAMETER IS INVALID

EXPLANATION:	ZDCMAXLOGFILESIZE specifies the log file size limit. It is used to rotate to a new log file when exceeded. The ZDClogfile pattern: ..*ZDCID_PID*.#.log. e.g. dt_DTV1ZDC4_MEPC_33620088.1.log The default is (10240) =10M The minimum is 10K, the maximum is 1048576K =1G.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct value for ZDCMAXLOGFILESIZE and resubmit.

ZDC210I zDC NO LONGER LINKED TO TCP/IP

EXPLANATION:	This informational message indicates that the TERMAPI macro was successfully executed to terminate TCP/IP communications.

SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC211E DTMSG_TRANBUFSIZE PARAMETER IS INVALID

EXPLANATION:	The DTMSG_TRANBUFSIZE parameter specifies a Transaction buffer total size in thousands of buffers and either a 2K or 4K buffer size. The default is (1,2) - 1 thousand 2K buffers. This 64-Bit SMO is used to buffer messages related to a transaction before being sent to the z/OS Unix agent. The minimum value is (1,2), maximum is (126,4) or (248,2) (512Megabytes).
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Choose a correct value for DTMSG_TRANBUFSIZE and resubmit.

ZDC212E zremoteagent= IS REQUIRED IN DTAGTCMD

EXPLANATION:	A remote agent is required. The remote agent isn’t specified in the DTAGTCMD parameter.
SYSTEM ACTION:	zDC is terminated
USER RESPONSE:	Add zremoteagent= to DTAGTCMD resubmit zDC.

ZDC215I TERMINATION STARTED FOR zDC @@

EXPLANATION:	This informational message indicates that termination processing has started for this zDC.
SYSTEM ACTION:	Termination processing continues.
USER RESPONSE:	None.

ZDC217I LONG TERM STORAGE STATISTICS FOLLOW

EXPLANATION:	This informational message is followed by messages that show the number of bytes of storage allocated in each of the designated areas. These numbers may only be approximate as dynamic storage areas are not included in them.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC218I PAGEABLE PRIVATE STORAGE @@

EXPLANATION:	The number of bytes of private zDC storage that is allocated long term is shown in this message. This number may only be approximate as dynamic storage areas are not included. Storage allocated both below and above the 16M line are included in this number.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC219I PAGEABLE COMMON STORAGE - CSA: @@ ECSA: @@

EXPLANATION:	The number of bytes of subpool 241 (pageable common) storage that has been allocated for the duration of zDC execution.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC220I FIXED COMMON STORAGE - SQA: @@ ESQA: @@

EXPLANATION:	The number of bytes of subpool 245 (fixed common) storage that has been allocated for the duration of zDC execution.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC221I NUMBER SP 241 PAGES PROTECTED: @@

EXPLANATION:	The number of pages in subpool 241 that have had the PGSER PROTECT macro issued against them. This message displays only if you have specified PROTECT(YES) in the SYSIN parameter data set.
SYSTEM ACTION:	Processing continues.
USER RESPONSE:	None.

ZDC302E Message @@ - BAD REC ADDRESS PASSED

EXPLANATION:	Validation routines invoked to ensure that each message to the z/OS Unix agent is valid has detected an error in the data.
SYSTEM ACTION:	If logging is active, the event is logged. This message is written to the user's JOBLOG and the event is discarded.
USER RESPONSE:	Report this message to support.

ZDC950E DT PARAMETER INVALID

EXPLANATION:	The parameter DTPIPEPATH, DTCHDIR, DTAGTCMD, or DTLOG is invalid. See comments in the supplied sample config member for a description of these parameters.
SYSTEM ACTION:	zDC execution ends.
USER RESPONSE:	Correct the parameter and reexecute zDC.

ZDC951I CICS property update too frequent, under 30sec., discarded

EXPLANATION:	There is a limit on how frequently the internal tables that contain CICS properties can be updated. The current limit is 30 seconds.
SYSTEM ACTION:	Informational, processing continues.

USER RESPONSE:	Re-apply the update later.

ZDC952W zDC z/OS Unix agent Data Space Queue is at @@% of capacity

EXPLANATION:	This message indicates that the queue is filling and you should be aware that if it reaches 100% that new messages will not be sent to the z/OS Unix agent, possibly causing PurePath corruption. This message will first be displayed when the queue reaches 70% and it will be displayed again at each 5% increments. When the utilization drops below 70% this message will no longer be displayed. Note that this message appears on the system console as well as in the SYSPRINT data set.
SYSTEM ACTION:	Execution continues.
USER RESPONSE:	Monitor these messages carefully. If you need to expand the Data Space, adjust the DTMSG_QSIZE parameter. DTMSG_QSIZE defaults to 1024K, the minimum value is 8K, and the maximum is 262144K.

ZDC953I ZIIP_ENABLE requires zremoteagent=, ZIIP_ENABLE(NO

EXPLANATION:	ZIIP enablement requires remote agent option.
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	ZIIP enablement requires both the zremoteagent= in the DTABGCMT command line and ZIIP_ENABLE(YES) in the ZDC SYSIN parameters.

ZDC974E Unexpected failure checking agent executable,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. Check that the agent file exists and is executable. See DTAGTCMD() parameter.
SYSTEM ACTION:	Processing terminates.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC975E z/OS Unix agent attach failed,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. Check that the DTAGTCMD parameter references an accessible and executable agent.
SYSTEM ACTION:	Processing terminates.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC976E Unexpected failure during change directory,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. Check that the zDC has read+write access to the DTCHDIR directory.
SYSTEM ACTION:	Processing terminates.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC977E Unexpected failure during z/OS Unix API,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. Check that the user ID that is running the zDC is defined to the security product. The userid is either the normally assigned ID, or overriden by the TCPIP_USERID parameter.
SYSTEM ACTION:	Processing terminates.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC978E Failed ZDC.FML Name/Token operation,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. The MVS Name/Token is used for internal processing. The operation failed.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC979E Waiting for Que Data Space failed,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. The Data Space used to pass messages to the z/OS Unix agent has not initialized.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC980E Waiting for OMVS failed,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. The OMVS kernel is not ready.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check why the OMVS address space is not operational. Restart OMVS or delay zDC startup until OMVS is ready.

ZDC981E DtmD.FlagBad internal error,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. A critical Data Space process has failed.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC982E DtmD.LenBad internal error,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. A critical Data Space process has failed.
SYSTEM ACTION:	Processing terminates with a dump.

USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC983E Timer routine critical error,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. A critical timer wait function has failed.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC984E Critical error in Wait routine,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. A Unix wait to check z/OS Unix agent status has failed.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC985E Poll for data failed,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. A Unix poll for z/OS Unix agent messages has failed.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC986E Unnamed pipe required,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. A Unix unnamed pipe is needed for z/OS Unix agent communication.

SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC987E Internal error unknown ECB,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. A critial internal call function has failed.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC988E Unexpected PC return,Abort

EXPLANATION:	This message is written to the SYSPRINT and Oper. A critial internal call function has failed.
SYSTEM ACTION:	Processing terminates with a dump.
USER RESPONSE:	Check joblog for additional error messages, correct them if possible. If still unresolved, refer this problem to support.

ZDC989E *Dbg*Developer build test message

EXPLANATION:	This message is written to the SYSPRINT and Oper. It is intended for internal diagnostic for dynaTrace loglevel:finer(1).
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC990I Internal DynaDiag

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal diagnostic for loglevel:finest(0).
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC991I Internal DynaDiag

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon diagnostic for dynaTrace loglevel:finer(1).
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC992I Internal DynaDiag

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon diagnostic for dynaTrace loglevel:fine(2).
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC993I Internal DynaDiag

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon diagnostic for dynaTrace loglevel:config(3).
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC994I Internal DynaDiag

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon diagnostic for dynaTrace loglevel:info(4)
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC995W Internal DynaDiag

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon diagnostic for dynaTrace loglevel:warn(5).
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC996E Internal DynaDiag

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon diagnostic for dynaTrace loglevel:severe(6).
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC997I Internal DynaDiag

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon diagnostic for dynaTrace loglevel:debug(7).
SYSTEM ACTION:	Informational, processing continues.

USER RESPONSE:	Informational, processing continues.

ZDC998I @@ @@ @@ @@

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon. Support may ask that system parameters be changed to cause these messages.
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

ZDC999I Internal DynaDiag: @@ @@ @@ @@

EXPLANATION:	This message is written to the SYSPRINT. It is intended for internal AppMon diagnostic and shouldn't appear under normal operation. Support may ask that system parameters be changed to cause these messages.
SYSTEM ACTION:	Informational, processing continues.
USER RESPONSE:	Informational, processing continues.

DTAX system messages

ZDTP001S Persistent Storage allocation failed.

Explanation	The CICS PLT program was unable to allocate a required persistent storage area.
System Action	The PLT program terminates and the CICS Agent is not enabled.
User Response	Determine why the CICS region is unable to provide storage for the DTAX transaction. If the problem persists contact support.

ZDTP002S Message Block allocation failed.

Explanation	The PLT program was unable to allocate a required Message block storage area.
System Action	The PLT program terminates and the CICS Agent is disabled.
User Response	Determine why the CICS region is unable to provide storage for the PLT program.

ZDTP003S ZDTSOAPH Load failed.

Explanation	CICS could not load ZDTSOAPH.
System Action	The PLT program terminates and the CICS Agent is disabled.
User Response	Determine why the CICS region is unable to provide storage for the DTAX transaction. If the problem persists contact support.

ZDTP004S ZDTDCCAL final block send failed. rc:<xx>.

Explanation	Message transfer from CICS Agent to zDC failed.
System Action	CICS Agent operation continues but with missing nodes in the PurePath.
User Response	Contact support.

ZDTP005S DTAX 5 minutes cycle START failed.

Explanation	The EXEC CICS START command used to initiate the re-execution of transaction DTAX in 5 minutes time failed.
System Action	N/A
User Response	Examine the more detailed message in the appropriate CICS joblog to look at the precise EIBRESP code that indicates why this EXEC CICS START command failed. If the problem persists, contact support.

ZDTP006S Inquire MQCONN <QMGR Name> returned with resp:<xx> resp2:<xx>.

Explanation	EXEC CICS INQUIRE for MQ Connection information failed.
System Action	Missing MQ Queue manager name in CICS attachment details.
User Response	With the help of `resp` and `resp2` codes, determine why INQUIRE MQCONN failed. If the problem persists, contact support.

ZDTP008S GETMAIN PLT Workarea failed.

Explanation	Storage allocation for PLT workarea failed.
System Action	The PLT program terminates and the CICS Agent is disabled.
User Response	Determine why the CICS region is unable to provide storage for the PLT program.

ZDTP009S PLT Name/Token storage not found rc: <xx>.

Explanation	Persistent storage for PLT program is not found.
System Action	The PLT program terminates and the CICS Agent is disabled.
User Response	Contact support.

ZDTP010S Turn ON RMI Hook failed with rc: <xx>.

Explanation	RMI hook initialization failed.
System Action	CICS Agent not initialized successfully.

User Response	Contact support.

ZDTP011S Turn OFF RMI Hook failed with rc: <xx>.

Explanation	Disabling RMI hook failed.
System Action	CICS Agent not disabled successfully.
User Response	Contact support.

ZDTP012S GWA Storage unavailable.

Explanation	Global work area storage for PLT program is unavailable.
System Action	Hooks are not initialized successfully.
User Response	Contact support.

ZDTP013S SOAP hook <nn> failed with rc: <xx>.

Explanation	SOAP Hook enable/disable failed.
System Action	Hook operation failed.
User Response	Contact support.

ZDTP014S LE hook <nn> failed with rc: <xx>.

Explanation	LE Hook enable/disable failed.

System Action	Hook operation failed.
User Response	Contact support.

ZDTP015S MRO hook <nn> failed with rc: <xx>.

Explanation	MRO Hook enable/disable failed.
System Action	Hook operation failed.
User Response	Contact support.

ZDTP016S ZDTHKIDR failed with rc: <xx>.

Explanation	Hook operation failed.
System Action	CICS Agent not initialized successfully.
User Response	Contact support.

ZDTP017S zDC INIT Failed.

Explanation	The PLT program is unable to connect to the collection system.
System Action	Execution continues. The PLT program tries to re-connect with the collection system during the next 5 minutes window.
User Response	Check if the Server, Collector and zRemote Agent are up and running. If not, start the service that is down. If the problem persists, contact support.

ZDTP018S CONF block allocation failed.

Explanation	Storage allocation failed for config block.
System Action	CICS Agent not initialized successfully.
User Response	Determine why the CICS region is unable to provide storage for the DTAX transaction.

ZDTP019S Message buffer allocation failed.

Explanation	Storage allocation failed for message buffer.
System Action	CICS Agent not initialized successfully.
User Response	Determine why the CICS region is unable to provide storage for the DTAX transaction.

ZDTP020S Get CICS Applid failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS ASSIGN for APPLID retrieval failed.
System Action	CICS Applid not displayed in DTAX screen and in log messages.
User Response	Determine the cause of the problem from `resp` and `resp2` codes.

ZDTP021S Get STARTCODE failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS ASSIGN for program startcode retrieval failed.
System Action	CICS Agent not initialized successfully.
User Response	Determine the cause of the problem from `resp` and `resp2` codes.

ZDTP022S CICS DELAY failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS DELAY failed.
System Action	N/A
User Response	Determine the cause of the problem from `resp` and `resp2` codes.

ZDTP023S Inquire DB2Entry failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS INQUIRE DB2ENTRY failed.
System Action	CICS Agent operation continues.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP024S CICS Agent could not locate a valid AgentId. Possibly a misconfigured or disconnected zRemoteAgent. Please check the zRemoteAgent or Server logs.

Explanation	CICS Agent did not get a valid AgentId after initialization.
System Action	CICS Agent disabled.
User Response	Check for error messages in zRemoteAgent or Server logs. If the Collector/Server is down, start the services.

ZDTP025S Inquire System failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS INQUIRE SYSTEM for CICS version retrieval failed.
System Action	By default CICS Agent for CICS Version 4.2 is loaded.

User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP026S Inquire Program failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS INQUIRE PROGRAM failed.
System Action	CICS Agent operation continues.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP027S Disable Exit program failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS DISABLE PROGRAM for all exits failed.
System Action	CICS Agent not disabled successfully.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP028S Disable RMI Exit failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS DISABLE PROGRAM for RMI exit failed.
System Action	RMI exit not disabled successfully.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP029S START Exit failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS ENABLE START failed for XPCREQC/XRMIIN/XRMIOUT exit.

System Action	Exits not started successfully.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP030S FREEMAIN failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS FREEMAIN failed.
System Action	Global Work Area for PLT program FREEMAIN failed.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP031S Enable Exit failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS ENABLE PROGRAM for enabling exits failed.
System Action	CICS Agent exits not enabled successfully.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP032S STOP Exit failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS DISABLE stop for exits failed.
System Action	CICS Agent exits not stopped successfully.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP033S Cancel ReqID failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS CANCEL REQID failed.
System Action	DTAX ICE transaction not cancelled.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP034S Inquire ReqID failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS INQUIRE REQID failed.
System Action	DTAX ICE transaction not started successfully.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP035S GETMAIN failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS GETMAIN for Global Work Area failed.
System Action	Global Work Area for PLT program not allocated. CICS Agent not initialized successfully.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP036S START DTAX ICE failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS START failed.
System Action	DTAX Interval control transaction is not started every 5 minutes.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP037S LOAD Agent failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS LOAD PROGRAM for ZDTAGTxx module failed.
System Action	ZDTAGTxx PTF and build date not printed in the log messages.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP038S LOAD ZDTSOAPH failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS LOAD PROGRAM for ZDTSOAPH module failed.
System Action	ZDTSOAPH PTF and build date not printed in the log messages.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP039S Release agent failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS RELEASE PROGRAM for ZDTAGTxx/ZDTSOAPH module failed.
System Action	CICS Agent operation continues.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP040S CICS Receive failed. resp:<xx>, resp2:<xx>.

Explanation	EXEC CICS RECEIVE DATA failed.
System Action	User command for DTAX transaction not received and hence not processed successfully.

User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP041S zOS SMFid unavailable.

Explanation	Error retrieving zOS SMFid.
System Action	CICS Agent not initialized.
User Response	Contact support.

ZDTP042S ZDTDCCAL INIT message send failed. rc: <xx>.

Explanation	CICS Agent initialization failed.
System Action	CICS Agent not initialized.
User Response	Contact support.

ZDTP043S Unknown AUTHTYPE:<xx>.

Explanation	EXEC CICS INQUIRE DB2CONN retreives an AUTHTYPE that is not one of GROUP, TERM, TX, OPID, USERID.
System Action	Authtype not specified in DB2 attachment data in Purepath.
User Response	Contact your DB2 Administrator.

ZDTP044S ZDTDCCAL Register Connection pool failed. rc:<xx>.

Explanation	Sending DB2 connection pool information from CICS Agent to zDC failed.

System Action	DB2 attachment details in PurePath is empty.
User Response	Contact support.

ZDTP045S ZDTDCCAL Register DLI Connection pool failed. rc:<xx>.

Explanation	Sending DL1 connection pool information from CICS Agent to zDC failed.
System Action	DLI attachment details in PurePath is empty.
User Response	Contact support.

ZDTP046S ZDTDCCAL Message send failed. rc:<xx>.

Explanation	CICS Agent could not send message to zDC.
System Action	No data passed from CICS to zDC.
User Response	Contact support.

ZDTP047S DTAX Transaction definition not found. Disabling dynatrace agent.

Explanation	The RDO definition for DTAX transaction is not found.
System Action	CICS Agent not enabled and the PLT program execution terminates.
User Response	Define DTAX transaction for the PLT program. Refer to CICRDO member in SZDTSAMP installation library for sample definition.

ZDTP048S Config block retrieval failed. rc:<xx>.

Explanation	CICS Agent could not find the configuration data with sensor settings.
System Action	CICS Agent not initialized.
User Response	Contact support.

ZDTP049S Config block not found.

Explanation	CICS Agent could not find the config block with CICS Sensor settings.
System Action	CICS Agent disabled.
User Response	Check if the CICS Transaction Server sensor is placed in the CICS Agent Mapping.

ZDTP050S DTAX Enable failed. rc:<xx>.

Explanation	CICS Agent Enable failed.
System Action	CICS Agent not initialized successfully.
User Response	Contact support.

ZDTP051S ZDTAGT Copyright Info not found.

Explanation	Could not find the Copyright block in ZDTAGTxx module.
System Action	CICS Agent continues. No PTF/Build date is printed in the zRemoteAgent log.
User Response	Contact support.

ZDTP052S Check Agent Mapping if the CICS Transaction Server sensor is placed.

Explanation	Configuration block with the CICS Sensor information is not found.
System Action	CICS Agent is disabled.
User Response	Check if the CICS Transaction Server sensor is placed in the CICS Agent Mapping. If the problem persists, contact support.

ZDTP053S Disabling CICS Agent.

Explanation	This message is an extention of ZDTP052S.
System Action	CICS Agent is disabled.
User Response	Follow ZDTP052S.

ZDTP001W Please update PLT, INITPARM and RDO to ZDTPLT.

Explanation	RDO definition for ZDTPLT with suffix (66/67/68/69/70) found.
System Action	Execution continues.
User Response	Update the RDO definitions for ZDTPLTxx and DTAX to refer to ZDTPLT without suffix. Refer to CICRDO member in SZDTSAMP installation library for sample RDO definitions.

ZDTP002W Extract Exit failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	There was a problem in retrieving the Global Work Area storage for exit program. EXEC CICS EXTRACT EXIT operation failed. This is also possible when an attempt was made to look up DTAX CONF when the CICS Agent is disabled.
System Action	N/A

User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

ZDTP003W zDC <name> and GWA unavailable.

Explanation	Exits are enabled but there was a problem in retrieving the Global work area because there was no zDC available.
System Action	N/A
User Response	Check if the zDC is up and connected to zRA. If the problem persists, contact support.

ZDTP004W zDC <zDC Name> unavailable.

Explanation	There was no zDC available for CICS Agent to connect to.
System Action	CICS Agent is disabled.
User Response	Start zDC and check if the zDC connects successfully to zRA. If the problem persists contact support.

ZDTP001I DTAX User Command: PING.

Explanation	User issued a DTAX PING command.
System Action	Check if the connection exists between the z/OS CICS Agent and the zRemote Agent.
User Response	N/A

ZDTP002I DTAX User Command: DISABLE.

Explanation	User issued a DTAX DISABLE command.

System Action	Disable all the exits used by CICS Agent and terminate the 5 minutes DTAX cycle
User Response	N/A

ZDTP004I DTAX User Command: CONF.

Explanation	User issued a DTAX CONF command.
System Action	Display the active CICS Sensor configuration in the DTAX panel.
User Response	N/A

ZDTP006I ZDTPLT[CPLT] Compiled on mm dd yyyy hh:mm:ss.

Explanation	This message is logged in zRemoteAgent log file when the CICS Agent connects during CICS Startup. This message gives ZDTPLT compilation and version information.
System Action	N/A
User Response	N/A

ZDTP007I ZDTAGT <PTF> <Build Date> <Build Time> VER <Product Release> <CICS Version>.

Explanation	This message is logged in zRemoteAgent log file when the CICS Agent connects during CICS Startup. This message gives CICS Agent module (ZDTAGT) compilation and version information.
System Action	N/A
User Response	N/A

ZDTP008I ZDTSOAPH <PTF> <Build Date> <Build Time> VER <Product Release>.

Explanation	This message is logged in zRemoteAgent log file when the CICS Agent connects during CICS Startup. This message gives ZDTSOAPH compilation and version information.
System Action	N/A
User Response	N/A

ZDTP009I INIT failed during PLT. Trying to reconnect...

Explanation	The PLT program is unable to establish connection with the zRemote Agent during PLT time.
System Action	Reconnect attempted during the next DTAX cycle, which starts immediately after the CICS region comes up.
User Response	N/A

ZDTP010I Screen size unavailable.

Explanation	There was a problem in retrieving the screen attributes.
System Action	Screen attributes are defaulted to Screen Width=80, Screen Height=24.
User Response	N/A

ZDTP011I DISABLE TRUE(ZDTAGTxx) with ENTRYNAME(ZDTAGENT).

Explanation	The exit program ZDTAGTxx associated with entryname ZDTAGENT is disabled. “xx” represents the CTS version.
System Action	N/A
User Response	N/A

ZDTP012I ENABLE TRUE(ZDTAGTxx) with ENTRYNAME(ZDTAGENT).

Explanation	The exit program ZDTAGTxx associated with entryname ZDTAGENT is enabled. “xx” represents the CTS version.
System Action	N/A
User Response	N/A

ZDTP013I ENABLE Exit program(ZDTAGTxx) with Exit([exit name]).

Explanation	Enable the global exit point specified by [exit name] associated with the ZDTAGTxx exit.
System Action	N/A
User Response	N/A

ZDTP014I START TRUE(ZDTAGTxx) with ENTRYNAME(ZDTAGENT).

Explanation	Start the user exit ZDTAGENT associated with the exit program ZDTAGTxx.
System Action	N/A
User Response	N/A

ZDTP015I START Exit program(ZDTAGTxx) with Exit([exit name]).

Explanation	Start the global exit point specified by [exit name] associated with the exit program ZDTAGTxx.
System Action	N/A

User Response	N/A

ZDTP016I STOP Exit program(ZDTAGTxx) and all associated exits.

Explanation	Stop and disable all the exit points associated with the exit ZDTAGTxx.
System Action	N/A
User Response	N/A

ZDTP017I DISABLE Exit program(ZDTAGTxx) with Exit([exit name]).

Explanation	Disable the global exit point [exit name] associated with the exit ZDTAGTxx.
System Action	N/A
User Response	N/A

ZDTP019I Config block retrieved successfully.

Explanation	Indicates CICS sensor configuration has been retrieved successfully after there was a problem in configuration block access. This message follows after ZDTP049S when the CICS Transaction Server sensor has been placed.
System Action	CICS Agent is initialized.
User Response	N/A

ZDTP020I Active Sensors: <TX | MQ | DB2 | DB2R | SOAP | CTG | DB2Fetch | DLI | DLIR | LE>.

Explanation	Indicates the CICS Sensors that are placed and active.

System Action	N/A
User Response	N/A

ZDTP021I CICS Sensors updated.

Explanation	Indicates that the CICS Sensors have been updated and the CICS Agent honored the changes during DTAX 5 minutes cycle.
System Action	N/A
User Response	N/A

ZDTP022I Inquire DB2Conn failed. resp:<xx>, resp2:<xx>, rcode:<xx>.

Explanation	EXEC CICS INQUIRE DB2CONN failed.
System Action	CICS Agent operation continues. No DB2 attachment details provided in the Purepath.
User Response	Determine the cause of the problem from `resp` and `resp2` codes. If the problem persists, contact support.

IMS Agent system messages

ZDTI032W Recovery routine entered.

Explanation	The IMS Agent's ABEND recovery routine was invoked by RTM.
System Action	The ABEND recovery process continues.
User Response	Contact support

ZDTI033W Successful ABEND recovery, agent disabled.

Explanation	The IMS Agent's ABEND recovery process was successful.
System Action	The IMS Agent is disabled and the IMS system continues to function.
User Response	Contact support

ZDTI034E Unable to obtain dynamic storage.

Explanation	The IMS Agent's ABEND recovery routine was unable to obtain the dynamic storage required to produce ABEND diagnostics.
System Action	The ABEND recovery process continues but some diagnostic information may not be available. A Software (SFT) Error Record further describing the ABEND is written to the zOS system SYS1.LOGREC data set.
User Response	Run the zOS EREP utility program to print the Software (SFT) Error Record associated with the ABEND. Contact support

ZDTI035W Retry not permitted, must percolate.

Explanation	The zOS system will not allow the IMS Agent's ABEND recovery process to continue.
System Action	The recovery process terminates and percolates (returns control to RTM).
User Response	Contact support

ZDTI036W ZDTIIInn zzzzzzzz yyyymmdd hh.mm VER vv.rr.mm ABEND at offset xxxxxx.

Explanation	An ABEND occured at offset xxxxxx in the IMS Agent module ZDTIIInn, where nn corresponds to the IMS version (i.e. 12, 13, 14). The remaining fields describe the maintenance level of the IMS Agent module.
System Action	A Software (SFT) Error Record further describing the ABEND is written to the zOS system SYS1.LOGREC data set. Optionally, an SVC dump may be taken. The ABEND recovery process continues.

User Response	Run the zOS EREP utility program to print the Software (SFT) Error Record associated with the ABEND. Contact support

ZDTI037W ZDTIIInn zzzzzzzz yyyymmdd hh.mm VER vv.rr.mm ABEND offset N/A, no SDWA provided.

Explanation	An ABEND occured in the IMS Agent module ZDTIIInn, where nn corresponds to the IMS version (i.e., 12, 13, 14). The remaining fields describe the maintenance level of the IMS Agent module. The ABEND offset could not be determined because no SDWA was provided by RTM.
System Action	The recovery process terminates and percolates (returns control to RTM).
User Response	Contact support

ZDTI038W Recovery routine of recovery routine entered.

Explanation	While recovering from an ABEND in the IMS Agent, the recovery routine encountered an ABEND.
System Action	The ABEND recovery process continues but does not attempt to produce further diagnostics. The IMS Agent is disabled and the IMS system continues to function.
User Response	Contact support

zOS Agent logs

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

CICS Agent logging

IMS Agent logging

zDC logging

zLocal Agent logging

zRemote Agent logging

Related pages

zOS Agent

zDC

zRemote Agent

System Information dashlet

zOS Agent troubleshooting

The zOS Agent includes several components. Each component has one (or more) logging mechanisms to aid in testing and troubleshooting. Most messages from each of the components go through the zDC to the zRemote Agent log. Download these messages by requesting the logs in the System Information overview panel of the AppMon Client. Analyzing the zRemote Agent log with the corresponding agent's log gives complete picture of the activity between components for better error determination.

CICS Agent logging

Control the CICS Agent log level either by the DTAX Transaction or by specifying an optional initparm at CICS region startup.

There are two different sets of CICS logs.

One set of messages occur when the DTAX issues the DISABLE and ENABLE command. These messages write to the CICS CSMT Transient Data Queue. View these messages in the CICS job log. DTAX also writes a set of messages to the CEEOUT SYSOUT statement when errors occur in the connection between the zDC and the DTAX transaction. View these messages in the CICS Job log. As long as the DTAX transaction can connect to the zDC, it logs its messages to the zRemote Agent.

The CICS Agent monitoring the CICS transaction activity routes its log messages to the zDC, and subsequently to the zRemote Agent. In this scenario with all the connectivity intact, all the component logs can be gathered for error diagnosis from the AppMon Client **System Information** overview panel. The log shows if there are any corrupted PurePaths, timeouts, or other errors that may have occurred. You may also see some statistical information in these logs.

If the CICS Agent has difficulties during startup, enable more granular logging for Agent startup by specifying FINE in the `INITPARM` for this unique CICS Region. INITPARM details are described here.

IMS Agent logging

There are two sets of IMS Agent logs.

The first set of IMS messages are from the IMS Injection job. These are messages that occur as IMS in injecting its Agent into the IMS Control Region. These messages only appear in the job log of the agent injection job. The second set of IMS messages is from the Agent as it monitors IMS activity. These messages are sent to the zDC which is then routed to the zRemote Agent.

zDC logging

`DTLOGLEVEL` sets the zDC log level specified in the zDC `SYSIN` parameters. View log message output in the zDC joblog. The joblog help determine errors that may occur during zDC start up or when connecting with the zRemoteAgent. Once the zDC successfully connects to the zRemote Agent, error messages from the zDC, CICS and IMS Agents route to the zRemoteAgent. You can change this log level value dynamically using a zDC modify command.

zLocal Agent logging

The `DTAGTCMD` parameter defined in the zDC `SYSIN` parameters sets the zLocal Agent log level for the zDC. The standard AppMon agent `loglevel` parameter is used, and this value can not be changed dynamically.

There are two sets of logs created for the zLocal Agent. Both are created in the OMVS file system. One set is a temporary set of logs only valid for the current execution of the zDC, and the location of these logs defaults to the home directory of the zDC user ID. You can override the location by using the `DTCHDIR()` parameter defined in the zDC SYSIN. Finsd a sample in member ZDCSYSIN in the sample PDS `&*HLQ* .LZDT70 0.AZDTSAMP` where *HLQ* is your high-level qualifier.

These log files can not be viewed from within the AppMon Client, but they are seldom used, primarily for issues with agent startup or abends. The zLocal Agent also creates a standard set of AppMon agent logs, one for bootstrapping, and one for the agent itself. These logs are located in the standard AppMon agent log locations on the OMVS file system. You can view these logs from the AppMon Client in the **System Information** overview panel.

zRemote Agent logging

The zRemote Agent can have two logs if using the Bootstrap agent. The zRemote Agent log level is set in the AppMon Client and can be changed as needed. The zRemote Agent logs are created on the machine where the zRemote Agent is installed, in the standard AppMon Agent log locations. You can view the zRemote Agent logs either directly on the machine hosting the zRemote Agent or by requesting them from the AppMon Client in the **System Information** overview panel.

The zRemote Agent bootstrap log should be similar to a Java Agent bootstrap log, containing the following:

- The initial connect messages.
- Any optional messages about downloading a new agent library.

The actual zRemote Agent log should be similar to a Java Agent log, containing the following messages:

- The actual zRemote Agent log messages once the the bootstrap agent loads the agent library.
- Log messages that are sent from all CICS/IMS Agents and the zDC.
- Log messages that are sent from the zLocal Agent.

If the zOS components communicate correctly, there is no need to look at JOB output of the zOS components. Access all component logs using the **System Information** overview pane in the AppMon Client.

zOS Agent troubleshooting

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Component set up verification

z/OS zLocal Agent could not be started

System under diagnosis (CICS region) could not connect to the zDC

Expected queues and/or transactions (including SOAP and CTG transactions) are not being monitored

Disabled IMS Agent

Migration troubleshooting

Related pages

Related topics

zOS Agent

CICS Agent installation and configuration

CICS Transaction Server sensor

Transaction measure

DTAX transaction

zDC

zOS Agent logs

zOS Agent messages

This page describes various techniques for troubleshooting and for collecting the necessary data for Customer Support.

Component set up verification

The following sections describe how to identify set up issues.

Verify the CICS Agent group and resources installed correctly

> **Note**
>
> The group name may be different, as well as the two-character suffix representing the CICS release of the Agent (for example, CTS52 uses 69).

From the CICS Region, look for these messages to validate the CICS resources have been defined:

```
CICSAPPL Install for group XXXX has completed successfully.
CICSAPPL OWNER CSSY Resource definition for ZDTAGT69 has been added.
CICSAPPL OWNER CSSY Resource definition for ZDTPLTSD has been added.
CICSAPPL OWNER CSSY Resource definition for ZDTPLT69 has been added.
CICSAPPL OWNER CSSY Resource definition for ZDTSOAPH has been added.
CICSAPPL OWNER CSSY Resource definition for ZDTDC2 has been added.
CICSAPPL OWNER CSSY Resource definition for ZDTDC2A has been added.
CICSAPPL OWNER CSSY TRANSACTION definition entry for DTAX has been added.
```

Verify the CICS Agent PLT program is invoked

The CICS Agent no longer logs its messages to the JESMSGLG DD.

Messages are now logged to the zRemote Agent log, accessible from within the AppMon Client, in the same manner as all the other Agent log files. Look for log entries similar to the following:

```
info    [native] ASID[221], smfID[CW06], sysid[C208], jobName[H06AC208],
        subagentid[c29ba1fd] snaId[USCWXN01.H06AC208], CICS release 42 was successfully
        initialized with protocol version=<dTMajor>.<dTMinorVersion>.0
info    [native] SENSOR_CICS ................. 03
info    [native] SENSOR_IMS .................. 02
info    [native] New subAgent registered successfully: c29ba1fd
info    [native] License = license ok;
info    [native] ASID[221], smfID[CW06], sysid[C208], jobName[H06AC208] Transaction[DTAX]
        contextPath[0000008C.C29BA1FD] - ZDTP005I - ZDTPLT Compiled on Mon DD YYYY
        HH:MM:SS.
info    [native] ASID[221], smfID[CW06], sysid[C208], jobName[H06AC208] Transaction[DTAX]
        contextPath[0000008C.C29BA1FD] - ZDTP005I - Transaction: CPLT.
info    [native] ASID[221], smfID[CW06], sysid[C208], jobName[H06AC208] Transaction[DTAX]
        contextPath[0000008C.C29BA1FD] - ZDTP003I - dynatrace CICS Agent initialized.
```

Verify the zDC started correctly

Verify that the zDC has started the correct version, initialized successfully and has started/attached to the z/OS zLocal (formerly USS – Unix System Services) Agent (messages in zDC SYSPRINT DD).

```
ZDC000I INITIALIZATION STARTED FOR zDC  VER 0<dTMajor>.0<dTMinorVersion>.00
ZDC065I zDC INITIALIZATION SUCCESSFUL. RETCODE=00/00 <dTMajor>.<dTMinorVersion>.0.<dTbuil
.
.
.
-------------------------------------------------------------------------------
Dynatrace Agent for z/OS Copyright (c) 2004-<currentYear> Dynatrace, www.dynatrace.com
-------------------------------------------------------------------------------
Version .................. <dTMajor>.<dTMinorVersion>.0.<dTbuild>, build date MMM DD JJJ
.
.
.
info    (native) ASID(448), smfID(CW01), sysid(MEPC), jobName(YOURZDC),
subagentid(xxxxxxxx) smfID.JobID(........) was successfully initialized
```

If the following error message appears in the z/OS zRemote Agent log file:

```
JJJJ-MM-DD HH:MM:SS [37ab2e00] severe [native] Exception while
connecting to Collector, info:<Could not register Agent with
Collector as the Collector declined service>
```

Then the Collector/Server log file should be searched for the following connection error:

```
JJJJ-MM-DD HH:MM:SS SEVERE [AgentListener] Connection for outdated Agent
 '<agentName>@<agentHost>:49
(dTMajor.dTMinorVersion.0.dTBuild)' (version: n/a) refused. Please update the Agent
```

or the following message:

```
JJJJ-MM-DD HH:MM:SS SEVERE [AgentListener] Connection for newer Agent
'<agentName>@<agentHost>:204
(dTMajor.dTMinorVersion.0.dTBuild)' (version: n/a) refused. Please
connect to a Dynatrace Collector with the same or a higher version.
```

Make the appropriate adjustments to the affected components.

If the following message appears in the zDC job log (your subsystem name may be different than MEPC):

```
11:55:30.419083 ZDC006E SUBSYSTEM MEPC ALREADY EXISTS AND IS ACTIVE
```

Follow the steps outlined in the Emergency ZDC Termination section.

Insufficient access authority message when starting the zDC

If the following messages (or something similar) appear in the zDC joblog:

```
H408I USER(xxxxxxxx) GROUP(xxxx) NAME(STARTED TASK )
BPX.FILEATTR.PROGCTL CL(FACILITY)
INSUFFICIENT ACCESS AUTHORITY
ACCESS INTENT(READ ) ACCESS ALLOWED(NONE )
```

And the associated messages appear in the z/OS zLocal Agent logfile:

```
JJJJ-MM-DD HH:MM:SS  3f8bbe02¨ info  native¨ Server requests us to use Agent
dTMajor.dTMinorVersion.0.dTBuild with a hash of 0317af199c1ab1a03dda2cee90c2ea61
JJJJ-MM-DD HH:MM:SS  3f8bbe02¨ info  native¨ Requesting Agent library from Server
JJJJ-MM-DD HH:MM:SS  3f8bbe02¨ info  native¨ Error setting Agent library program
controlled: EDC5139I Operation not permitted.
JJJJ-MM-DD HH:MM:SS  3f8bbe02¨ info  native¨ Loading Agent
/dt/dynatrace-<dTMajor.dTMinorVersion.0>/agent/downloads/<dTMajor.dTMinorVersion.0.dTBuil
/native/zos-s390-64/libdtzagent.so
```

These can be safely ignored. After downloading the core Agent initially, the bootstrap Agent is trying to set a flag for the downloaded library that is needed in certain circumstances. It doesn't try to set this flag during the second run because it doesn't re-download the library if it's already there.

Verify the z/OS zLocal Agent is running and connected to the zDC and AppMon Collector/Server (in zDC SYSPRINT)

> **Note**
>
> Agent status information and Agent log files for connected Agents are available through the AppMon Client, but this process can help to diagnose Agent connection failures.

The standard AppMon Agent logfile contains the standard AppMon Agent messages relating to startup, logging, versioning, and connectivity to the Server/Collector. The zDC SYSPRINT contains the `stdout` for the z/OS zLocal Agent, and this contains the location of the z/OS zLocal Agent log file.

If the SYSPRINT does not contain this information, instruct the zDC to display it using the following command:

```
Modify zdc,dt1 stdo

info (native) Log level console ........... info
info (native) Log level file .............. info
info (native) Log file .................... <dtInstallFolder>/log
/dt_<agent_name>\_<process_id>.log
```

Once that log file is located, it is accessible by the standard techniques. Use OMVS, ssh/telnet, or view it in the AppMon Client like a regular Agent log file.

This log file should be searched for the following items to verify that all the channels are connected properly.

```
info    [native] dynaTrace z Remote Agent data channel connected successfully, performinc
info    [native] dynaTrace z Remote Agent client handshake performed.
info    [native] dynaTrace z Remote Agent data channel handshake successful, version[rr.r
info    [native] dynaTrace z Remote Agent control channel connected successfully, perform
info    [native] dynaTrace z Remote Agent handshakes are complete, all channels are fully
```

The corresponding zRemote Agent log file should be searched for the following (the values inside the bracket should reflect your live data).

```
info    [native] dynaTrace z Local Agent handshakes are complete, all channels are fully
info    [native] Data client socket listener thread started
info    [native] ASID[48], smfID[S0W1], sysid[Z208], jobName[AF71Z208], subagentid[da57ff
info    [native] zDC[Z208] SMO is initialized with size=10M.
info    [native] Registering the zdc[48]
```

If the following appears in the zRemote Agent log file:

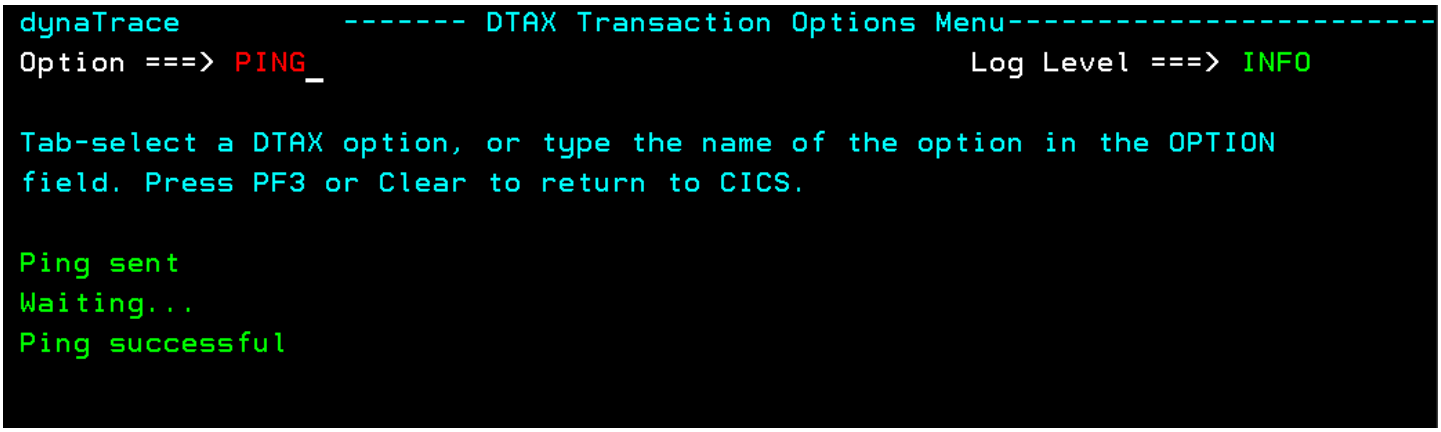
```
info    [native] Starting without instrumentation, agent did not match a System Profile
```

This indicates that the name specified in the zRemote Agent startup does not match a name defined in any System Profile defined to the AppMon Server. You must alter the name parameter to match your agent mapping, or define a new agent mapping that matches the name parameter.

Verify the connectivity loop by issuing a test ping

1. Log in to CICS.
2. Execute the transaction 'DTAX' with a parameter of 'PING'.
3. Verify that the PING traveled from CICS to zDC to z/OS zRemote Agent and back again from z/OS zRemote Agent to zDC.

DTAX Transaction output



Log message in the Agent log file:

```
info    [native] ASID[228], smfID[CW06], sysid[C208], jobName[H06AC208] Ping data=CA72106
```

Verify the z/OS zRemote Agent received the CICS Sensor configuration

Log messages are in the zRemote Agent log file. You will need to enable the 'config' log level.

Values for SENSOR_CICS:

00 = not placed
01 = placed and disabled
03 = placed and enabled

The default settings:

```
info    [native] ZAgent capturing ..... enabled
info    [native] SENSOR_CICS ..... 03
```

Sample of customized settings:

```
config    [native] SensorProperty: MQ.Enabled = true [bool]
config    [native] SensorProperty: MQ.QueueNameInclude[0] = MY.QUEUE.NAME [string from
          string list]
config    [native] SensorProperty: MQ.QueueNameInclude[1] = PROD.QUEUE.* [string from
          string list]
config    [native] ZAgent capturing ..... enabled
config    [native] SENSOR_CICS ................ 03
```

z/OS zLocal Agent could not be started

Check the job log of the zDC for any of the following error messages.

> **Note**
>
> These do not appear in the z/OS zLocal Agent log file, since the Agent could not be started.

Invalid permission on the z/OS zLocal Agent folder or libraries

```
ZDC996E      Atx5!DTAGTCMD attach failed
ZDC997D      RetRcRsn FFFFFFFF 0000006F 5B400002 EACCES: Permission is denied
ZDC997D      Notice: unknown modid, reason text may be incorrect
ZDC992I      IniA!z/OS Unix agent attach failed
ZDC996E      @Abrtz/OS Unix agent attach failed,Abort
```

Correct the permission on the z/OS zLocal Agent folder/binaries, so they are at least readable and executable.

TCP/IP access denied

```
ZDC996E      Atx5!DTAGTCMD attach failed
ZDC997D      RetRcRsn FFFFFFFF 0000006F EF076015 EACCES: Permission is denied
ZDC997D      Description: SAF CKACC returned error.
ZDC996E      @Abrtz/OS Unix agent attach failed,Abort
```

If your security system, such as Top Secret or ACF2, denies access to TCP/IP by default, add a rule to permit the zDC user ID to access TCP/IP.

Invalid location specified for the z/OS zLocal Agent

```
ZDC996E      Atx5!DTAGTCMD attach failed
ZDC997D      RetRcRsn FFFFFFFF 00000081 0516006C ENOENT: No such file, directory, or IPC
             member exists
ZDC997D      JRFileNotThere: The requested file does not exist
ZDC992I      IniA!z/OS Unix agent attach failed
ZDC996E      @Abrtz/OS Unix agent attach failed,Abort
ZDC975E      z/OS Unix agent attach failed,Abort
```

Correct the DTAGTCMD parameter in the SYSIN DD of the zDC to point to the correct z/OS zLocal Agent binary location.

System under diagnosis (CICS region) could not connect to the zDC

Check the job log of the affected CICS region(s) for the following message, where xxxx = the subsystem ID of the zDC that the CICS region is trying to connect to, or blank if it is trying to connect to the default subsystem that is configured with the DEFAULT(YES) parameter. It is advisable to simply search for the error message code.

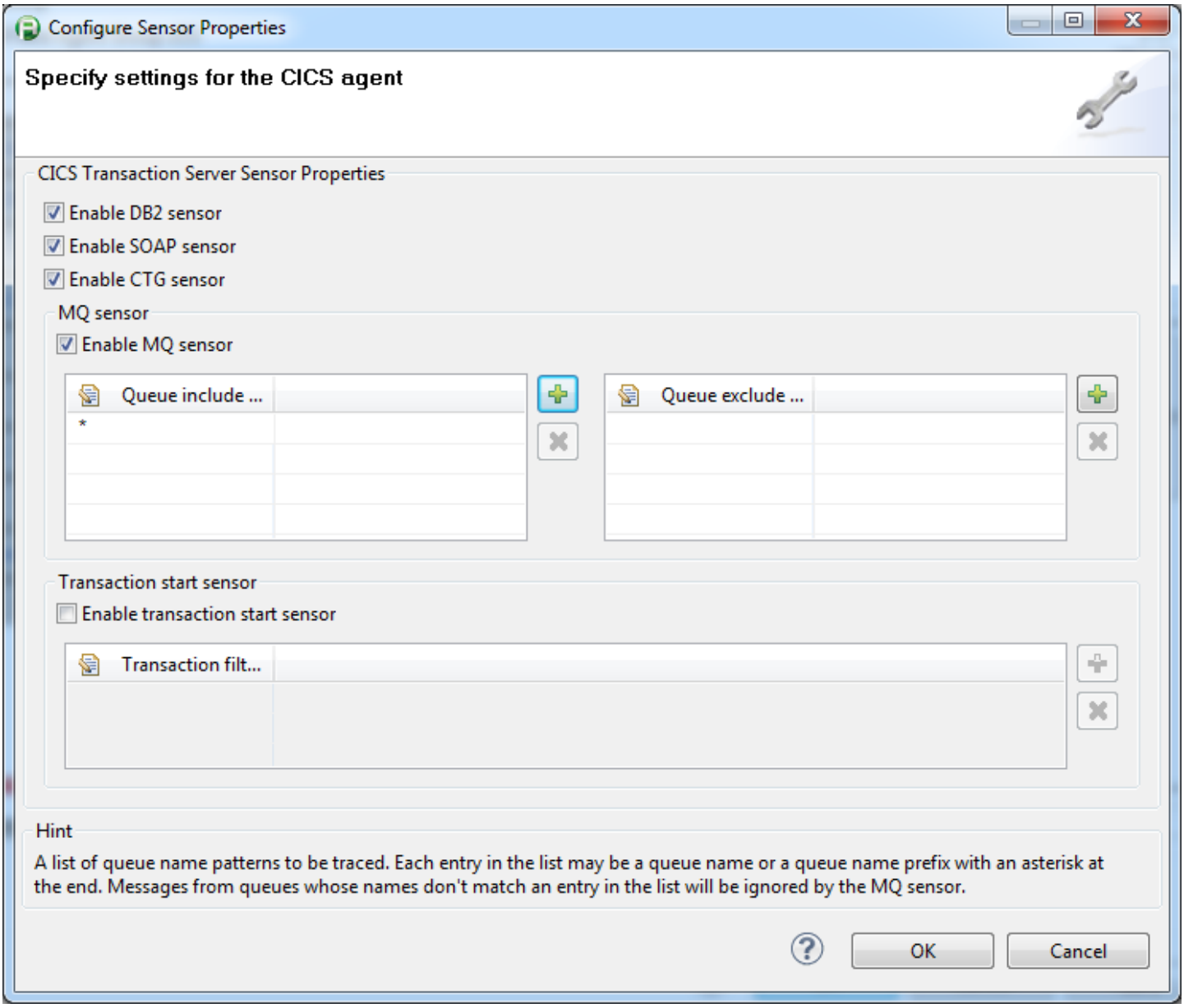
```
ZDTP002W zDC xxxx unavailable or CICS exits disabled
```

Verify that the zDC with that subsystem ID is started. If so, then try to issue a DTAX Enable command to re-enable the connections.

Expected queues and/or transactions (including SOAP and CTG transactions) are not being monitored

Verify that the CICS Sensor is configured correctly, and that the monitored CICS region's configuration matches that of the CICS Sensor in the system profile.

Edit the CICS Sensor Properties for the System Profile under diagnosis.



Issue a DTAX CONF command to display the following output:

```
dynaTrace          ------- DTAX Transaction Options Menu-----------------------
Option ===> CONF_                                    Log Level ===> INFO

Tab-select a DTAX option, or type the name of the option in the OPTION
field. Press PF3 or Clear to return to CICS.

zDT Agent: MEPM Logic Errors: 0, API arguments: 0
Version: 0
Last Configuration Update Date & Time: 03/28/2013, 12:20:20
Last Ping Date & Time: 00/00/0000, 00:00:00

Enabled Sensors:    MQ   DB2   SOAP   CTG
MQ.QueueNameInclude: *
```

Disabled IMS Agent

If an ABEND occurs in the IMS Agent code, the recovery process produces ABEND diagnostics if possible, then disables the IMS Agent. The IMS system continue to function. When this occurs a series of WTO messages are written to the system log for the IMS Control Region and/or IMS dependent region. A sample normal message set follows:

```
ZDTI032W Recovery routine entered.
ZDTI036W ZDTIII14 BDCSBK00 20160713 10.15 VER 06.05.00 ABEND at offset 001DB6.
ZDTI033W Successful ABEND recovery, agent disabled.
```

Different or additional messages may be issued if abnormal conditions are encountered by the recovery process, such as when dynamic storage cannot be obtained, retry is not permitted, or no SDWA was passed. All of the messages related to the ABEND recovery process are documented in the zOS Agent Messages section.

A Software (SFT) Error Record further describing the ABEND is usually written to the zOS system SYS1.LOGREC data set. You should run the zOS EREP utility program to print the Software (SFT) Error Record associated with the ABEND.

Optionally, an SVC dump may be taken during recovery, depending on the ABEND recovery option specified or defaulted to when the IMS Agent was injected. The default action is to not capture an SVC dump when ABEND recovery is driven. This option can be specified as a parameter when the IMS Agent is initially injected, or specified in conjunction with the Modify or Enable function parameters to toggle dump capture during recovery on or off for a previously injected IMS Agent. See IMS Agent Injection and Configuration.

When the IMS Agent becomes disabled as a result of the ABEND recovery process, it remains disabled until explcity re-enabled using the IMS Agent injection program. See IMS Agent Injection and Configuration.

When the IMS Agent becomes disabled as a result of the ABEND recovery process, you should contact support for a resolution.

Migration troubleshooting

In general, do not connect newer z/DCs to an older AppMon Server or newer Agents to older z/DCs. Newer AppMon Servers with older other components should normally be fine – no matter if Agents are bootstrapped or not (except a “wild” Agent mix across CICS regions).

Agents started by the bootstrap part update on application restart.

zDC stats

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

zDC

Host Monitoring Agent

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Installation and configuration

Agent bulk installation

Related pages

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Database Agent

OneAgent

Overview

You only need a Host Monitoring Agent if no other Agent runs on that particular machine, because any other active Agent except Node.js delivers those metrics.

The Web Server master Agent only delivers when a slave / sub- Agent is connected to it.

With many Agents on one machine, you may want to switch off delivery of host monitoring data from non-Host-Agents to reduce traffic from Agent to Collector. See support for the best strategy.

Restart Host Agents from the Client's **Agent Overview / Status dashboard > Host Agent** context menu.

For AppMon 6.5 and later, the Host Monitoring Agent can monitor availability of any process running on the host. For further information on process availability monitoring see information on configuration of hosts/host groups on the Infrastructure page.

Installation and configuration

Windows

The Windows full package installer installs the Host Agent, but the service is not registered due to the potential redundant Host Agent role mentioned previously. You can register the Host Agent with `" <DT_HOME>\agent\lib\dthostagent" -service install`. Copy and paste this example for 32-bit Host Agent service from a 64-bit AppMon full installation:

```
"C:\Program Files\dynaTrace\dynaTrace 7.0\agent\lib\dthostagent" -service install
```

Unregister the Agent with `sc delete "dynaTrace Host Agent7.0"`.

With the Agent-only installer, you must custom install the Host Agent to be installed and registered for auto-start. This lets you monitor the machine with no other Agent running on it. The default `<DT_HOME>` is `C:\Program Files (x86)\dynaTrace\dynaTrace Agent 7.0`, including the 64-bit libraries in `agent\lib64`.

*NIX

On *NIX, use `cd /opt` and `java -jar <DownloadDir>/<dynatrace(Agent)Installer.jar>`. The installer defaults to the current directory and asks if you want to change the destination. Make sure you have the correct permissions for installation and to write to the target location.

`<DT_HOME>` is `/opt/dynatrace-7.0` or `/opt/dynatrace-agent-7.0`, for the full or Agent-only installer respectively. The `<DT_HOME>/init.d/dynaTraceHostAgent` shell script should be copied to or referenced in (link) `/etc/init.d` and added to the system boot by using something similar `chkconfig` or `update-rc.d`, depending on your distribution.
Configuration must be done by editing the `<DT_HOME>/agent/conf/dthostagent.ini`:

Key	Default	Description
Name	host	Agent name, can be used to map an Agent to a specific Agent Group.
Server	localhost:9998	The AppMon standalone or Server-embedded Collector to which this Agent should connect. The argument name “Server” stems from past releases when the Collector was Server-integrated. Most licenses allow only standalone Collectors, except for testing.
Loglevel	info	Define log level.
ConsoleLoglevel	info	Define console log level.
LogfilePath	<agent location>/log	Specify a custom log file.
OverrideHostname	<not set>	Specify a custom hostname, if set this prevents the Agent from auto detecting the hostname.
usessl	false	Enables SSL encryption between the Agent and the Collector. Encrypted communication uses the same port as non-encrypted. To enable SSL you need to set this parameter as true before launching the Agent.

Example basic dthostagent.ini

The following is an example `<DT_HOME>/agent/conf/dthostagent.ini` file. This file configures the Host Monitoring Agent named **host** (default) to connect to the AppMon Server (actually stand-alone or Server-integrated / embedded **Collector**) with the fully qualified name **dtcollector.localdomain.com** on port **9998**.

```
Name host
Server dtcollector.localdomain.com:9998
```

Agent bulk installation

To monitor a large number hosts in the hundreds, you can automate the Host Agent deployment.

Windows

Perform a (custom) installation with the full or Agent-only `.msi`, or use `dthostagent.exe` and `dtagentcore.dll` and `dthostagent.ini` in the directory structure.

If you use the installer with `msiexec /passive /i <path>\<dynaTraceInstallerFileName>.msi ADDLOCAL="HostAgent"`, then Host Agent registers as an auto-starting service with the default settings from `dthostagent.ini`.
You can rewrite `dthostagent.ini` and un/register the service with the new arguments.

If you just use the `.exe` and `.dll`, you can write your own `dthostagent.ini` and register the service to auto-start with `<DT_HOME>\agent\dthostagent.exe -service install`.

***NIX**

```
#extract dthostagent and libdtagentcore.so from agent/lib64/ in the jar
#extract init.d/dynaTraceHostagent from the jar

#mimic Agent directory structure on the host
mkdir dynatrace-agent-7.0.0
mkdir dynatrace-agent-7.0.0/agent
mkdir dynatrace-agent-7.0.0/agent/lib64
mkdir dynatrace-agent-7.0.0/agent/conf

#copy dthostagent and libdtagentcore.so to agent/lib64/
#write a basic dthostagent.ini (see above) to agent/conf
#copy dynaTraceHostagent shell script to /etc/init.d
#be sure the Name specified in dthostagent.ini is mapped in some tier to avoid an error m
#try the Agent before you add it to boot
#/etc/init.d/dynaTraceHostagent start
chkconfig add dynaTraceHostagent
```

Database Agent

Related pages

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

OneAgent

The **Database Agent** directly runs on the AppMon Collector and as such is delivered out-of-the-box and no other components are needed. It connects to a database instance and retrieves data every 30 seconds (measures) or on-demand (live view, execution plan).



Please see Database Agent configuration to learn how to configure the Agent.

OneAgent

On this page

Agent Platform Overview

OneAgent for Appmon

Related pages

OneAgent configuration

OneAgent rollout

Set up agents

Sensors

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

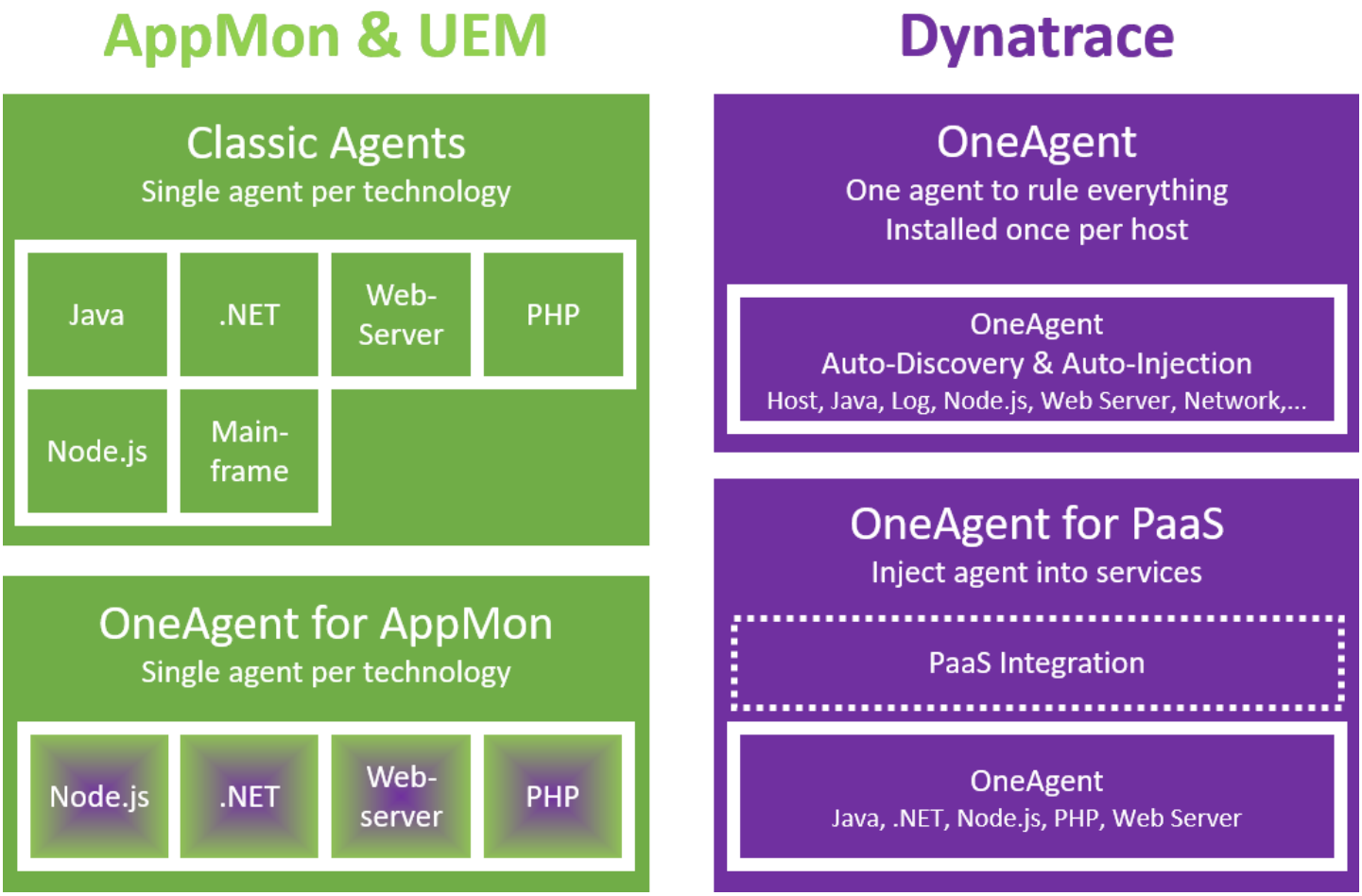
JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

Agent Platform Overview



Major differences

- The **AppMon Classic Agents** are the origin technology specific agents available in AppMon.
- The **OneAgent for Appmon** is based on a *newarchitecture* shared with the Dynatrace Agents.
- The **Dynatrace OneAgent** is a single, fullstack Agent which installed once per host, and then automatically discovers all subsystems (processes, containers, processes within containers, network, databases) running on it and automatically instruments them. Read more about Dynatrace Oneagent
- The **Dynatrace OneAgent for PaaS** is used in environments with limited access on the host-level such as containerized or Platform-as-a-Service (PaaS) environments. Read more about Dynatrace PaaS integrations

OneAgent for Appmon

Upgrade benefits from Classic Agent



- End-to-end encryption via https
- Code-instrumentation on the Agent
  - Faster agent startup time
  - Less collectors required
  - Less/no pressure on collectors
- Easy traffic tunneling via https based protocol
- No webserver master agent required
- Support for new platforms, technology & features which are not available with the Classic Agents.

Compare agent availability with classic agent

Agent	OneAgent Status	Windows	Linux x64	Linux PPC LE	Linux PPC BE	Linux Musl
Node.js	GA					
Webserver						
Apache	GA					
NGINX	GA					
IIS	BETA					
PHP						

PHP 5.x	BETA					
PHP 7	BETA					
.NET						
.NET Framework	BETA					
.NET Core	BETA					
Java	BETA					

Legend

 Classic Agent/OneAgent

 available

 not available, will *not* be available in further releases

 not available now, will be available in further releases

Agent Status

GA - Generally Available

The Agent is complete. Keep in mind that the Features/Sensors known from Classic Agent can (must not though) behave different.

BETA

OneAgent released as BETA functionality for AppMon 2017 May. Please be aware that:

- Not all features/sensors available in Classic Agent are already available in the OneAgent for AppMon
- Features/Sensors known from Classic Agent can behave differently

To use OneAgent in AppMon 2017 May you need to install Classic Agents first, and then roll them out to the OneAgent. No additional configuration needed. Before you switch to OneAgent, double check your major requirements with the available feature set.

> **Warning**
>
> You must **not** use the OneAgent in your production environment. The aim of BETA is to test it in pre-production environment, and see how the new Agent affects your custom dashboards/configurations you use to monitor your production environment.

BETA Agents are publicly available in the same release as GA agents and supported by Dynatrace technical support team.

Agent Groups

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What are Agent Groups?

What are the benefits of Agent Groups?

Configure Agent Groups

Agent mapping workflow

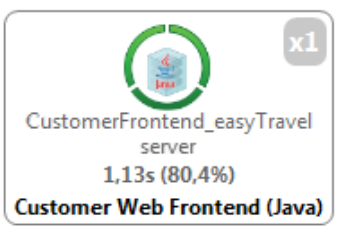
Disabling System Profiles

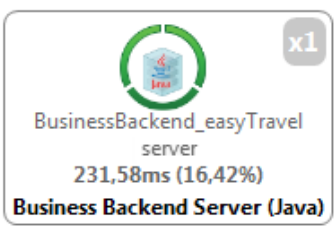
What are Agent Groups?

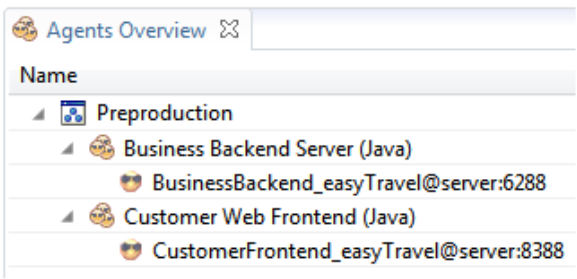
Use Agent Groups to automatically assign AppMon Agents to groups that share the same configuration settings for Sensor Placement, Sensor Configuration, and Measure Subscription. Use Agent Mappings to map AppMon Agents based on name or host.

What are the benefits of Agent Groups?

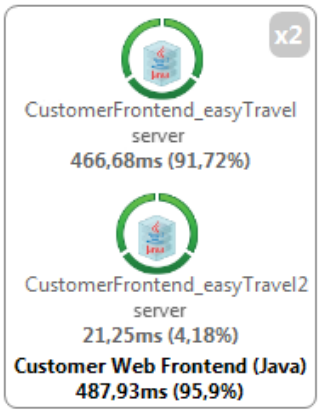
AppMon is initially deployed and configured in a test or pre-production environment. These environments are similar to their production counterparts but with fewer SUDs and fewer AppMon Agents: two application servers in preproduction versus four in production.

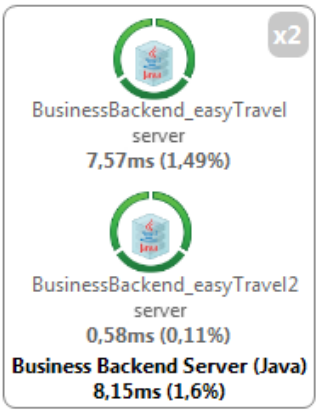


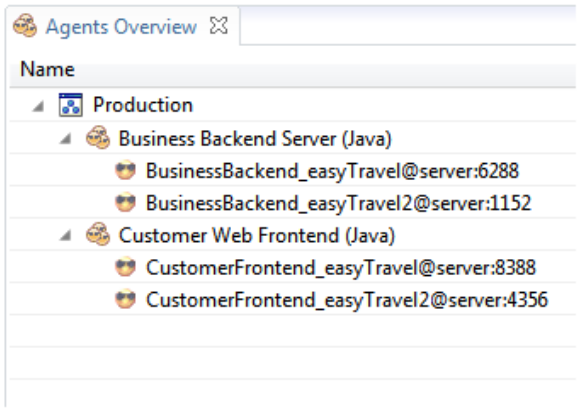




When AppMon is deployed in production, the AppMon configuration must be migrated. Use a layout of Agent Groups and Agent Mappings to simplify this process. The production AppMon Agents are dynamically mapped to Agent Groups as they connect to the AppMon Server. They share the same Sensor Placement and Sensor Configuration. The Measure Subscriptions are shared on-the-fly with zero configuration.







Configure Agent Groups

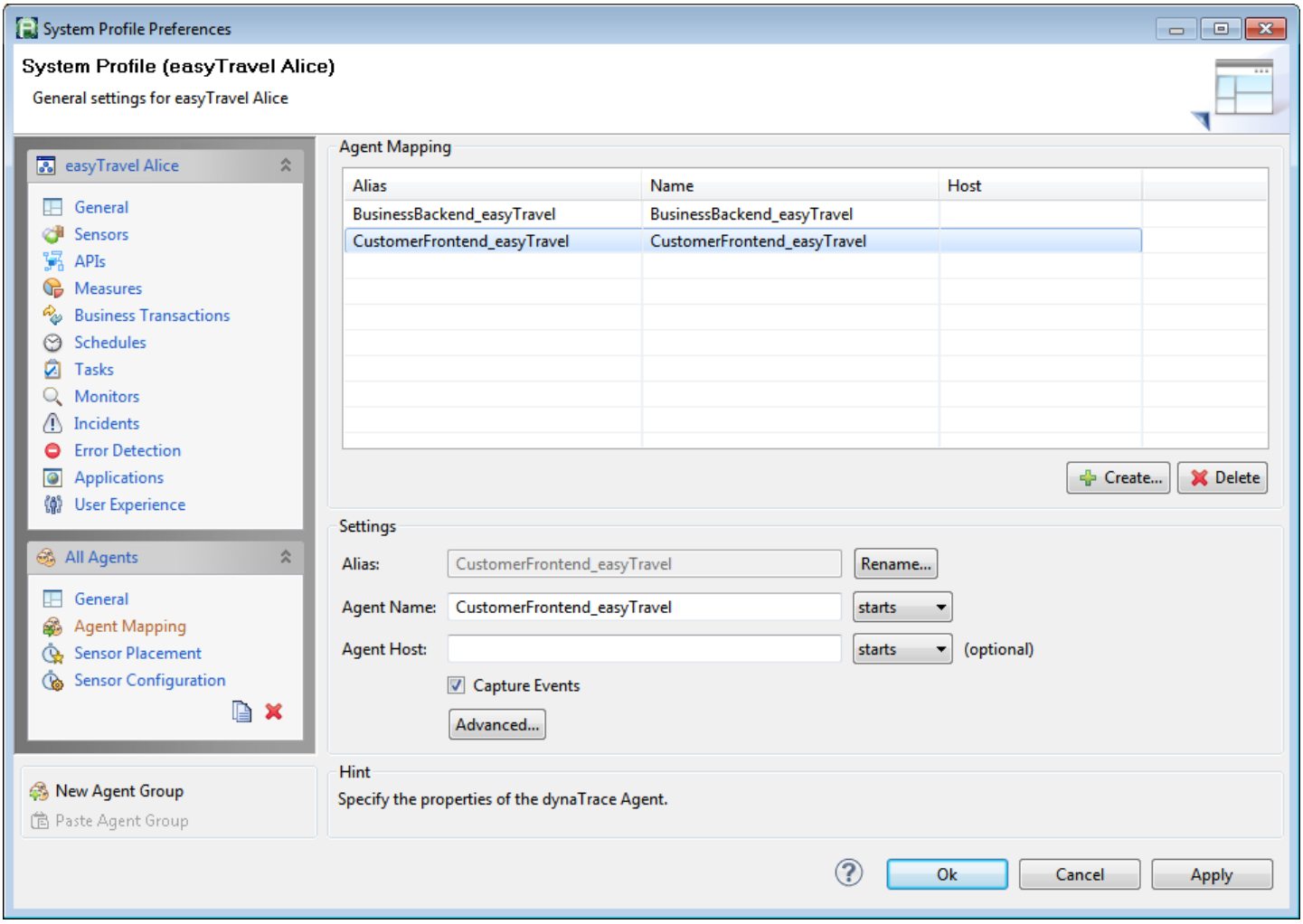
An Agent Group defines:

- Which Sensors to place (Knowledge Sensor Packs, Method and Memory Sensor Rules).
- How the Sensors are configured, such switch on/off parameter capturing for the Servlet Sensor.
- Which Measures are subscribed: either for the specific Agent Group, a set of Agent Groups or all Agent Groups.

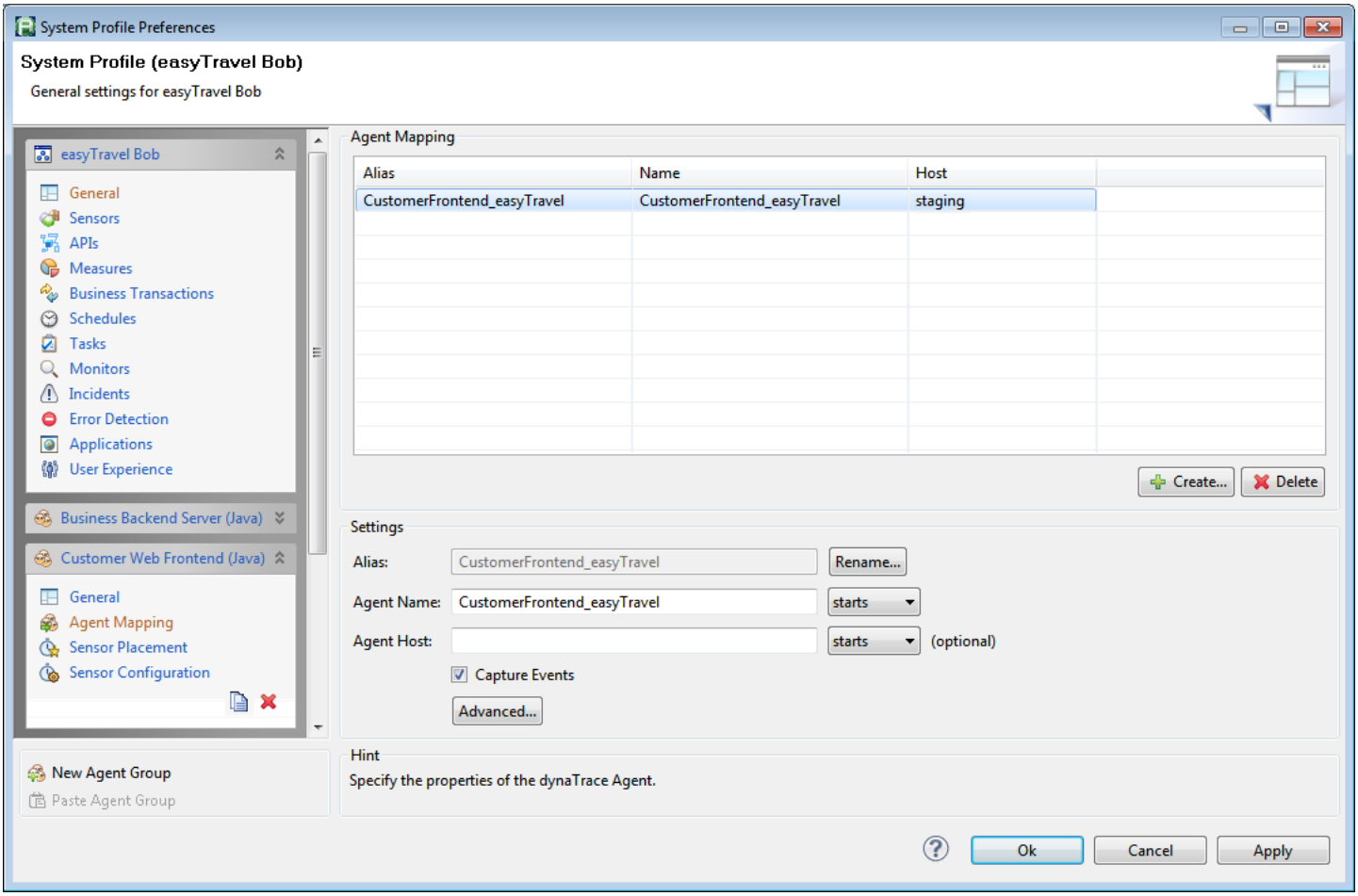
The Agent Mapping specifies which AppMon Agents belong to an Agent Group based on Agent Name and optionally the Agent Host. Additionally, you must configure an alias and use it throughout the AppMon Client for user-friendly visualization since AppMon Agents are similarly named to the SUD (e.g. node42) or infrastructure (e.g. prod-100-xserver).

Agent mapping workflow

It is important to understand how AppMon Agents are mapped to Agent Groups and assigned to System Profiles. Consider the following example:



The System Profile *easyTravel Alice* has a single Agent Group *All Agents* configured. It contains two Agent Mappings, *CustomerFrontend_easyTravel* and *BusinessBackend_easyTravel*, which map all AppMon Agents starting with *CustomerFrontend_easyTravel* and *BusinessBackend_easyTravel* from all hosts (the Agent Host setting is empty) to this Agent Group and System Profile.



In contrast, *easyTravel Bob* has two Agent Groups *Customer Web Frontend (Java)* and *Business Backend Server (Java)*. Each contains one Agent Mapping *CustomerFrontend_easyTravel* and *BusinessBackend_easyTravel* respectively, which map AppMon Agents named *CustomerFrontend_easyTravel* to the Agent Group *Customer Web Frontend (Java)*. Analogously, AppMon Agents named *BusinessBackend_easyTravel* map to *Business Backend Server (Java)*.

Putting it all together

When an AppMon Agent initially connects, the AppMon Server (Collector) iterates over all System Profiles in alphabetical order. Then it evaluates all Agent Groups and their respective Agent Mappings one after another, from top to bottom (as they have been defined), preferring Agent Mappings that match both name and host over name-only matches.

Let's deploy both System Profiles *easyTravel Alice* and *easyTravel Bob* to the same AppMon Server and examine the mechanics when various AppMon Agents connect:

Example 1: An AppMon Agent named *CustomerFrontend_easyTravel1* connects from host *staging*

It initially matches to *easyTravel Alice's* Agent Group *All Agents*. Since the host value is empty, there may be a better match to one of the other Agent Mappings; the search continues. The Agent Mappings of *easyTravel Bob* match AppMon Agents starting with *CustomerFrontend* _easyTravel but also *staging* has been explicitly specified, and a better match was found. Thus, the AppMon Agent is associated with *easyTravel Bob's* Agent Group *Customer Web Frontend (Java)*.

Example 2: An AppMon Agent named *BusinessBackend_easyTravel32* connects from host *testing*

It initially matches to *easyTravel Alice's* Agent Group *All Agents*. Since no host value is specified, the search continues. *easyTravel Bob*'s Agent Groups explicitly match AppMon Agents connecting from *staging*; no better match is found. The AppMon Agent is associated with *easyTravel Alice*'s Agent Group *All Agents*.

Example 3: An AppMon Agent named *myAgent* connects from *staging*

It neither matches to any of *easyTravel Alice*'s nor *easyTravel Bob*'s Agent Mappings; it is not associated with any of the System Profiles' Agent Groups. Nevertheless, the AppMon Agent shows up in the Agent Overview and **System Overview** dashlets of all System Profiles to increase the user awareness because this typically indicates misconfiguration. After appropriate Agent Mappings are created, the AppMon Agent must be restarted.

Disabling System Profiles

Multiple System Profiles with similar Agent Mappings are confusing. In big deployment scenarios, we recommend that you pick meaningful names for all AppMon Agents and map them into clearly defined Agent Groups. You cannot do this if you copy or import a System Profile.

You can disable System Profiles to find configuration problems easier. When disabled, the System Profile is skipped during the evaluation of Agent Mappings for all incoming AppMon Agents. Additionally, any Scheduled Tasks or Monitors are not executed until the System Profile is activated again.

See Enable or Disable a System Profile.

What happens to assigned AppMon Agents?

An AppMon Agent that is assigned to a System Profile is bound to it until it is restarted. If the System Profile is disabled or its Agent Mappings are modified so that it no longer matches the AppMon Agent, the Memory/Method Sensors and Sensor Configuration/Placement stay the same, even though the AppMon Agent is not assigned to this System Profile anymore.

This guards against incorrect configuration, but you have to restart the AppMon Agent to match to a different System Profile.

Sensors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is a sensor?

Hot Sensor Placement

Entry points

Auto sensors

User-defined sensors

Related pages

Sensor Configuration

Sensor properties

Java Agent

.NET Agent

Apache Agent

IIS Agent

NGINX Agent

PHP Agent

JavaScript Agent

zOS Agent

Host Monitoring Agent

Database Agent

OneAgent

Agent Group - Agent Mapping Advanced Settings

Watch and learn

Managing sensors

Advanced sensors

What is a sensor?

A *Sensor* is a small piece of code that is injected into the monitored applications at certain points to extract the relevant data. For example, a Sensor transports tagging information from one host to another when a web service call is executed. The AppMon installation includes a set of built-in Knowledge Sensor Packs for a wide set of application servers and frameworks. These Knowledge Sensor Packs are maintained by AppMon, but you can import/export and activate/deactivate them as needed.

Hot Sensor Placement

Whenever a class is loaded in a monitored application, the Agent sends this class over the network to the AppMon Collector for instrumentation. Sensors are injected if necessary. The class is sent back to the Agent if it has been manipulated. If a sensor is placed or removed while the monitored application is running, you must reload the affected classes to enforce re-instrumentation. This *Hot Sensor Placement* is provided as an explicit operation in Sensor maintenance.

Entry points

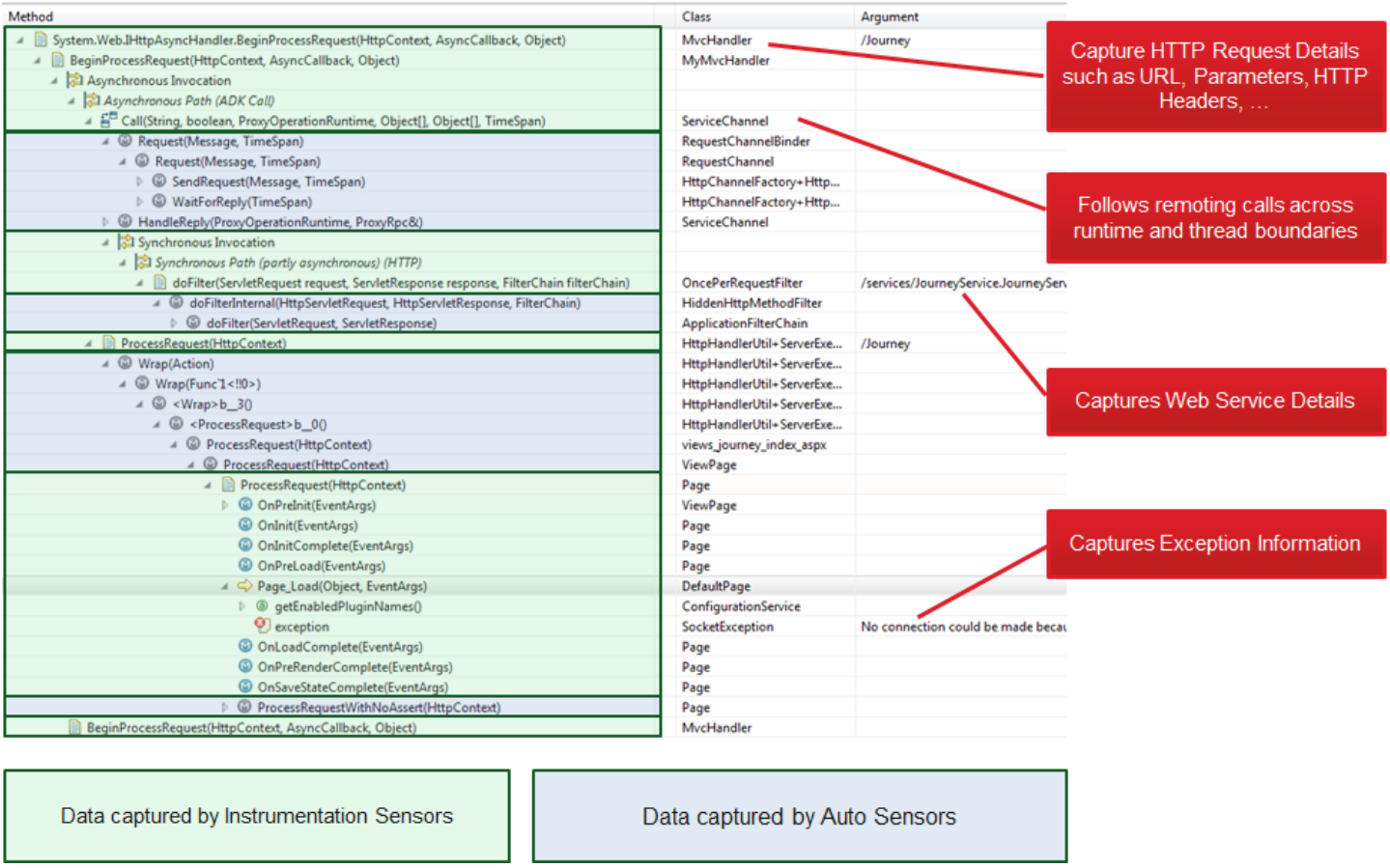
An *entry point* is the starting point of a new PurePath. It is defined by a placed and active sensor with the appropriate configuration. For most users, the Knowledge Sensor Packs (e.g. Servlet, Webservice, ASP .NET) already define the correct entrypoints. For more information, see Sensor Configuration.

Auto sensors

AppMon provides additional information on PurePaths without the need to do specific configuration. Auto Sensors automatically retrieve additional call hierarchy information and place this information on the respective PurePaths. This information includes class and method names, parameter and return types, and access modifiers. Auto Sensors are enabled by default.

Because Auto Sensors do not inject code into the monitored application, the timing information that is provided is not as accurate as the timing information delivered by explicitly placed Sensors.

The following figure illustrates the information captured by Auto Sensors.



Auto Sensor data

Auto Sensor resolution

Auto Sensors adjust automatically to ensure they do not incur more overhead than the percentage setting. The amount of detail captured depends on the complexity of the application or environment. You may need to adjust the resolution settings until you see the desired level of detail for a tolerable amount of overhead. See Agent Group - Agent Mapping Advanced Settings to learn how to adjust sensor resolution.

User-defined sensors

You can manually create additional sensors by defining *Sensor rules*. Sensor Rules specify a Sensor's location through class and method patterns. The rules also enable the Sensor to access data such as method parameters or return values. User-defined Sensors are typically used to specify additional Entrypoints or access specific information in a method's parameters. For more information, see Sensor Configuration.

Dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Charting

Dashboards - AppMon Client vs web

Event specific dashlets

Functional Health dashlet

Load Test Overview dashboard

Monitoring Overview dashboard

Other dashlets

Performance specific dashlets

AppMon client dashboards

Runtime Specific dashlets

Status overview dashlets

Transaction Flow dashlet

User Experience dashlets

Web dashboards

Web Page Performance Analysis dashboard

Overview

Dashboards provide a powerful tool for preparing diagnostic data for different kinds of use cases in the lifecycle of a software application. In the dashboards, you can filter, aggregate, and annotate PurePath-related information for experts in different fields.

AppMon currently support two types of Dashboards:

- AppMon Client Dashboards
- Web Dashboards

See Dashboards - AppMon Client vs Web for more information.

See related pages to get more information on Dashboards.

Dashboards - AppMon Client vs web

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

AppMon client dashboards

Web dashboards

This page compares AppMon client and web dashboards.

Many concepts are common to both AppMon Client and AppMon Web dashboards. For example, you can select the System Profile and time-frame to display, filter criteria, and customize both Client and Web dashboards. Although looks are quite distinct, tiles are for web dashboards what dashlets are for dashboards in the AppMon Client.

AppMon client dashboards

- The traditional way to serve information / insights are the dashboards in the AppMon Client.
- For AppMon 6.5 and later, many dashboards are only available in the Client and are more configurable, so it is often worth the seconds to:
  - Start a webstart client. (via https://<DynatraceServer>:8021,…)
  - Have a portable client available on portable media and connect it to the AppMon Server.
  - Install the client for continued use and the whole lot of configuration tasks.
- You can share dashboards by storing them on the AppMon Server.

Web dashboards

- More recently dashboards have become available in the browser, available instantly on any conceivable device.
- Fire them up in the browser at the blink of an eye without any installation necessary. (https://<DynatraceServer>:9911)
- They adapt well to now ubiquitous mobile devices.

- Pivotal dashboards like the transaction flow one are available now.
- You can share them across the web via a URL (and an account and permissions on that AppMon Server).

AppMon client dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Create a dashboard

Open a dashboard

Save a dashboard

Dashlets

Adding dashlets

Reuse and share dashboards

Drill down from dashboards

Filter dashboard and dashlet data

Customize dashboard names & descriptions

Link dashboards to a dashlet

View in kiosk mode

Related pages

Cockpit dashlet

Dashboard and dashlet properties

Dashboard vs dashlet

Getting data for a dashboard

Dashboards - AppMon Client vs web

Web dashboards

Watch and learn

Understanding dashboards

This topic describes the common tasks you can perform using dashboards and dashlets.

Create a dashboard

To create a dashboard just click the **New Dashboard**

icon, or press Ctrl+N.

Open a dashboard

You can open a dashboard:

- **From your local PC** by selecting **My Computer** in the **Open Dashboard** dialog box.
- **From a specific AppMon Server** by selecting the Dynatrace Server icon in the **Open Dashboard** dialog box.

Open default dashboards

Alternatively, you can open default dashboards and dashlets from the **Start Center** by clicking on the Start Center icon in the toolbar.

Learn more about the Start Center.

Save a dashboard

You can save a dashboard:

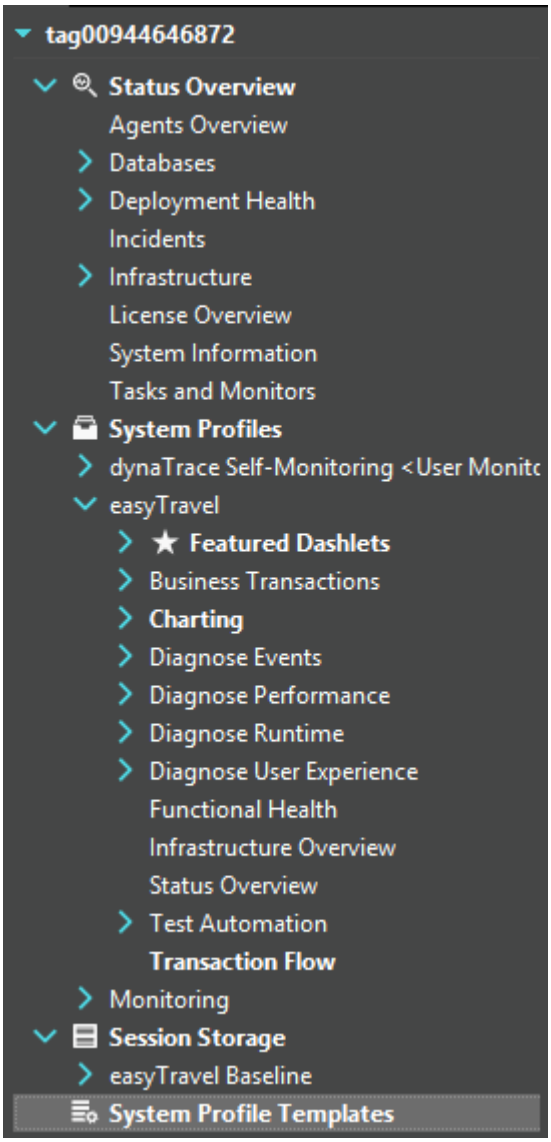
- **On your local PC** by selecting **My Computer** in the **Save Dashboard As** dialog box.
- **On a specific AppMon Server** by selecting the Dynatrace Server icon in the **Save Dashboard As** dialog box.

Dashlets

A Dashboard can contain several dashlets.

Cockpit sidebar

The cockpit sidebar provides an overview of all available data sources.



Cockpit dashlet

The **Cockpit** dashlet is a specialized dashlet for managing AppMon sessions and corresponding System Profiles for a specific data source. A data source is either an AppMon Server or the offline Performance Warehouse of the AppMon Client.

As part of its dashlet-specific properties, the **Cockpit** dashlet provides a filter for System Profiles. This filter can be used to configure Cockpits for user groups that are only interested in specific System Profiles.

1. Right-click the dashlet and select **Dashlet Properties** from the context menu
2. In the **Dashlet Properties** dialog box, select the **Filter** tab.
3. Selecting a checkbox for a System Profile **excludes** it from the Cockpit. Select or clear the checkboxes as needed, then click **OK**.

Adding dashlets

Double clicking on a dashlet in the Cockpit Sidebar normally adds the dashlet to the active dashboard. If **Add cockpit dashlets and drill down to new dashboard** is selected in the active dashboard's properties, the dashlet opens in a new dashboard.

Reuse and share dashboards

You can either store your dashboard documents on your computer or share them with other users via an AppMon Server. The AppMon Server allows you to configure viewing and editing permissions for your dashboards based on users and user groups that are managed by the Server. See User Permissions and Authentication.

If you share a dashboard with another user, you are only sharing a common view, independent from the actual data. A dashboard document does not contain any diagnostic data. It contains only the configuration of the view with reference to required data sources. A user that wants to reuse a saved dashboard either needs to have access to the same data sources (AppMon Servers) or local copies of the same sessions in the offline Performance Warehouse of the user's AppMon Client. If a dashboard is opened that references a data source that cannot be accessed, the user will be informed.

Drill down from dashboards

If you are exploring PurePaths, you can use the **Drill Down** context menu to navigate from a selected entry in a dashlet to a new dashlet with more detailed information. By default, the target dashlet is added to the same dashboard.

When you press the **Shift** key while selecting the drilldown target in the context menu the dashlet is added to a new dashboard.

You can override the default behavior for all drilldowns through the **Dashboard Properties** dialog box: On the **General** tab, select **Add cockpit dashlets and drill down to new dashboard**.

Filter dashboard and dashlet data

The AppMon Client provides the ability to set filters on dashboards and dashlets. Filters set on a dashboard are inherited by its dashlets unless a specific dashlet is configured to override the dashboard filters.

To define the dashboard filters, right-click the dashboard name and select **Edit Filter** from the context menu.

To define a dashlet filter, right-click the dashlet name and select **Edit Filter** from the context menu. The **Edit Filter** command is not available for some dashlets.

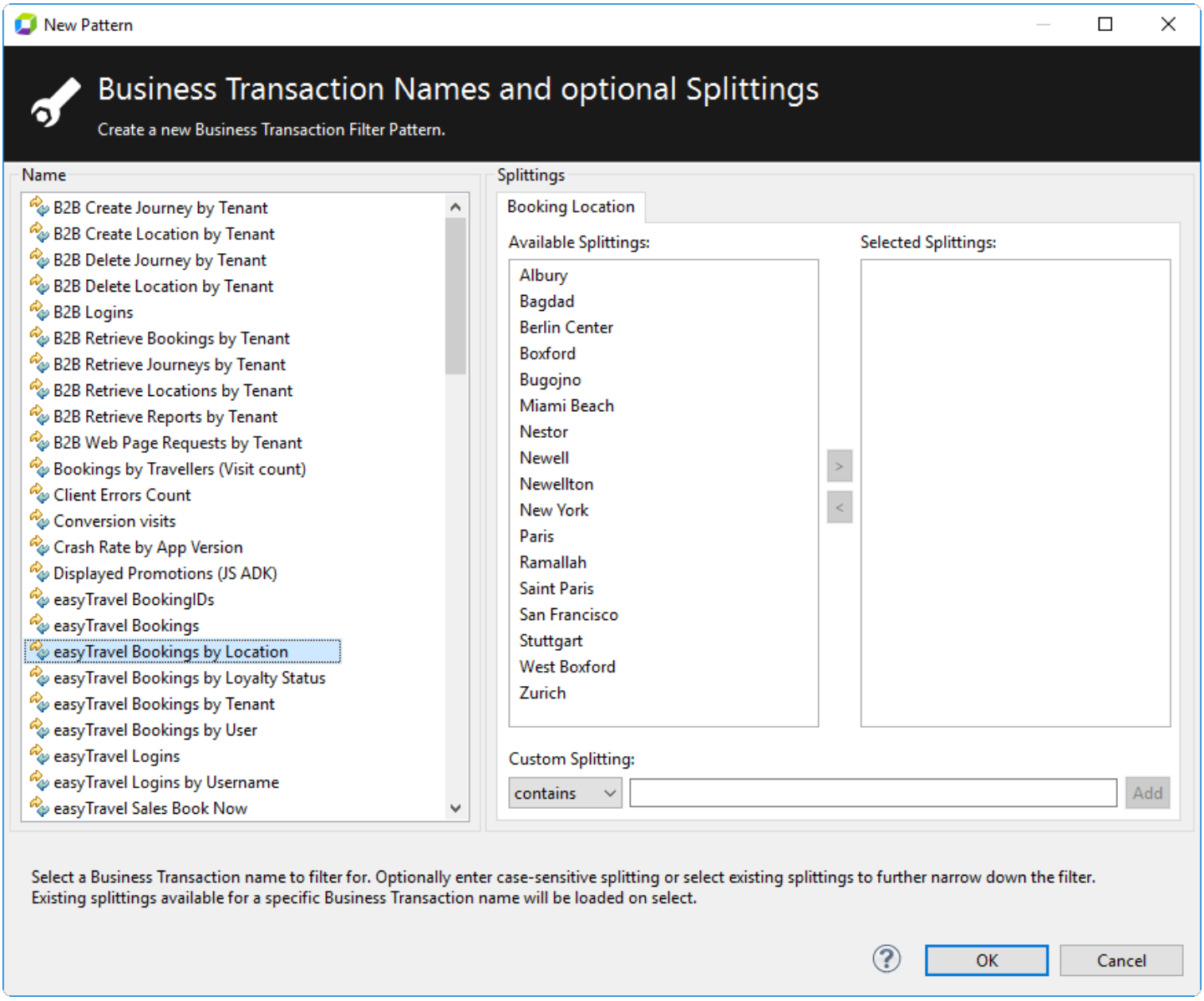
The filters provided by the AppMon Client are generally designed to filter out the underlying PurePaths. That means filter settings are not applied on visible data in views but instead are applied on the underlying PurePaths. The only exception is the Top X filter, which is exclusively applied on the visible data.

For example, when a **Methods** dashlet is opened, it has an **Agent** column in its table. When an Agent filter is applied on this dashlet, the Agent column will probably still show other Agents than those selected by the filter, because the filter is not applied on the view but on the underlying PurePaths. The Agent filter only ensures that the listed methods are part of PurePaths that contain the selected Agent.

The AppMon Client's filter mechanism consists of two domains: dashboard filters and dashlet filters. The filters available on the dashboard are predefined and contain the most commonly used filters. Filters set on the dashboard will only affect a dashlet if the dashlet supports that filter type and does not override the settings of that filter.

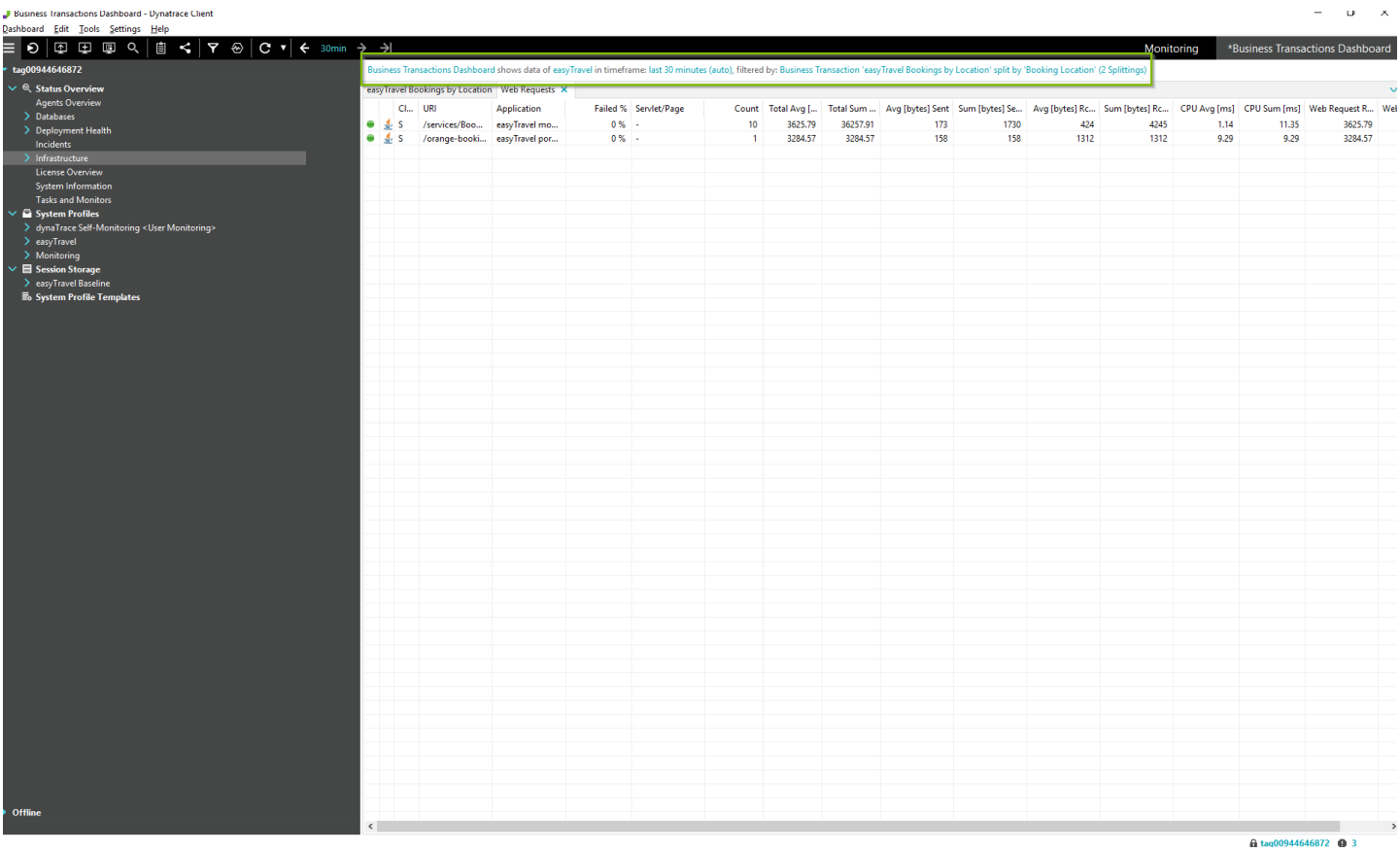
Filtering example

To investigate a single Business Transaction, for example bookings by location in an easyTravel session, search for the Business Transaction “easyTravel Bookings by location”. Then and select **Open in new dashboard** in the context menu. This creates a new **Business Transaction Dashboard** with a Business Transaction filter. Then configure the Business Transaction filter for the dashboard.



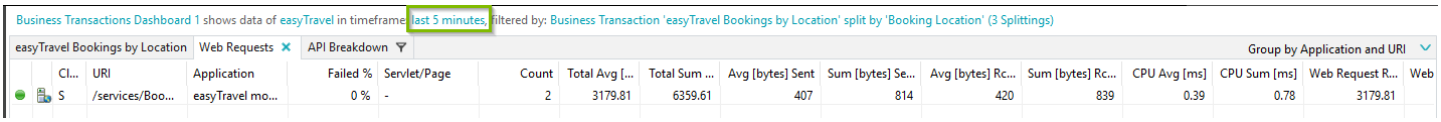
Every dashlet added to the dashboard inherits the filter setting by default.Because of this each shows only PurePaths covered by the selected Business Transaction.

Adding a **Web Requests** dashlet shows only Web Requests from PurePaths covered by the Business Transaction.



Dashlet Filtered by Default

In the **PurePath** dashlet, to further investigate the Business Transaction PurePaths from the last five minutes, define an additional Timeframe filter.



Drill down from a PurePath to an API breakdown. This adds an **API Breakdown** dashlet to the dashboard. Instead of using the custom filter settings for the dashlet, the AppMon Client has set the PurePath filter to limit the data shown in the **API Breakdown** dashlet to the single PurePath we have selected for drilldown before.

Dashlet-Specific Filter{figure}

Customize dashboard names & descriptions

In the dashlet and dashboard properties, you can change the names and add custom descriptions to single dashlets or complete dashboards. Custom dashlet names are displayed in the header of the dashlet. Descriptions are available as tooltips or in the custom dashlet header. To change the dashlet name and description, press F2 while the mouse pointer is inside the dashlet.

Text and Image dashlet

Another way to add additional information to a dashboard is to use Text and Image Dashlet dashlets. A Text and Image dashlet provides an easy way to add formatted headings and to integrate additional notes, company logos, and images into a dashboard.

Link dashboards to a dashlet

To document common navigation paths, single dashlets can be linked with specific dashboards. The links are configured in the **Dashboard Links** tab of the **Dashlet Properties** dialog box. Linked dashboards can be opened through the context menu of the dashlet under “Show in Dashboard”

View in kiosk mode

In *kiosk mode*, the AppMon Client displays in full screen. The cursor is hidden and the Client rotates through all open dashboards at an interval specified in seconds (the default interval is 30 seconds).

You can open the AppMon Client in Kiosk Mode by adding a command-line parameter on startup.

Argument	Value	Description
`<DT_HOME%>\dtclient -kiosk [interval]`	Optional: Positive integer value for the interval (in seconds, default 30, minimum 1)	Start the Client in kiosk mode. Optionally specify the interval in se displaying each screen. For example `-kiosk 10`. If the  **Application Overview** was active when you exited interactive m included in kiosk mode; otherwise, it does not display. You can use the Set up the Webstart Client in kiosk mode by inclu arguments in the Webstart URL: `http://localhost:8020/webstart/Client/cli argument=-kiosk&argument=10` See Client Com Arguments and Set up the Webstart Client for more information.

If you open and close dashboards while in kiosk mode, these dashboards are included in or excluded from the rotation.

Use the following keys to control the display:

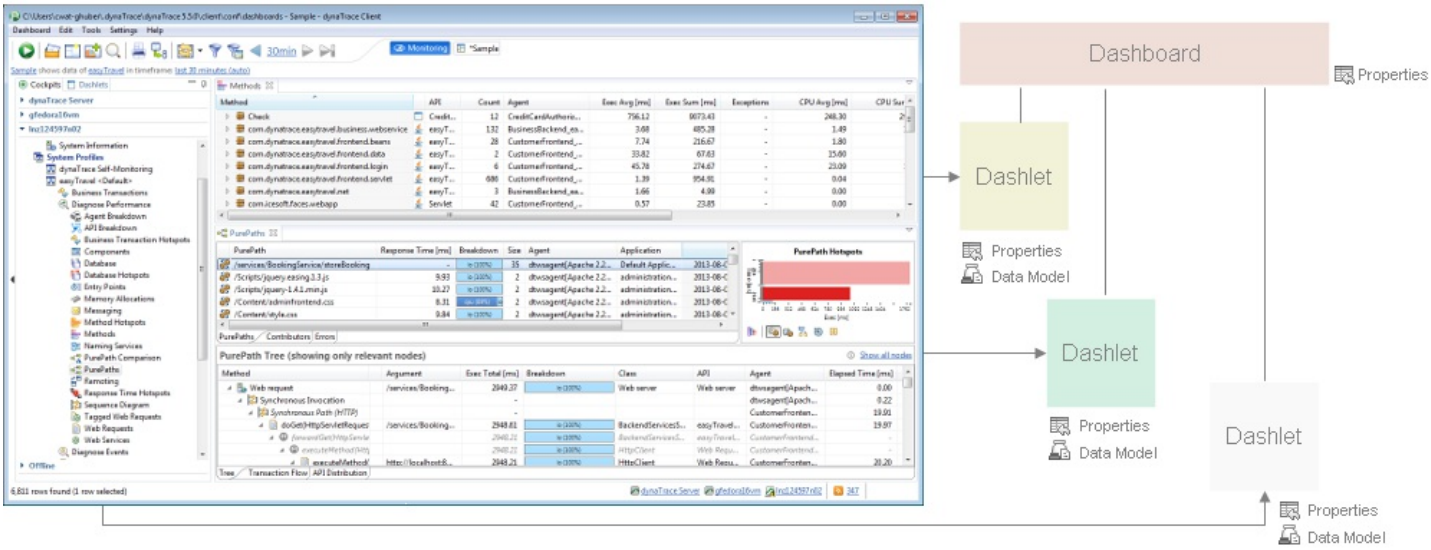
- ESC: Pause or resume the rotation.
- PAGE UP: Go to the previous dashboard.
- PAGE DOWN: Go to the next dashboard.
- F11: Toggle full-screen mode.

If user input such as a mouse movement is detected, the automatic rotation stops. It resumes After two minutes without user interaction.

Dashboard vs dashlet

A *diagnosis dashlet* defines a reusable and configurable view of one or more AppMon sessions. You can choose from a rich set of different diagnosis dashlet types, each focusing on specific aspects of diagnosis. Each diagnosis dashlet has its own data model that can be refreshed separately. A data model can be refreshed either manually or automatically at definable refresh intervals. The refresh behavior is part of the dashboard configuration. The refresh interval is configured for the dashboard, but each dashlet can be configured with a separate refresh interval.

A dashboard represents a specific view of one or more AppMon data sources.



Example dashboard

Example Dashboard

The data that is visualized on a dashboard is not stored in the dashboard, but in a session.

- The layout and arrangement of the dashlets.
- The common default data source(s) for the dashlets.

A dashboard is comprised of *dashlets* , the building blocks of a dashboard that contain specific types of data. In addition to defining which dashlets to display, a dashboard also defines:

Dashboard and dashlet properties

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Dashboard and dashlet common properties and inheritance

Design settings

Dashboard mode

Dashlet toolbar

Session comparison

Dashlet-specific properties

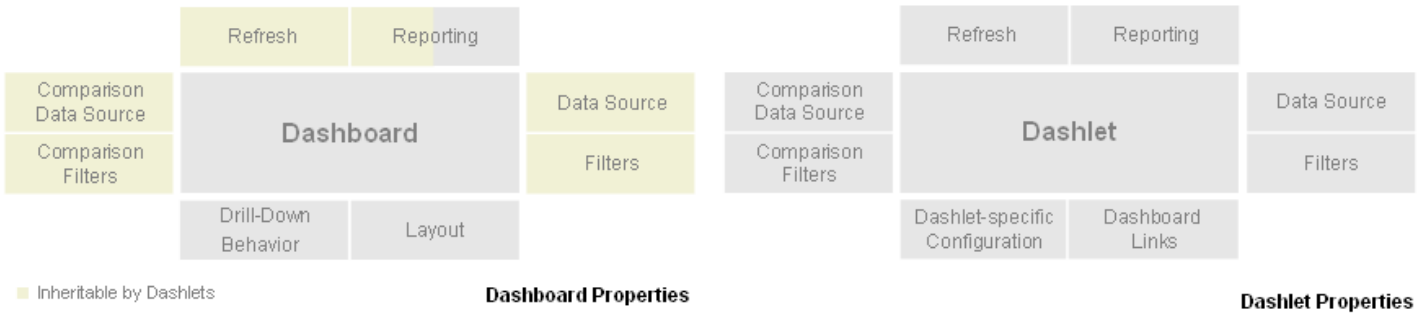
This page describes dashboard / dashlet properties / settings / layout and their relationship.

Dashboard and dashlet common properties and inheritance

A dashboard has a configuration layer above its dashlets that allows you to manage the layout and common properties of the dashlets.

A dashboard defines common properties that can be inherited its dashlets. A dashboard configuration defines default values for new dashlets and provides a means of changing common dashlet properties. A new dashlet inherits all the properties defined for the dashboard. Changes to inherited properties are automatically propagated to all dashlets in the dashboard. This simplifies the reuse of dashboard configurations for multiple applications.

The following figure illustrates the relationship of dashboard properties and dashlet properties.



Dashboard and Dashlet Properties

Access the dashboard's properties by selecting **Dashboard > Properties** or by selecting **Properties** from the dashboard's context menu.

Access the dashlet's properties by selecting **Dashlet Properties** from the dashlet's context menu.

For properties that can be defined for the dashboard and the dashlet, the dashlet inherits the dashboard properties. Each common tab in the **Dashlet Properties** dialog box includes an option to **Enable Dashlet specific <property>**. If this option is **not** selected, you can click the tab's **Edit Dashboard** link to change the settings in the corresponding **Dashboard Properties** tab. When you save the dashboard changes, the settings are automatically changed in the **Dashlet Properties**.

Design settings

The **Design** settings customize the dashlet headers. The options for displaying headers are:

- **Default**: Tabs across the top of the dashboard display the dashlet names.
- **Customized**: No tabs appear. The title and description of the active dashlet are displayed at the top of the dashlet. You can customize the font and alignment separately for the dashlet title and description.
- **No Header**: The active dashlet name is not displayed, and no tabs are displayed.

At the dashboard level (though not for individual dashlets), you can also select **Show in Night Mode**, described above in Dashboard Mode.

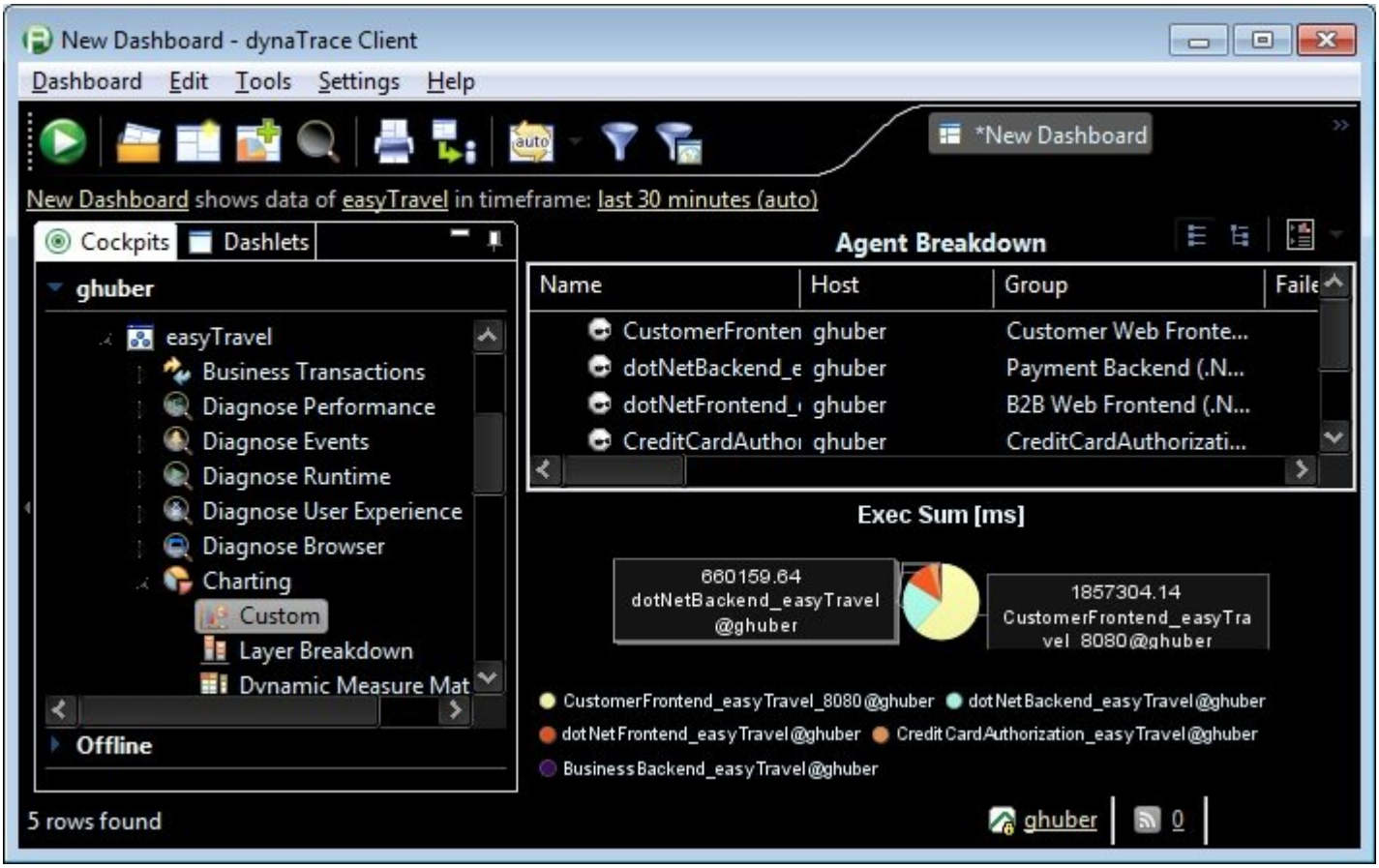
You can set a global style at the dashboard level that is inherited by all its dashlets. You can override these global settings with individual dashlet settings.

The **Customized** style integrates the dashlet with the AppMon Client background. You should use the **Design** settings instead of Text and Image Dashlet dashlets to create customized titles and descriptions. Keyboard Shortcuts are available for editing titles and descriptions inside the dashlets.

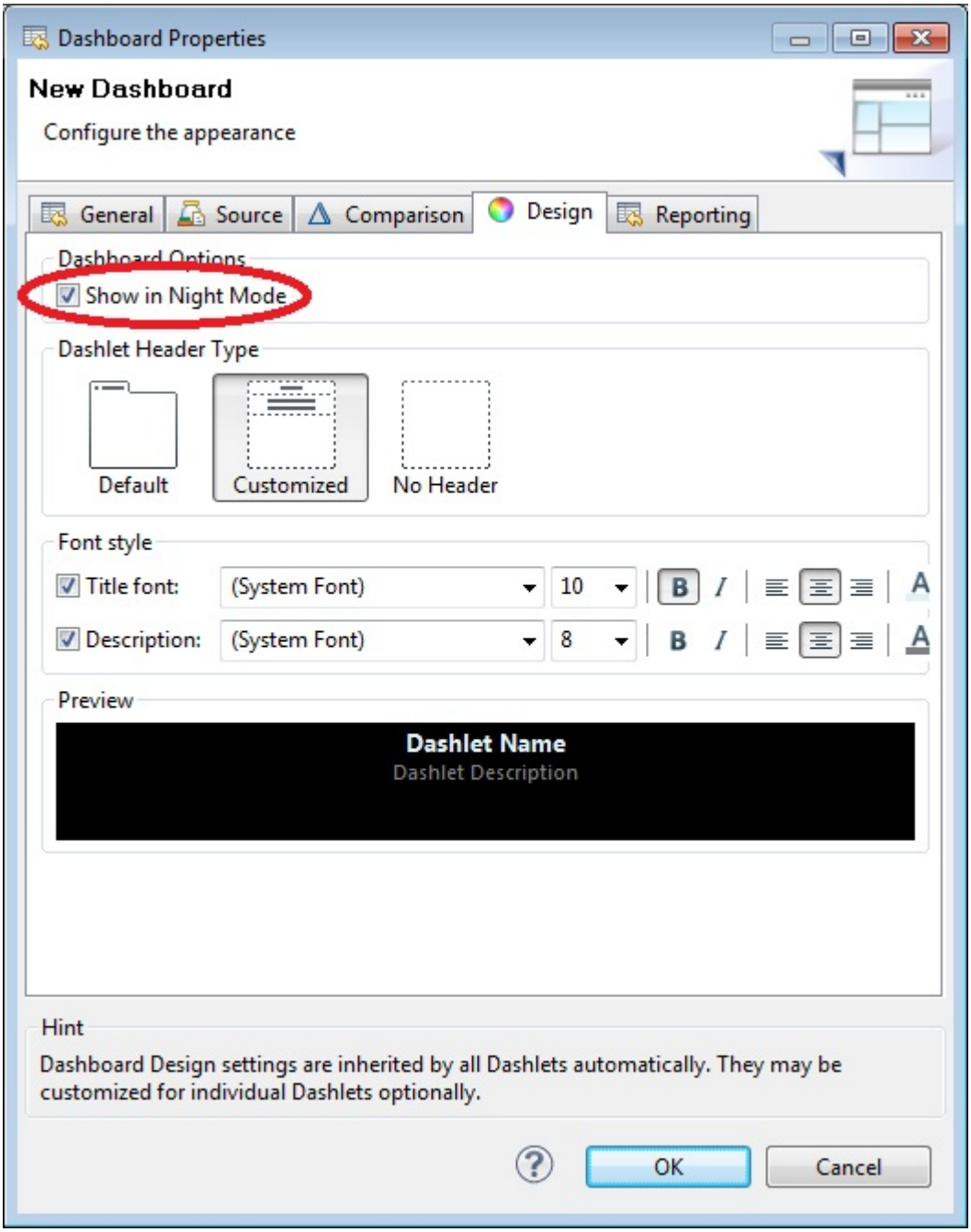
When the **Customized** style is applied, only the header of the active dashlets is visible. To switch the active dashlet, press Ctrl+H to toggle through the **Design** options until the tabs are visible, and then select the desired dashboard.

Dashboard mode

When you select **Use for Monitoring** in a newly created Dashboard, dashlets are added to the dashboard in *Night Mode* . Night Mode is enabled by default when you open a Deployment Health dashboard through **Start Center > Monitoring**.



Night Mode is a property of dashboards, not individual dashlets. To change the dashboard to Day Mode (Analysis Mode), right-click the dashboard tab and select **Properties** from the context menu. In the **Dashboard Properties** dialog box, select the **Design** tab and clear the checkbox for **Show in Night Mode**.



Dashlet toolbar

Most dashlets include a toolbar (right top)



to perform actions in the dashlet. This toolbar is only visible when the dashlet is in focus / when you hover over it.

Session comparison

See Session Comparison.

Dashlet-specific properties

The **Dashlet Properties** dialog box may contain tabs that are specific for that dashlet in addition to the tabs that are shared by the dashlet and dashboard.

Cockpit dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

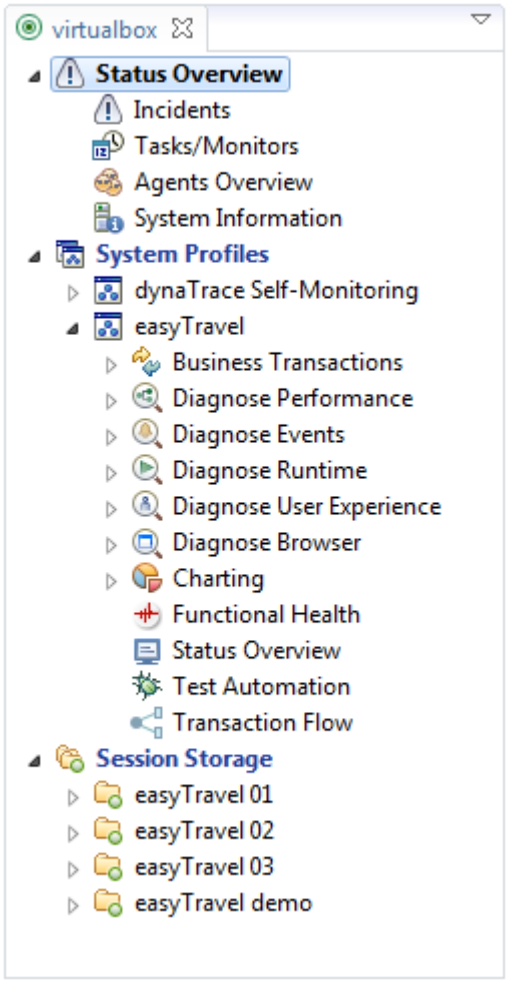
Overview

Offline sessions

Overview

As the figure shows, when you open the **Cockpit** dashlet, the dashlet tab displays the name of the connected AppMon Server. The tree is identical to the tree in the **Cockpits** tab of the sidebar.

Cockpit Dashlet



You can use the **Cockpit** dashlet in the same way as the Cockpit in the sidebar. In particular, you can perform the following tasks:

- Manage an AppMon Server. Right-click the **System Profiles** node and select **Manage Server**.
- Manage a System Profile. Right-click a System Profile and select **Edit System Profile**.
- Open profile-specific dashlets. Expand a System Profile tree. The lowest-level nodes are dashlets. Double-click a dashlet to add it to the dashboard.

Offline sessions

Depending on the dashboard settings, the **Cockpit** dashlet is connected to an AppMon Server so you can explore the live sessions and stored sessions on the connected server.

To use the **Cockpit** dashlet to work with offline sessions, override the dashboard **Source** setting in the **Dashlet Properties** dialog box. To open the dialog box, right-click the dashlet tab and select **Dashlet Properties**.

When you switch the dashlet's **Source** property, the dashlet tab changes to the selected source. Instead of System Profiles, it displays the **Local Sessions** tree.

Getting data for a dashboard

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Quick vs. Full Analysis

Analyzer progress visualization

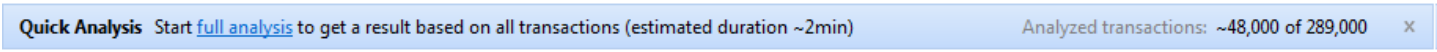
Analysis stopped due to PurePath limit reached

Analysis cache

This page describes AppMon strategies to quickly display relevant data in the dashboard of choice and how you can influence things.

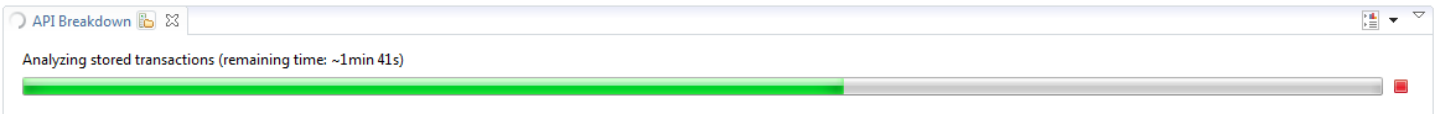
Quick vs. Full Analysis

A partial analysis is performed to provide instant feedback when opening or refreshing a dashlet is estimated to take a long time. (due to the amount of data and / or filter criteria) Then shown are only *representative* data from the selected time frame. To give an indication of the accuracy of the result, the number of analyzed vs. total transactions is displayed.
A quick analysis is never performed in dashlets where it is important to see every transaction, such as **PurePath**, **User Actions**, or **Visits**. When AppMon performs a quick analysis, the dashlet displays the message as shown in the following figure. Click **full analysis** to analyze all transactions.

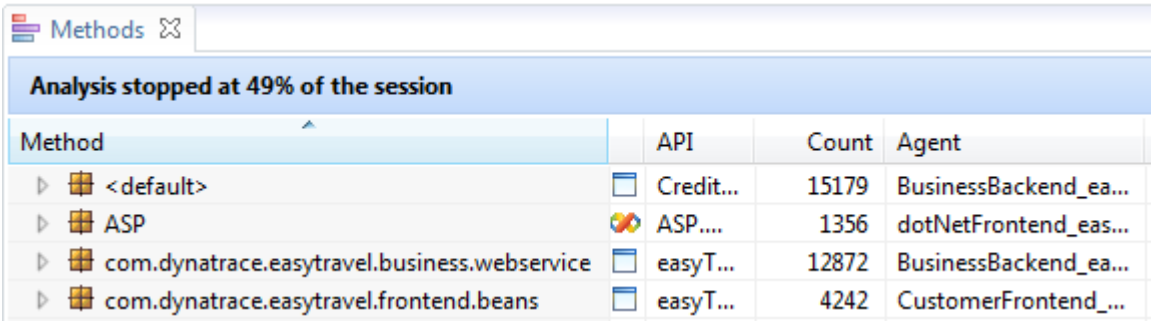


Analyzer progress visualization

While an analyzer job is running, its progress is visualized at the top of the dashlet.



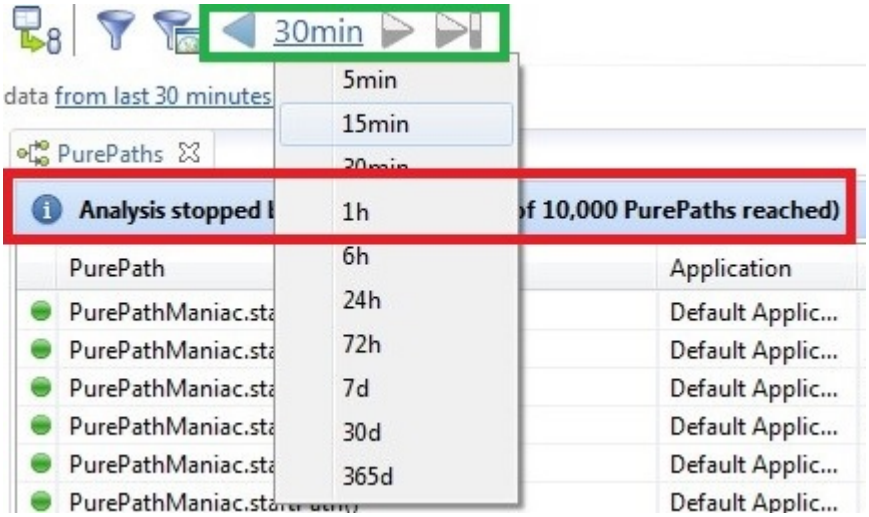
The results for the data analyzed so far are shown in the dashlet. A blue information bar shows how much of the session was analyzed when the analysis was stopped.



Analysis stopped due to PurePath limit reached

The analysis is stopped by the AppMon Server if the PurePaths limit is reached. This limitation is not related to the number of PurePaths you are able to store on disk. This PurePaths limit ensures that the AppMon Client does not run out of memory when displaying the data.

A blue information bar indicates that the analysis was stopped by the Server and shows the number of analyzed PurePaths. The recommended response is to switch to a smaller timeframe for analysis. Use the arrow buttons in the toolbar to navigate through the PurePath history.



Analysis cache

In addition to quick / partial, AppMon continuously does an analysis of the incoming data in the background and caches the results.

Web dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Dashboards for every use

Easy to collaborate

Know and control data context at all times

Understand the real meaning of the user's experience

Ensure infrastructure and environment health

Responsive design

Real-time data display

Shared web dashboards

Related pages

Access AppMon web

Applications web view

AppMon Web FAQ & troubleshooting

Built in web dashboards

Control access to AppMon Web dashboards

Custom web dashboards

Drilldown to the AppMon Client

Embed custom content into a web dashboard

Export and import web dashboards

Operations web view

Share web dashboards

User experience management

Watch and learn

Dynatrace Web

AppMon Web uses modern web technologies to display intuitive and visually engaging dashboards for end-to-end visibility of real user experience, application performance, and key infrastructure metrics. Designed for real-time data visibility by business and technical users, AppMon Web has the following features and benefits:

Dashboards for every use

Business and **App Owner** dashboards give you a big-picture view of where your application's users are and how they see application performance, while you can quickly spot and run down environment problem hotspots using the **Operations** dashboards. Share any web dashboard with any users all users to communicate issues and help you stay on the same page with others.

Zero-configuration built-in web dashboards display key metrics for your application performance and system health, errors and failures, insight into mobile usage, and CDN and third party performance.

See App Owner dashboards for more information.

Can't find a built-in web dashboard to fit your needs? You can create your own custom dashboards and pull in exactly the type of data you want. Arrange tiles in a custom dashboard any way you like for a personalized touch. Choose from a wide variety of charts and graphs and even tie in a social component to your data by including custom HTML or JavaScript from multiple sources.

Business dashboards

The **Business Analytics** dashboards gives comprehensive overviews of your system's real user experience, application performance, and user visits. Use it to understand the bigger picture of how user experience application performance. Drill down in the tiles on the dashboard and analyze specific site visits in detail to help pinpoint user experience issues.

The **Business Analytics** dashboard is always available to all users and displays by default when you start AppMon Web. Click or tap a tile to open a full screen view for a deeper-dive into related metrics and analytics data.

See User Experience Management to learn more.

Operations dashboards

The **Operations** dashboards provide a one-stop for keeping in touch with the health of your environments. The **Operations** dashboard and its detailed views include key infrastructure metrics to help you determine where your problem hotspots are, and provide a starting point for analysis and resolution. You can share any operations data to keep team members aware of where environment issues are impacting health.

See Operations dashboard to learn more.

Built-in web dashboards

Zero-configuration built-in web dashboards display key metrics for your application performance and system health, resource utilization for hosts, errors & failures, insight into mobile usage, and CDN & 3rd party performance. These dashboards also provide you with a great starting point for learning about the available tile types you can use in your custom web dashboards.

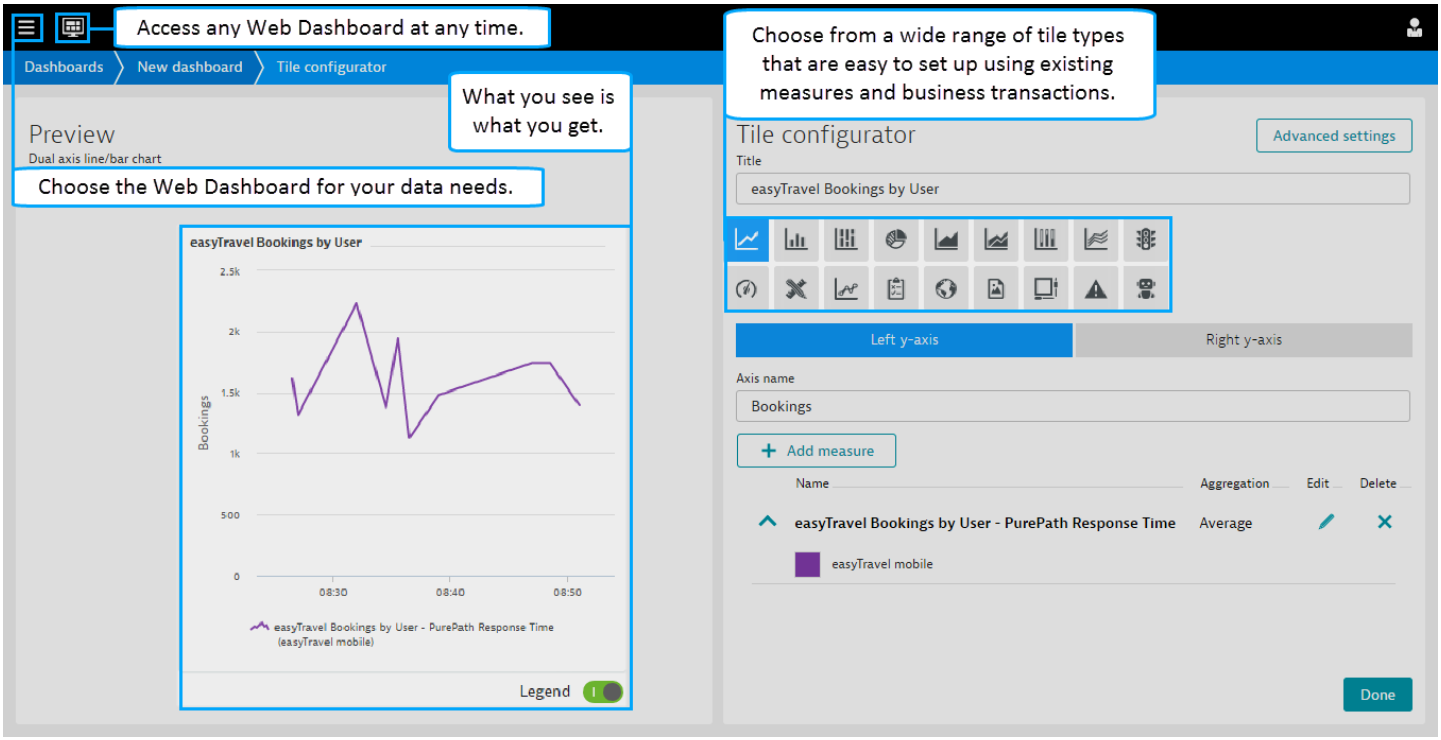
Learn more about built-in web dashboards.

Custom web dashboards for your data and your team

Create your own custom web dashboards to display just the data you need. Uncover new business insights and identify and prevent problems from occurring.

Use the drag-and-drop editor and choose from a wide variety of different tile types to create a customized dashboard layouts. Each tile can be sized to take up as much or as little space as you want.

Share your custom web dashboard with the rest of your team by providing a unique URL for your dashboard or by exporting a snapshot image of the dashboard to include in a report or presentation.



Learn more about custom web dashboards and sharing web dashboards.

Easy to collaborate

Working together just got easier. AppMon Web lets you share web dashboards on-the-go. Your APM champions can customize infographics to provide insight into user behaviors and experience, infrastructure health, and application performance and quickly communicate that insight to anyone in the organization. You can also export a group of dashboards so others can import them, or copy dashboards created by others, and modify then to suit your needs. This allows teams to work more easily together and tightens team member collaboration, helping team members stay on the same page.

Learn more about sharing web dashboards.

Know and control data context at all times

Whether reviewing user behavior analytics in a built-in dashboard, getting insight into user behavior, or combining data in a custom dashboard, you always know the scope of your data. Change or adjust the system profile, which applications to monitor, the data time frame on the go to fine tune the range of data. These are always visible and changeable from any dashboard, so you always know instantly the exact bounds of your data analysis.

Learn more about controlling data scope on AppMon Web dashboards.

Understand the real meaning of the user's experience

The tiles on the AppMon Web Business Analytics dashboard provide the full context of user experience for your applications. From these tiles, you can drill down to open more informative views that help you quickly understand and respond to user experience performance issues.

Learn more about User Experience Management.

Ensure infrastructure and environment health

As an operator you often have to check on the health of your environments, whether you are in a meeting or on your desk. Tiles on the AppMon Web Operations dashboard let you see all your healthy and unhealthy hosts and processes, and let you drill down and spot where problem hotspots come from, so you can determine if issues stem from infrastructure or application. Infrastructure and application alerts display for configured alert thresholds. See exactly what is generating alerts, then drill down to and act on or assign specific incidents.

Learn more about infrastructure and environment health.

Responsive design

The responsive design of AppMon Web lets you see visually-rich charts and infographics for your business' key metrics on a tablet, desktop browser, or extra-wide monitor in your boardroom.

Real-time data display

Web dashboards display live data that immediately update for the time frame you select and auto-refreshes every minute.

Note

Only *live session* data and measure data in the AppMon Performance Warehouse appears on web dashboards. *Stored session* data, which is a memory dump, thread dump, sampling or a former PurePath session that is stored on a hard drive is not available for display on web dashboards. This data is available for display in AppMon Client dashboards. Learn more about session data.

Shared web dashboards

You can access custom web dashboards created by other team members via a shared custom web dashboard URL. See Share Web Dashboards.

Access AppMon web

On this page

Technical requirements

Permissions necessary

Related pages

Permissions mapping

User experience management

Web dashboards

You can access AppMon Web from the AppMon Client. Select **Dashboard > New** and click **Dynatrace Web**. You can also use the following links in any supported browser:

- Go directly to the AppMon Web home with `https://<AppMonServer>:9911`.
- Use the AppMon Web link on Services home: `https://<AppMonServer>:8021` (SSL) or `http://<AppMonServer>:8020`. This page also contains links to the Webstart Client and other administrative features.

In case of port conflicts, you can change the AppMon web port in the AppMon Client. To do so, click **Settings menu > Dynatrace Server > Services item > Dynatrace Web** tab, and specify the new port in the **HTTPS listen port** field.

Technical requirements

AppMon Web is built for viewing on **tablet devices** and **desktop machines** that have a minimum browser resolution of 1024 x 768 pixels (e.g., iPad mini physical resolution 2048 x 1536 vs. browser resolution 1024 x 768). More details for your device can be found here.

AppMon Web is supported and tested with the following browsers:

- Edge 14+
- Firefox 50+
- Chrome 55+
- Safari 10+

Drilldowns to the **AppMon Client** require a running AppMon Client. See the AppMon Client requirements for more details.

Permissions necessary

Access to the web UI is granted by assigning certain permissions to a user role. The permissions configuration takes place in the **Users** item of the **Dynatrace Server Settings** dialog box. Permissions allow administrators to control access to certain dashboards, workflows, and data in the web UI. See User Permissions and Authentication for information how to create user accounts and assign permissions.

Permissions use cases

Limit users to only view web dashboards, without access to analytics data or workflows

The permission **View web dashboards** grants access to shared dashboards. It is also possible to prevent users from changing the datasource of the dashboard.

Engineers and developers

The permission **Allow application analysis in web UI** grants access to workflows related to application backend analysis, like PurePaths, transaction flow, errors. It also allows to view web dashboards.

Business analytics and User experience management

The permission **Allow business analysis in web UI** grants access to workflows related to business analytics, like world map, user analytics. It also allows to view web dashboards.

Operators and IT

The permission **Allow infrastructure monitoring in web UI** grants access to information relevant to operations, like hosts and processes, and databases. It also allows to view web dashboards.

> **Note**
>
> All of the permissions listed in use cases above are limiting access for certain System Profiles. The permission **Create web dashboards** is a basic permission, which assigned on its own, allowing users to create web dashboards for all System Profiles.

Default user roles

By default, AppMon has these roles for different levels of access control, sorted from less to more powerful:

- No Permission: doesn't have any permissions.
- Guest: allowed to view certain application-related dashboards, but cannot modify them or create new. Doesn't have access for self-monitoring dashboards. Can't modify System Profile and server settings. Typically suitable for business users, who wants to see the data, but doesn't need to perform any configurations.
- User: allowed to view certain application- and system-related dashboards, but cannot modify them or create new. Allowed to work with stored sessions. Can't modify System Profile and server settings. Typically suitable for experienced business users.
- Power User: allowed to view and modify certain application- and system-related dashboards. Can modify System Profile, can't modify server setting. Typically suitable for developers, who should have extended access, but should not affect server settings.
- Administrator: has all the permissions

See Permission mapping for more information about permissions and full list of permissions for these roles.

AppMon Web secure login requires at least the permission **View web dashboards** for at least one of the installed systems profiles. This permission is included in the default Guest, and all more powerful user roles.

To view information, marked as confidential, the **Read Confidential Strings** is required. This permission is included in the default User, and all more powerful user roles.

Built in web dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Web dashboards

Access AppMon web

Share web dashboards

AppMon Web FAQ & troubleshooting

Built-in web dashboards give you a starting point for learning about the available tile types you can use in your custom web dashboards. Refer to the description of tiles and out-of-the-box measures used in each built-in dashboard for specific guidance in setting up your web dashboards.

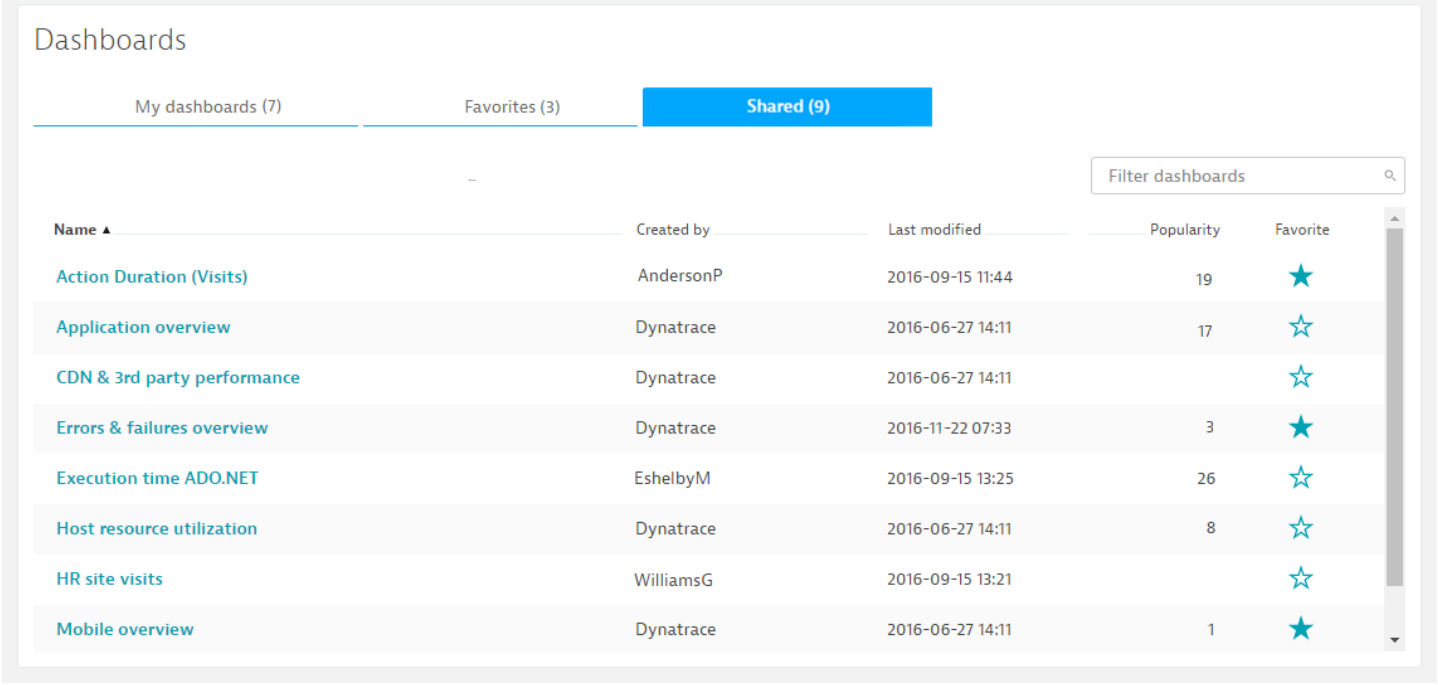
Built-in Web dashboards are automatically shared so all users can access them.

In AppMon Web, click the **Dashboards**

button, then click the **Shared** tab to see a list of the built-in dashboards. The built-in dashboards in the list have the name **Dynatrace** in the Created by column and include the following:

- Application overview
- CDN & 3rd party performance
- Errors & failures overview
- Host resource allocation
- Mobile overview
- User behavior analytics

> **Tip**
>
> The shared dashboards list also contains dashboards that are shared by other users in your organization.


Dashboards
My dashboards (7)
Favorites (3)
Shared (9)
Filter dashboards
Name ▲
Created by
Last modified
Popularity
Favorite
Action Duration (Visits)
AndersonP
2016-09-15 11:44
19
Application overview
Dynatrace
2016-06-27 14:11
17
CDN & 3rd party performance
Dynatrace
2016-06-27 14:11
Errors & failures overview
Dynatrace
2016-11-22 07:33
3
Execution time ADO.NET
EshelbyM
2016-09-15 13:25
26
Host resource utilization
Dynatrace
2016-06-27 14:11
8
HR site visits
WilliamsG
2016-09-15 13:21
Mobile overview
Dynatrace
2016-06-27 14:11
1

Custom web dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Before you begin

Create a custom web dashboard

Related pages

Add and edit measures

Tile types

Web dashboard timeframe and resolution

Web dashboards

Access AppMon web

Share web dashboards

AppMon Web FAQ & troubleshooting

Analyze and monitor big data solutions

Overview

You can create custom web dashboards to visualize the data most important to you and your team. Click the dashboards

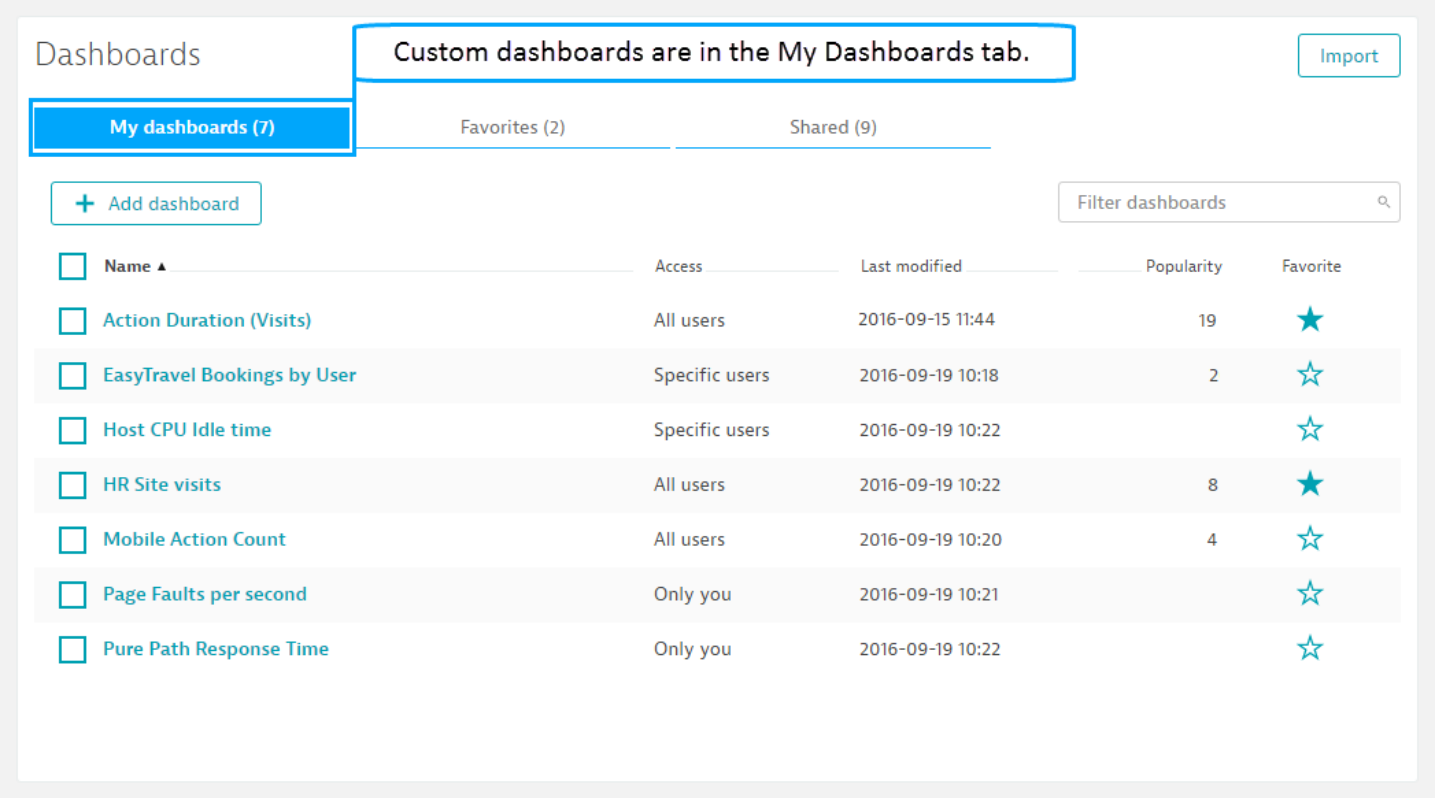


button and access your custom web dashboards in the **My dashboards** tab.

Custom web dashboards are private unless you share them with others by providing a unique URL for each custom web dashboard. Opening a web dashboard via a shared link requires a Dynatrace user account to login. Learn more about sharing dashboards.

Other important features and considerations include:

- Custom web dashboards display live session data only. They do not display stored session data.
- A global time frame setting is used for all tiles on a dashboard. Auto refresh rate is set a 1 minute.
- 40 tiles allowed per dashboard. Tile size is configurable.
- Each tile can contain up to 40 measures. In case of splitting each part counts as one measure.



Before you begin

Before creating a custom web dashboard, identify and create the measures or business transactions that contain the data you want to display on your dashboard. Create measures and business transactions using the AppMon Client.

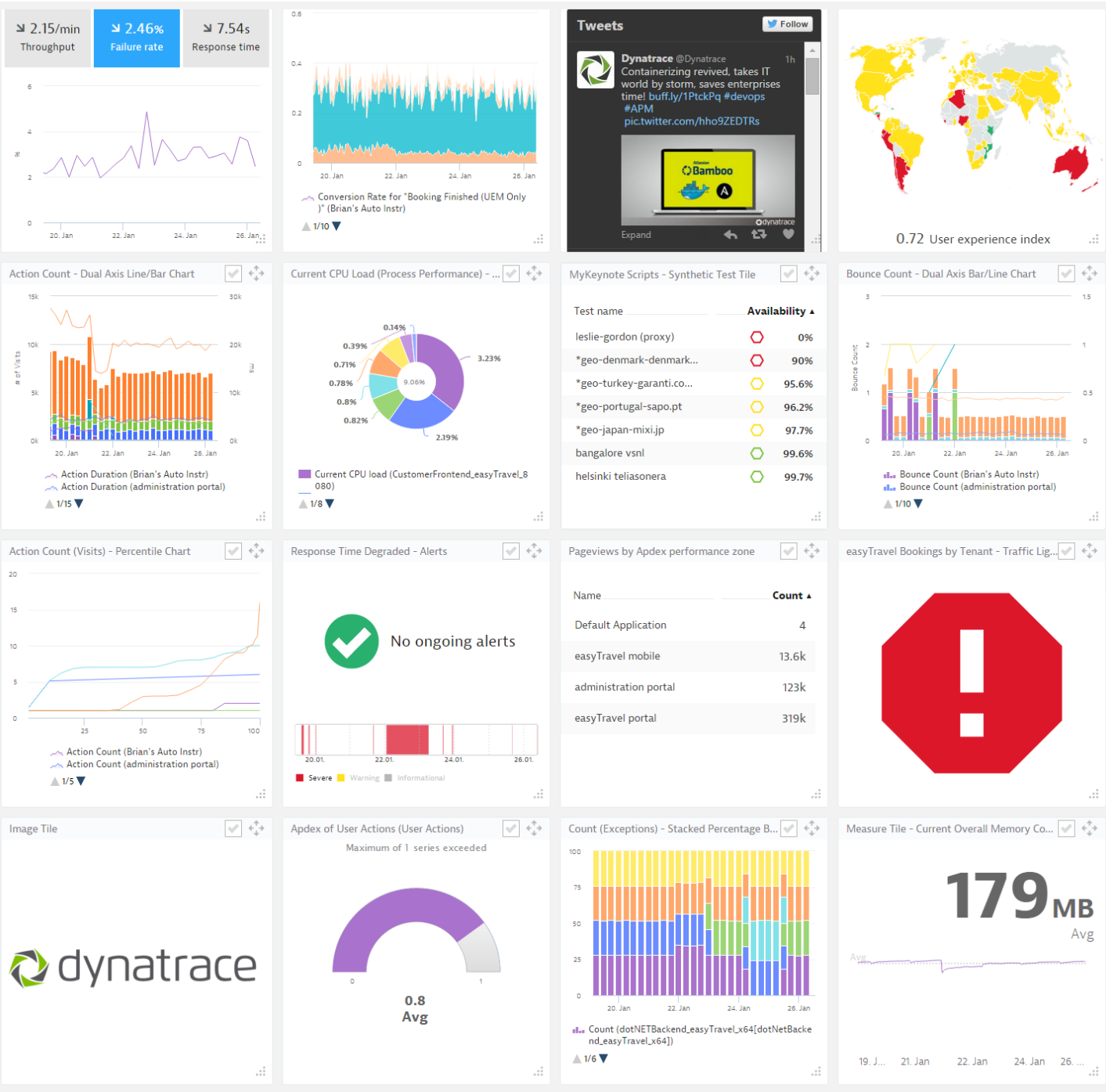
What's a measure and a Business Transaction?

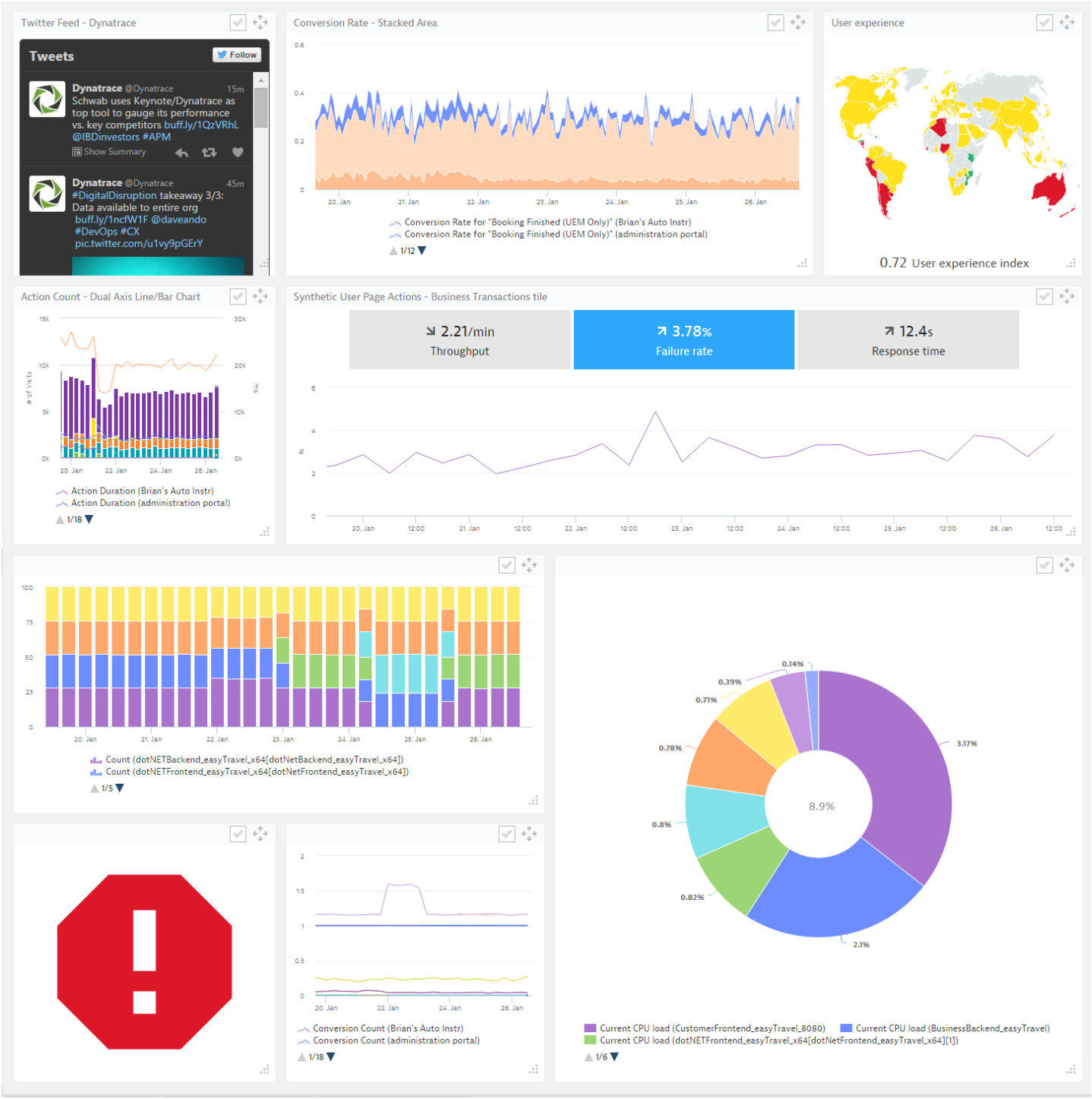
Measures in AppMon are metrics, or data points, that are collected and recorded in the AppMon Performance Warehouse either periodically or based on transactions. Measures are used for long-term charting, trend analysis, and as the basis for configuring incidents.

A *business transaction* is a set of automated interactions between IT systems that execute a business task or business process. In AppMon, a business transaction is a categorization of PurePaths defined by filter or grouping parameters. When a PurePath matches those parameters, one or more evaluation metrics are calculated to create the result set of the business transaction.

When designing your dashboard layout, you have the flexibility to arrange and resize up to 16 tiles in a grid layout. Tiles can be resized independently from other tiles to optimally display data. The dashboard grid layout supports a maximum width of four tile columns and a maximum height of four tile rows. Tiles cannot overlap.

Customizable Layout - Drag and Drop to Arrange & Resize Tiles


2.15/min
Throughput
2.46%
Failure rate
7.54s
Response time
Tweets
Follow
Dynatrace @Dynatrace 1h
Containerizing revived, takes IT world by storm, saves enterprises time! buff.ly/1PtckPq #devops #APM pic.twitter.com/hho9ZEDTRs
Expand
0.72 User experience index
Action Count - Dual Axis Line/Bar Chart
Action Duration (Brian's Auto Instr)
Action Duration (administration portal)
Current CPU Load (Process Performance) - ...
Current CPU load (CustomerFrontend_easyTravel_8080)
MyKeynote Scripts - Synthetic Test Tile
Test name
Availability
leslie-gordon (proxy) 0%
*geo-denmark-denmark... 90%
*geo-turkey-garanti.co... 95.6%
*geo-portugal-sapo.pt 96.2%
*geo-japan-mixi.jp 97.7%
bangalore vsnl 99.6%
helsinki teliasonera 99.7%
Bounce Count - Dual Axis Bar/Line Chart
Bounce Count (Brian's Auto Instr)
Bounce Count (administration portal)
Action Count (Visits) - Percentile Chart
Action Count (Brian's Auto Instr)
Action Count (administration portal)
Response Time Degraded - Alerts
No ongoing alerts
Severe Warning Informational
Pageviews by Apdex performance zone
Name
Count
Default Application 4
easyTravel mobile 13.6k
administration portal 123k
easyTravel portal 319k
easyTravel Bookings by Tenant - Traffic Lig...
Image Tile
dynatrace
Apdex of User Actions (User Actions)
Maximum of 1 series exceeded
0.8
Avg
Count (Exceptions) - Stacked Percentage B...
Count (dotNETBackend_easyTravel_x64[dotNetBackend_easyTravel_x64])
Measure Tile - Current Overall Memory Co...
179 MB
Avg



Create a custom web dashboard

1. Click the dashboards

   button and access your custom web dashboards in the **My dashboards** tab
2. Click the **Add dashboard** button.

3. Name your dashboard, and then select the System Profile and data timeframe. The timeframe determines the measure data that appears on your dashboard. This value is applied globally to every tile.
4. Optionally, click the add logo icon  next to the **Dashboard name** box to link to the URL of your favorite custom logo. The logo replaces the logo icon.
5. Click the **Edit title** link on the default tile. The **Tile configurator** appears.
6. Configure the tile. See Tile Types, Editing a Measure, and the Timeframe and Resolution for details about the settings for each tile type.
7. Click **Done** to close the **Tile configurator** and return to the dashboard editor.
8. Click **Add tile** to add additional tile to your dashboard.
9. Click **Done** in the top right to see your dashboard at full resolution with the maximized tile view feature enabled.

Your can make copies of custom dashboards for your own use, and share custom dashboards with others. Select the dashboard you want to copy from your custom dashboards list in the **My dashboards** tab, click the **More Actions** button  and choose **Copy**. In the box that appears, name the dashboard copy and click **Copy and open**. The Dashboard copy show in the **My** tab.

> **Tip**
>
> The name of the dashboard to copy defaults to [copied dashboard name] - Copy.

See Share Web Dashboards for details on sharing custom dashboards with others.

Tile types

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Dual Axis Line/Bar Chart & Dual Axis Bar/Line Chart

Stacked Percentage Bar Chart

Pie chart

Area Chart / Stacked Area Chart

Meter chart

Percentile Chart

Traffic Light Chart

Gauge Chart

Measure

Business Transaction

Business Transaction List

World Map

Image

External content

Alerts

Synthetic tests

Text and measures

Set up a Business Transaction List tile

Related pages

Web dashboards

Custom web dashboards

Add and edit measures

Web dashboard timeframe and resolution

Access AppMon web

Share web dashboards

AppMon Web FAQ & troubleshooting

The following list describes the available tile types that you can add to a custom web dashboard.

Dual Axis Line/Bar Chart & Dual Axis Bar/Line Chart

Visualize a wide variance in the values of two data series or mixed types of data.

For the **dual axis line/bar chart**, the left y-axis displays the value of the data series as a line chart and the right y-axis as a bar chart. For the **dual axis bar/line chart**, the opposite is true.

Stacked Percentage Bar Chart

Compare parts in a series as a percentage of each value that a part contributes to the total series.

Pie chart

Plot a single data series where none of the values are zero or less.

Area Chart / Stacked Area Chart

Area chart: Visualize a series on top of each other as independent values, which is useful for visualizing the relative difference between multiple data series.

Stacked bar chart: Visualize the percentage of each part that comprises a data series.

Meter chart

Visualize and analyze data in a relative comparisons between measures.

Percentile Chart

Analyze a data series as a percentile from 0 to 100 for a selected time frame.

Tip

If the chart has many measures, use a logarithmic y-axis to increase readability.

Traffic Light Chart

See the result of a measure threshold conveyed by traffic light colors that represent the result:



OK: no threshold violation detected.



Warning: represents a minor threshold violation within a selected timeframe.



Problem: represents the most severe threshold violation within the selected timeframe.

Notes

- The selected measure must have a threshold configured for it to use the traffic light chart.
- The threshold violation state is calculated against the configured aggregation.

Gauge Chart

Chart a single configured measure result represented in a combined doughnut chart and pie chart.

Measure

Visualize data in a relative comparisons between measures.

Business Transaction

See measure results (response time, failure rate, and throughput) for a selected business transaction along with a line graph for each.

This interactive tile contains three tabs, one for each result measure. You can set which tab appears by default by selecting the result measure for the default.

Notes

- If a result measure was not configured for the selected business transaction, an **N/A** message appears on the respective tab to indicate that data is not available.
- This tile does not support the appearance of custom measures of a business transaction.

Business Transaction List

See data captured by a business transaction in a table view sorted by splitting value, result measures, and aggregation types (such as sum, average, maximum). See Set up a Business Transaction List Tile to learn how.

World Map

Gain high-level insight into overall global performance by visualizing visits or user actions within a geographical representation of the world. This tile represents a real-time view of user experience for the applications of the selected and time frame.

Image

Display an image defined by a URL linked to any type of image format. You can also link a URL to an image that when clicked opens a new tab displaying the target URL.

> **Tip**
>
> If the size of the image is greater than the tile size, then the image is resized to fit the tile size.

Use the image tile to link to your favorite shared dashboards. See Share Web Dashboards to learn how.

External content

Lets you see information and content from externally linked web URLs in a web dashboard tile.

To control the click behavior:
Show detail view: Displays the contents of the linked URL in a web dashboard tile. When the tile is clicked, the detail view expands to size of the dashboard and displays the content of the linked URL.
Open in a new window: Opens the linked URL in new browser tab or window.

See Embed Custom Content Into a Web Dashboard to learn how.

Alerts

Get an at-a-glance indication of the result of one or more configured incident conditions.

Synthetic tests

View and analyze data from selected AppMon or Keynote Synthetic Monitoring test scripts runs.
See Synthetic Monitoring Integration for more information.

Notes

- If the connection to the appropriate Synthetic Monitoring server is not configured, the **Use webstart client** button appears in the Tile Configurator pane. Click this button to download the `client.jsp` file. When downloaded, open the file to install an instance of the AppMon Client . Use that instance of the AppMon Client to configure Synthetic Monitoring integration. SeeSynthetic Monitoring Configurationto learn how.
- If you do not see data in the Synthetic tests tile or if you see invalid data, you may have not configured the Synthetic Monitor Server settings in the correct instance of the AppMon Client. Ensure that you have used the AppMon Client installed from the downloaded `client.jsp` file.

Text and measures

View the value of a measure, along with the custom text. See Text and measure tile for more information.

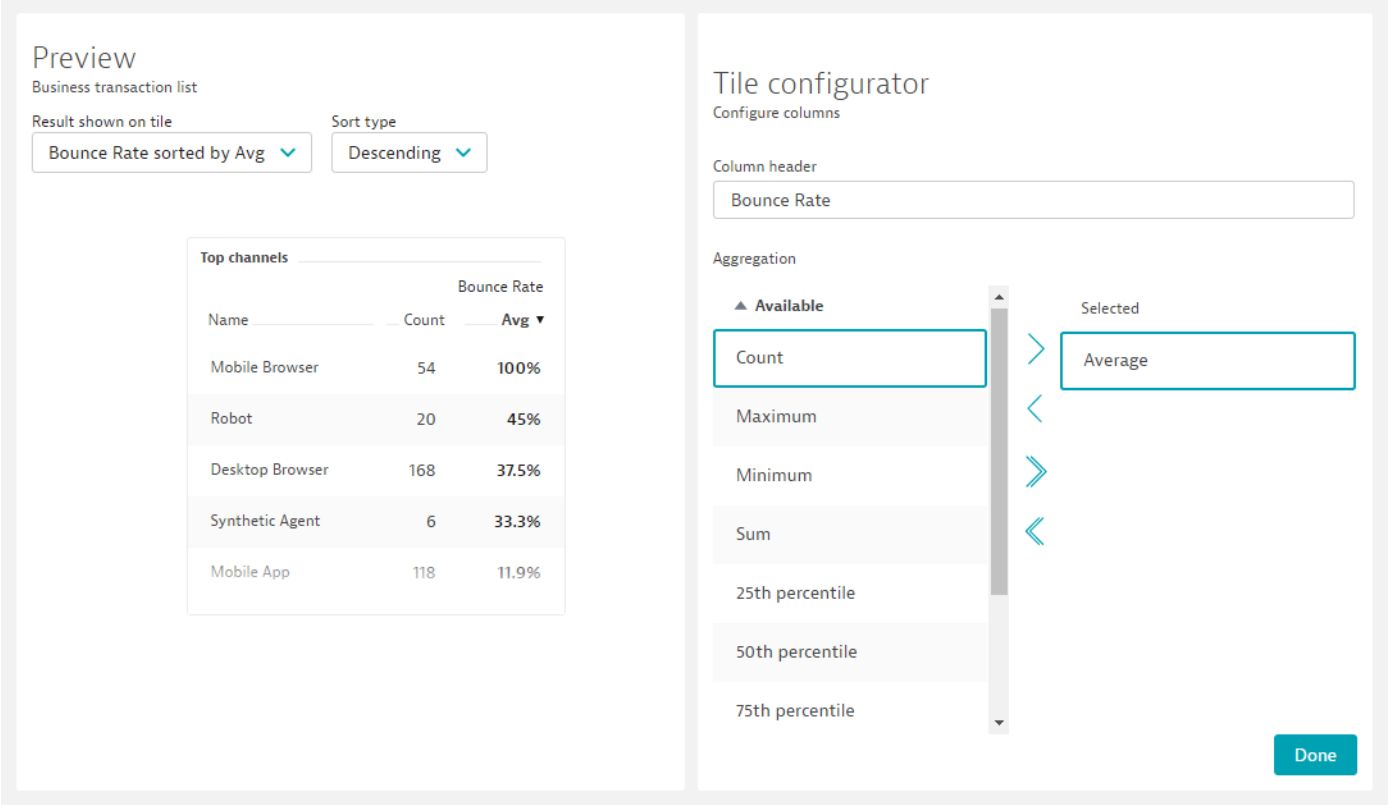
Set up a Business Transaction List tile

1. In the Tile configurator, click **Select a business transaction**, and then search or browse for a business transaction to display on the tile.

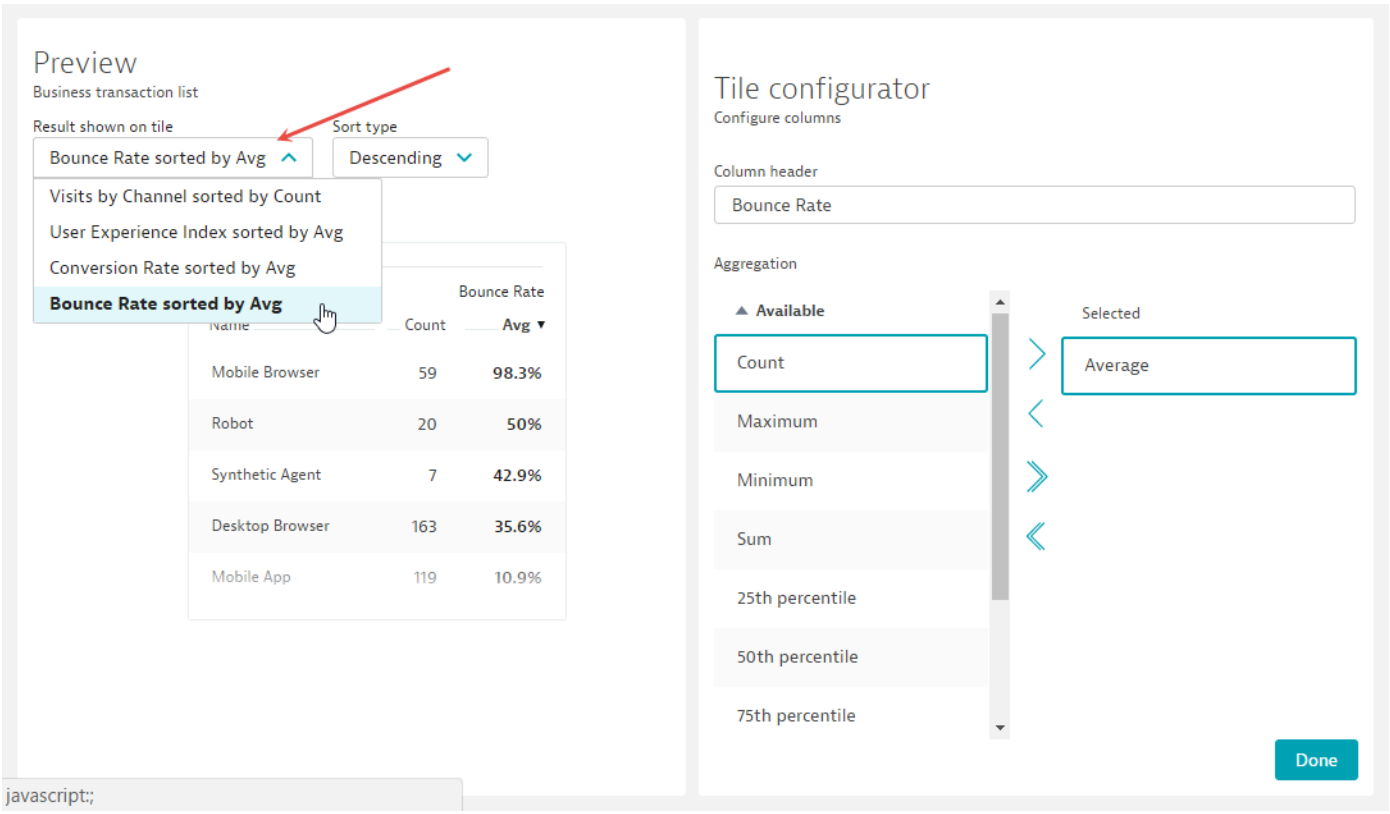
2. Next, choose a splitting value to increase the granularity of data that appears under the **Splitting** column on the tile. For an aggregate view, keep the default value of **No splitting**.

3. Optionally, choose one or more result measures to add additional data columns on the tile. Next, choose one or more aggregation types (sum, average, maximum, minimum, count) for each result measure. Each selected aggregation type appears as a subcolumn under the main result measure column header.

   In the following example, the aggregation type **Average** is selected for the **Bounce Rate** result measure and appears under the **Bounce Rate** column header, as shown in the preview tile below.

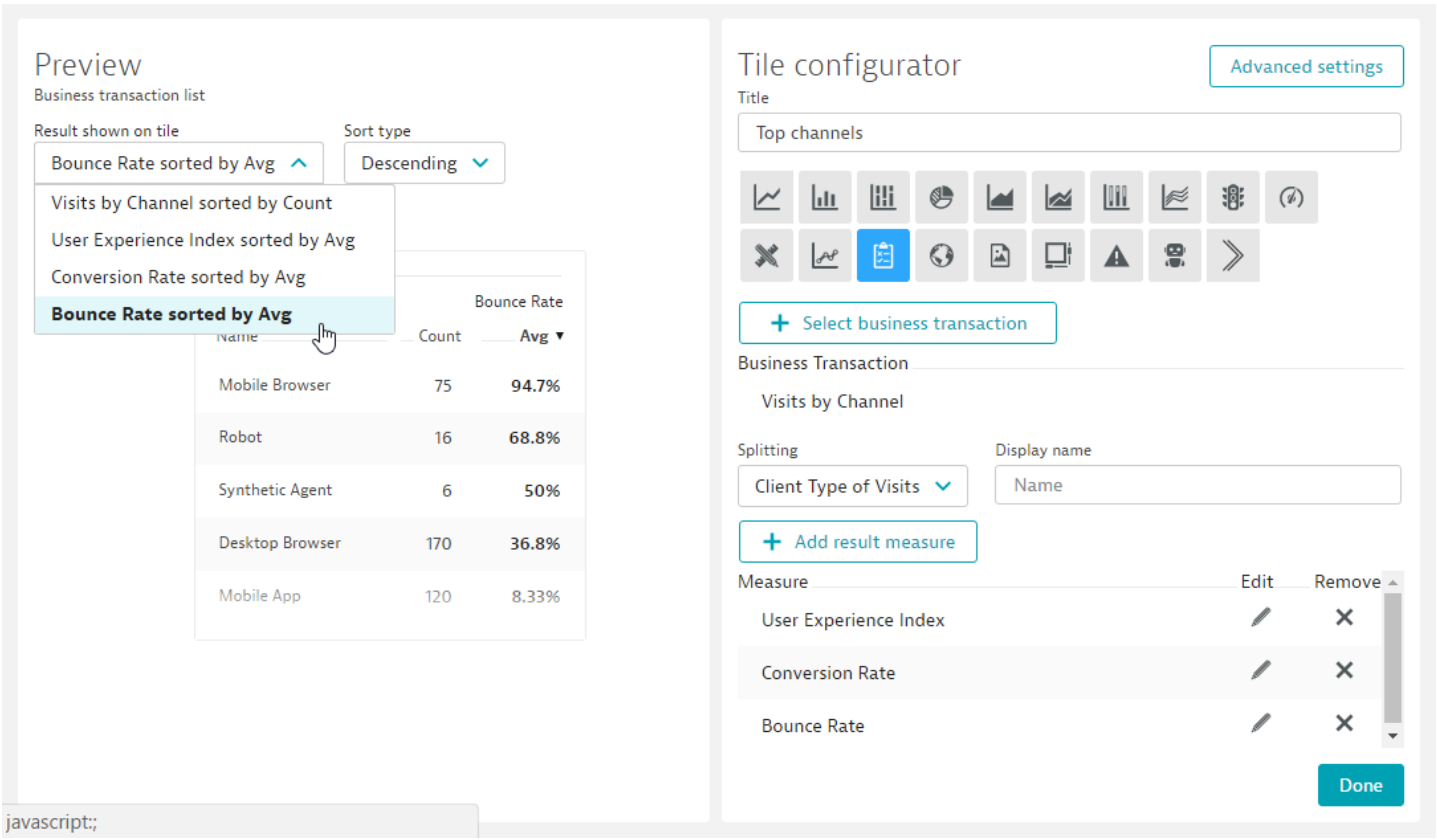


4. To complete the set up, you have the option of selecting the default sort order for data appearing on the tile. In the drop down list at the top of the preview tile, choose the result measure and aggregation type by which to sort the data.



In the following example, the Top channels tile used in a copy of the User behavior analytics built-in dashboard shows the total count for the Visits by Channel business transaction. It also shows data columns for the result measures for User Experience Index, Conversion Rate, and Bounce Rate.

These additional data columns are set up by selecting the measures of the same name (User Experience Index, Conversion Rate, and Bounce Rate), and then choosing the aggregation type, such as **Average** and **Maximum**.


Preview
Business transaction list
Result shown on tile
Bounce Rate sorted by Avg
Sort type
Descending
Visits by Channel sorted by Count
User Experience Index sorted by Avg
Conversion Rate sorted by Avg
Bounce Rate sorted by Avg
Bounce Rate
Count
Avg
Mobile Browser 75 94.7%
Robot 16 68.8%
Synthetic Agent 6 50%
Desktop Browser 170 36.8%
Mobile App 120 8.33%
javascript:;
Tile configurator
Advanced settings
Title
Top channels
Select business transaction
Business Transaction
Visits by Channel
Splitting
Client Type of Visits
Display name
Name
Add result measure
Measure
Edit
Remove
User Experience Index
Conversion Rate
Bounce Rate
Done

Add and edit measures

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Add and edit a measure

Measure aggregation methods

Measure splittings

Related pages

Custom web dashboards

Web dashboard timeframe and resolution

Share web dashboards

AppMon Web FAQ & troubleshooting

FAQ - measures and charting

Measures are the data source for the various chart tiles. You can select a single measure or multiple measures provided the data contained in the measure does not exceed the display capability of the tile type.

Add and edit a measure

1. Open the Tile Configurator, and then click **Add Measure** (
   

).
2. Search or browse for the measure(s) to add, and then click the **Done** button at the bottom of the list of measures.

3. After adding measures, click the **Edit** icon



next to the desired measure to change the display name for the measure, the aggregation method, and the splitting value. See the descriptions below for more details.

> **Note**
>
> Editing a measure in AppMon Web does not modify the measure configuration originally made in the AppMon Client. It only modifies the name and appearance of data on AppMon Web tiles.

Measure aggregation methods

Events captured for a measure on the AppMon Server are recorded at 10 second intervals and aggregated into the following measurements. When you edit a measure, you can choose the type of aggregation method to display on a tile. Not all aggregation methods are available for every measure.

- **Average**: The average of all incoming event values per interval
- **Minimum**: The minimum value measured per interval
- **Maximum**: The maximum value measured per interval
- **Sum**: The sum of event values per interval
- **Count**: The number of times an event occurs per interval

Measure splittings

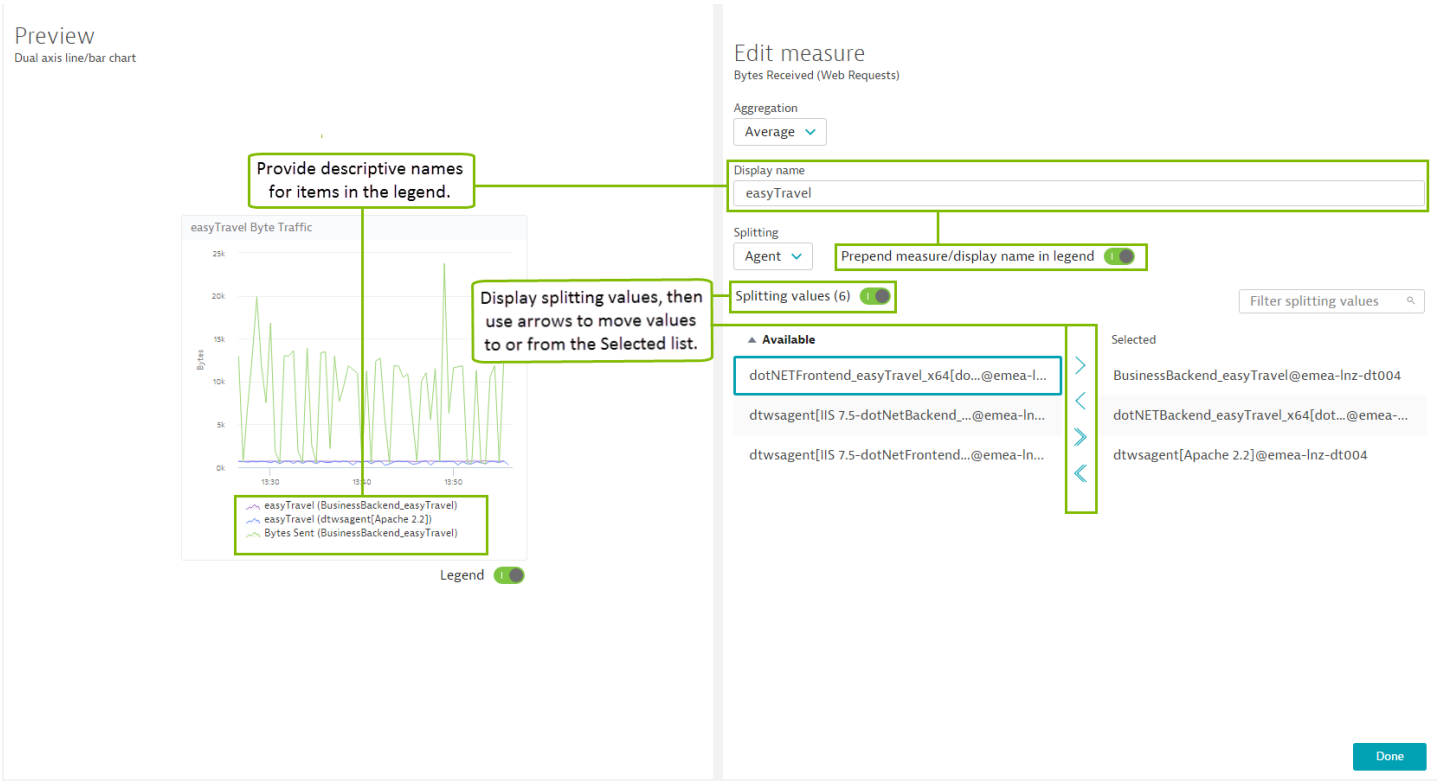
Some measures contain *splittings*, which are used to identify a subset of measure data. For example, the Memory Used out-of-the-box measure has an agent host splitting. For this measure, you can either display the total memory used for all agent host machines, or choose to split by agent host to display the memory used for each agent host machine.

Chart tiles support the display of up to ten splittings per tile. If your splitting contains more than ten values, choose the most import ten to display on your tile.

To select splitting values to display on a chart tile:

1. Click the **Edit** icon
   
   to the right of the measure name.
2. Select the type of aggregation from the Aggregation list.
3. Optionally type a descriptor that applies to your splitting values to prepend to each splitting value name in the tile's legend.
4. From the **Splitting** list, select the splitting type.
5. Use the **Prepend measure/display name in legend** to add or omit the measure name and display name (if supplied) to each splitting value name in the tile's legend.
6. To select a specific splitting value, click the slider next to **Splitting values (#)** to the right to make the values visible.
7. Use the arrows to move splitting values to or from the Selected column. Click **Done** below the list of values after selecting the desired splitting values.

> **Tip**
>
> Use the Filter splitting values box to quickly find a specific value for long splitting lists.



Web dashboard timeframe and resolution

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Timeframe

Chart resolution

Related pages

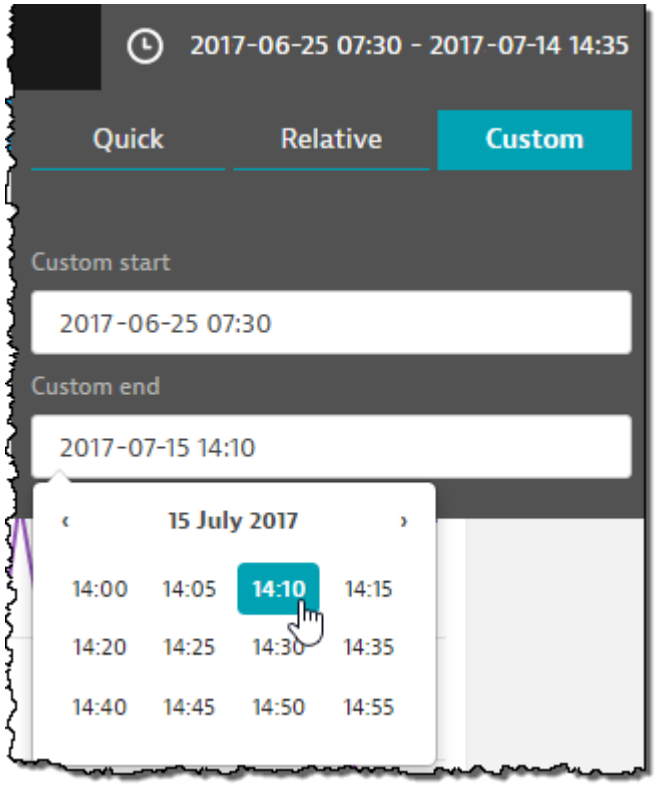
Web dashboards

Access AppMon web

Share web dashboards

AppMon Web FAQ & troubleshooting

Timeframe



The Timeframe drop-down is available at the top of all web dashboards and displays the time frame currently used for the dashboard. Click the dropdown and use the **Quick**, **Relative**, and **Custom** tabs to set the time period for information that displays on your web dashboards.

- **Quick**: Click a button in the Quick tab to use a set time range relative to the present time, or set the time frame to **Today** or **Yesterday**.
- **Relative**: Set a time range between 15 minutes and 999 days, starting from either the present time or a period from 15 minutes to 999 days in the past.
- **Custom**: Click in the Custom start and Custom end fields and use the calendar and time selectors to set a time range.

The time frame for all dashboards is dynamic and automatically refresh at a 1 minute intervals for all tiles.

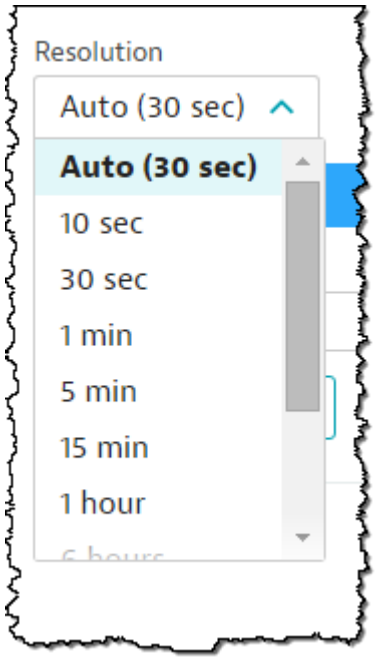
The time frame set for the dashboards applies to all its tiles unless a separate time frame is applied to a specific tile. With AppMon 6.5 and later, you can set a specific time frame for an individual tile within a custom dashboard. Click **Advanced Settings** in the Tile configurator, then select **Tile specific timeframe** and use the Timeframe dropdown to set a time frame for the tile.

Increasing the time frame to larger period of time, such as 30 days, may result in a slight delay in the rendering of web dashboard content depending on the amount of data in your selected measures.

Note

The time frame of a web dashboard and its chart resolution is set using the time zone of the AppMon Server not the time zone of the device on which you are viewing AppMon Web . This is different from the charts appearing in the AppMon Client, which display data using the local time zone.

Chart resolution



The *chart resolution* drop-down defines the interval between data points that appear on a dashboard tile. By default, the resolution is set to the optimal interval value for the chosen dashboard time frame. For example, if the dashboard time frame is **1 hour**, then the resolution is automatically set to **1 min**.

Note

The optimal interval value for selected dashboard time frame always appears as **Auto** in the Resolution list.

You can adjust the resolution to increase or decrease the number of data points on your charts, however, keep in mind that changing the dashboard time frame after setting a custom resolution value may distort your data and make it difficult to interpret. Additionally, the data point calculation is relative to the *preceding* data point **not** the *first* data point of the time frame. As a result, increasing the resolution also increases any aggregate values of each data point as there are more data points in a given interval.

> **Tip**
>
> As a best practice, keep the default resolution value set to **Auto** to ensure that whenever the dashboard time frame is modified the resolution optimally displays.

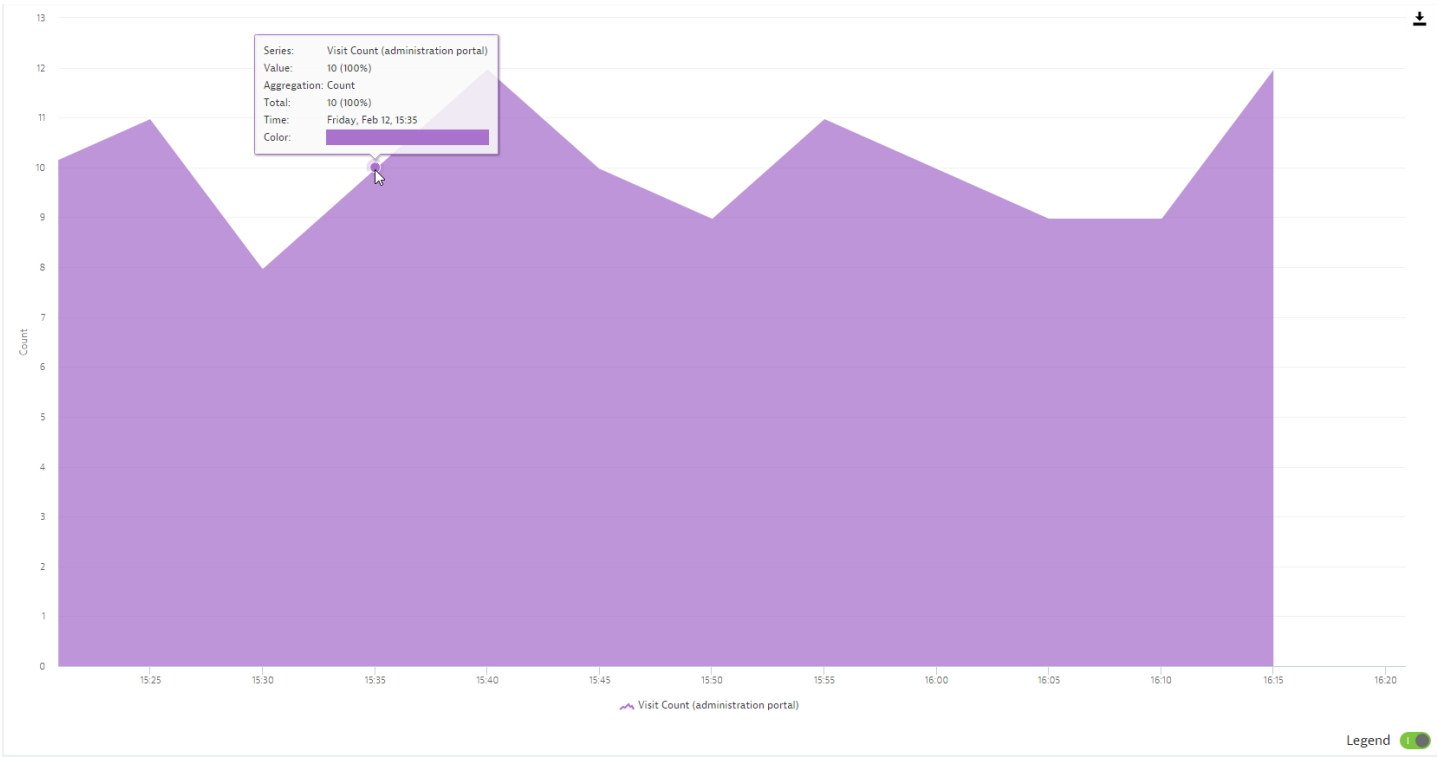
Resolution example

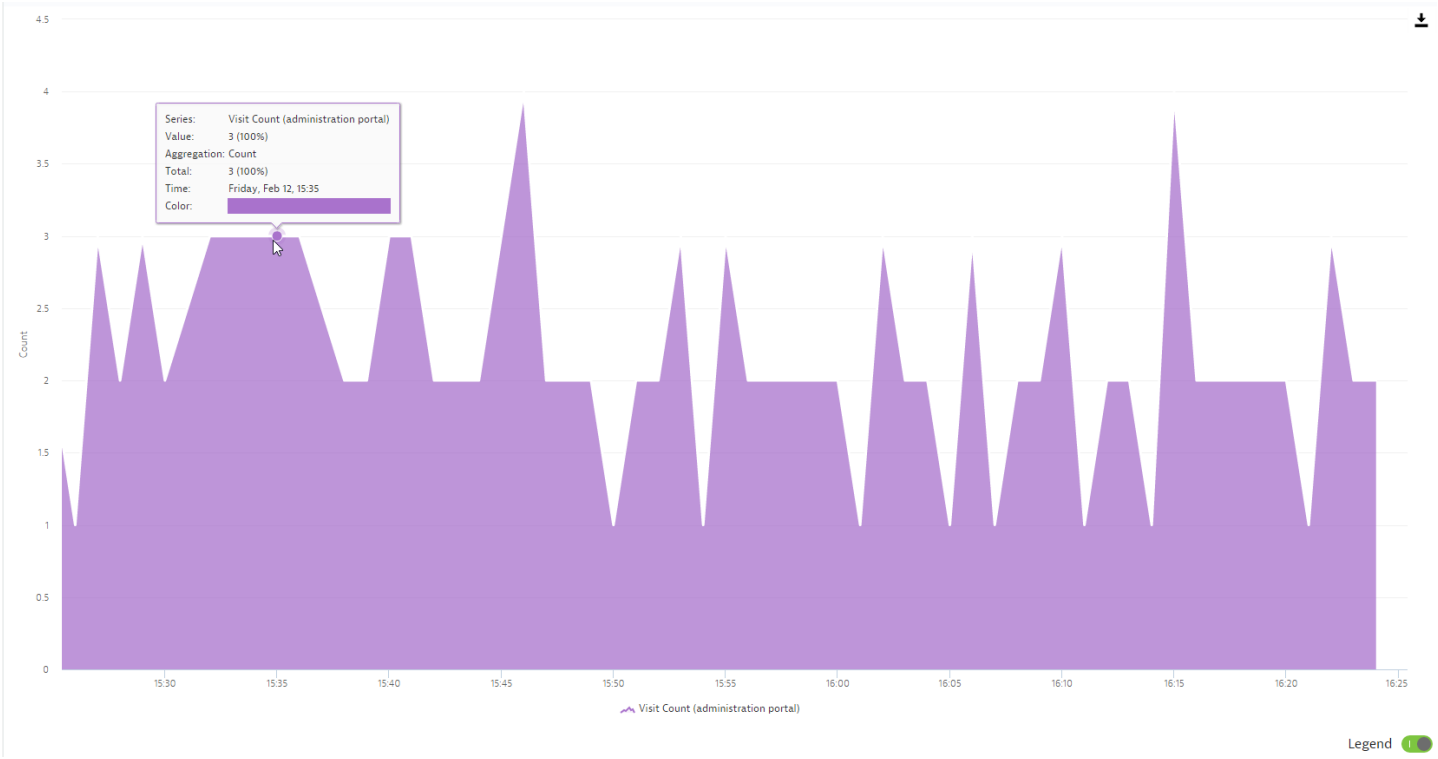
The following example shows the effect that changing the resolution has on the number of data points displayed on a chart **and the aggregation values** for each data point. In this example, the following settings are used:

- Dashboard time frame = 1 hour
- Chart #1 resolution - 5 min
- Chart #2 resolution = 1 mins

In addition to **fewer data point intervals** appearing on the 5 minute resolution chart, the **value of the data points are higher**. For example, the 15:35 data point on the 5 minute resolution chart has a count of 10, but on the 1 minute resolution chart, it has a count of 3. The difference is because the cumulative aggregation of data for an interval is relative to the previous data point, not the first data point in the chart.

In this example, the data point in the 5 minute resolution chart includes the count of all data from 10:45 am to 11:00 am. In the 1 minute resolution chart, the data point includes only the count of data from 10:59 am to 11:00 am.


Series: Visit Count (administration portal)
Value: 10 (100%)
Aggregation: Count
Total: 10 (100%)
Time: Friday, Feb 12, 15:35
Color:
Count
Visit Count (administration portal)
Legend


Series: Visit Count (administration portal)
Value: 3 (100%)
Aggregation: Count
Total: 3 (100%)
Time: Friday, Feb 12, 15:35
Color:
Count
Visit Count (administration portal)
Legend

Share web dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Capabilities

Sharing a dashboard

Mark a Web Dashboard as a Personal Favorite

Publicly share a web dashboard

Other reporting features

Related pages

Web dashboards

Access AppMon web

AppMon Web FAQ & troubleshooting

Custom web dashboards that you create are only accessible to you until you share them with other users. The basic concept of sharing views or dashboards is a persistent URL that you can send off or include in other assets like documents or wiki pages.

Capabilities

- **Sharing with specific users** via e-mail invite. Only they can see the dashboard.
- **Sharing with all users having access to the AppMon Server**. The popularity of dashboards shows how many users have added the dashboard to their personal favorites.
- **Sharing with users that do not have an AppMon user account**. See the guide in the section below.

A dashboard that has been shared with you appears in the **Shared** tab in the dashboard overview every time you log into AppMon Web.

> **Note**
>
> Opening a shared web dashboard requires an AppMon user account.

Sharing a dashboard

To share a web dashboard:

1. Click the **Share**  button in the top right corner of the screen.

2. Select the access level:

   - **Only you**: Nobody else can see it.
   - **Specific users**: Only users, who have the direct link to the dashboard, will see it. Copy the link from the below to share the dashboard.
   - **All users**: All users with sufficient permissions see the dashboard at the **Shared** tab.

3. If needed, activate the **Fixed system profile and application for other users** button, to prevent change of data source by other users.

Mark a Web Dashboard as a Personal Favorite

Whenever a user shares a web dashboard with value to you, you can add it to your personal favorites. If the dashboard is shared with all users, this will increase the popularity of the dashboard and its rank in the list, so your team members easily see which dashboards are most valuable.

Publicly share a web dashboard

A web dashboard can be publicly shared and made accessible without login. The dashboard's share URL has to be extended by the username and password of the user account intended for public sharing. Thus, a dedicated user account is recommended in order to keep full control over which System Profiles are publicly visible.

Syntax for extending the share URL:

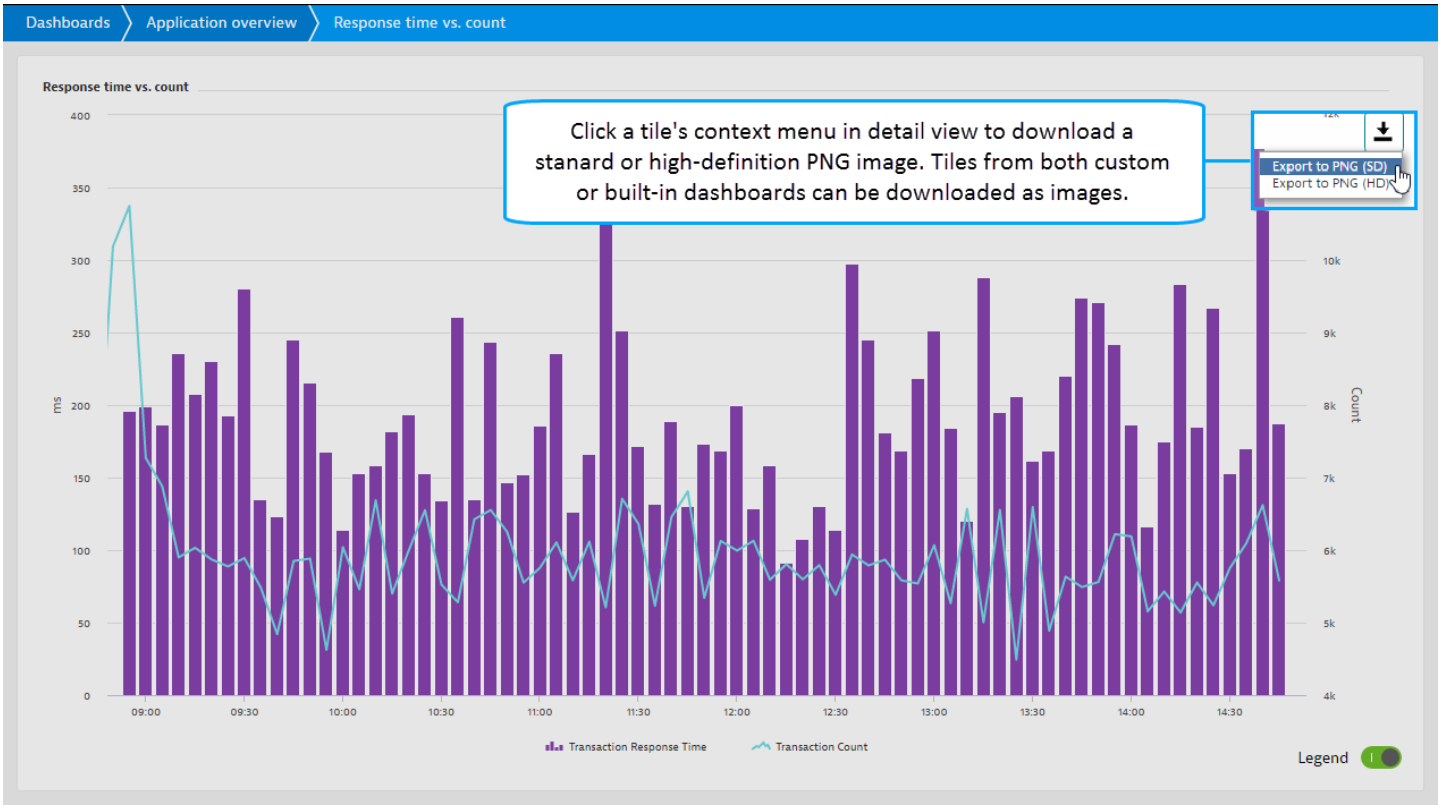
```
https:/<dt server host/ip>:9911/index.jsp?username=<username>&password=<password>#dashboa
```

Other reporting features

Use tile image download feature to create a distributable PNG or SVG image file of chart tiles to include in a presentation, post on a wiki page, or send in an email.

To download a tile image, click the context menu

of a built-in or custom web dashboard tile, and then select the image format you want to download.



Export and import web dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Export web dashboards

Import web dashboards

Delete custom web dashboards

Related pages

Custom web dashboards

Share web dashboards

Built in web dashboards

Embed custom content into a web dashboard

Control access to AppMon Web dashboards

Web dashboard exporting and importing presents another way to share your dashboards to others in your organization. Exporting is useful when a large number of dashboards need to be shared with others. Hundreds of dashboards can be exported and imported at one time. Only Custom dashboards can be exported or imported.

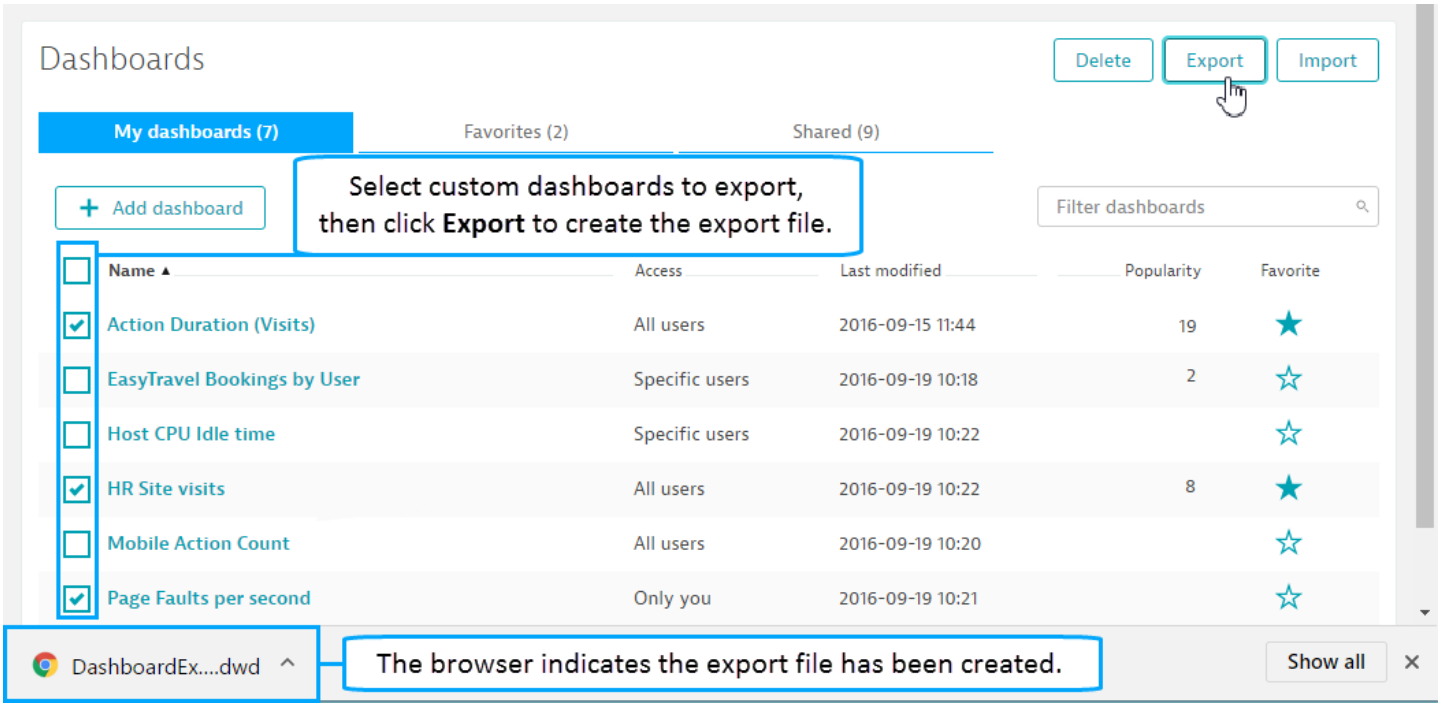
Export web dashboards

In AppMon Web, click **Dashboards**, then select the **My** tab. Your list of custom dashboards displays, each with a check box next to the dashboard entry.

Click in the check box of each dashboard you want to export, then click the **Export** button. A file named `DashboardExport.dwd` downloads to your assigned downloads folder. Use Windows to send the file to others with whom you want to share the dashboards. Once an export file has been copied to an accessible location, others can use it to import the Web dashboards.

Tips

- The **Export** button only appears when dashboards are selected.
- Click the check box next to the Custom Dashboard headers to select all the custom dashboards in the list.



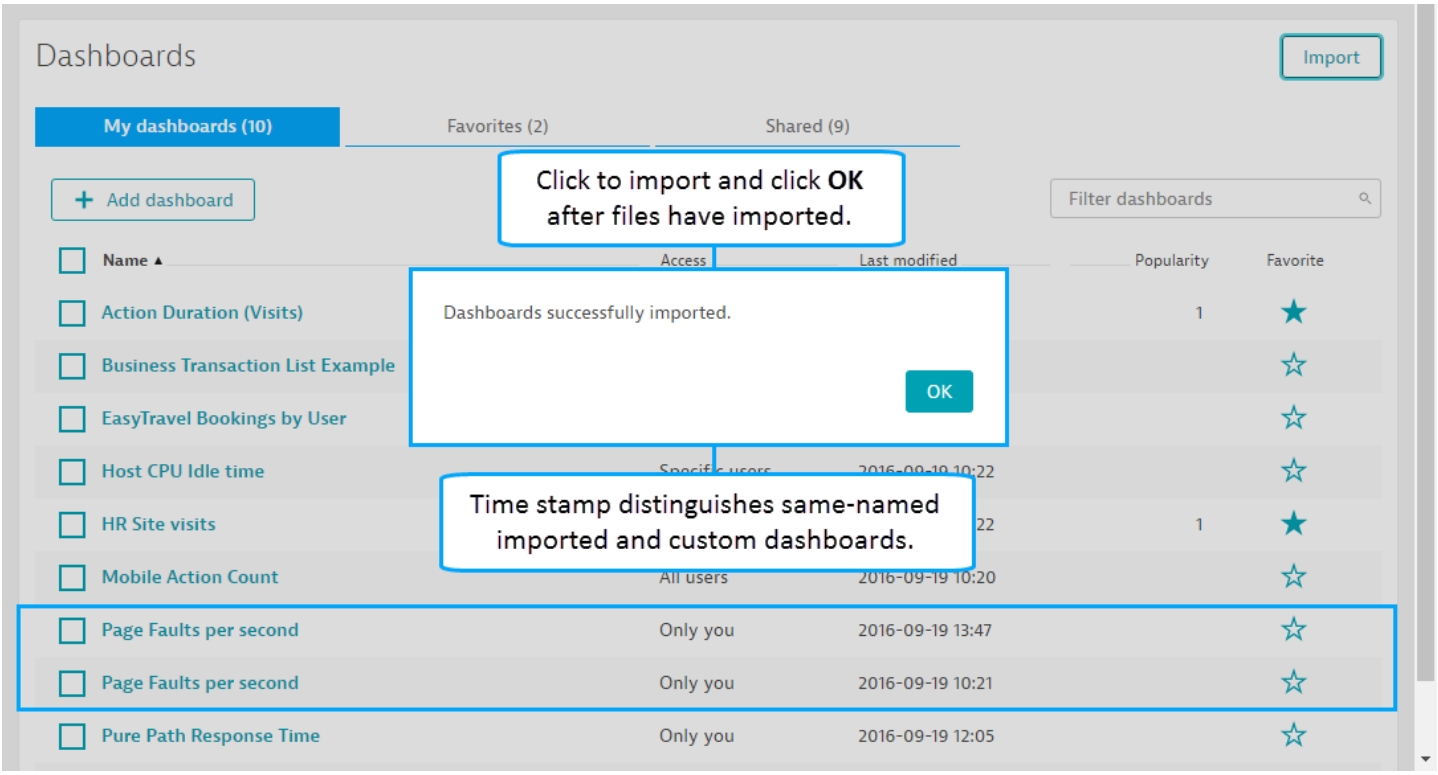
Import web dashboards

You can import a web dashboard file from any accessible file location. All exported Custom Web dashboards in the file import to your Custom dashboards list.

To import a Custom web dashboard file, click **Dashboards** then select the **My** tab to display your custom dashboards and click the **Import** button.

Use the operating systems file browser to navigate to the location of the export file and select it.

When the message box appears indicating the dashboards are successfully imported, click **OK**. The imported dashboards appear in the Custom dashboards list.



Note

If you Import a dashboard that has the same name as one of your existing Custom Web dashboards, it imports with the name intact. The only noticeable difference is that the imported dashboard has a different time stamp in the **Last Modified** column. You can change the name of any Custom Web dashboard at any time for differentiation.

Delete custom web dashboards

You can delete dashboards to help remove obsolete or unneeded custom dashboards and help manage your custom dashboard list.

Simply click in the check box of each dashboard you want to delete, then click the **Delete** button. Click **OK** in the message box that appears to confirm the deletion.

Tips

- The **Delete** button only appears when dashboards are selected.
- Click the check box next to the Custom Dashboard headers to select all the custom dashboards in the list.

Embed custom content into a web dashboard

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Embed content using HTML

Create and define the custom tile

Define the custom content and save the tile

Related pages

Custom web dashboards

Share web dashboards

Built in web dashboards

Export and import web dashboards

Control access to AppMon Web dashboards

Integrate user experience or infrastructure data in your dashboards with social aspects of your business by embedding custom content into a custom web dashboard using the Web tile type.

Add informative or news-based YouTube or other video content to compliment data in other tiles on the dashboard, Using the Web tile this way enhances the use of AppMon Web the single portal you use to gain an overview of your environments and applications. See Tile Types for more information on the Web tile.

Embed content using HTML

An inline frame places another HTML document in a frame. Unlike an `object` element, an inline frame can be the target frame for links defined by other elements, and it can be selected by the user agent as the focus for printing, viewing its source, and so on. The element's content displays as alternative text if the browser does not support iframes.

Create and define the custom tile

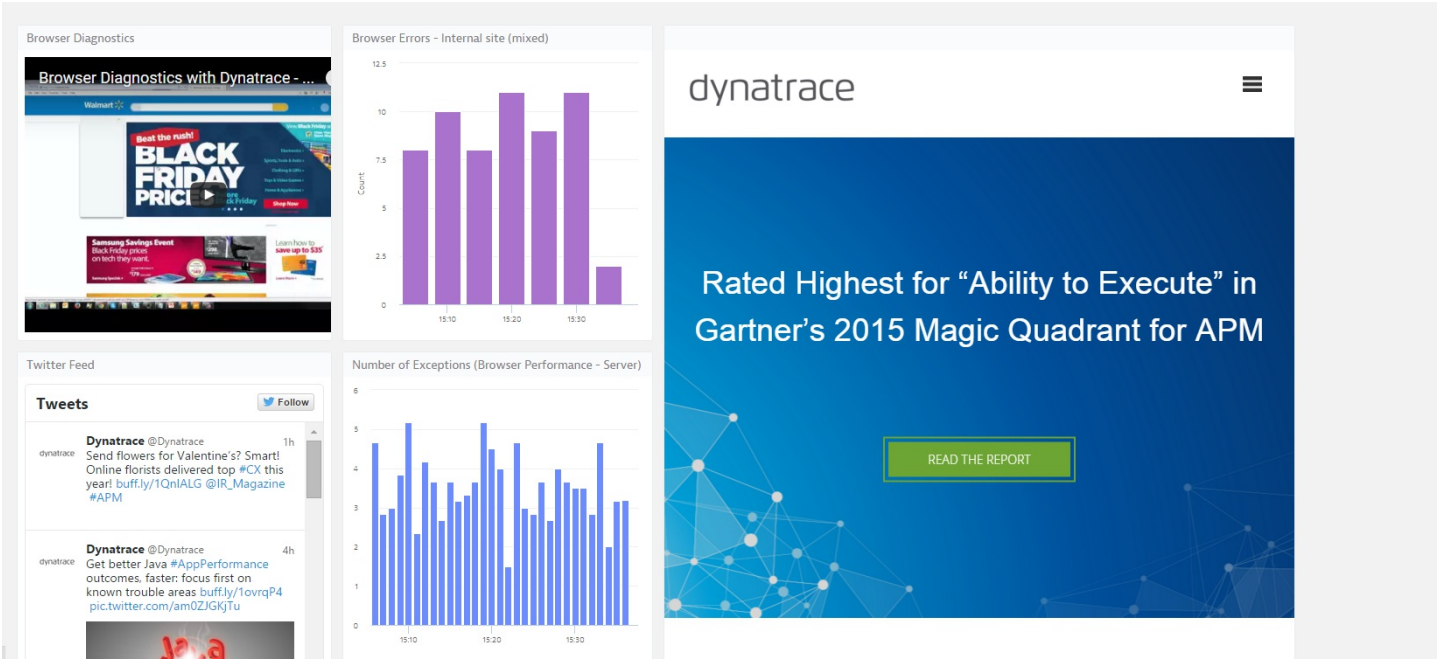
1. Open the dashboard into which you want to add the custom HTML or URL content and click the **Edit** button.
2. Click the **Add** Tile button to add the tile, then click the tile to edit its contents in the Tile configurator.
3. Select the Web tile
   
   in the Tile Configurator.
4. Optionally type a title for the tile.
5. In the **Show content from** list, choose from the following values:
   - **URL**: Embed a URL page in the tile. Any URL used must be secure (https://).
   - **HTML**: Embed custom HTML code, such as a widget or other page embed code.

The text field below the **Show content from** list changes based on the selected value.

Define the custom content and save the tile

1. Do one of the following:
   - **URL**: Copy the URL if the Web page you want to show in the tile and paste it in the **External content URL** box.
   - **HTML**: Copy your external content embed HTML-compliant code from its source and paste it in the **Custom content HTML** field.
2. Set the tile behavior when clicked in the **Click behavior** list.
   - **Show detail view**: Show the tile as a single detailed page, where you can save it as an image for sharing.
   - **Open in New Window**: Opens the tile as a separate web page.
3. Click **Done** to save changes to the tile.
4. Click **Done** again to save the changes to the custom dashboard.

This example shows a dashboard with tiles containing custom content, including a Twitter feed created with Twitter widget code copied from a Twitter page, another tile containing a widget that houses a playable YouTube instructional video, and another tile with a company homepage URL.


Browser Diagnostics
Browser Diagnostics with Dynatrace - ...
Browser Errors - Internal site (mixed)
Count
12.5
10
7.5
5
2.5
0
15:10
15:20
15:30
dynatrace
Rated Highest for “Ability to Execute” in Gartner’s 2015 Magic Quadrant for APM
READ THE REPORT
Twitter Feed
Tweets
Follow
Dynatrace @Dynatrace 1h
Send flowers for Valentine's? Smart! Online florists delivered top #CX this year! buff.ly/1QnIALG @IR_Magazine #APM
Dynatrace @Dynatrace 4h
Get better Java #AppPerformance outcomes, faster: focus first on known trouble areas buff.ly/1ovrqP4 pic.twitter.com/am0ZJGKjTu
Number of Exceptions (Browser Performance - Server)
6
5
4
3
2
1
0
15:10
15:20
15:30

Control access to AppMon Web dashboards

Related pages

Web dashboards

Built in web dashboards

Custom web dashboards

Share web dashboards

Built in web dashboards

Export and import web dashboards

Embed custom content into a web dashboard

You can control Web dashboard access by assigning permissions to a user role. Configure permissions in the server settings dialog box. AppMon supports different levels of access control depending on the version.

For AppMon 6.5.6 and earlier, the permission Use System Profile in Dynatrace Web controls data access for specific system profiles. With this permission granted, all workflows are visible to the user including visit details, PurePath, and infrastructure views such as hosts or processes.

Applications web view

Application monitoring in AppMon Web

Application incidents

PurePath analytics web view

PurePath web view

Transaction flow web view

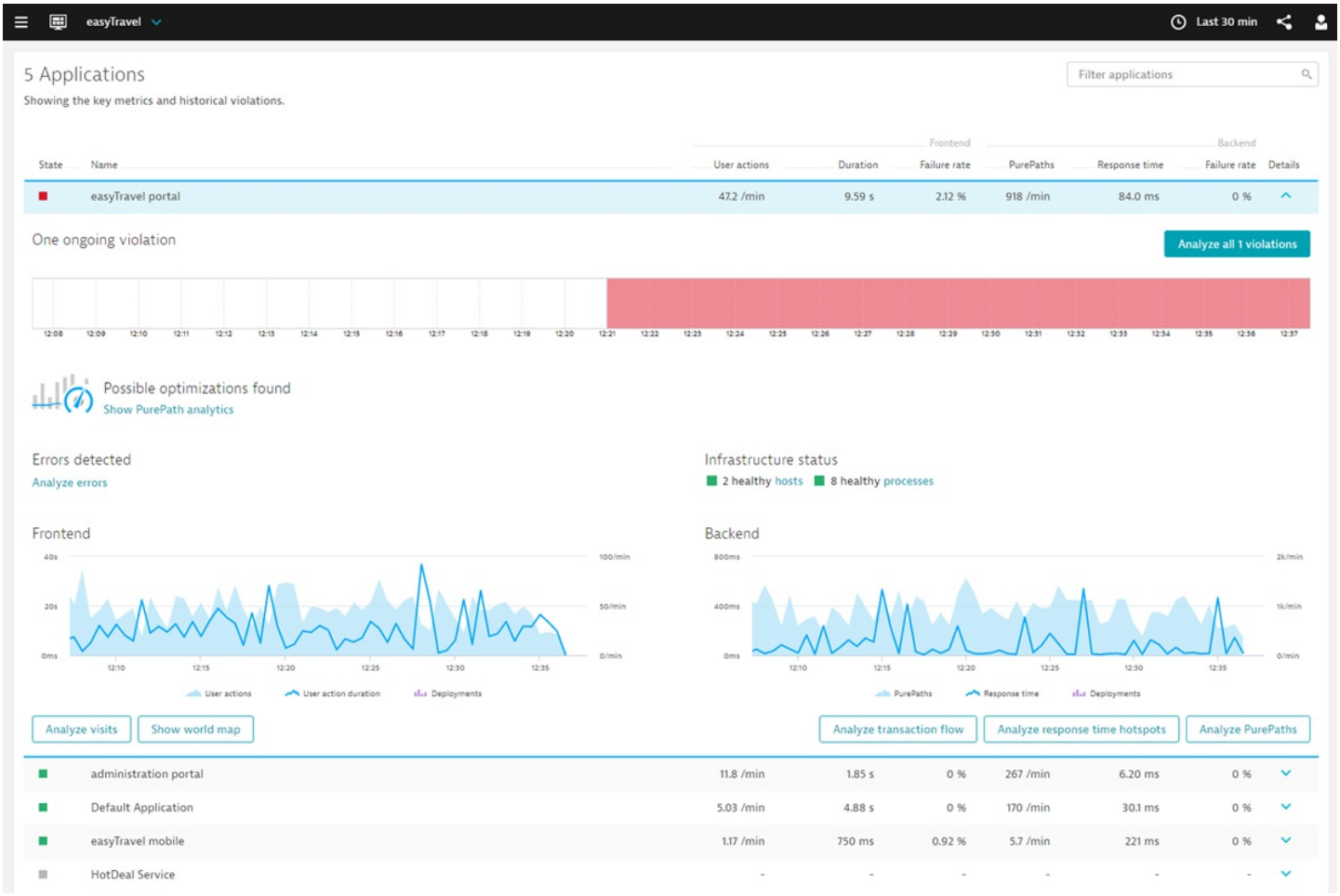
Application monitoring in AppMon Client

Monitoring Overview dashboard

Application Overview dashboard

Application incidents

Application owners need to know how their applications are doing from many perspectives. The **Applications** web view provide all the required insights, from big picture to in-depth drill down into performance or user actions. See which application is unhealthy, analyze PurePaths, response times and transaction flow to see where problem is coming from. To access the **Applications** web view, in the AppMon Web click the burger button and then **Applications**.



The Application overview allows you to quickly view application state, frontend and backend related data — in pure numbers and graphically. For frontend you can see the amount, duration, and failure rate for user actions. For backend you can see purepaths, their response time and failure rate. For each parameter you can quickly go to detailed view with just one click. Additionally, expanded application view displays status of the related infrastructure.

Moreover, the violations history shows a quick overview of current and past problems with your application and allows you to drill down into the violation for detailed information about those problems.

You can easily share application overview and other app owner dashboards with others. See Share Web Dashboards to learn how.

All violations, response time hotspots, errors, transaction flows, and PurePaths can be opened in detailed views for in depth problem analysis. The detail views show the most important properties in the selected timeframe.

On this dashboard you can:

- View response time hostpots and quickly drill down into their PurePaths.

- View errors, detected by error rules, and into their PurePaths or user actions.
  See System profile - error detection for learn how to configure error rules, and Error analysis for general info about error analysis.
- See the aggregated view and dependencies between tires of you application.
- Analyze PurePaths.
- See how PurePath data change over time.

Alert notifications can also be viewed on custom web dashboards using the **Alert** tile. See Tile Types for more information.

Transaction flow web view

Related pages

Follow your transactions with Transaction Flow

PurePath web view

PurePath analytics web view

Application monitoring in AppMon Client

Monitoring Overview dashboard

Application Overview dashboard

Transaction Flow dashlet

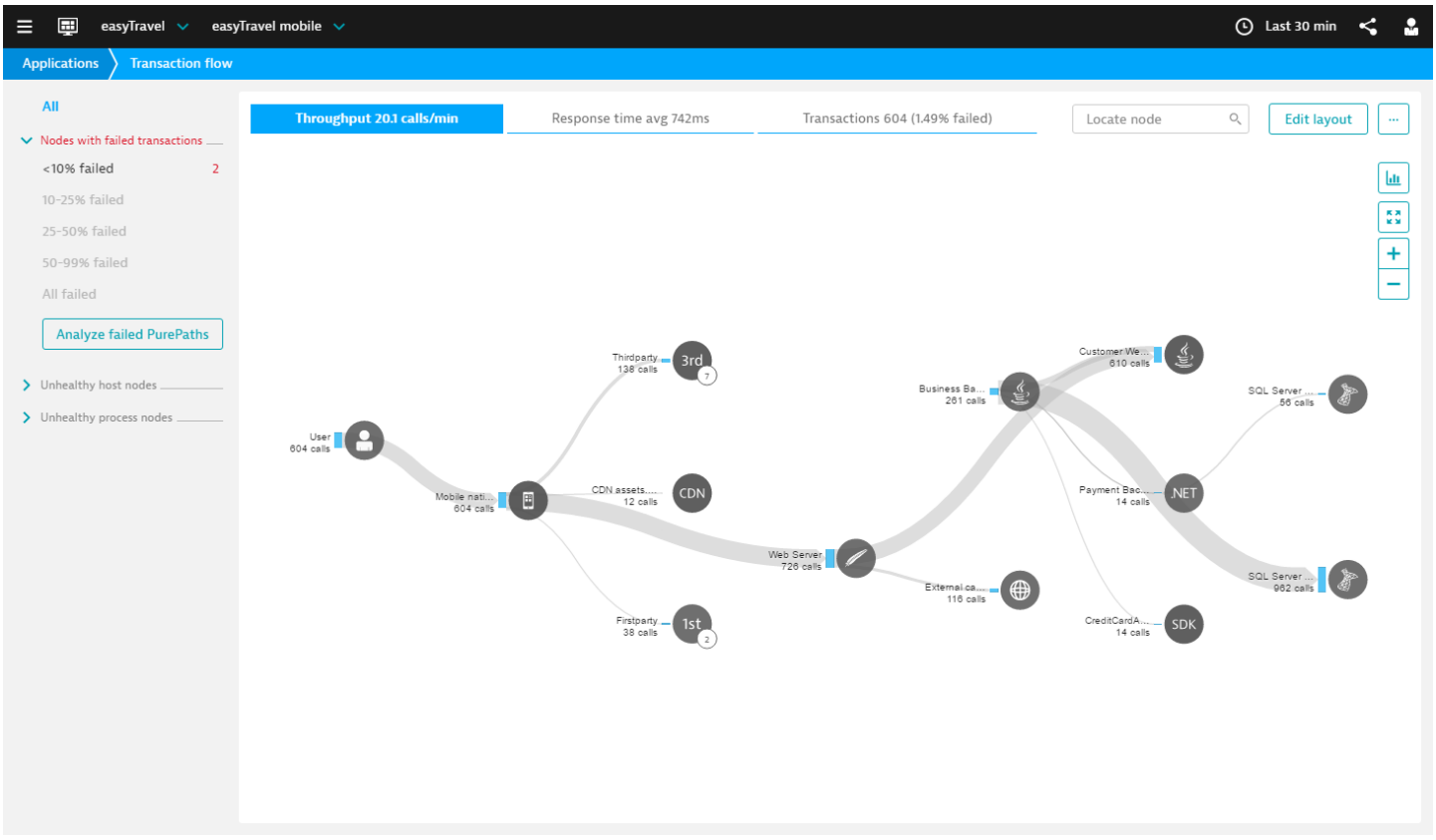
Watch and learn

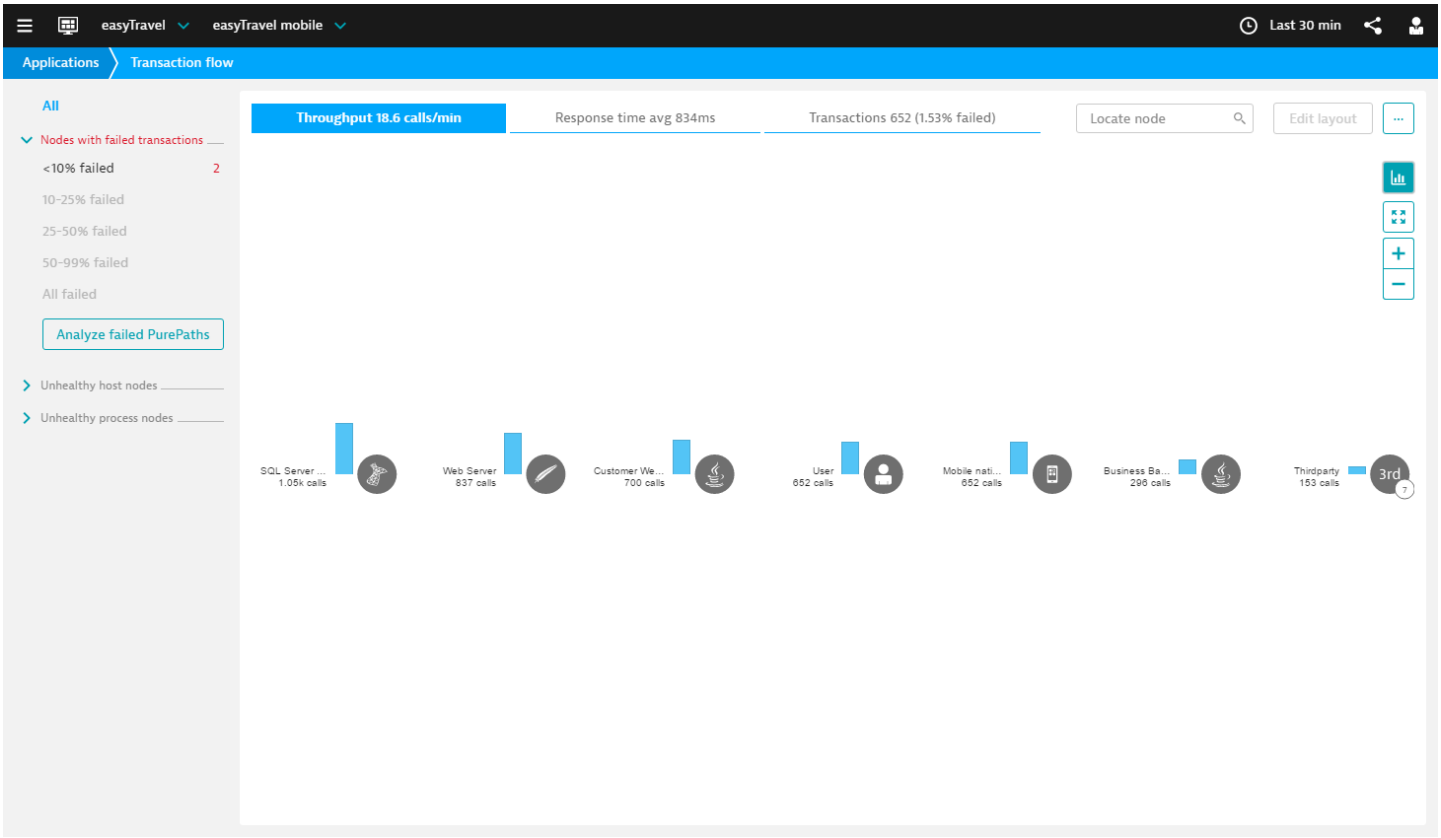
Analyzing Transaction Flow in AppMon Web

The Transaction Flow graphically represents the tiers of a monitored application in the current time scope for the selected System Profile. You can view it in two ways: as a topology view, or as a bar chart. Both views display:

- **Throughput between nodes**
  - Topology: Connections represent calls between nodes. The thickness of the lines between nodes increases with the number of calls. Blue bars adjacent to the nodes represent incoming calls.
  - Bar chart: Blue bars represent incoming calls. Grouped nodes are shown as one node, with overall number of calls in the group. You cannot ungroup or re-sort them.

- **Response time**.
  - Topology: Purple bars adjacent to the nodes represent node's contribution to the average response time.
  - Bar chart: Purple bars represent node's contribution to the average response time. Grouped nodes are shown as one node, with overall contribution of the group. You cannot ungroup or re-sort them.
- **Transactions**.
  - Topology: Connections represent PurePaths between nodes. The thickness of the lines between nodes increases with the number of PurePaths. The bar adjacent to the nodes represents the overall amount of PurePaths; purple bars are incoming PurePaths and dark purple are unique PurePaths.
  - Bar chart: Bars adjacent to the nodes represent the overall amount of PurePaths; purple bars are incoming PurePaths and dark purple are unique PurePaths. Grouped nodes are shown as one node, with overall number of PurePaths of the group. You cannot ungroup or re-sort them.





You can filter the nodes by one or several criteria, listed in the left pane. Affected nodes will be highlighted in red in topology and bar chart views.

Hover the cursor over a node to highlight its incoming and outgoing connections. Incoming connections are highlighted in blue, and outgoing connections are highlighted in orange. If you activate the **Failed links** under the **…** button, connections with failed links will be highlighted in red.

Click a node to view details about related response time, transactions, and infrastructure. From here you can drill down into visits or PurePaths. In details view of a node group, click **Expand** to view separate nodes.

Edit topology layout

You can customize the topology view to make it more comfortable for you to read. You can change node names, move nodes, group them, or ungroup auto-group nodes. One group can contain up to 25 nodes.

To customize view, click **Edit layout**, and drag the nodes as you like. When finished, click **Done**.

To group nodes, simply drug one above the other. Some nodes are grouped automatically, such as tiles of one server, or third party nodes, or nodes of one technology, etc. You can still ungroup them manually: in the group details view, click the **…** button for the required node and select **Ungroup**.

You can automatically align all nodes to a grid. To do so, click the **…** button, and select **Align nodes**.

To restore the original layout, click the **…** button, and select **Reset all**.

Transaction flow print

You can print both topology and the bar chart views. Unlike the standard browser print option, which prints only currently visible part of the view, the **Print** option under the **…** button always prints the whole view. The view is a vector graphic and can be unlimitedly scaled. The print also contains information about metric (throughput, response time, transactions), selected System Profile and Application, and the current timeframe. In addition, the print has QR-code with the link to the printed view.

PurePath web view

Related pages

Transaction flow web view

PurePath analytics web view

UEM-PurePath integration for Synthetic Monitoring

PurePath explained

Application monitoring in AppMon Client

Monitoring Overview dashboard

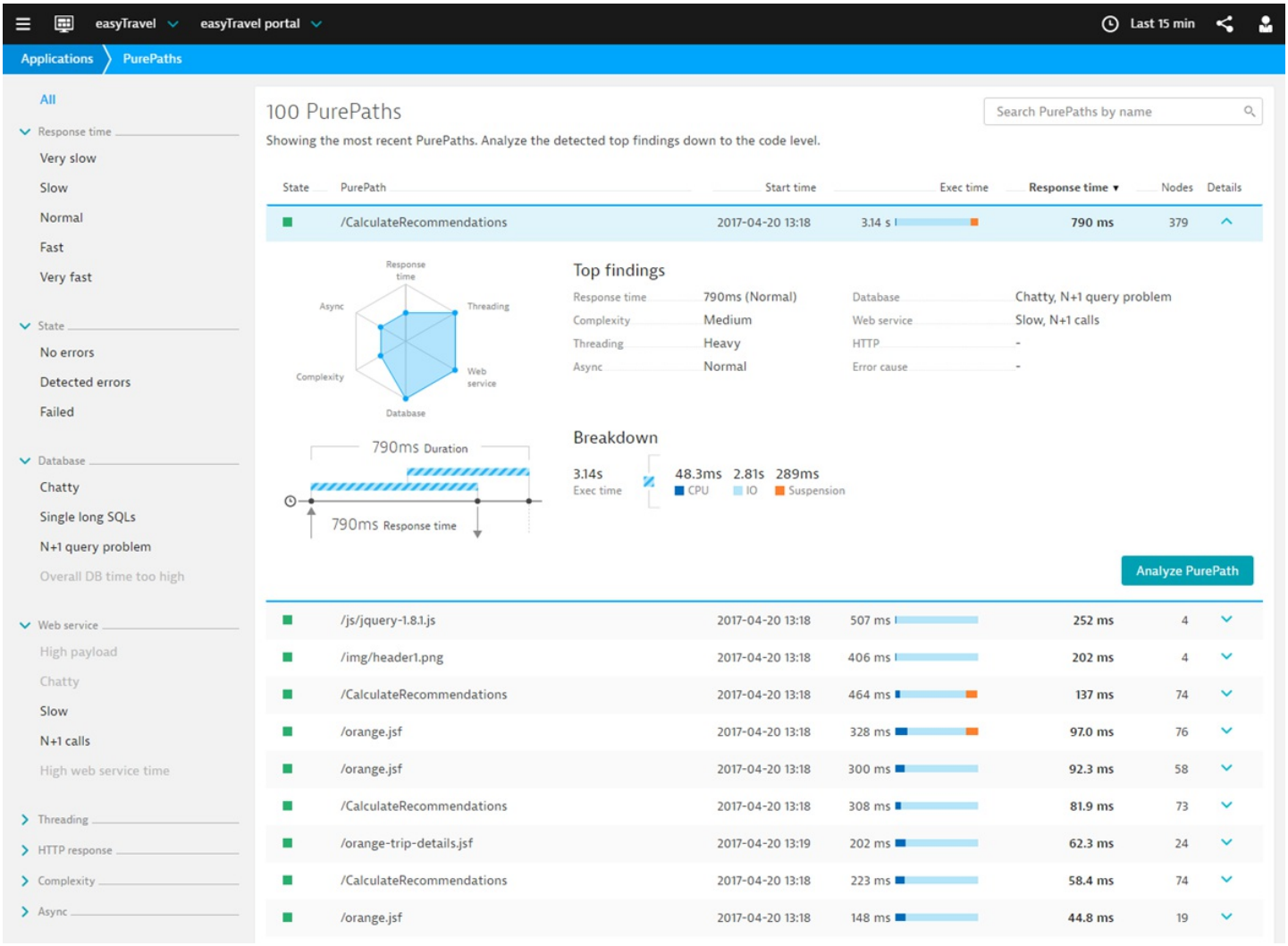
Application Overview dashboard

PurePaths dashlet

PurePath Comparison dashlet

User Action PurePaths dashlet

The PurePath web view graphically represents all captured PurePaths in the current time scope for the selected System Profile. Here you can identify methods and business transactions which caused performance issues.



You can filter PurePaths by the problem patterns, listed in the left side panel. Hover over the pattern to view a pop-up tip. See problem patterns detection for more information.

The table displays the most recent PurePaths for the current problem pattern. Click a column header to re-sort PurePath. Expand the item to view the detailed information about PurePath:

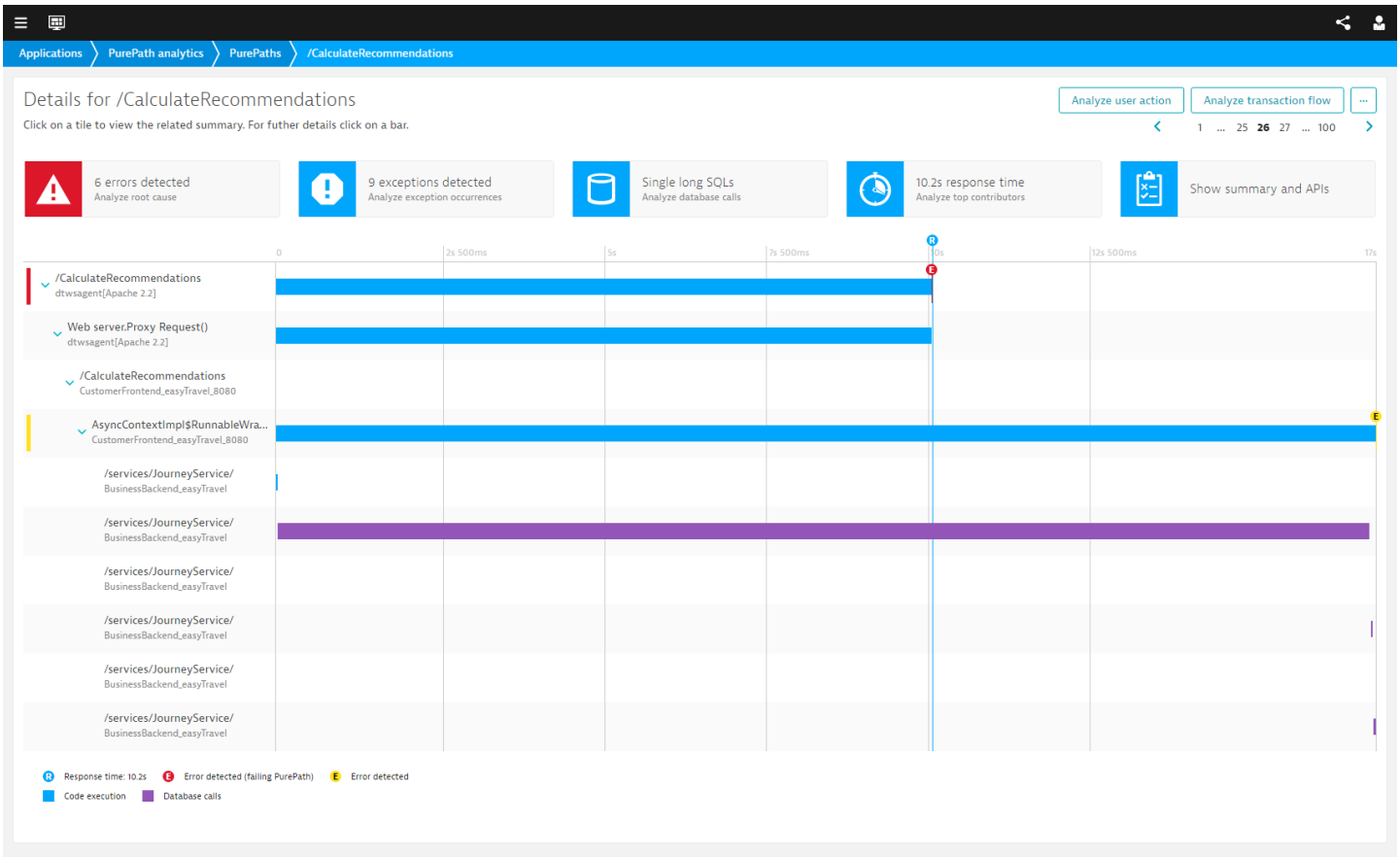
- The spider chart displays severity level for main categories.
- The top findings display performance info, and give insight for possible improvement
- Breakdown shows details about execution times

Click **Analyze PurePath** to open the PurePath details view for deeper analysis.

PurePath details view

The PurePaths details view contains the swim lane of the PurePath, displaying response time for each call from the call stack.

From here you can switch to user action analysis, and view the transaction flow of the PurePath, by clicking the appropriate button.



Tiles on top of the page display numbers for the PurePath top findings: detected errors, exceptions, and database patterns; plus top contributors to the response time. Click the tile to view these findings. Click the tile again to switch back to the swim lane. By clicking **Show summary and APIs** tile, you can view the summary of the PurePath.

You can also view these findings for each call separately. Just click the call you want to analyze and then the required tab. Here you can also view the PurePath tree for the call.

Session export

You can export a PurePath session, to store it on your computer and use later. To do it, click the **…** button, and select the **Export session**.

PurePath analytics web view

Related pages

Transaction flow web view

PurePath web view

UEM-PurePath integration for Synthetic Monitoring

PurePath explained

Application monitoring in AppMon Client

Monitoring Overview dashboard

Application Overview dashboard

PurePaths dashlet

PurePath Comparison dashlet

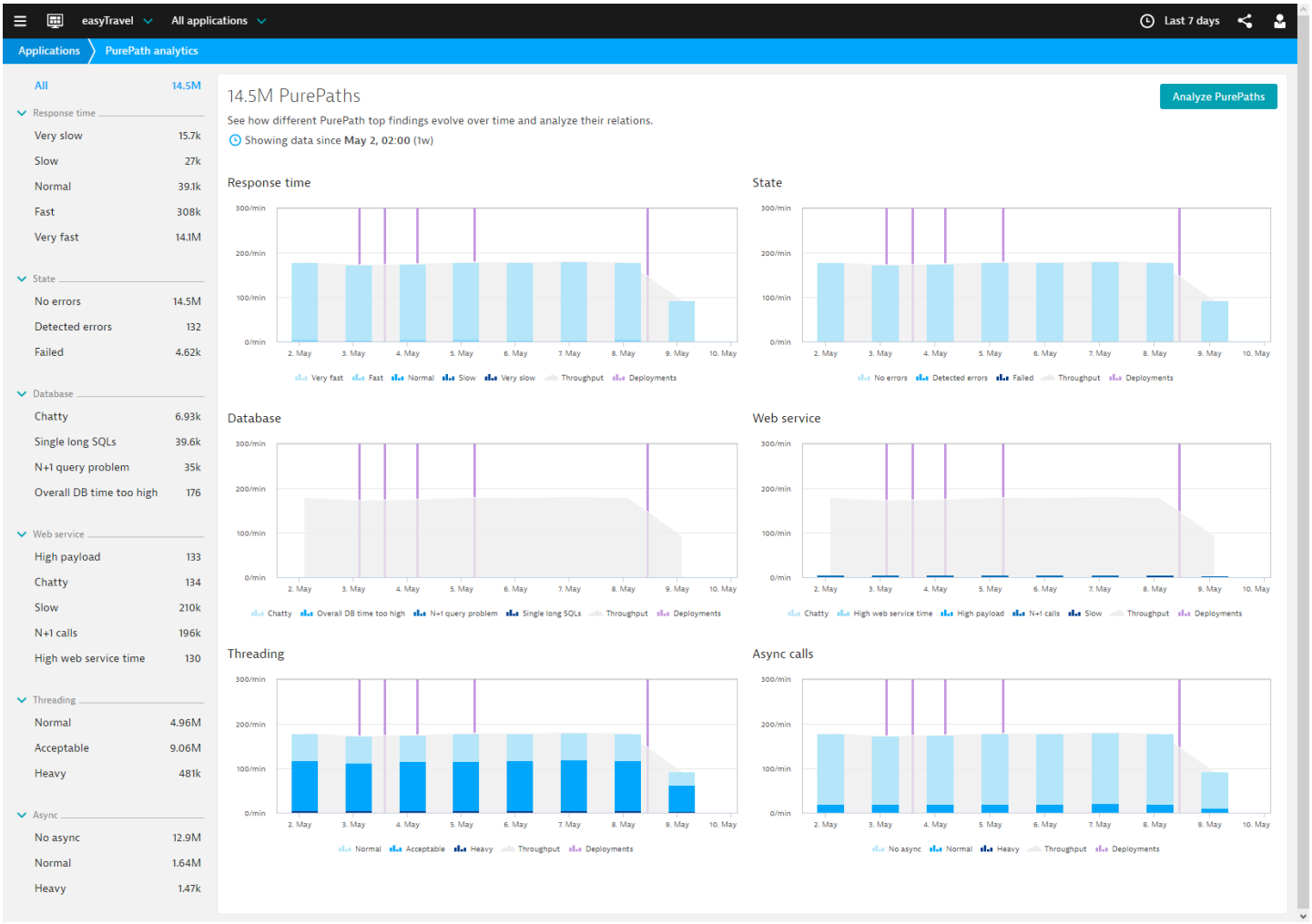
User Action PurePaths dashlet

The PurePath analytics web view graphically represents PurePaths data in retrospective in the current time scope for the selected System Profile. You can select one particular application, or view data for all applications in the System Profile.

Each chart contains marks for the application deployment, allowing you to easily see how the new version of your app affects performance. Hover over the mark to see which application and which version of it was deployed.

Each chart divides data by problem patterns. See the color legend below the chart. Click the bar to filter data by this problem pattern. Problem patterns are also listed in listed in the left side panel, and can be used for filtering as well. Hover over the pattern to view a pop-up tip. See problem patterns detection for more information.

By clicking **Analyze PurePaths**, you'll switch to the PupePath web view with the same filtering criteria applied.


easyTravel
All applications
Last 7 days
Applications
PurePath analytics
All 14.5M
Response time
Very slow 15.7k
Slow 27k
Normal 39.1k
Fast 308k
Very fast 14.1M
State
No errors 14.5M
Detected errors 132
Failed 4.62k
Database
Chatty 6.93k
Single long SQLs 39.6k
N+1 query problem 35k
Overall DB time too high 176
Web service
High payload 133
Chatty 134
Slow 210k
N+1 calls 196k
High web service time 130
Threading
Normal 4.96M
Acceptable 9.06M
Heavy 481k
Async
No async 12.9M
Normal 1.64M
Heavy 1.47k
14.5M PurePaths
Analyze PurePaths
See how different PurePath top findings evolve over time and analyze their relations.
Showing data since May 2, 02:00 (1w)
Response time
State
Database
Web service
Threading
Async calls
300/min
200/min
100/min
0/min
2. May
3. May
4. May
5. May
6. May
7. May
8. May
9. May
10. May
Very fast
Fast
Normal
Slow
Very slow
Throughput
Deployments
No errors
Detected errors
Failed
Chatty
Overall DB time too high
N+1 query problem
Single long SQLs
High web service time
High payload
N+1 calls
Acceptable
Heavy
No async

Application incidents

Related pages

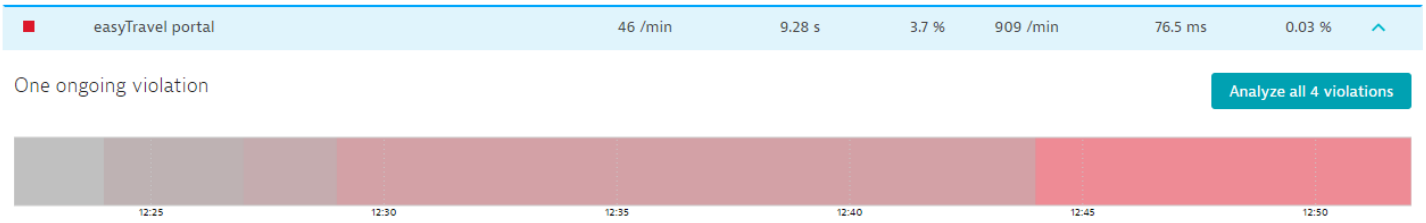
Act on infrastructure and application violations using alerts

Incidents and alerting

Application monitoring in AppMon Client

Incidents dashlet

If the application has any incidents within the current timeframe, the violations history shows a quick overview of current and past problems with your application.

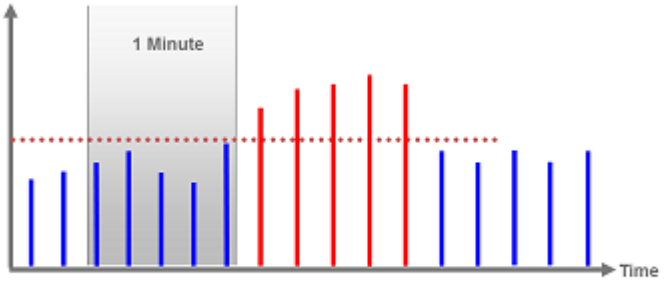


Click it to view the history of all incidents for the current timeframe along with their statuses. Click the incident to switch to the source event for the detailed examination.

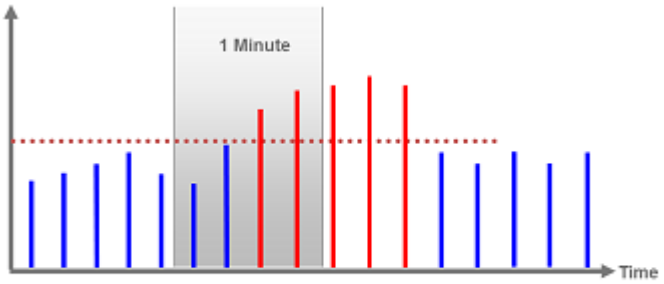
Incident statuses explained

In the incident rule you specify the **Evaluation Timeframe**, which is duration for the incident conditions evaluation. The incident conditions must be fullfilled during this timeframe for an incident to arise. When this happens, the incident is created. It remains open, until the incident measures are below thresholds for the same timeframe.

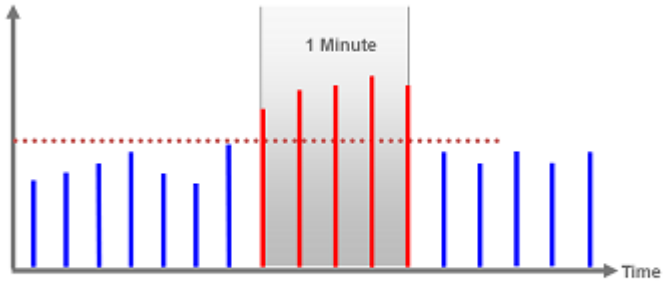
For example, if you select a timeframe of one minute and use a PurePath duration measure with an average aggregation as input for this incident rule, the average PurePath duration of the last minute is calculated and checked for violation every 10 seconds. Measures remain in memory for one hour. The following figures illustrate how the timeframe is used to check for incidents.



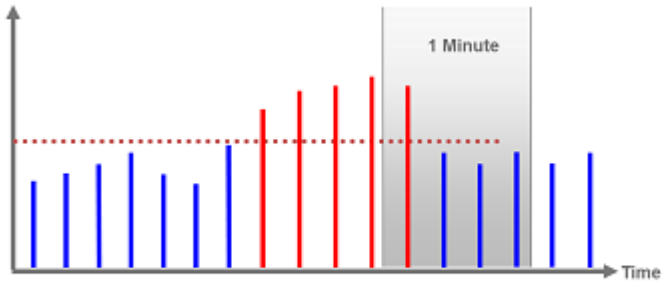
1.6 sec average



1.9 sec average



2.5 sec average
Incident condition fulfilled



2.0 sec average
Incident condition still fulfilled

Incidents are closed only when their condition hasn’t been met for a minute, to better deal with measurements oscillating around their thresholds. Once this timespan has passed without further violation, incidents are closed with the closing time set to fit actual measurements.

Operations web view

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Act on infrastructure and application violations using alerts

Database health web view

Host health web view

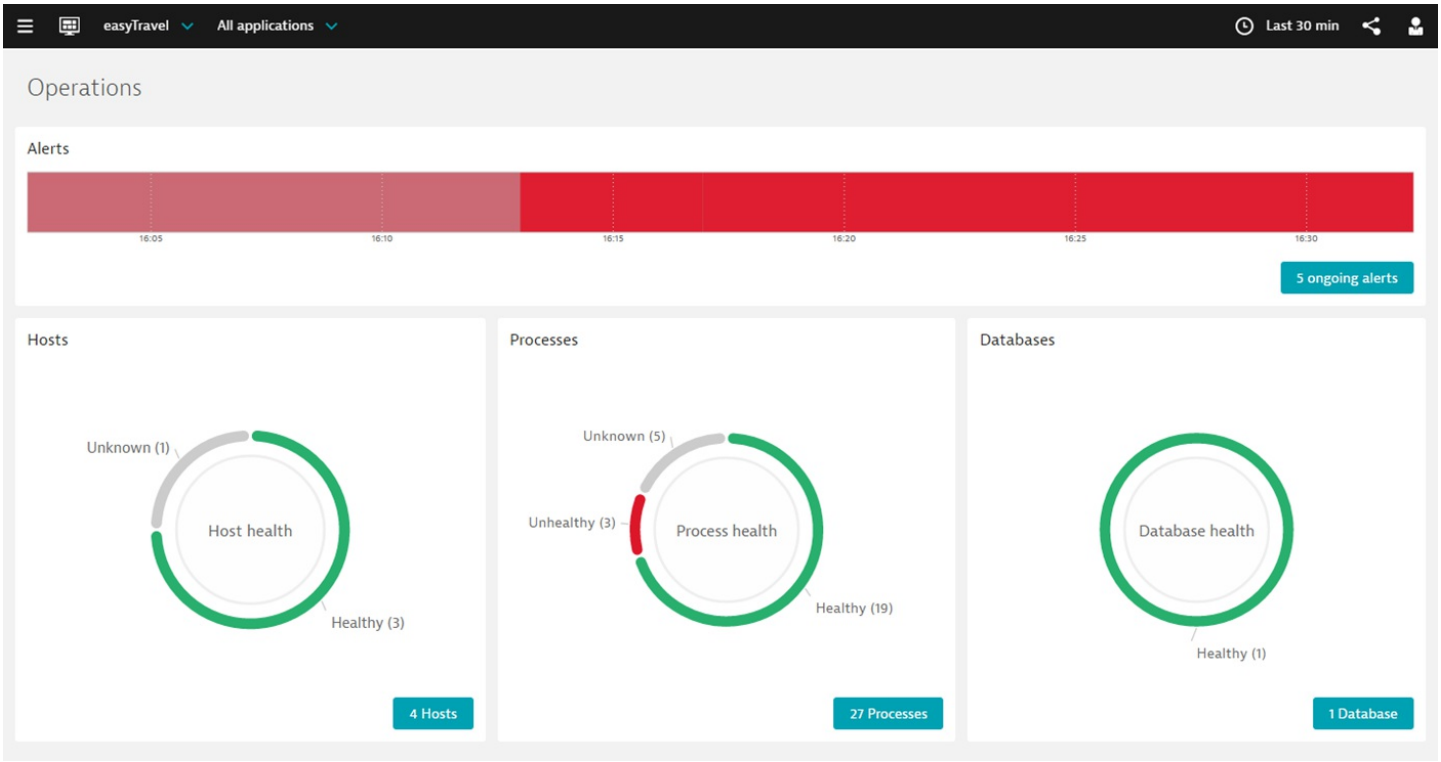
Process health web view

Infrastructure

Infrastructure monitoring

Infrastructure monitoring dashboards

Operations staff require continuous updates about environment health wherever they are. The **Operations** web view lets you see hot spots at a glance. Know what's causing the hot spots, and share your insights across teams to resolve problems. See if problems originate in your infrastructure or within specific applications, track down critical alerts, then drill down for alert details, alert assignment and root cause analysis. To access the **Operations** web view, in the AppMon Web click the burger button and then **Infrastructure**.



Infrastructure monitoring with web dashboards

The operations overview allows you to quickly filter hosts, processes and databases by their health state and to subsequently further narrow down the hosts, processes or databases you are interested in by applying additional filter criteria before drilling down for more detailed analysis to the hosts, processes and databases view, respectively.

Additionally, the alert history shows a quick overview of current and past problems with your hosts and processes and allows you to drill down to the alert view for more detailed information about those problems.

You can easily share host and process views with others. See Share Web Dashboards to learn how.

All alerts, hosts, processes and databases in the **Operations** dashboard can be opened in detailed views for in depth problem analysis. The detail views show the most important properties of each host, process, or database in addition to key metrics visualized in charts covering the selected timeframe.

Tiles on this dashboard:

- Help you to monitor host, process, and database health.
- Determine if issues are infrastructure or application related and respond to alerts.

Alert notifications can also be viewed on custom web dashboards using the Alert tile type. See Tile Types for more information.

Host health web view

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Act on infrastructure and application violations using alerts

Process health web view

Database health web view

Share web dashboards

Infrastructure monitoring in AppMon Client

Host health monitoring

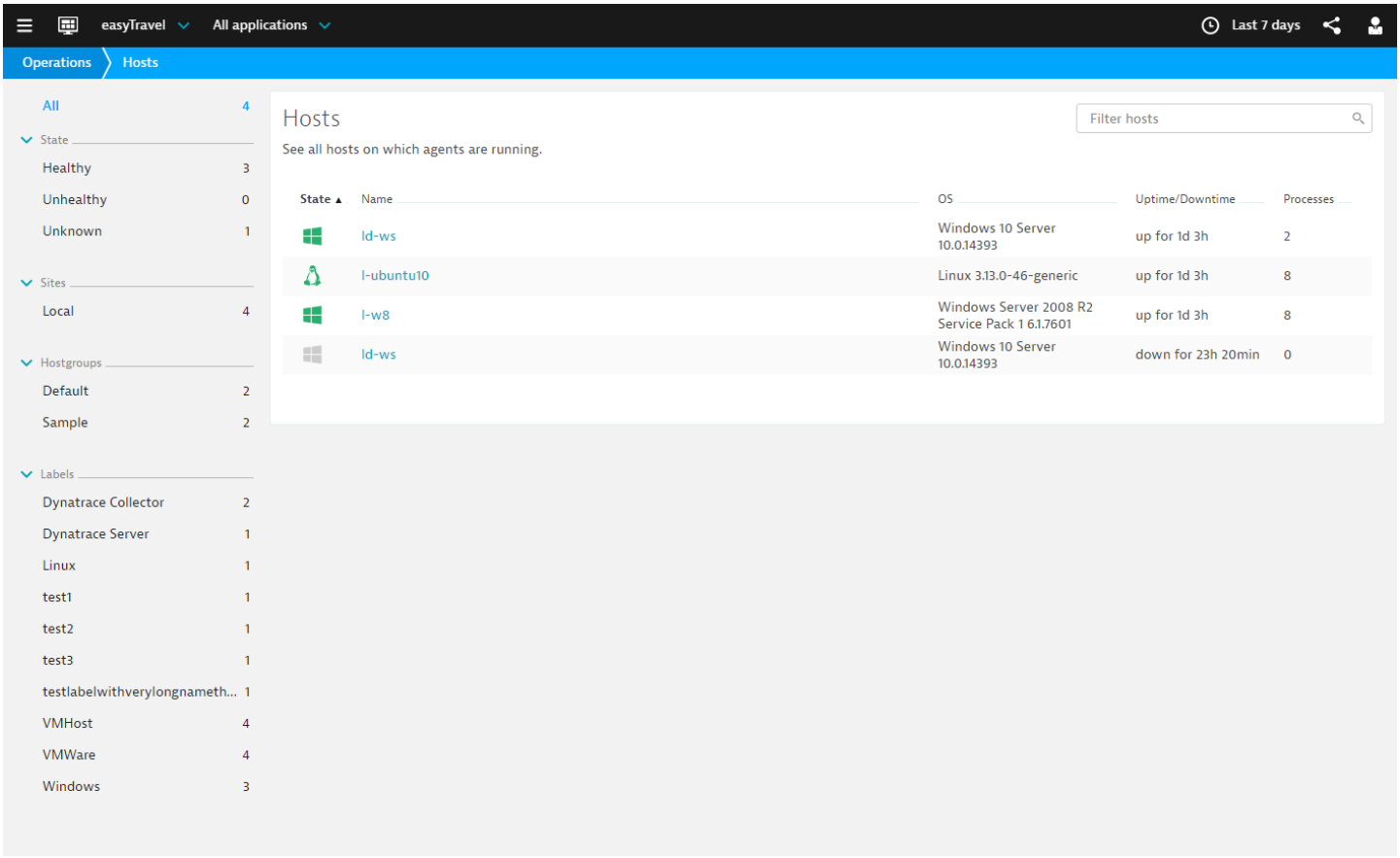
Process health monitoring

Using host and process health monitoring

Watch and learn

Monitoring infrastructure health in AppMon Web

The **Hosts** web view lets you instantly know the state of all monitored hosts in the current time scope for the selected System Profile and application. For each host list contains a pictogram, showing its operational system and color-coded health, the host name, the host OS info, and the amount of unhealthy/overall processes. You can filter hosts by one or several criteria, listed in the left pane.

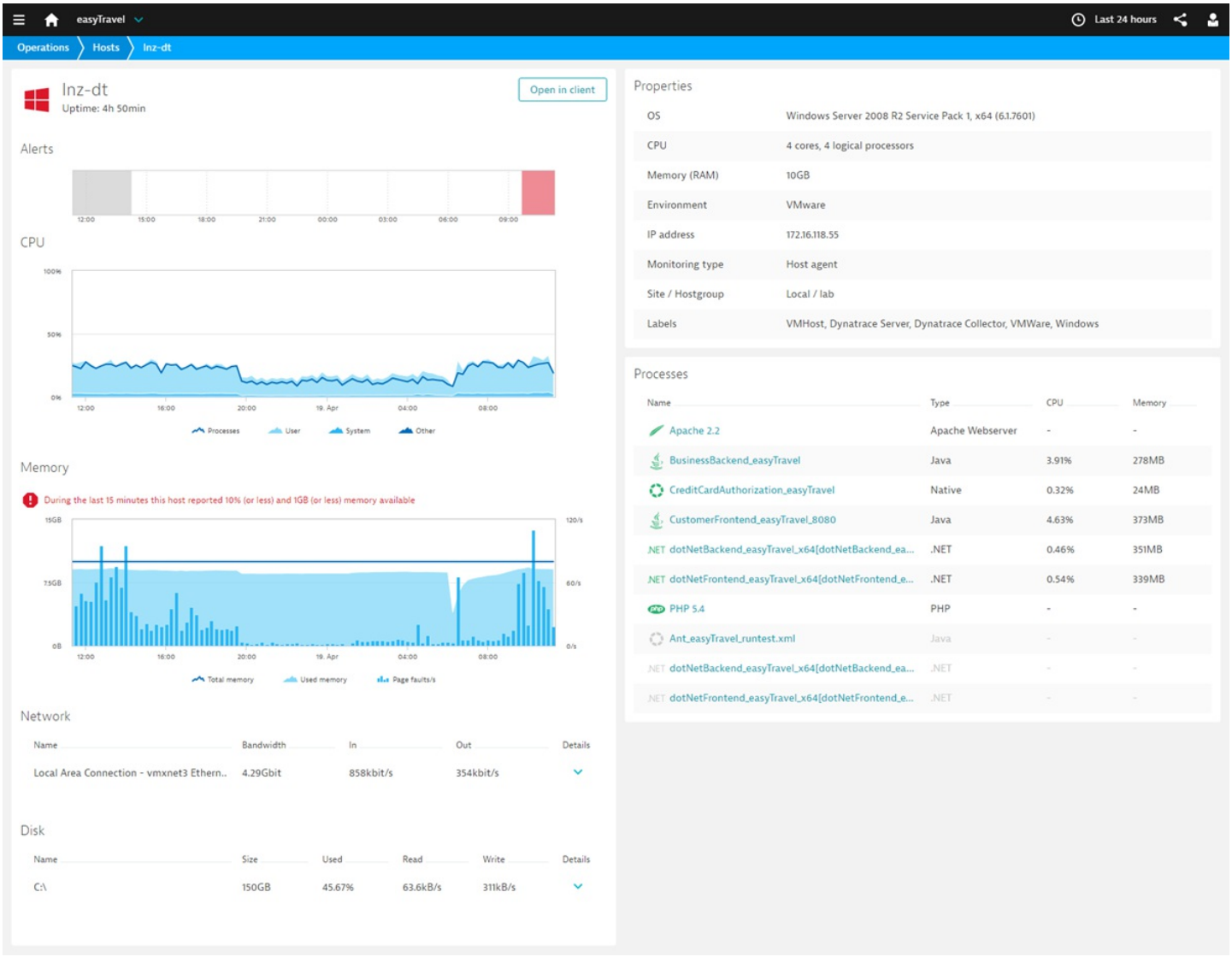


Host health web view

CPU, memory, network, and disk metrics captured by AppMon Agents calculate the host health. Health is based on thresholds defined for those metrics. Edit threshold values for the applicable host or host group in the **Dynatrace Settings > Infrastructure** section's **Host** or **Host groups** tabs. See Infrastructure to learn how.

Host details view

For a deeper drill down and root cause analysis of a failed host state, click the name of the host in the list to open the host details view. The host detail view contains charts for key metrics such as memory usage, CPU utilization, as well as disk and network usage for each hard disk and network interface. If any of the metrics violated thresholds within a selected time scope, a warning message will be displayed above its chart, telling what's wrong.



Host details view

The host details view also has a list of instrumented processes with basic metrics to quickly identify which processes uses the most memory or CPU on that host. To navigate from a host detail view to the detail view of a process, click the name of the process in the list of processes shown in the host detail view.

Additionally the host details view displays alarm history for the selected time scope. Click it to view the detailed alarm list. In the list, click **Infrastructure alerts** to switch to the alerts page.

Click **Open in client** to drill down for analysis in the AppMon Client, where you can create thread and memory dumps, and CPU samplings for efficient host issue resolution. See Host Health Monitoring in AppMon Client for more information. Please note that your AppMon Client or Webstart Client should be running for this function to work.

Process health web view

Related pages

Act on infrastructure and application violations using alerts

Host health web view

Database health web view

Share web dashboards

Infrastructure monitoring in AppMon Client

Host health monitoring

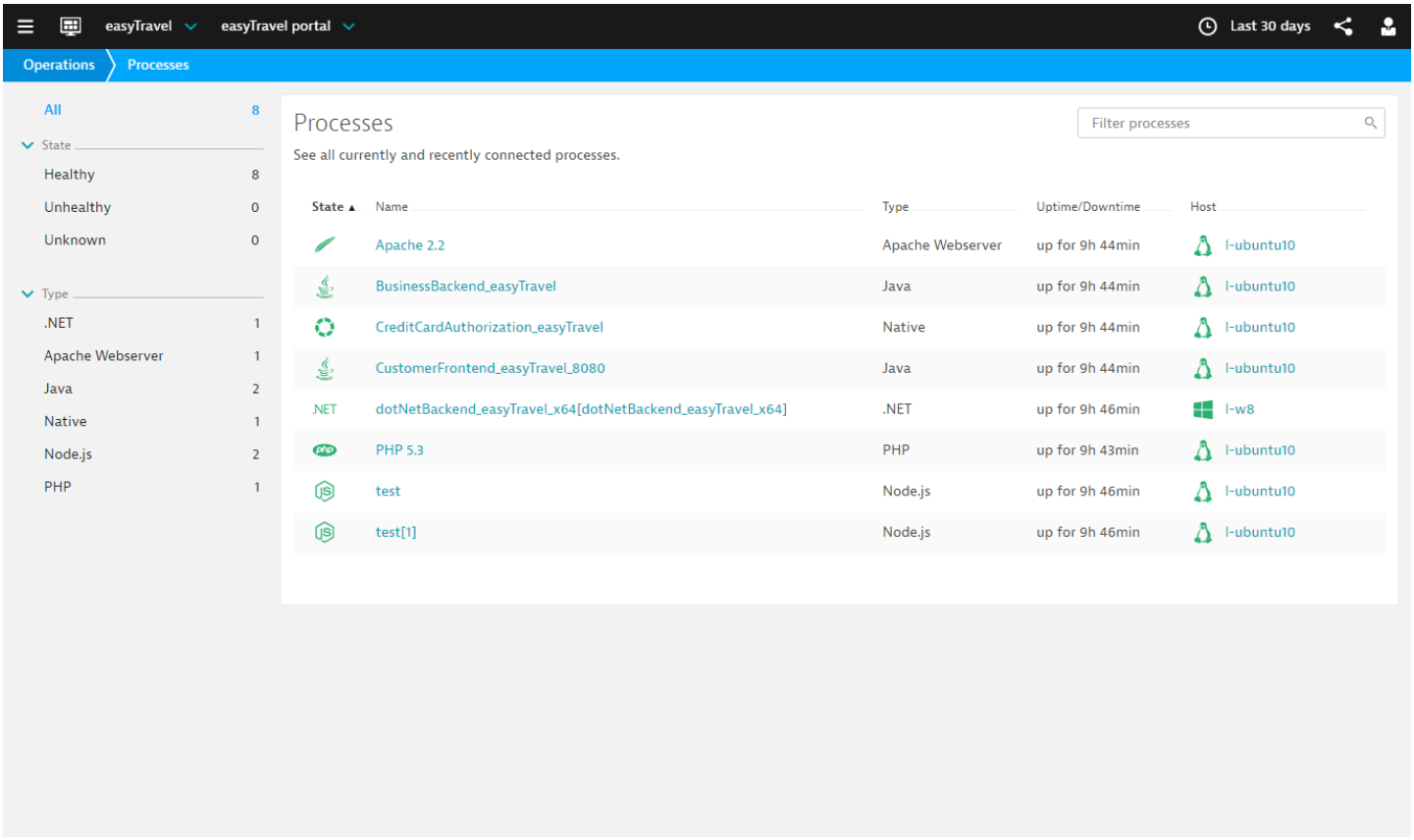
Process health monitoring

Using host and process health monitoring

Watch and learn

Monitoring infrastructure health in AppMon Web

Knowing the state of running processes lets you narrow down the cause of performance problems and bottlenecks. The **Processes** view shows all instrumented and monitored processes for a selected System Profile and lets you see at-a-glance which processes are unhealthy or not running. Each process listed includes pictogram for the process type and color-coded health, process name, how long the process has been running (for currently running processes), and the host to which it belongs. You can filter processes by one or several criteria, listed in the left pane.

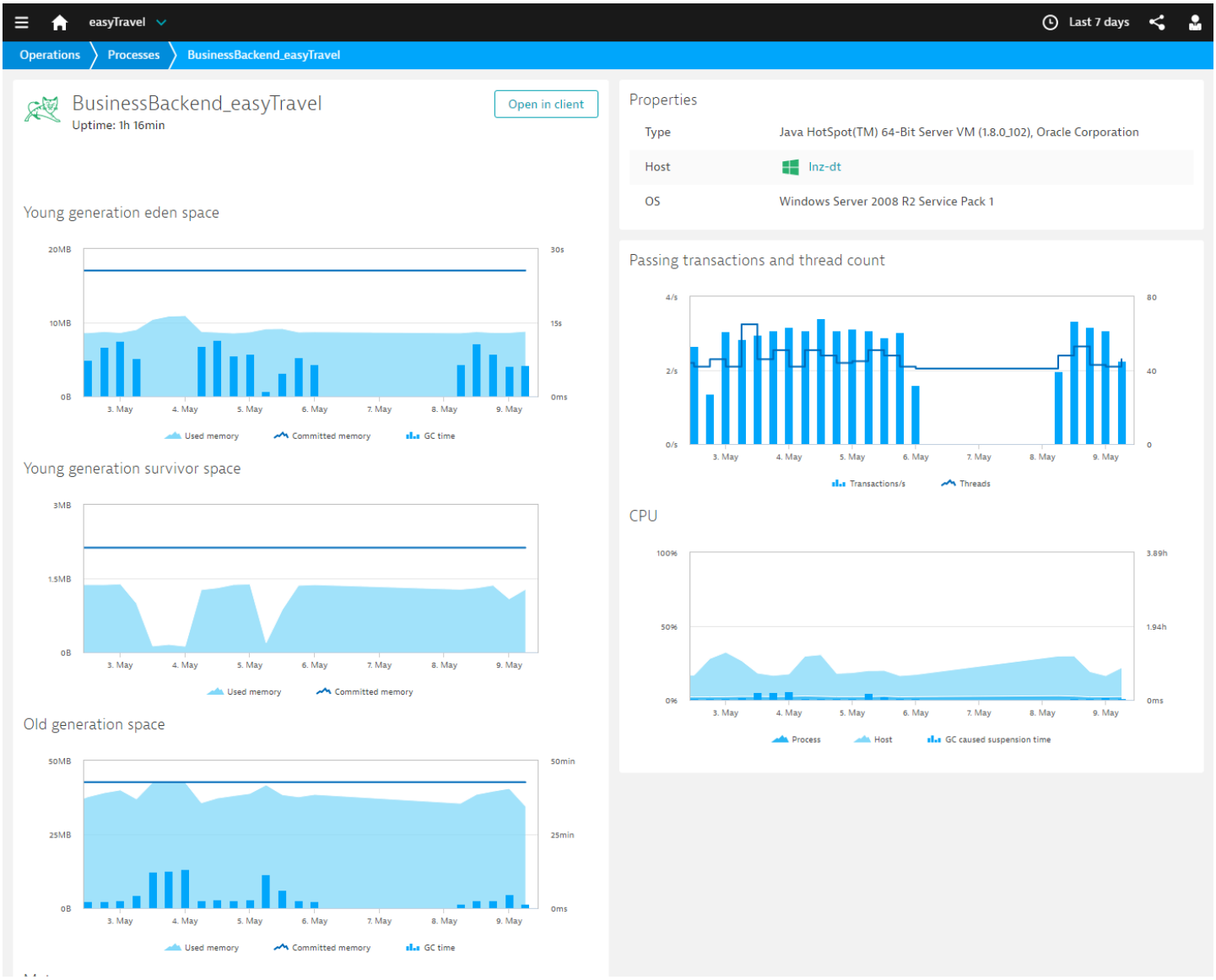


Process health wrb view

- A process is considered healthy (green label) if it spends less than 15% of its execution time in Garbage Collection (GC). GC naturally doesn't apply to native, just run state.
- The label is red if GC is above threshold or the process was terminated unexpectedly.
- Gray label: Information about the known process is unavailable, e.g. its Agent or Collector is disconnected.

Process details view

Click the process name to open its detailed view, where the key metrics are charted. To navigate from a process detail view to the detailed view of a host, where the process is running, click the name of the host in the process properties. The process detailed view contains charts for key metrics of the process.



Process details view

Click **Open in client** to drill down for analysis in the AppMon Client, where you can examine transaction flow of passing transactions, and create CPU samplings for efficient process issue resolution. See Process health monitoring in AppMon Client for more information. Please note that your AppMon Client or Webstart Client should be running for this function to work.

Tips

- Most Agents automatically deliver process health information. On Windows, Host Agent services must be in the Performance Monitor Users group to deliver process health information.
- The Host Monitoring Agent can monitor the availability of any process running on the host. Running processes defined within the process availability of hosts are listed along with processes monitored by other AppMon agents. Once no process of a configured process pattern is running, an incident will be created and the process pattern will be considered unhealthy.

Database health web view

Related pages

Act on infrastructure and application violations using alerts

Host health web view

Process health web view

Share web dashboards

Infrastructure monitoring in AppMon Client

Host health monitoring

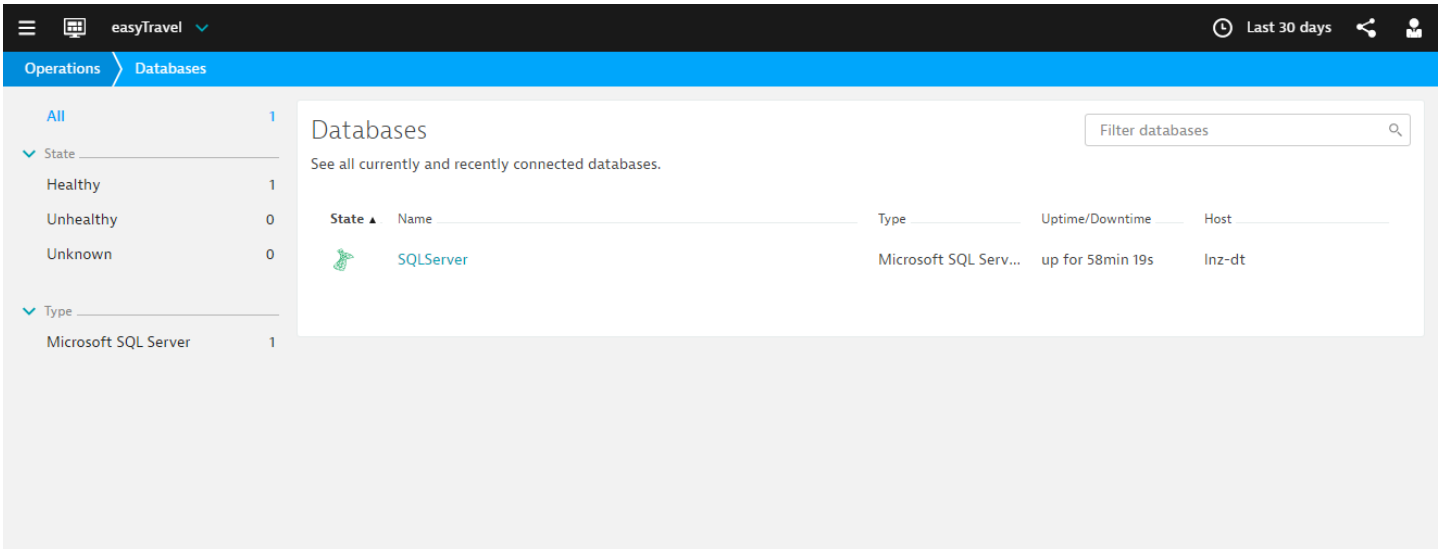
Process health monitoring

Using host and process health monitoring

Watch and learn

Monitoring infrastructure health in AppMon Web

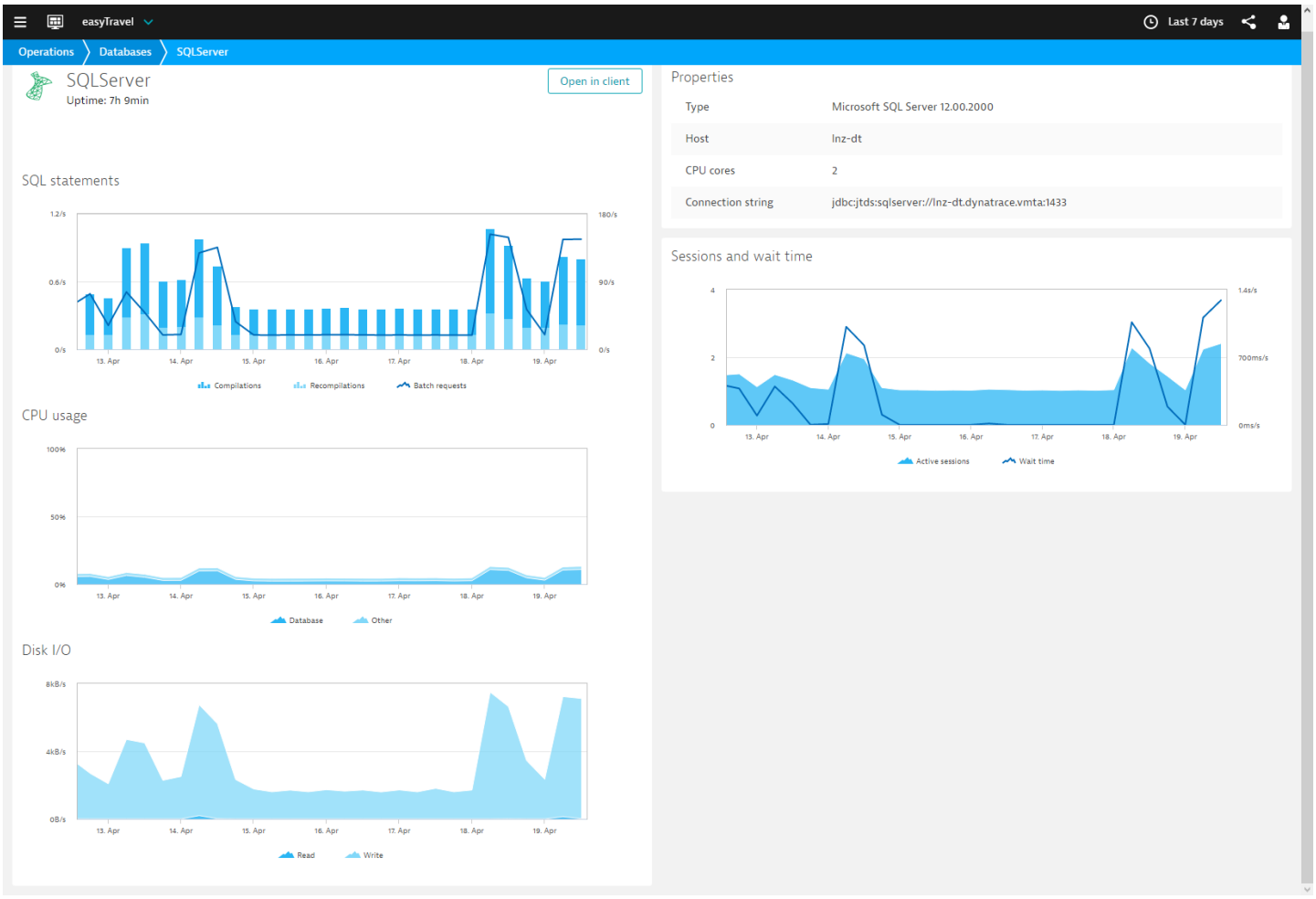
Knowing the state of databases lets you prevent performance problems and bottlenecks. The **Databases** view shows all instrumented and monitored databases for a selected System Profile and lets you see which databases are unhealthy. Each database listed includes pictogram for the technology type and color-coded health, database name, how long it has been running, and the host to which it belongs. You can filter databases by one or several criteria, listed in the left pane.



Database health web view

Database detailed view

Click the database name to open its detailed view, where the key metrics are charted.



Database details view

Click **Open in client** to drill down for analysis in the AppMon Client, where you can examine transaction flow of passing transactions, and create CPU samplings for efficient process issue resolution. See Process health monitoring in AppMon Client for more information. Please note that your AppMon Client or Webstart Client should be running for this function to work.

Act on infrastructure and application violations using alerts

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Create Incident Rules to set up alerts

Alert tile on the operations dashboard

Viewing additional incident details

View and set the scope of alerts

Focus on specific alerts for analysis

Analyze an incident in the AppMon Client

Related pages

Incidents and alerting

Host health web view

Process health web view

Database health web view

Infrastructure monitoring in AppMon Client

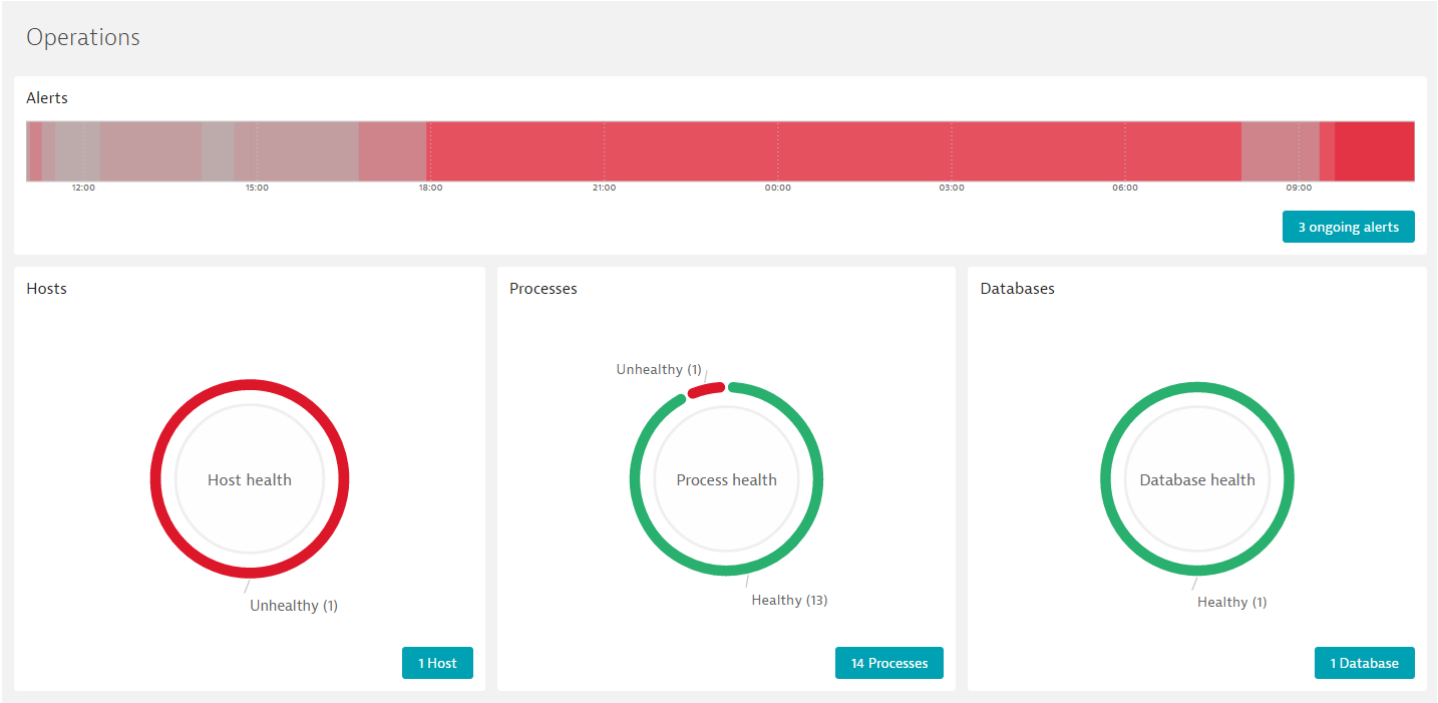
Incidents Overview dashlet

Host health monitoring

Process health monitoring

Using host and process health monitoring

Checking on and ensuring the health of your environments is an essential operations task wherever you are and at any point in the day. AppMon Web's Operations dashboard shows alerts from triggered Infrastructure and Application incidents, and the number of currently ongoing alerts. You can view the Operations dashboard from any browser-enabled device.



Create Incident Rules to set up alerts

After subscribing to measures and configuring thresholds for the measures, create incident rules to define whom to notify when the thresholds are exceeded and how to act on violations. See Incidents and Alerting to learn how.

What is an incident rule?

An *incident rule* in AppMon is a mapping between threshold violations of measures and the actions taken. Incident rules can be based on multiple measures and different severity levels of threshold violations to trigger notifications. Threshold checks are done every 10 seconds. The threshold may be for cyclic measures like CPU and memory that are sampled every 10 seconds, or for transactions.

If all the thresholds defined in the incident rule are violated, an incident is created and can be viewed in the Alert overview, even if no actions have been configured for the incident rule.

Once done, the **Alerts** tile of the **Operations** dashboard will display alerts.

Alert tile on the operations dashboard

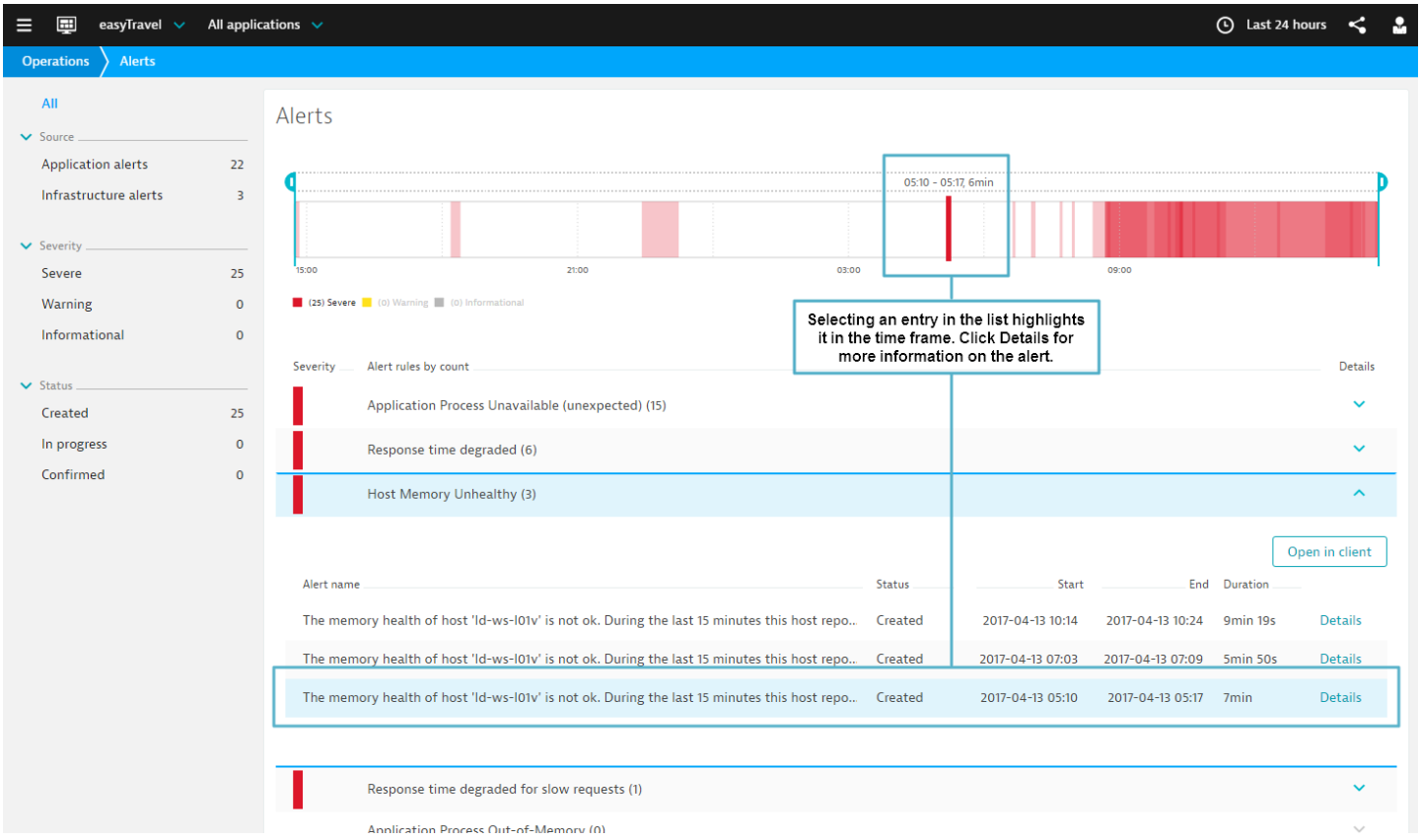
Click the **ongoing alerts** button in the **Operations dashboard** to open the **Alerts** page.

Using the alerts as an initial launching point in diagnosing or analyzing incidents, you can spot problem hotspots, where they come from, and easily share your insight across teams to resolve issues. Because alerts in the Alert Overview can be easily filtered to show infrastructure or application alerts, you can quickly determine if problems originate in your infrastructure or certain applications.

Viewing additional incident details

The time line and alert list in the **Alerts** overview are synchronized. Selecting an entry in the list highlights the incidents hash in the time line. The time of the incident and the incident's duration display above the hash. In the same way, selecting a hash in the time line highlights the alert entry for the incident in the Alert list.

Click **Details** on the right of the desired alert entry in the Alert list to view additional details about the incident. The **Alert Details** message displays the alert status information and the name of any person assigned to analyze the incident. The alert name and description also display. Click **Close** to close the **Alert Details** message.



View and set the scope of alerts

Alerts that display in the **Alerts** list of the **Alerts** page are those that fall within the range set by the alert severity and status. Severities are part of defining measure thresholds associated with rules. This information is set when configuring incident rules. See Incidents and Alerting to learn how.

Use the filter sidebar of the **Alerts** page to set the scope of viewed incidents based on alert severity and alert state.

Types of sources include:

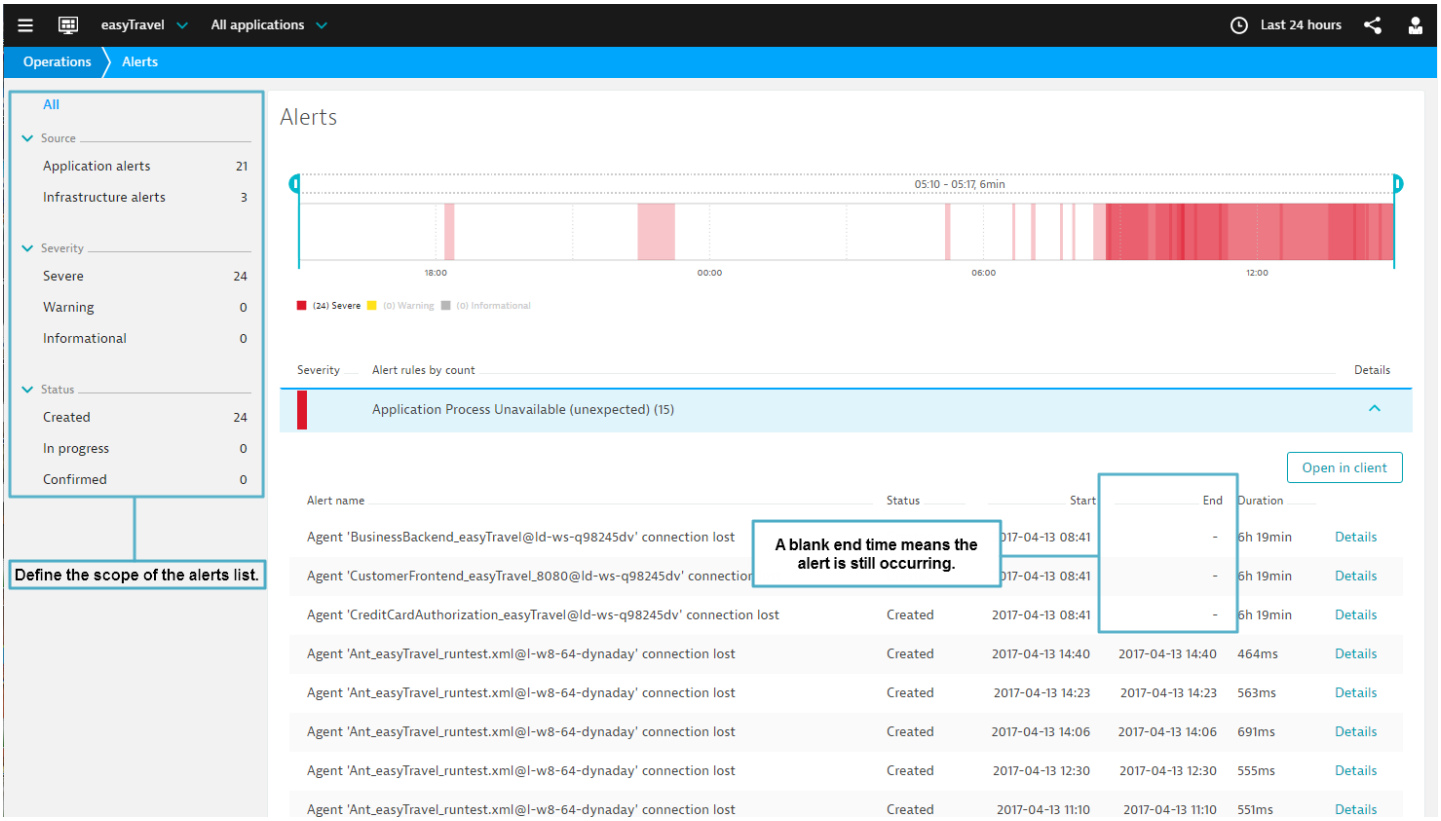
- **Applications alerts**
- **Infrastructure alerts**

Types of severity include:

- **Severe**: Displays only alerts whose threshold violations trigger a severe state.
- **Warning**: Display only alerts whose threshold violations trigger a warning.
- **Informational**: Display only alerts whose not defined as **Warning** or **Severe**.

Status reflects the current incident state, and includes:

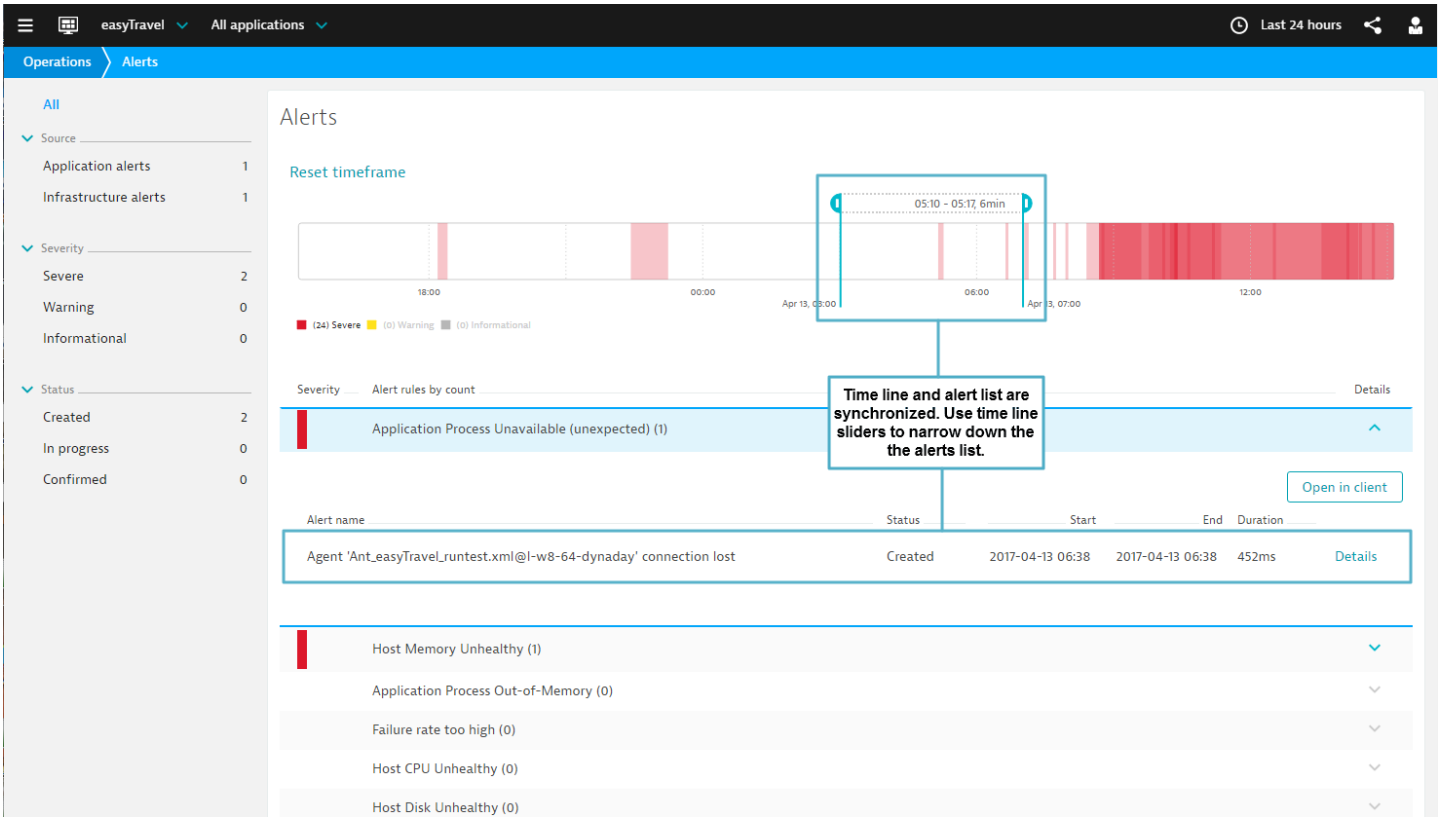
- **Created**: The incident has been created, but no statuses have been given to the incident.
- **In Progress**: The incident status has been changed to **In Progress** in the AppMon Client.
- **Confirmed**: The incident has been changed to **Confirmed** in the AppMon Client.



Focus on specific alerts for analysis

The **Alerts** page displays a running timeline with hash mark indicators for each incident that is configured to show alerts, along with a listing of qualified alerts that have been generated for incidents. These alerts are for incidents triggered during the time frame set for the data scope in AppMon Web. There are several ways to trim the alerts list to help find a specific alert:

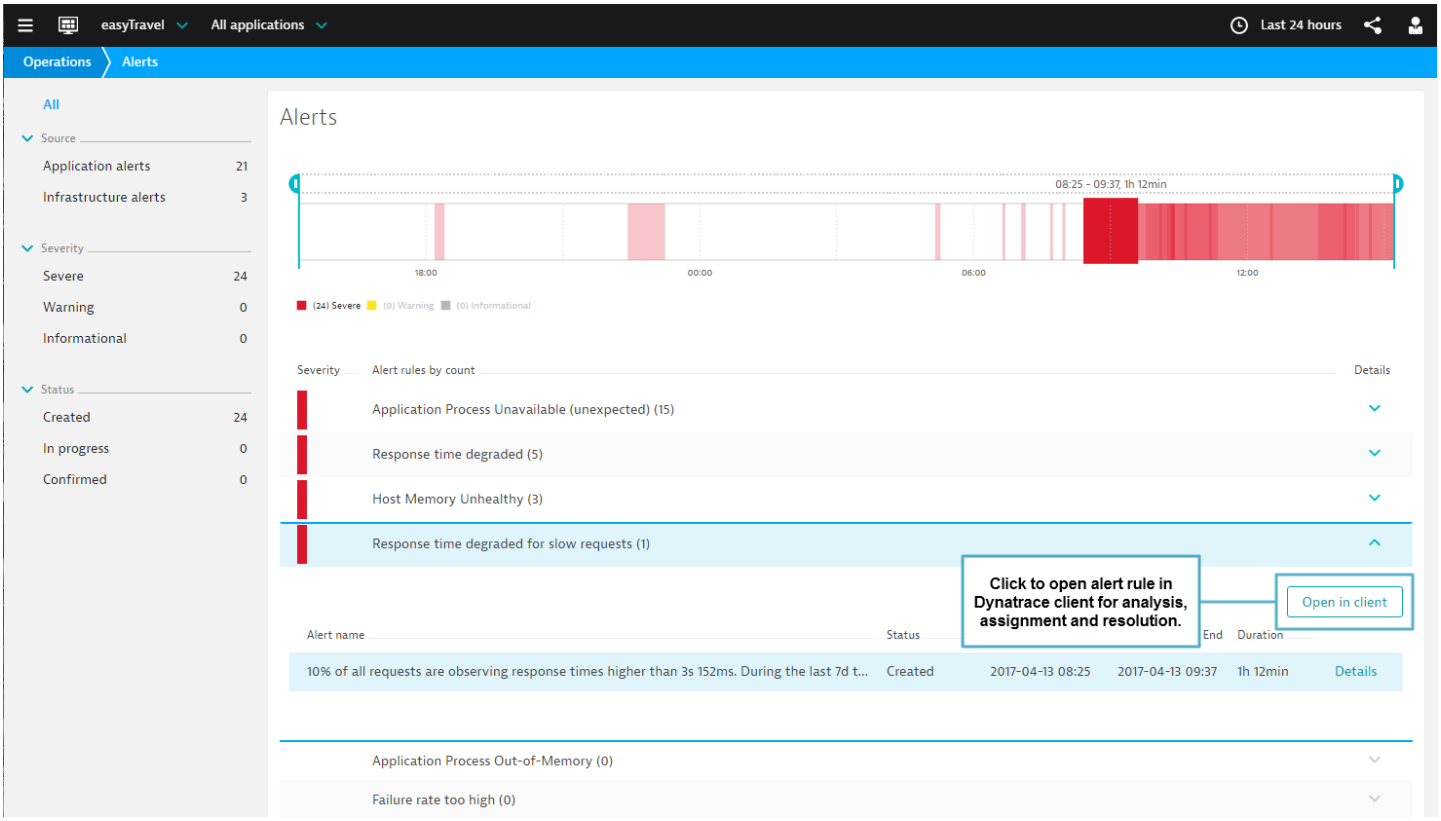
- Use the sliders to either side of the time line to reduce the time frame and exclude alerts from the list that fall outside of the time line. After moving the sliders, click **Reset timeframe** to move the sliders back to their original positions.
- In the Alert list, click the hash in the Severity column of the Alerts list to expand or collapse a list of alerts under each alert rule.

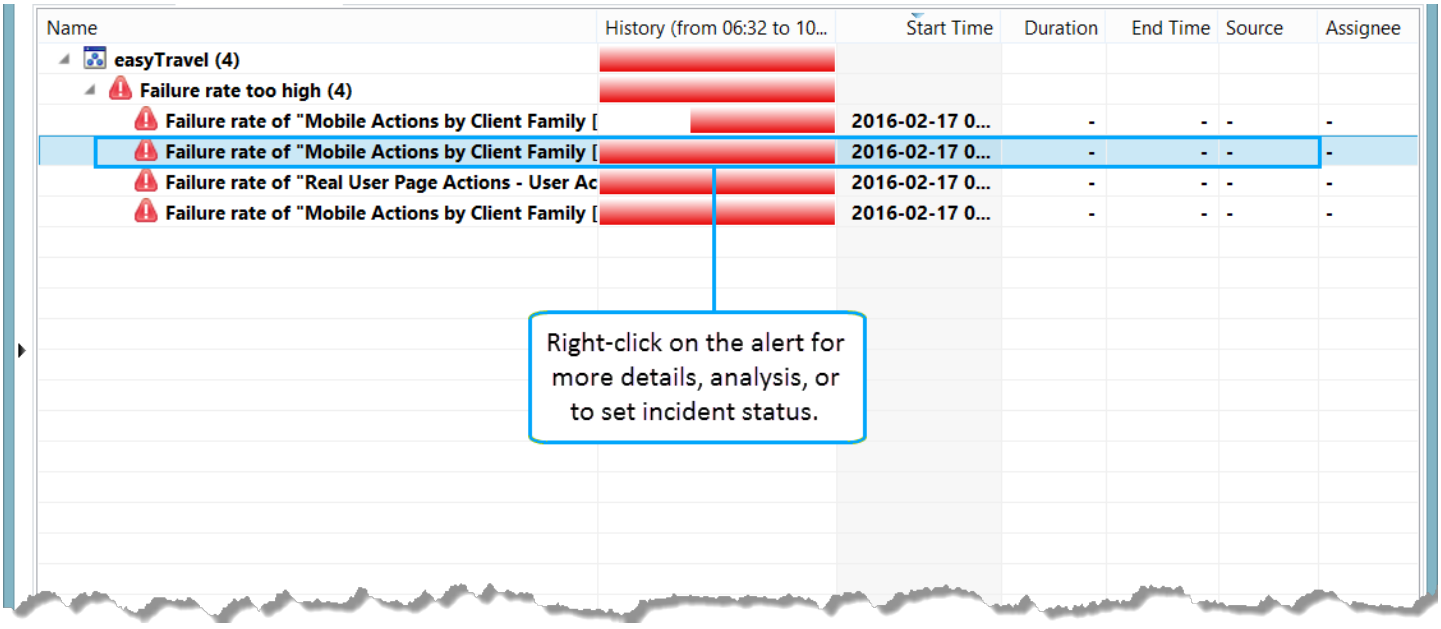


Analyze an incident in the AppMon Client

When you find the desired alert, click the **Open in Client** button for the Alert rule to drill down for detailed analysis. This opens an **Incidents** dashboard in the AppMon Client, where you can perform rule and incident analysis activities, including:

- See detailed information for the duration, description, status, state, and involved agents for each incident captured for the incident rule.
- Edit or disable the incident rule.
- Assign, confirm, or set all incidents or individual incidents within in a rule to In Progress.
- Export the incident session.





See Incidents Dashlet for more information in the Incidents dashboard in the AppMon Client.

Drilldown to the AppMon Client

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Client setup and drilldown

Share the open in client link

Related pages

Web dashboards

Set up the Webstart Client

Share web dashboards

Built in web dashboards

Custom web dashboards

AppMon Web FAQ & troubleshooting

AppMon Web provides convenient user experience management workflows and infrastructure monitoring together with the AppMon Client for fast and informative to root cause analysis. Several views in AppMon Web let you drill down from specific AppMon Web items to open the corresponding or detailed data in the AppMon Client.

Client setup and drilldown

1. Choose to use either the AppMon Client or AppMon Webstart Client.

   To use a locally installed **AppMon Client**, you must **start the Client before** using the drill down feature. Also, make sure that the AppMon Client is connected to the same AppMon Server as AppMon Web. You may also need to perform a browser configuration step.

To use the **Webstart Client**, you do not need to start it before using the drill down feature.

2. In AppMon Web, open the page containing the item for which you want to drill down. You can drill down on the following:

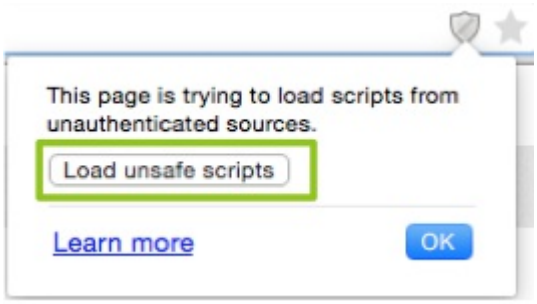
View/Action in AppMon Web	AppMon Client opens with
Business analytics > Visits > Visit details	Visits by UserExperience, UserExperienceReason
Operations > Infrastructure > Select an alert or alert rule	Incidents
Operations > Hosts > Select a Host name	Incidents
Operations > Hosts > Select a Host name	Host Information
Operations > Processes > Select a Process name	Process Information

Note

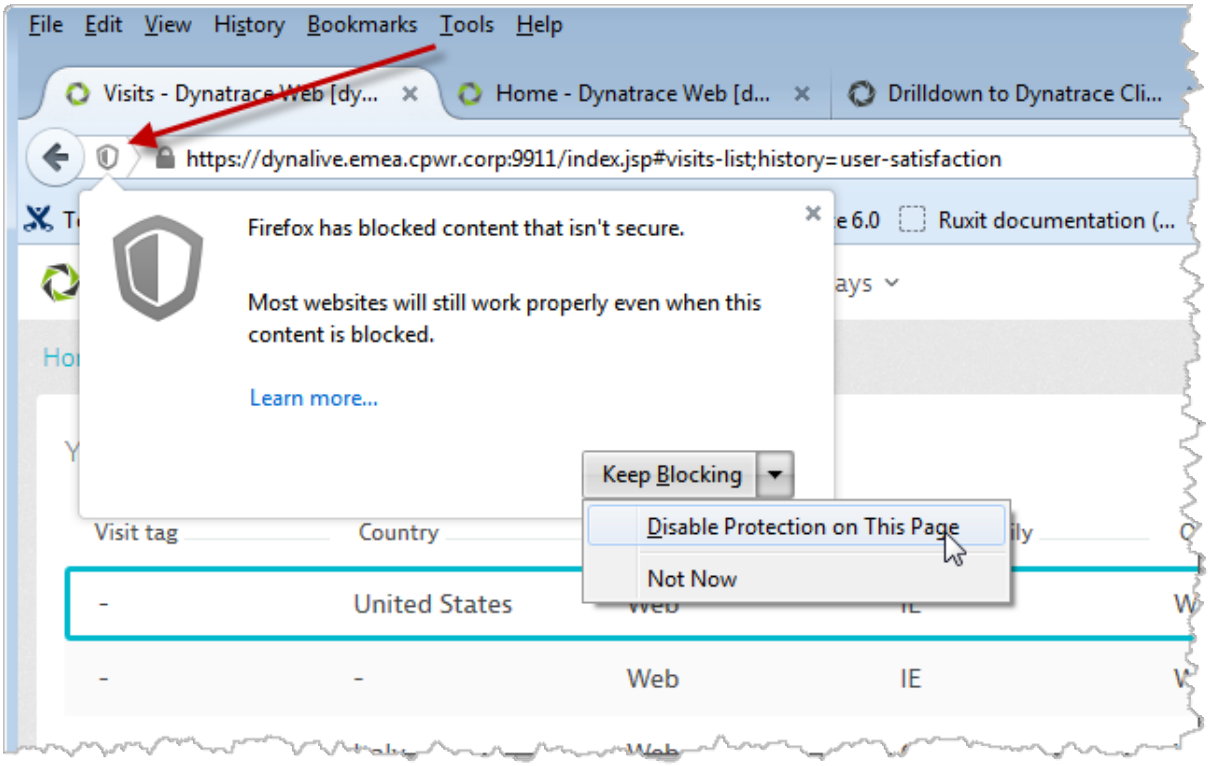
Depending on your browser, you may be prompted to authorize the loading of unsafe scripts, insecure content, or a certificate. If you want to drill down using the AppMon Client, you must comply with the browser authorization request. If you do not, then only the Webstart Client opens by default.

Browser authorization steps

Chrome Click the shield icon located at the far right of the URL box, and then select **Load unsafe scripts**.



Firefox Click the shield icon located at the far left of the URL box, and then select **Disable Protection on This Page**.



Internet Explorer Click **Show all content** in the **Security report shield**.



Safari No browser authorization configuration required.

3. Click the **Open in client** button located below the visits table.

The AppMon Client starts if it is running. If not, the is Webstart Client starts. A message box asks what the browser should do with the `client.jnlp` file. Open it with the suggested Java Web Start Launcher to start the Client.

The client opens and displays details for the selected information. with the selected visit information displayed.

Share the open in client link

After selecting a visit in AppMon Web, you can share the link to the Visits dashboard in the Webstart Client by clicking the **Share**



button. Copy the URL path and share with another person. When they click the link, they are prompted to download the `client.jnlp` file required to run the Webstart Client.

AppMon Web FAQ & troubleshooting

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Web dashboards

Why do I get a security warning when I try to access AppMon Web?

When attempting to log into AppMon Web, your browser may display a security warning. This is because your browser doesn't recognize the AppMon Web security certificate. If this message appears, you have several options depending on your browser type:

1. For all browsers, you can dismiss the security warning message and proceed to log into AppMon Web.
2. For Firefox, Internet Explorer and Safari you can add the Dynatrace certificate to the browser's list of trusted certificates.
3. For Chrome, you can ask your system administrator to install the AppMon Web security certificate in your system's keystore. See Certificate Management and sub-pages for more information.

Why do I get a JavaScript error when logging in?

To avoid login errors, make sure you are use a supported browser. See System Requirements.

Can someone else edit my custom web dashboard?

Custom web dashboards are only editable by the user who originally created the dashboard. In addition, shared custom web dashboards are only accessible from the link you provide. They do not appear in the My Dashboards list on the Dashboards view of AppMon Web. As a result, make sure to remind people to bookmark the link you send to them for future access. Learn more about sharing web dashboards.

Is it possible to make web dashboards available for people without a Dynatrace account?

Yes. But in order to keep control of which Data can be accessed - A user account intended for public sharing has to be created with permissions for the required System profile that needs to be accessed. Learn more.

Does stored session data appear in web dashboards?

No. Only *live session* data and measure data in the AppMon Performance Warehouse appears on web dashboards. A *stored session* is a memory dump, thread dump, sampling, or a former PurePath session that is stored on a hard drive. Learn more about session data.

Can I convert my AppMon Client dashboards to web dashboards?

No. At this time, dashboards created using the AppMon Client must be recreated as web dashboards. You can, however, create an HTML report of a AppMon Client dashboard. See Reporting for more details.

Can I use a different port than 9911 for AppMon Web?

You can modify the default AppMon Web port (`9911`) by setting the **webuihttpsport** attribute in the **server.config.xml** file. This file is located in the **/server/conf** folder on the AppMon Server machine. To modify the default AppMon Web port, the AppMon Server must be stopped and restarted.

See AppMon Web Administration for more details.

Status overview dashlets

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Agents Overview dashlet

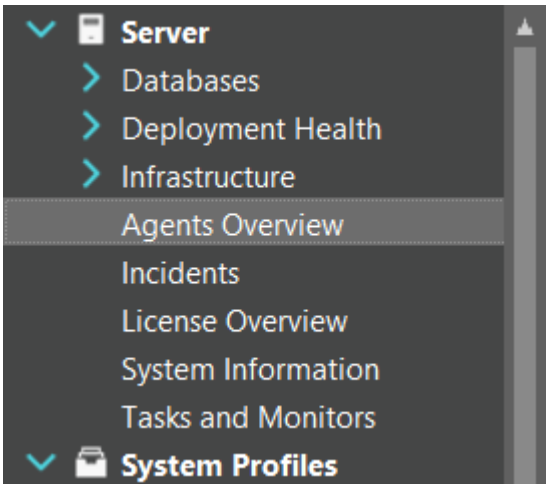
Incidents Overview dashlet

License Overview dashlet

System Information dashlet

Tasks and Monitors dashlet

The **Status Overview** node in the Cockpit contains dashlets that provide overall information about your system.



Dashlet	Description
Agents Overview dashlet	Lists information on all Agents registered with the AppMon Server.
Incidents dashlet	Lists the incident rules defined in the System Profiles and occurrences of thresholds violations.
License Overview dashlet	Lists license status and other general license information along with used agents and license consumption details.

System Information dashlet	Lists all the log and configuration files, dashboards, and other information for your AppMon installation.
Tasks and Monitors dashlet	Displays an overview of maintenance tasks and system Monitors for all System Profiles.

Agents Overview dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Dashlet details

Related pages

Status overview dashlets

Incidents Overview dashlet

License Overview dashlet

System Information dashlet

Tasks and Monitors dashlet

Agent Group - Agent Mapping

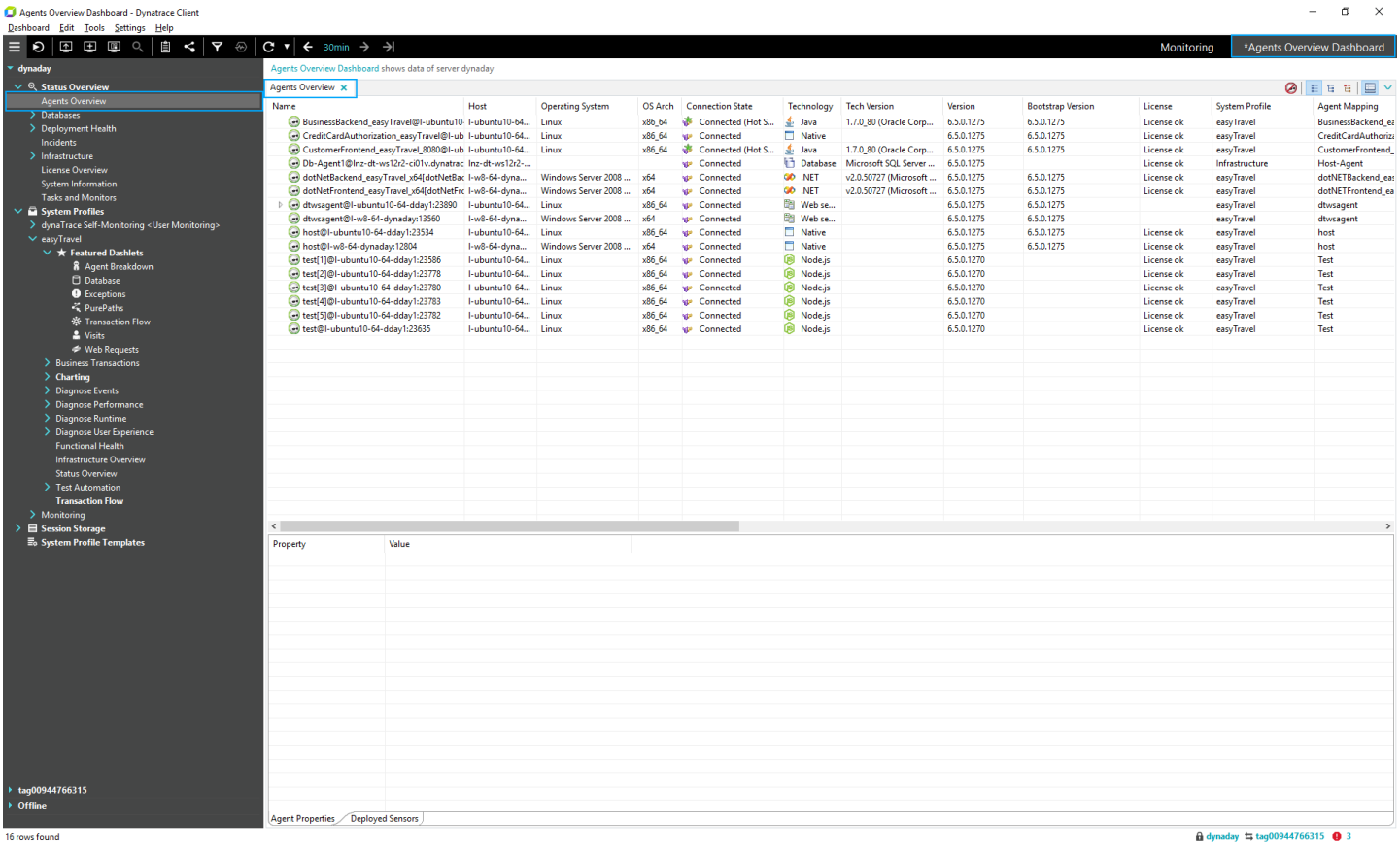
Agent timers

The **Agents Overview** dashlet provides information on the Agents registered on the AppMon Server, including their properties and deployed sensors.

Access the **Agents Overview** dashlet the following ways:

1. On the **Cockpits sidebar > Status Overview branch**.
2. When you got the **Monitoring Overview** dashboard or one of its drill-downs displayed (a System Profile or the Infrastructure selected) you find it in the cogwheel menu (right top).

From the **Status Overview** section in the Cockpit Dashlet:



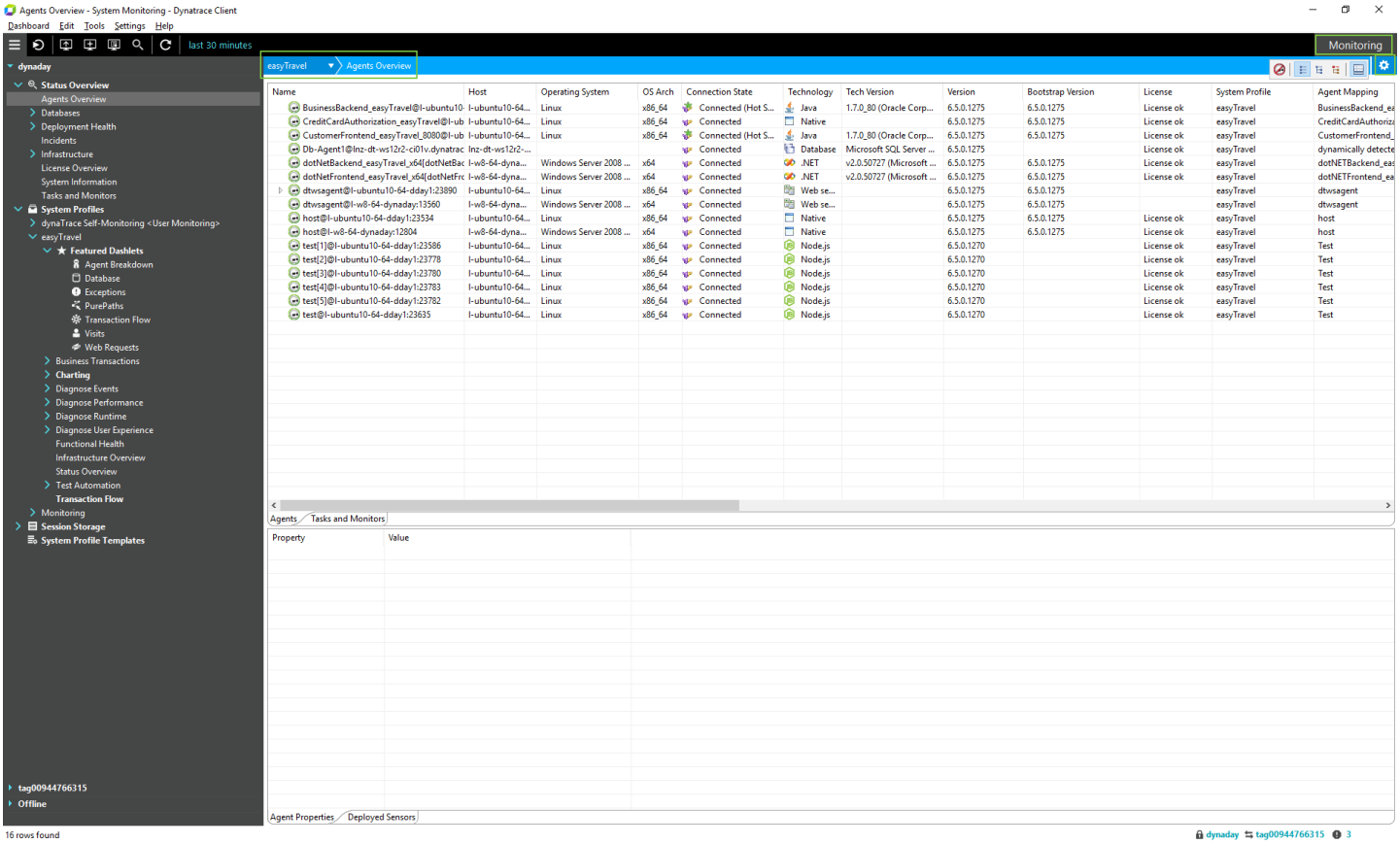
Double-click Agents Overview in the status overview branch

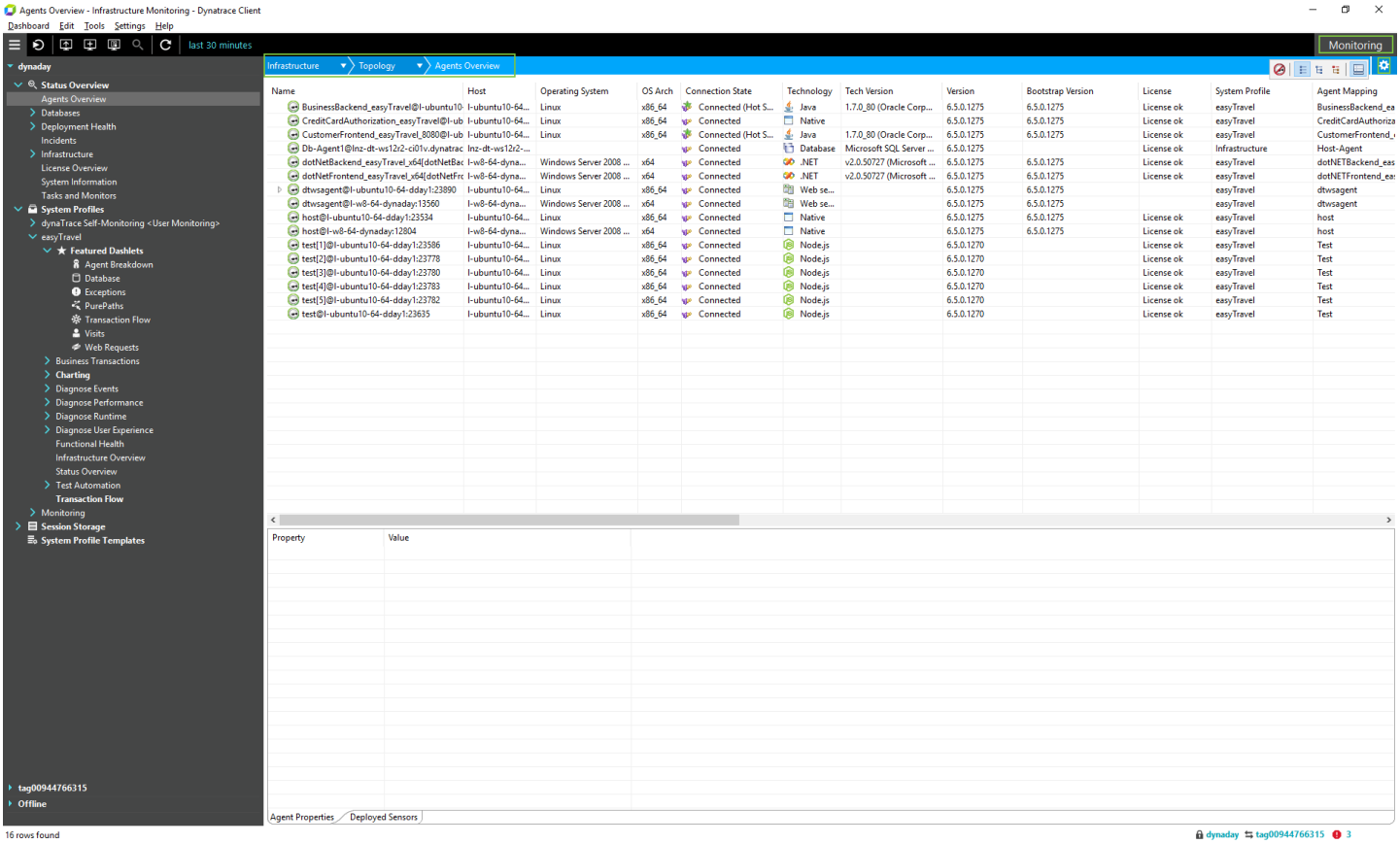
When on the Monitoring (blue with eye) dashboard / any drill-down, select Agents Overview from the Configure cogwheel menu.

Note

Agents Overview from **Cockpits > Status Overview** and **Monitoring > Infrastructure** display all Agents.

Monitoring > <SystemProfileName> displays only information for the Agents that are connected to the selected System Profile and for the Tasks and Monitors configured for it.





Dashlet details

The **Agent Overview** dashlet has two main sections. The Agent list shows all Agents that are registered on the connected Server. Below this section the **Details** pane provides information about the selected Agent.

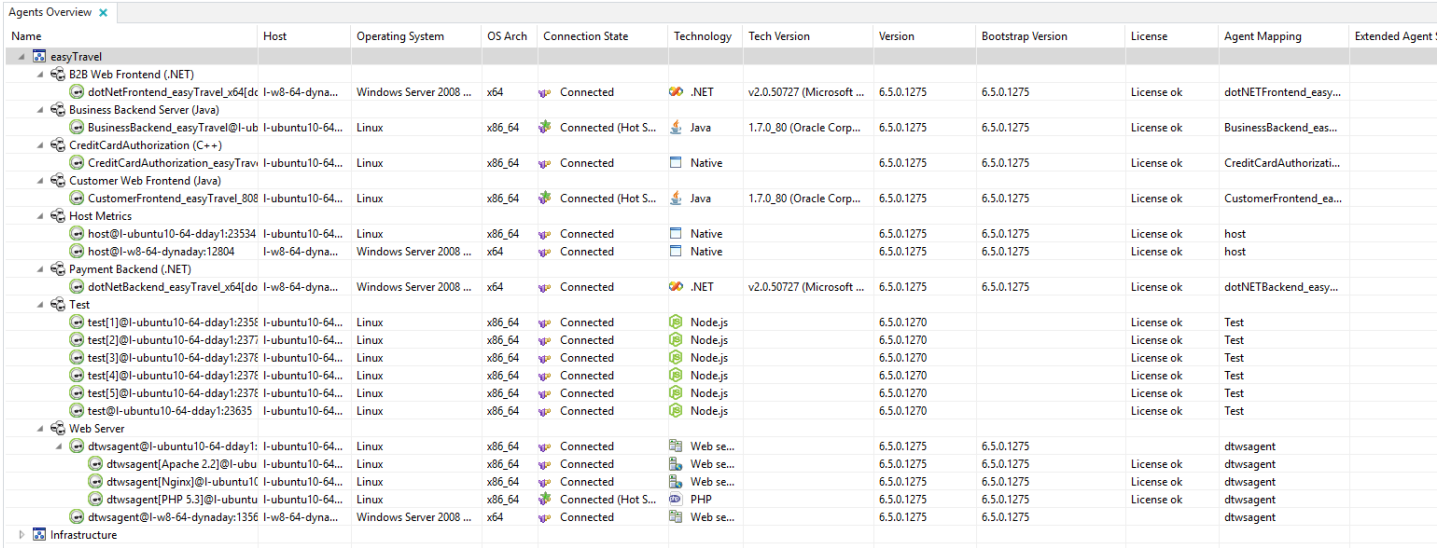
Toolbar

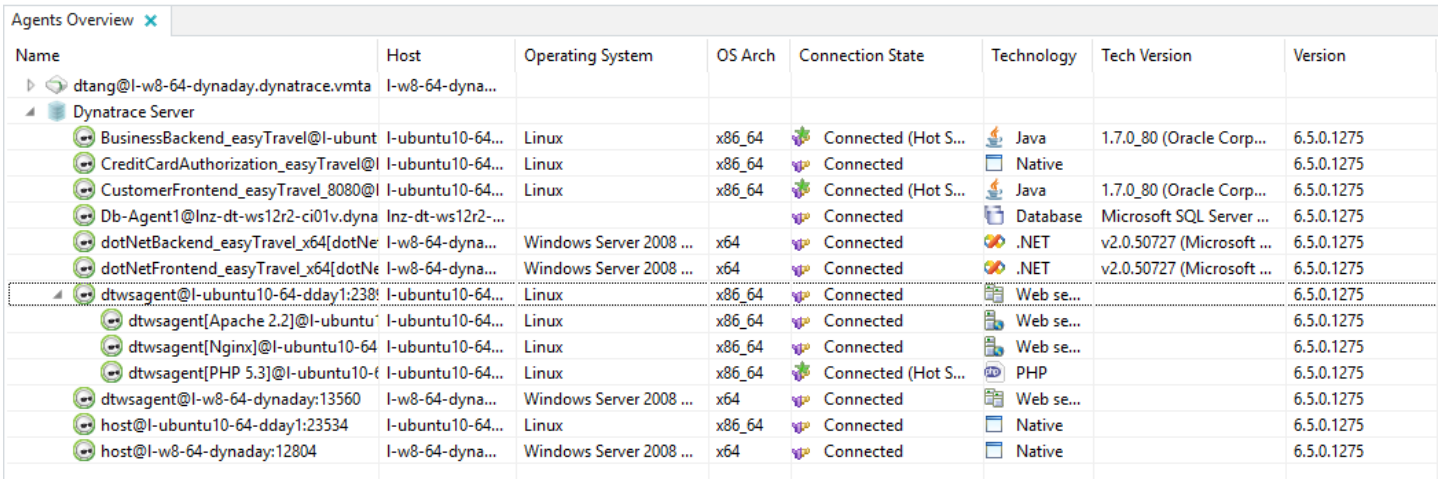
When the cursor is in the dashlet, the toolbar appears at the top left corner. It provides the following functions:



Agents Overview toolbox

- **Show offline Agents**: List Agents that once were connected to a System Profile but are no longer connected. Offline Agents are removed automatically after 72 hours.
- Organize table by **Agents**: This is the default table view, shown in the figure above. The Agents are listed in alphabetical order.
- Show **Logical Hierarchy**: Agents are organized by System Profile and Agent Group, as shown in the first figure below.
- Show **Physical Hierarchy**: Agents are listed under the Collector to which they are connected, as shown in the second figure below.
- **Show or hide the Details Pane** at the bottom of the dashlet.





Agent list

Agent states

Icon	Description
	The Agent is connected and up to date.
	The Agent is not connected.
	There is a problem with the Agent (e.g., it does not match any System Profile). To get more information, right-click the Agent and select **Details**.
	The application monitored by the application must be restarted to get the most recent Agent version injected.
	The Agent is outdated and is not bootstrapped. You need to manually configure the Agent to use the current Agent library.
	The Agent is connected to an outdated AppMon Collector. You need to restart the Collector.
	The Agent does not match the rollout criteria of the controlled update rollout. Therefore, it is not updated (see Updates for further details).
	The agent will not be updated after a restart because automatic Agent Updates are disabled.

The columns in the Agent list vary depending on the selected view. Most column widths can be adjusted, but you may need to scroll right to view all columns.

Click here for descriptions of all columns

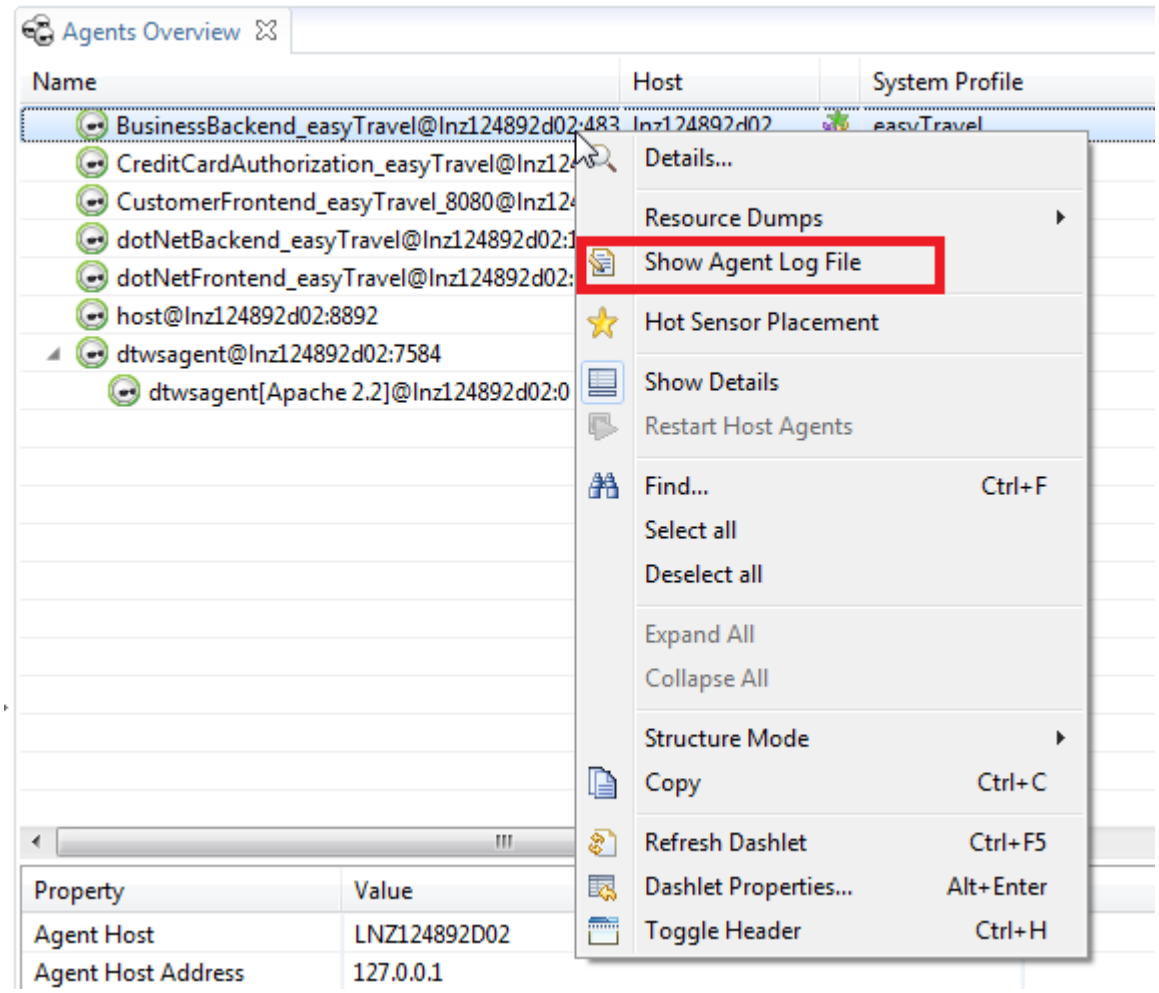
Column	View	Description
Name	Agents	List of agents in alphabetical order. The presented icon symbolizes the state of the agent (see table above). Depending on the technology type, supplemental data, such as the version or app name pool, is appended to the user-defined agent name in brackets. For example, **Apache 2.2** is automatically appended after the agent name **dtwsagent**, as shown in the following agent name example: `dtwsagent**[Apache 2.2]** @l-ubuntu10-64-1313` See the list below for a description of the syntax and origin of data appearing in brackets per technology type. Note that this data is not configurable. Java • WebLogic: [<node>] ◦ <node>: taken from “-Dweblogic.Name=” parameter in command line • WebSphere: [<cell>-<node>-<server>] ◦ <cell>: taken from 3rd last parameter in command line OR “cell_name” environment variable ◦ <node>: taken from last parameter in command line OR “node_name” environment variable ◦ <server>: taken from “server_specific_name” environment variable .NET: [<appdomain>] • <appdomain>: taken from “-ap” parameter in command line Apache: [Apache <version>-<node>] • <node>: taken from ApacheNodeName value in httpd.conf IIS: [IIS <version>-<apppool>] • <apppool>: queried from internal API Nginx: [Nginx] PHP: [PHP <version>] Mainframe • IMS Connect: [ICON-<jobname>] • IMS Application: [IAPP-<region>] • CICS Application: [CICS-<suffix>] ◦ <suffix>: can be configured via Client to be either subsystem ID, SNA ID, or jobname • else: [<type>-<subsystem>] ◦ <type>: can be either DB2, ICTL, ZDC
	Logical Hierarchy	System Profile > Agent Group > Agent tree.
	Physical Hierarchy	Collector > Agent tree.
Host	all	The host to which the Agent is connected.
Connection State	all	This column, to the right of the **Hosts** column, does not have a heading. It displays an icon that indicates the Agent’s connection state.

System Profile	Agents Physical Hierarchy	The System Profile used by the Agent.
Agent Mapping	all	The configuration of the System Profile used by the Agent.
Agent Group	Agent Physical Hierarchy	The Agent Group into which the Agent is mapped.
Technology Type	all	This column, to the right of the **Agent Group** column, does not have a heading. It lists the technology type: Java or .NET.
Version	all	The AppMon version: *major.minor.revision.build* .
Bootstrapped	all	The value is **true** if the Agent is bootstrapped, otherwise **false**.
License	all	Indicates whether the license is valid.
Started	all	The date and time the Agent was started.
Event Count	all	Number of events sent by the Agent so far.
Class Load Count	all	Number of classes loaded by the JVM or CLR associated with the Agent.
Total Class Load Count	all	Number of classes loaded by the associated JVM or CLR since startup.
Total PurePath Count	all	Number of PurePaths recorded by the Agent.
Total Execution Time	all	Total execution time in milliseconds for all instrumented methods.
Total CPU Time	all	Total CPU time in milliseconds for all instrumented methods.
Connected	all	The value is **true** if the Agent is connected, otherwise **false**.
Capture	all	The value is **true** if capturing is enabled, or **false** if it is disabled.

Total Skipped Events	all	Total number of events that were skipped because the Agent's saturation threshold was exceeded.
Skipped PurePaths	all	Total number of PurePaths that were skipped because the Agent's saturation threshold was exceeded.
Collector	all	Name of the Collector to which the Agent is connected.
Data Sent	Physical Hierarchy	Data sent to the Collector, in Bytes per second.
Data Received	Physical Hierarchy	Data received from the Collector, in Bytes per second.
MPS	all	Average number of nodes collected per second.
Average PurePaths Length	all	Average length of all the PurePaths stored in memory.
Completed PurePaths	all	Number of completed PurePaths per interval.
Skipped Events	all	Average number of skipped events per interval.
Send Buffer Queue Used	Logical Hierarchy Physical Hierarchy	Size of the Send Buffer Queue, in megabytes.
Send Buffer Queue Committed	Logical Hierarchy Physical Hierarchy	Number of megabytes committed for the Send Buffer Queue.
Agent ID	Logical Hierarchy Physical Hierarchy	The unique identifier for the Agent on the AppMon Server.

Show agent log file

You can access an Agent's log file directly from the Agents Overview by right-clicking the Agent and selecting **Show Agent Log File** from the context menu. The **System Information Dashlet** dashlet is added to the dashboard and the first log file of the selected agent is automatically displayed. For sub-agents, this command also opens also the first log file of the parent agent. The log files of any sub-agent are also stored within the log files of their parent agents.



Context Menu – Show Agent Log File

Note

Because the **first** log file in the System Information Dashlet tree displays, you may need to manually search the tree for the desired log file. It should be found within the same hierarchy level as the opened file.

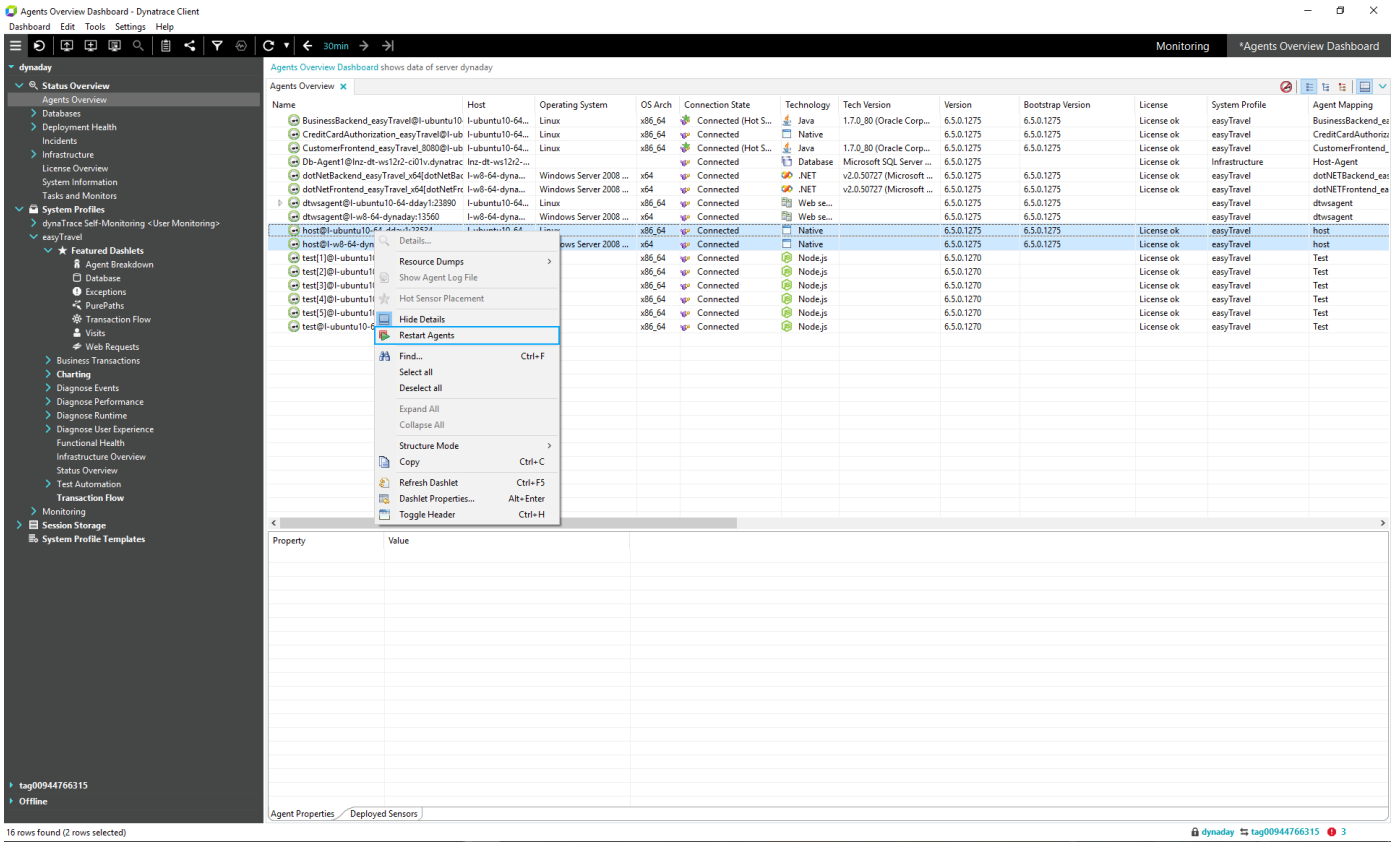
Restart host agents from client

After an upgrade or patch, you must restart your Host Agents. You can perform the restart from the AppMon Client. Host Agent restarts are limited to three tries every 30 seconds, to prevent an erroneous infinite loop.

> **Note**
>
> For any Agent that is at an AppMon version earlier than 5.5, this feature is not implemented by the auto-update. You must manually upgrade the Agent to use this feature.

Host Agents can be restarted from the Client on all supported platforms. The running Host Agent process keeps its PID. Windows Services are not informed about the internal restart.
Only a user with the Administrator role can use the Host Agent restart function.

1. Select one or more Host Agent(s) in the **Agent Overview** dashlet.
   Selected Agents that are not Host Agents are not affected by the restart mechanism, so, you if you want to restart all Host Agents you can just select all the Agents.
2. Right-click and select **Restart Host Agents** from the context menu.



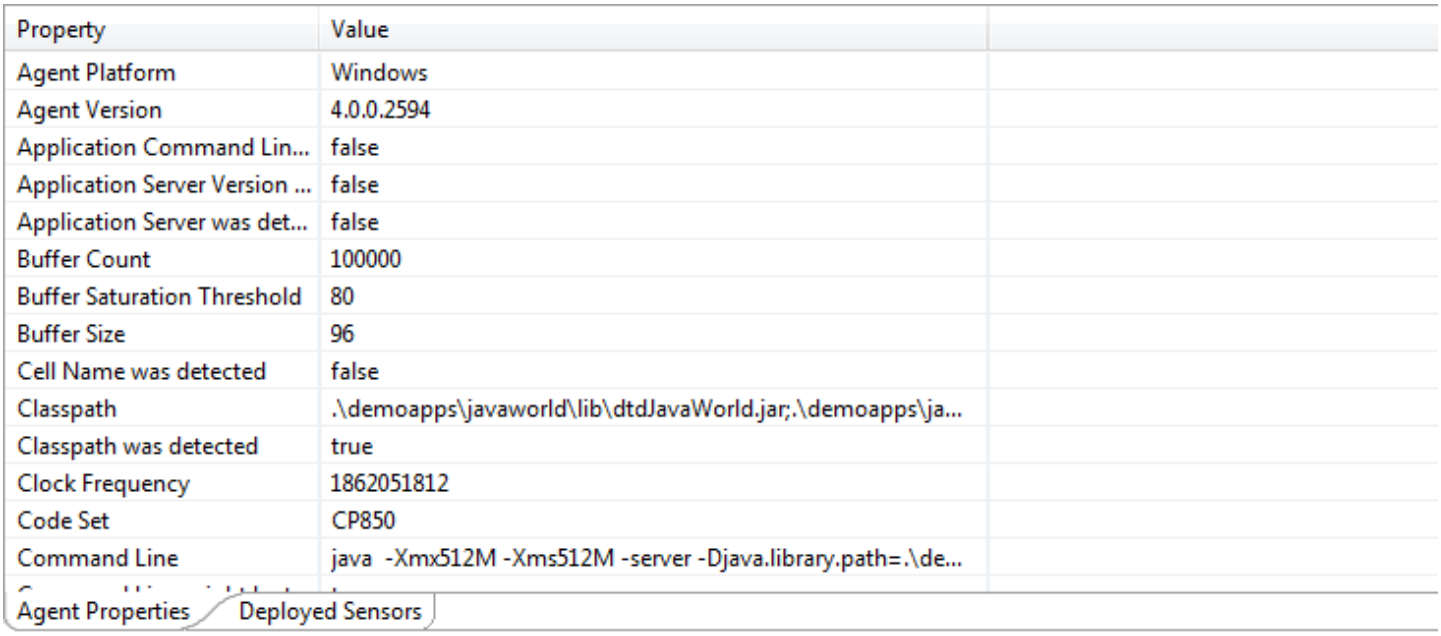
3. When prompted, confirm the restart.

Agent details

The Agent Details section contains tabs to list values for the **Agent Properties** of the selected Agent and information about the **Deployed Sensors** of the selected Agent.

Agent properties

This tab provides detailed information about the environment of the AppMon Agent currently selected in the Agents table.



Property	Value
Agent Platform	Windows
Agent Version	4.0.0.2594
Application Command Lin...	false
Application Server Version ...	false
Application Server was det...	false
Buffer Count	100000
Buffer Saturation Threshold	80
Buffer Size	96
Cell Name was detected	false
Classpath	.\demoapps\javaworld\lib\dtdJavaWorld.jar;.\demoapps\ja...
Classpath was detected	true
Clock Frequency	1862051812
Code Set	CP850
Command Line	java -Xmx512M -Xms512M -server -Djava.library.path=.\de...

Agent Properties | Deployed Sensors

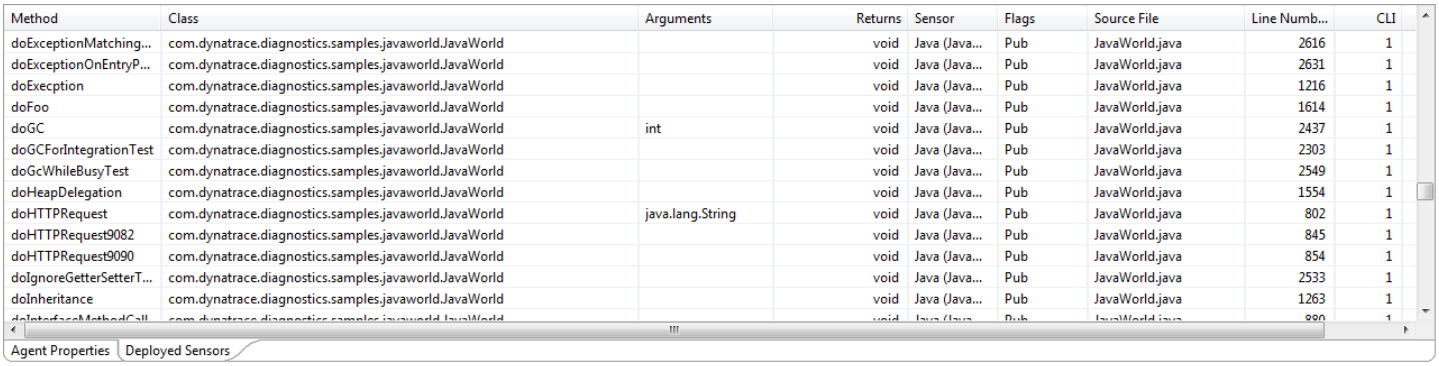
Agent Properties Tab

Deployed sensors

This tab provides information about the instrumented methods for the selected AppMon Agent instance.

- **Method**: The method name.
- **Class**: The name of the class containing the method.
- **Arguments**: Argument types of the method.
- **Returns**: Return type of the method.
- **Api**: API(s) assigned to the method.
- **Sensor**: Sensor(s) applied to the method.

- **Flags**: Access modifiers for this method * Pub = public
  - Pro = protected
  - Pri = private
  - Sta = static
  - Nat = native
  - Fin = final
  - Syn = synchronized
  - Str = strict
- **Source File**: Name of the source file that contains the method.
- **Line Number**: Line number of the method within the source file.



Deployed Sensors Tab

Through the **Deployed Sensors** tab, you can view decompiled versions of the instrumented classes. To view the decompiled code of a class, right-click the method and select the appropriate item on the **Source Lookup** cascading menu:.

- **Open in IDE**: Display the source code in Eclipse or Visual Studio. Only available if the appropriate Plugin is installed. See Eclipse Integration Plugin and Visual Studio Extension.
- **Decompile Byte Code**: Open a view that displays the byte code representation of the selected method.
- **Decompile Source Code**: Open a view that shows the source code representation of the selected method. Only available if JAD decompiler is installed.
- **Decompile mixed**: Open a view that shows a mixed representation, containing both byte code and source code of the selected method. Only available if JAD decompiler is installed.
- **Save Byte Code**: Save the byte code of the class containing the selected method. Only available if JAD decompiler is installed.
- **Open in Reflector**: Open the decompiled IL Code in the .NET Reflector. Only available if .NET Reflector is installed.

The original byte code is saved and used for decompiling, so no instrumentation code is visible.

You can use the commands on the **Add Sensor Rules** cascading menu to define Sensor rules for the selected methods:

- **Refine with Callee Methods**: All methods that are called by the selected methods are retrieved and can be used as input for new Inclusion Sensor Rules.
- **Exclude Selected Methods**: All selected methods can be used as input for new Exclusion Sensor Rules.
- **Include Selected Methods**: All selected methods can be used as input for new Inclusion Sensor Rules.

Incidents Overview dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Incident state and incident rule state

Incident count

History

Drilldowns

Table columns

Context menu

Related pages

Status overview dashlets

Agents Overview dashlet

License Overview dashlet

System Information dashlet

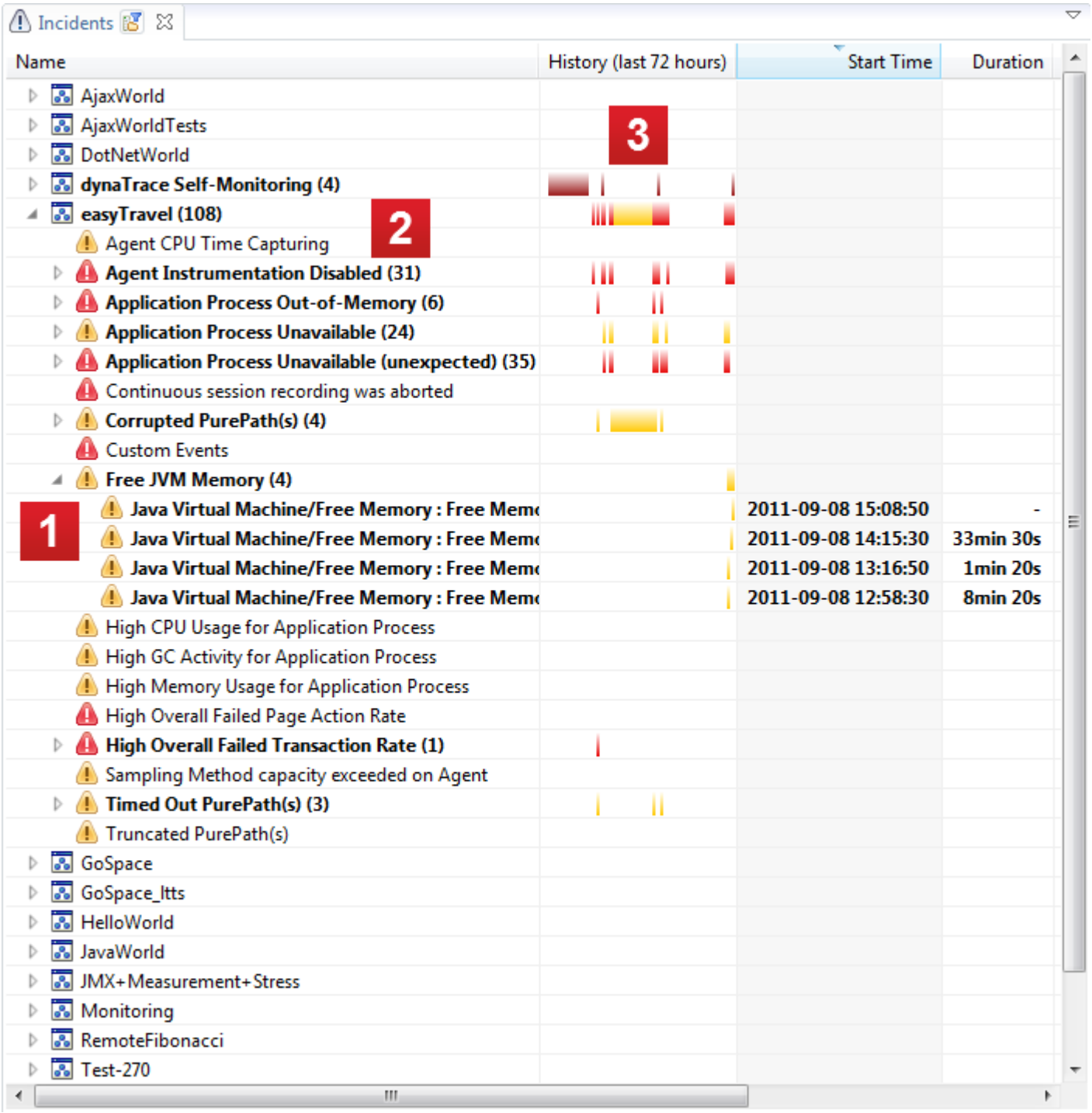
Tasks and Monitors dashlet

The **Incidents** dashlet is the entry point for managing and querying incident information across all System Profiles.

By default, the dashlet filter is configured to show new and unconfirmed incidents of the last 3 days. Notification based on incident rules provides more flexibility than notification based on thresholds, because incident rules can define complex conditions, so they provide the capability to generate notifications only if two or more measure values exceed their thresholds simultaneously.

Incident notifications are useful starting points for performance analysis. See Incidents and Alerting for more information about incidents.

As the following figure shows, the **Incidents** dashlet lists each incident rule that was violated.



Incidents dashlet

To display details in a popup window, double-click the incident or right-click it and select **Details** from the context menu.

Incident state and incident rule state

The icon to the left of an incident or incident rule (
1
) denotes the incident's severity.

Icon	Description
	Informational
	Warning
	Severe

Incident count

If an incident rule was violated multiple times, all incidents that have a state of **new** and a severity of **warning** or **severe** are listed individually below the incident rule. The incident count is appended in parentheses to the incident rule name.

Incidents with a state of **new** are listed in **boldface**. The total of all new incidents is appended to the System Profile name (
2
).

History

The **History** column (
3
) shows a heat field that charts the occurrences of an incident in the configured time frame. Hold the mouse pointer over a bar to display details in a popup window.

Drilldowns

To copy an active incident into a new **Incidents** dashlet, right-click the incident and select **Drill Down > Incidents**.

To chart an active incident in a **Charts** dashlet, double-click the incident to open the **Details** popup window, and click **Show Measures in Chart**.

Table columns

These columns are displayed by default:

- **Name**: The names of the System Profiles, incident rules, and incidents,
- **History**: The heat field representation of incidents in the specified timeframe.
- **Start Time**: The date and time the incident began.
- **Duration**: The duration of the incident.
- **End Time**: The date and time the incident ended.
- **Source**: The incident source (<Agent>@<Host>).
- **Confirmed By**: The user that confirmed the incident.

The following columns are hidden by default.

- **Confirmation Message**: The confirmation message entered by the user that confirmed the incident.
- **Sensitivity**: The configured sensitivity setting of the incident rule.
- **Conditions**: The conditions that trigger the incident rule.
- **Thresholds**: The measure threshold settings of the incident rule.
- **Actions**: The actions triggered by the incident rule upon violation.

To hide or display columns:

1. Right-click the dashlet and select **Dashlet Properties** from the context menu.
2. In the **Dashlet Properties** dialog box, select the **Columns** tab.
3. Clear or select the **Visible** checkboxes for the columns, as needed.

Context menu

Depending on the selected table entry, the context menu provides the following functions.

Menu Item	Description	System Profile	Incident Rule	Incident
Details	Open the **Details** window with information about the selected table entry.	+	+	+
Create Downtime	Schedule downtime to suppress incident reporting for planned downtimes like system backups, database mirroring, etc. See Incident Downtimes for more information.	+	+	
Create Incident Rule	Configure a new incident rule for the selected System Profile. See Incidents and Alerting for more information.	+		
Confirm All Incidents	Set all incidents associated with the selected System Profile or incident rule to the **confirmed** state.	+	+	
Set All Incidents In Progress	Set all incidents associated with the selected System Profile or incident rule to the **in progress** state.	+	+	
Delete All Incidents in Timeframe	Delete **all** incidents associated with the selected System Profile or incident rule.	+	+	
Create Custom Incident…	Create an incident manually, for example for logging purposes.		+	
Show Measures in Chart	Open a **Chart** dashlet to chart the measures configured for the incident rule in the incident's timeframe.			+
Confirm Incident	Set the selected incident to the state **confirmed**. You can select multiple incidents to use this command.			+
Set Incident in Progress	Set the selected incidents to the state **in progress**. You can select multiple incidents to use this command.			+
Delete	Delete the selected item.		+	+
Edit Incident Rule	Alter the properties of the incident rule. An altered incident rule takes effect only after confirmation of the incidents of this incident rule.		+	
Disable/Enable Incident Rule	Disable an active incident rule, or enable a disabled incident rule. A disabled incident rule remains in the incidents table but is dimmed.		+	
Edit Incident	Alter the message, severity, and duration of the selected incident. This command is available only for active incidents.			+
Drill Down	Copy an active incident to a new **Incidents** dashlet.			+

Export Incident Session	Export the session for the timeframe when an incident occurred. See Usage of Sessions for more information. To get a significant session that contains only PurePaths that are directly related to the incident, drill down to the PurePaths before you export the session.			+
Structure Mode	Change the grouping of the incidents. See below for details.	+	+	+

Structure modes

The **Structure Mode** command displays a cascading menu that provides the following options for organizing the **Incidents** table:

- Group by System Profile/Incident Rule
- Group by Severity
- Group by Start Time
- Group by State
- Group by Progress
- Flat

License Overview dashlet

Related pages

Status overview dashlets

Agents Overview dashlet

Incidents Overview dashlet

System Information dashlet

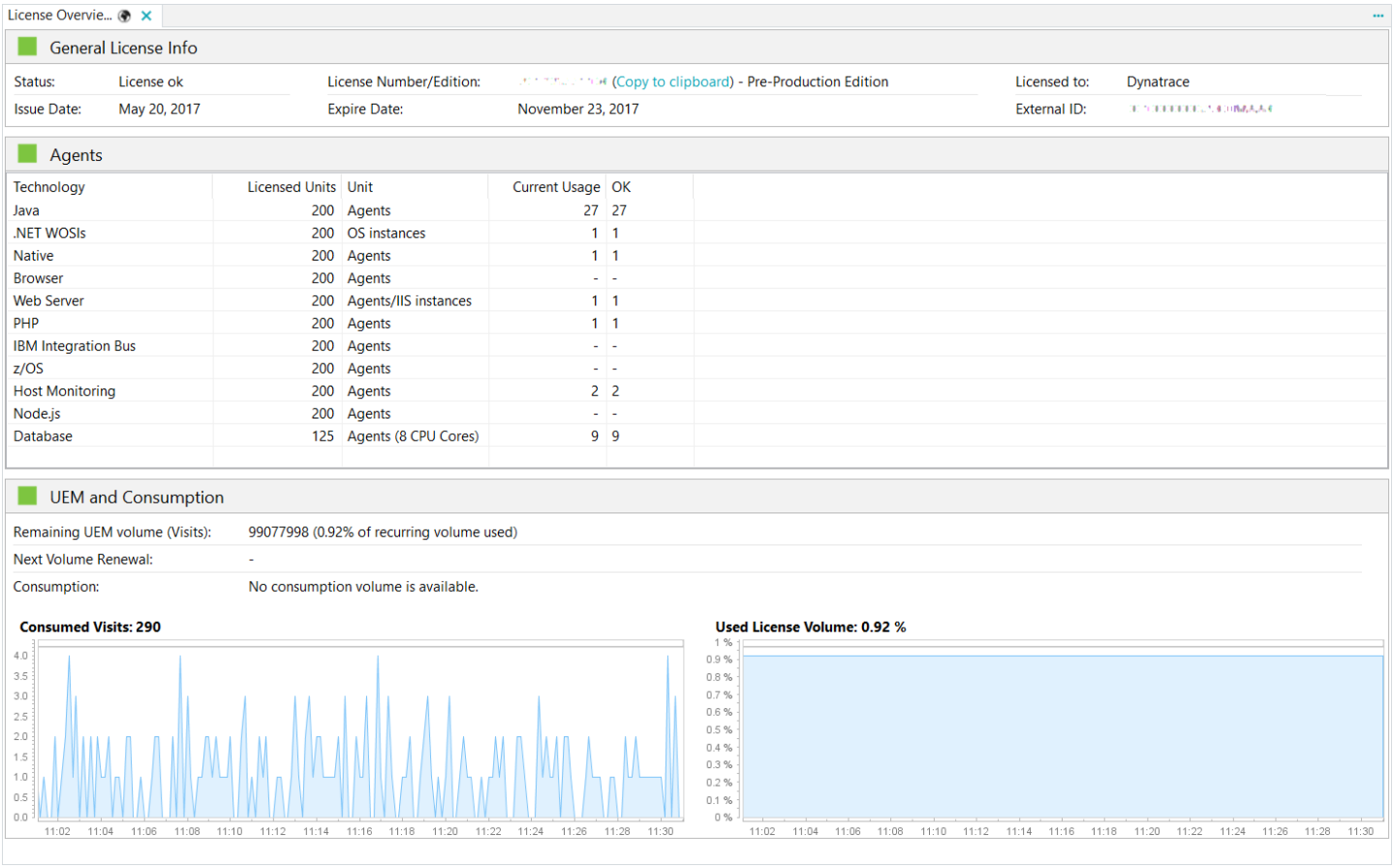
Tasks and Monitors dashlet

Licensing

Manage UEM visits volume

The **License Overview** dashlet provides a quick way to see essential information regarding the license for the currently connected AppMon Server. The dashlet displays general licensing information, including the license status and the license number, which you can copy for inclusion in support emails. All current agents for the license are listed, as well as UEM consumption information.

See Licensing for descriptions of licensing information shown in the License Overview dashlet and status types. See Manage UEM visits volume for more information on UEM volume consumption.



License Overview dashlet

System Information dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Create a support archive

Related pages

Status overview dashlets

Agents Overview dashlet

Incidents Overview dashlet

License Overview dashlet

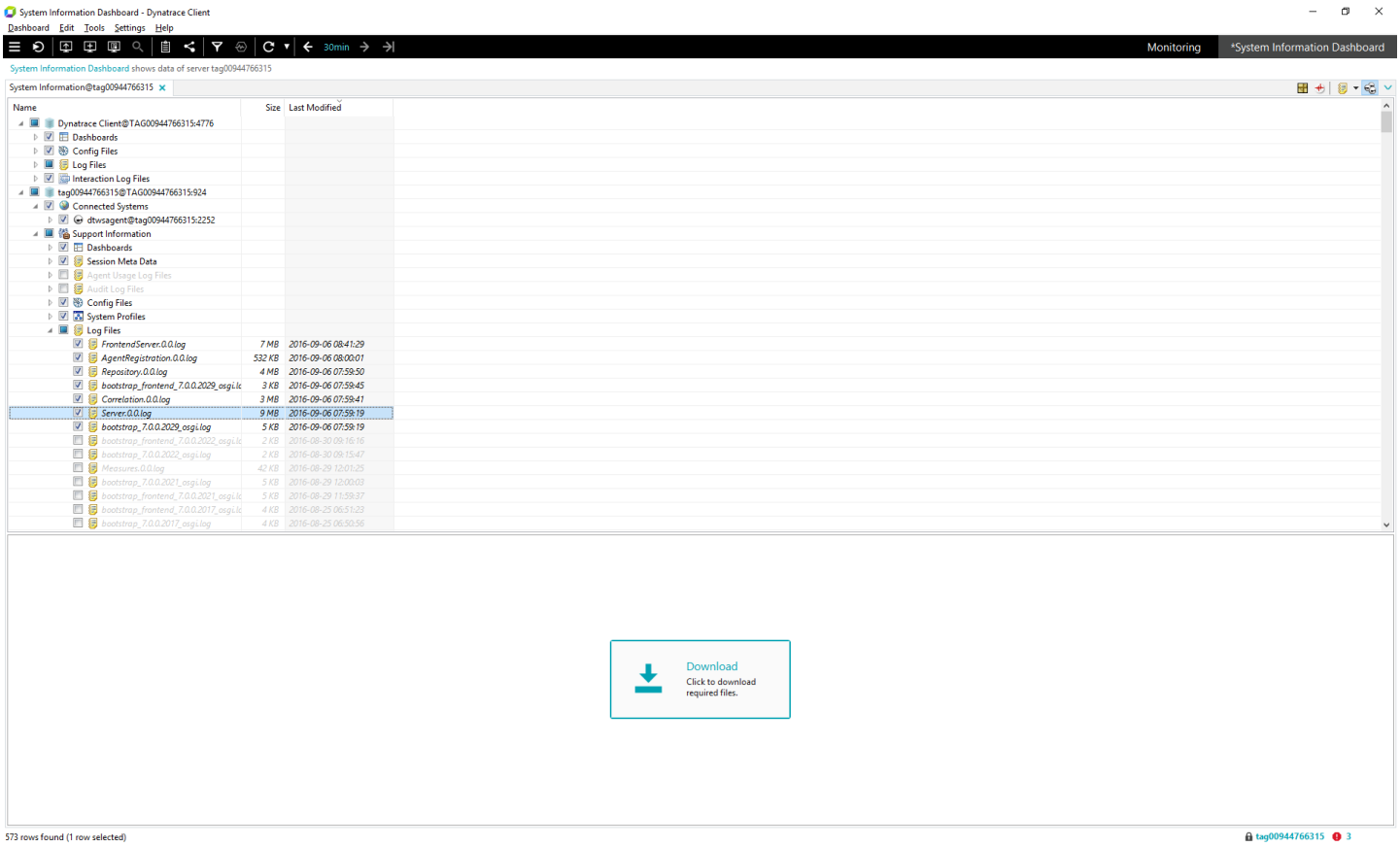
Tasks and Monitors dashlet

Getting support

Overview

The **System Information** dashlet provides information you need to provide when you contact AppMon Support.

The dashlet lists all configuration and log files of your AppMon installation.



Create a support archive

If you discover problems with your AppMon environment, the log files and other information available through this dashlet can help your AppMon administrator or the AppMon Support team to analyze and resolve the problem.

To create a support archive in `.zip` format:

1. Select or clear the checkboxes to specify which items to include in the archive. If an item is unavailable (dimmed), select the parent item and then clear the checkboxes for the child items you don't want to include.
2. Click the package icon
   
   in the toolbar at the top right of the dashlet to open the **Save As** dialog box.
3. Navigate to the folder where you want to save the archive, type a name for the archive and click **Save**.
   By default, the archive is saved as `<name>supportarchive.zip`.

Tasks and Monitors dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Current status icons

Last status icons

Related pages

Status overview dashlets

Agents Overview dashlet

Incidents Overview dashlet

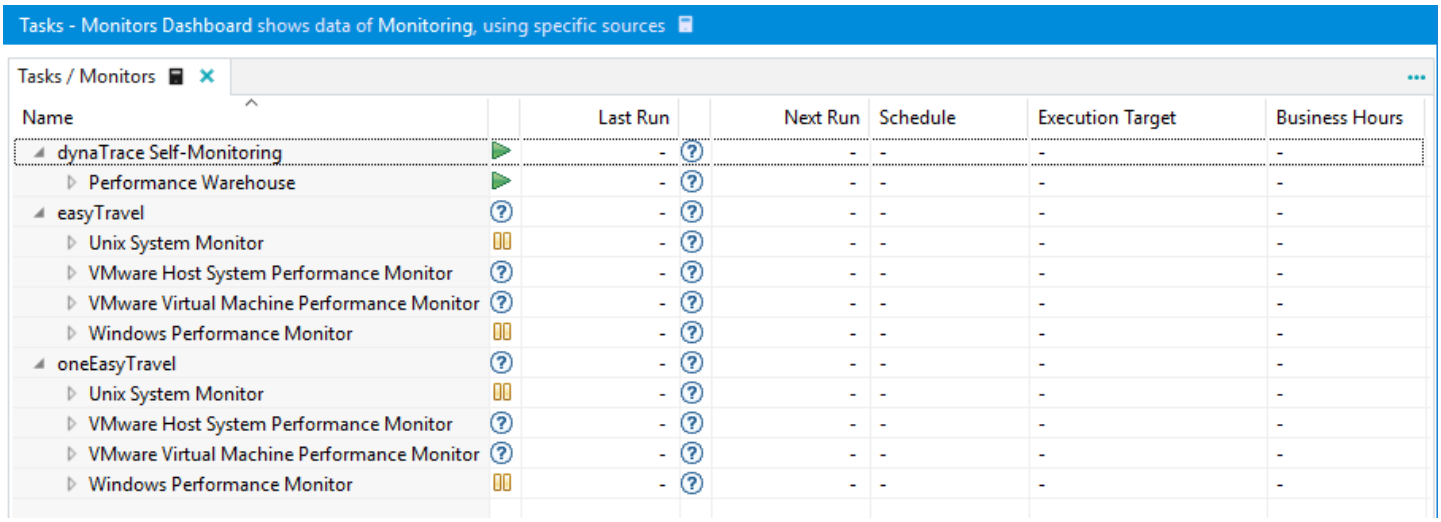
License Overview dashlet

System Information dashlet

Overview

The Tasks/Monitors dashlet displays information about all Tasks and Monitors configured on your AppMon environment. It displays the tasks and Monitors from all System Profiles and AppMon maintenance tasks. The dashlet provides a quick overview of the running states, results, and schedules of your tasks and Monitors. You can manage the task and Monitor configurations through commands on the context menu.

Tasks / Monitors Dashlet



Current status icons

The icons in the column to the right of the **Name** column indicate the current status.

Icon	Description
	Execution is suspended.
	The task or Monitor is scheduled for execution.
	The task or Monitor is currently running.
	The status is unknown.
	The System Profile System Profile is disabled, so no tasks or Monitors are executed.

Last status icons

The icons in the column to the right of the **Last Run** column indicate the status of the most recent run.

Icon	Description
	The status is unknown. For example, the task has not yet executed.

The last run was successful.
The status is Incomplete. For example, a master Monitor has two worker Monitors; if one was successful and the other failed, the state of the master monitor is incomplete because only one of the two monitors was executed successfully.
The last run was not successful.

Monitoring Overview dashboard

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Welcome page - configure your application environment

Monitoring overview dashlets

Questions & answers

Related pages

Application Overview dashboard

End User Experience dashboard

Infrastructure monitoring dashboards

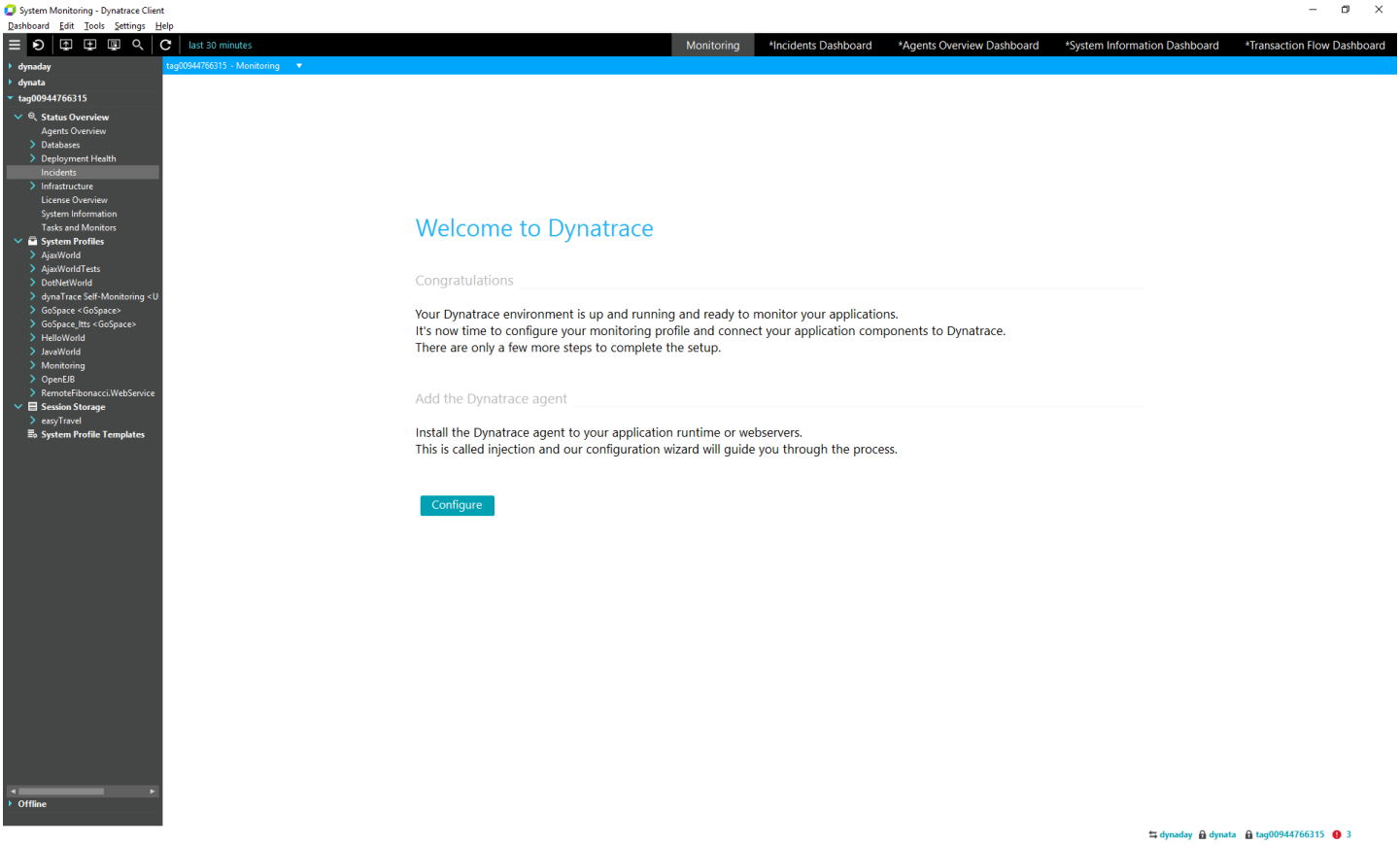
Mobile Applications dashboard

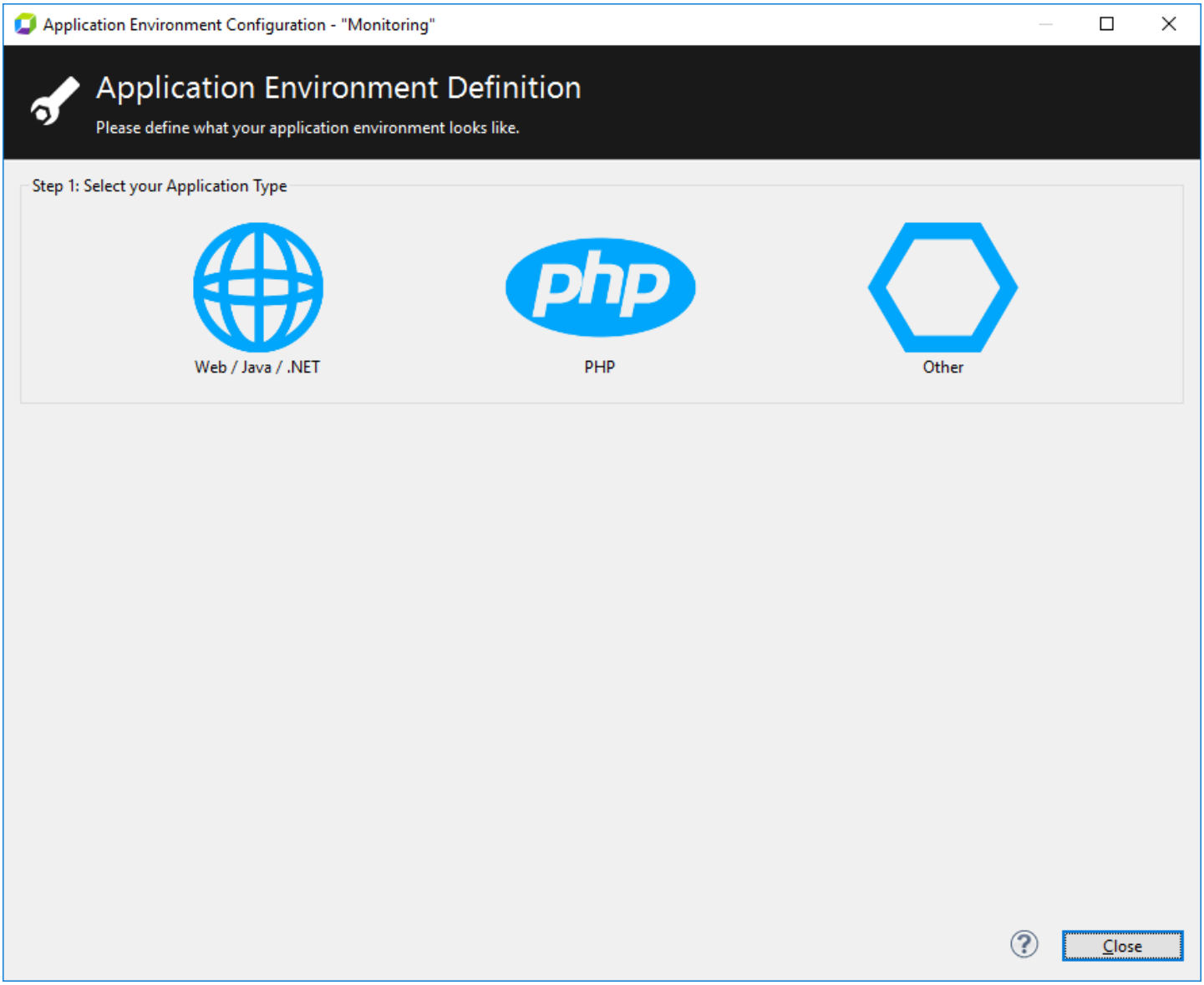
Transaction Flow dashlet

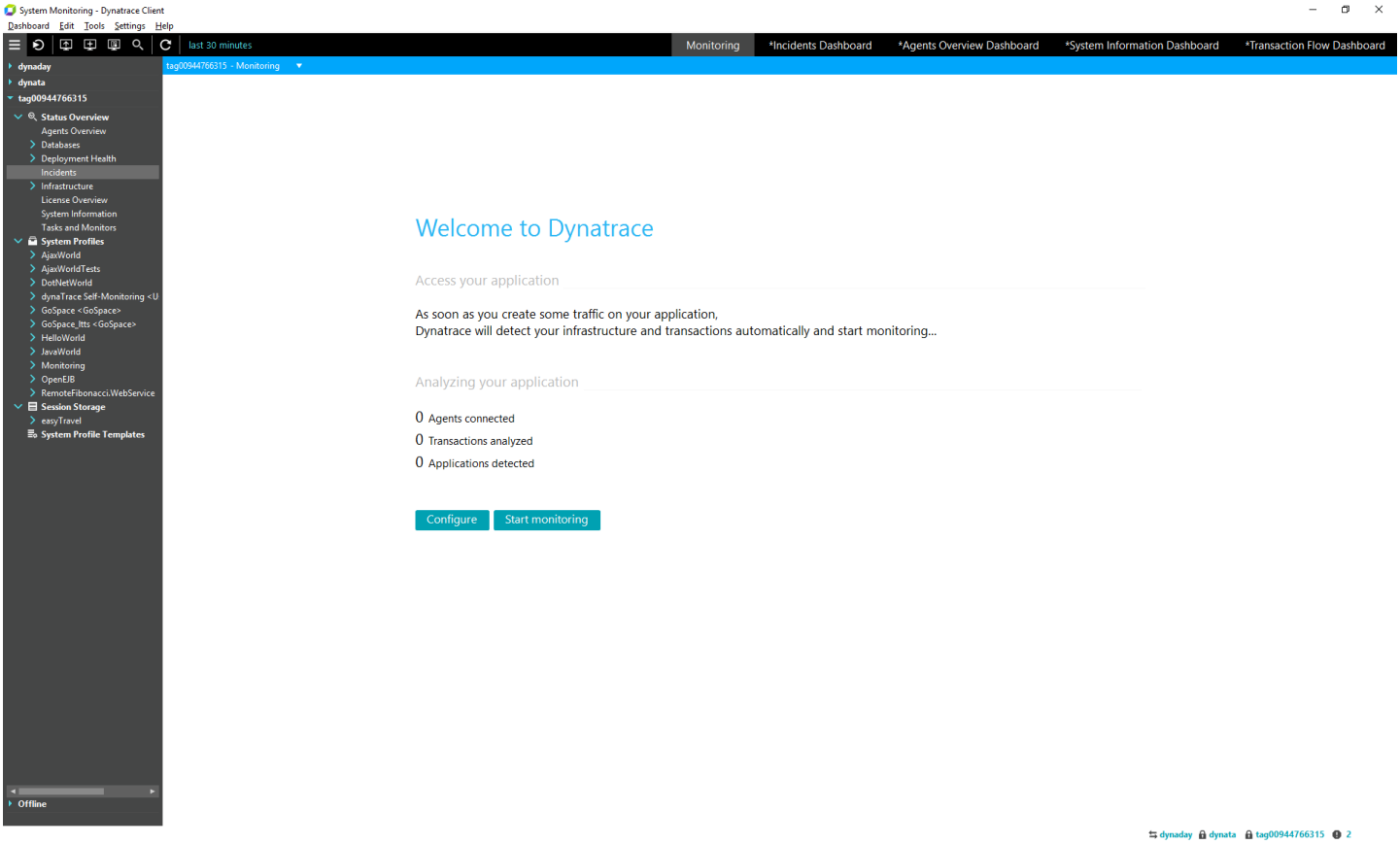
User experience management

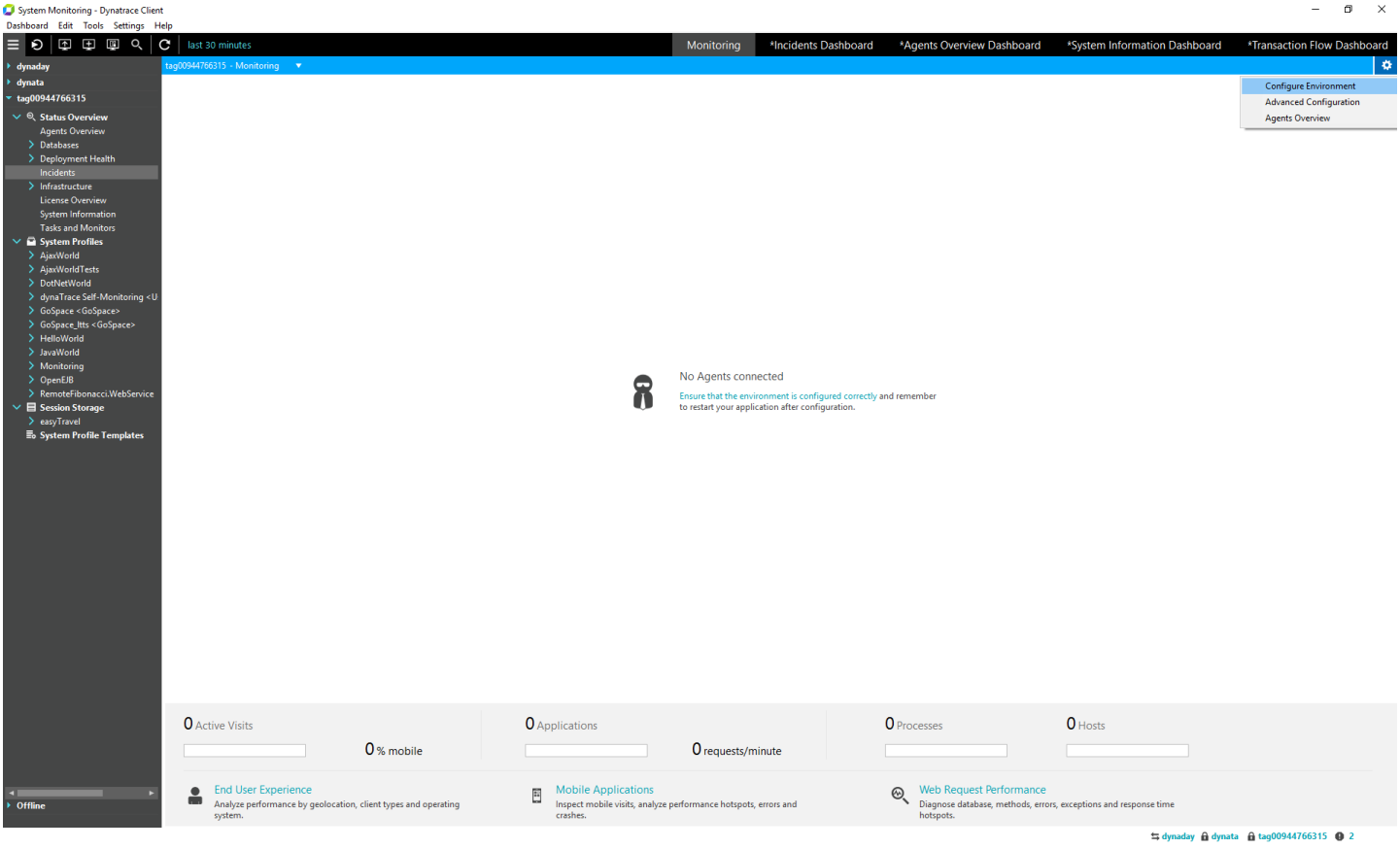
Welcome page - configure your application environment

The Welcome page guides you through System Profile configuration when starting the Client if no tier is configured. Use the step-by-step wizard to configure and connect AppMon Agents to your apps. This lets you collect data and analyze your transactions. See Application Environment Configuration for more information.


System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
last 30 minutes
Monitoring
*Incidents Dashboard
*Agents Overview Dashboard
*System Information Dashboard
*Transaction Flow Dashboard
dynaday
dynata
tag00944766315
Status Overview
Agents Overview
Databases
Deployment Health
Incidents
Infrastructure
License Overview
System Information
Tasks and Monitors
System Profiles
AjaxWorld
AjaxWorldTests
DotNetWorld
dynaTrace Self-Monitoring <U
GoSpace <GoSpace>
GoSpace_ltts <GoSpace>
HelloWorld
JavaWorld
Monitoring
OpenEJB
RemoteFibonacci.WebService
Session Storage
easyTravel
System Profile Templates
Offline
tag00944766315 - Monitoring
Welcome to Dynatrace
Congratulations
Your Dynatrace environment is up and running and ready to monitor your applications.
It's now time to configure your monitoring profile and connect your application components to Dynatrace.
There are only a few more steps to complete the setup.
Add the Dynatrace agent
Install the Dynatrace agent to your application runtime or webservers.
This is called injection and our configuration wizard will guide you through the process.
Configure
dynaday dynata tag00944766315 3


Application Environment Configuration - "Monitoring"
Application Environment Definition
Please define what your application environment looks like.
Step 1: Select your Application Type
Web / Java / .NET
PHP
Other
Close


System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
last 30 minutes
Monitoring
*Incidents Dashboard
*Agents Overview Dashboard
*System Information Dashboard
*Transaction Flow Dashboard
dynaday
dynata
tag00944766315
Status Overview
Agents Overview
Databases
Deployment Health
Incidents
Infrastructure
License Overview
System Information
Tasks and Monitors
System Profiles
AjaxWorld
AjaxWorldTests
DotNetWorld
dynaTrace Self-Monitoring <U
GoSpace <GoSpace>
GoSpace_ltts <GoSpace>
HelloWorld
JavaWorld
Monitoring
OpenEJB
RemoteFibonacci.WebService
Session Storage
easyTravel
System Profile Templates
Offline
tag00944766315 - Monitoring
Welcome to Dynatrace
Access your application
As soon as you create some traffic on your application,
Dynatrace will detect your infrastructure and transactions automatically and start monitoring...
Analyzing your application
0 Agents connected
0 Transactions analyzed
0 Applications detected
Configure
Start monitoring
dynaday dynata tag00944766315 2


System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
last 30 minutes
Monitoring
*Incidents Dashboard
*Agents Overview Dashboard
*System Information Dashboard
*Transaction Flow Dashboard
tag00944766315 - Monitoring
Configure Environment
Advanced Configuration
Agents Overview
No Agents connected
Ensure that the environment is configured correctly and remember to restart your application after configuration.
0 Active Visits
0 % mobile
0 Applications
0 requests/minute
0 Processes
0 Hosts
End User Experience
Analyze performance by geolocation, client types and operating system.
Mobile Applications
Inspect mobile visits, analyze performance hotspots, errors and crashes.
Web Request Performance
Diagnose database, methods, errors, exceptions and response time hotspots.
dynaday dynata tag00944766315 2

Once you have configured at least one AppMon Agent, the Monitoring Overview Dashboard replaces the Welcome page, which is the starting point to analyze the application performance in a specific System Profile.

To access the Monitoring Dashboard, click the

Monitoring

button on the right top.

Don't confuse the

Monitoring

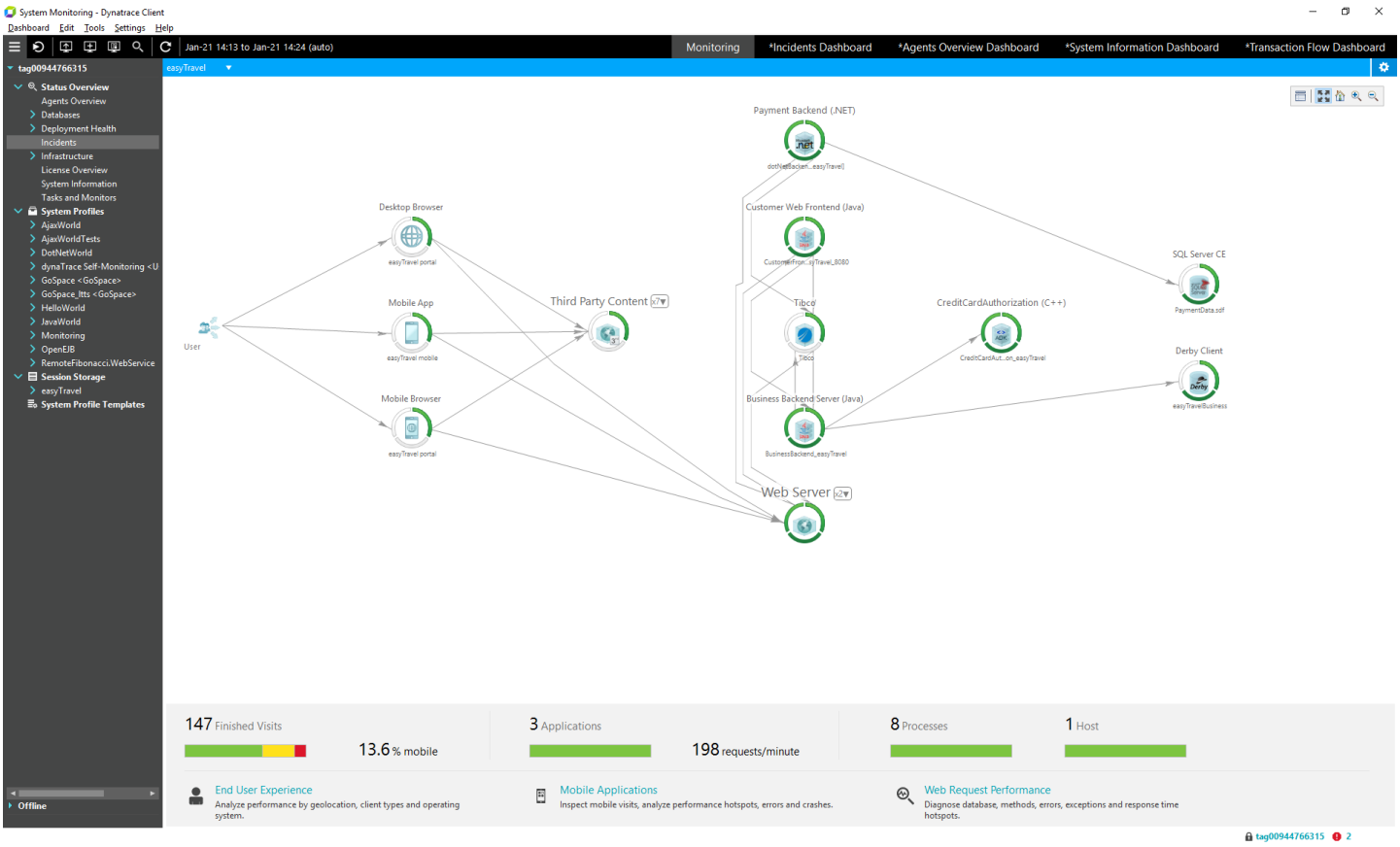
button with the root breadcrumb button / drop-down combo box

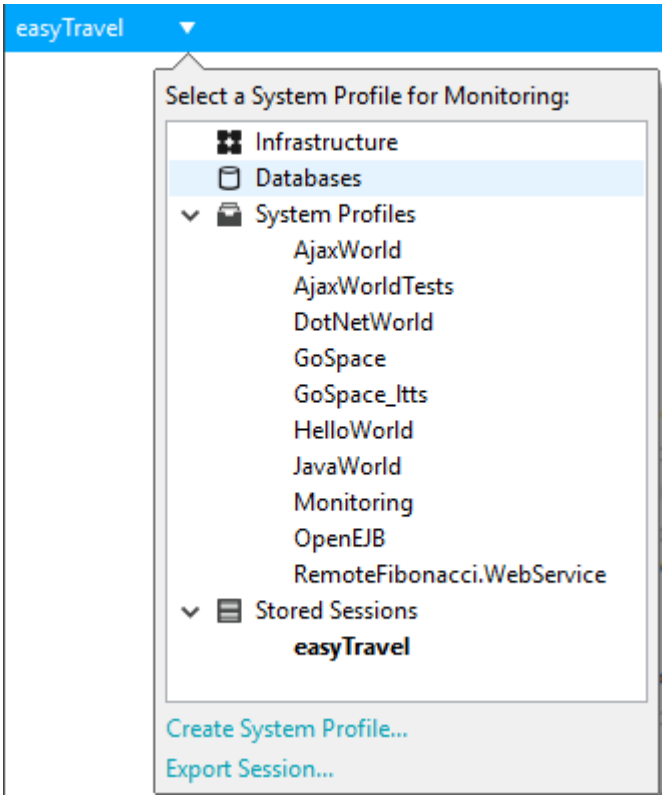
Monitoring

(left top), showing the name of the default System Profile – which happens to be named "Monitoring". By clicking the down-arrow you can select a different System Profile (if you have one; see below) and its name

easyTravel

will appear on the root breadcrumb.





Monitoring overview dashlets

The Monitoring Overview Dashboard aggregates the following types of monitored data that display in the Monitoring Overview dashlets into one view. This view lets you quickly see your system status and drill down for more specific information.

Transaction Flow

The Monitoring Overview embeds a Transaction Flow Dashlet to show an overview of your environment featuring OMNI-channel (the main entry points grouped by channel (client types) and application), Agent Groups, databases, and external services.

> **Note**
>
> The embedded Transaction Flow is optimized for quick data access and for application monitoring. For advanced analysis and to access all drill down options, use the default Transaction Flow Dashlet.

Key performance indicators

The KPI sections below the Transaction Flow link to separate views that provide more detail about each monitored aspect.

- **(Active or Finished) Visits**: The Visit section shows the currently active visits for live sessions or finished visits for stored sessions and their User Experience Index (Satisfied, Tolerating or Frustrated). See How does UEM work? for a detailed explanation of these categories.
- **Applications**: The Application section shows the selected System Profile's applications and their health states (OK, Critical or Not Connected) and the total requests per minute of all applications.
- **Processes and hosts**: The Processes and Hosts section shows the selected environment's health state (OK, Critical or Not Connected).

Dashboard links

- **End user experience**: Provides detailed information about the performance aggregated by geo-location, client type and operating system.
- **Mobile applications**: Provides detailed information about mobile visits, performance hotspots, errors and crashes.
- **Web request performance**: Provides detailed information about databases, methods, errors, exceptions and response time hotspots.

Configuration

Click the gear icon / Configure menu at the top right to change the selected System Profile.

Questions & answers

What is an active visit?

An active visit is a visit in progress, which means that the representative user is actively using your application. If the user has left your application or is inactive for a certain time, the visit is finished.

See also a description of a visit in User experience management.

Application Overview dashboard

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is an application in the AppMon context?

Overview

Displaying the application overview

Related pages

Application Details

Baseline chart

Business Transaction details

Identify problems in your application

User experience management

Baseline and smart alerting explained

Infrastructure monitoring

What is an application in the AppMon context?

See System Profile - Applications for more information.

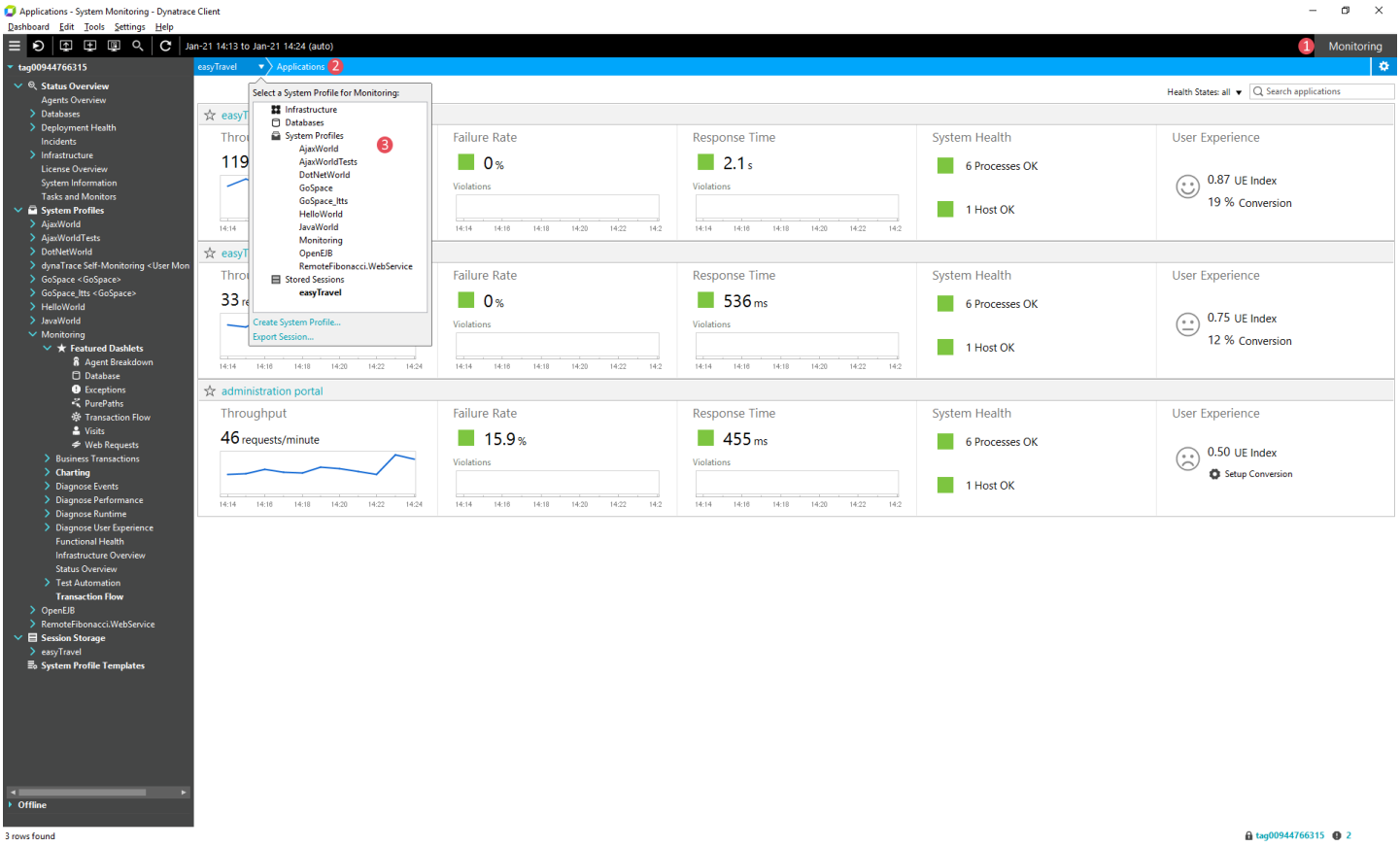
Overview

The **Application Overview** lists your system's applications, sorted by relevance. For each application, it displays the current status and the history within the selected time frame.

The following walkthrough explains how to use the **Application Overview** to analyze performance problems. Also see Identify Problems in your Application.

Displaying the application overview

To display the **Application Overview**, click the **Monitoring** button, as shown in the screenshot below. If you have multiple System Profiles, use the drop-down list for the first breadcrumb to switch the active System Profile. You can also select stored sessions and offline sessions.



Selecting a System Profile for the Application Overview

Violation indicators

A Violation Indicator shows where there are current baseline violations for the measurement for at least one Business Transaction of the application.

There are violating Business Transactions.

Everything is OK.

Violation histories

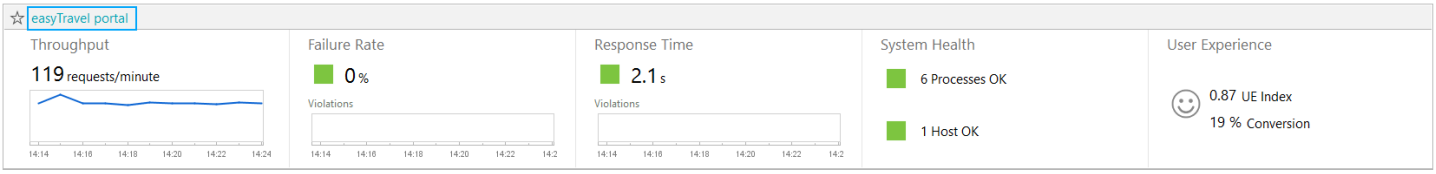
The violation history shows all violations and their duration in the currently selected timeframe. Violations are marked red in the timeline.



Violation history

Application name and details

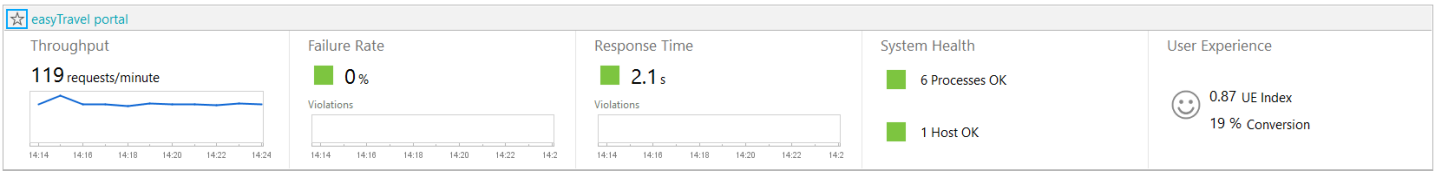
Click the application name or the Application Details button, as shown in the figure, to open the Application Details dashlet.



Opening the Application Details dashlet

Application favorite

You can mark applications as favorite by toggling the favorite star to the left of the application name. Your favorite applications are listed above the other applications. This feature is especially useful to keep an eye on important applications.



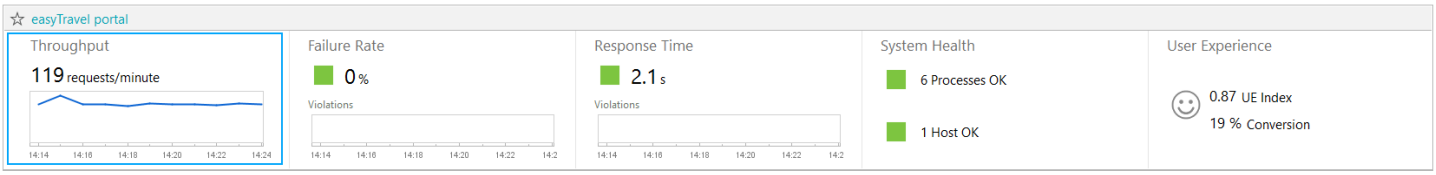
Favorite icon

This feature is user-specific. Each user AppMon can define different favorite applications. Other users are not affected by your favorites.

See Application Ranking for more information.

Throughput

The chart shows the *throughput count* (requests per minute) and the expected range, based on the data for the last week.



Throughput chart

The throughput value is the sum of all Business Transactions of the application. By default, these are the out-of-the-box **Web Page Requests** Business Transactions. In this case, the throughput is the number of web requests that are processed by your application, excluding requests to static content, synthetic requests, and requests from search robots.

Throughput values higher or lower than expected do not automatically show that there is something wrong with your system. For this reason, no violation indicators are available for the throughput.

If throughput is zero, your application may have an availability problem. See Availability Monitoring for more information.

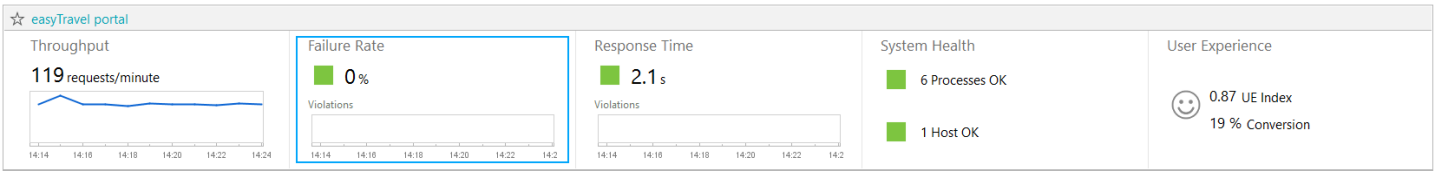
Read more about how throughput is calculated in Baseline and Smart Alerting Explained.

Important

The **Application Overview** dashboard can be configured to show any set of Business Transactions. A single user action may fit more than one of the configured Business Transactions. For example, if a user searches for *car*, this action might contribute to a Business Transaction **Search** as well as to **Guest Visits**. In this example, the single user action would count as two requests for the throughput calculation.

Failure rate

The *failure rate* is the average percentage of failed requests. A violation is indicated if at least one Business Transaction currently produces more failures than expected, based on the data for the last week.

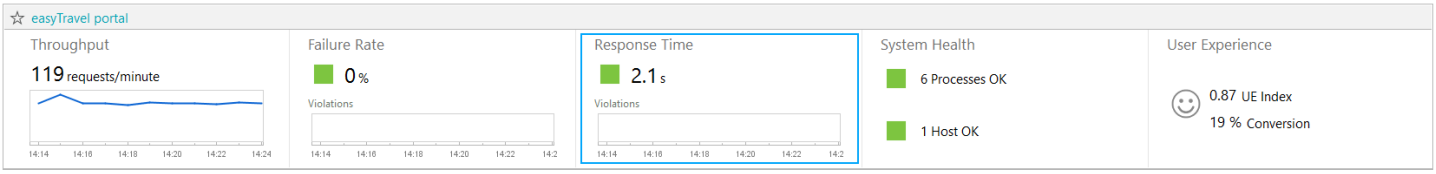


Failure rate chart

Read more on how the failure rate is calculated in Baseline and Smart Alerting Explained.

Response time

The *response time* charted in the **Application Overview** is the median response time of current requests.

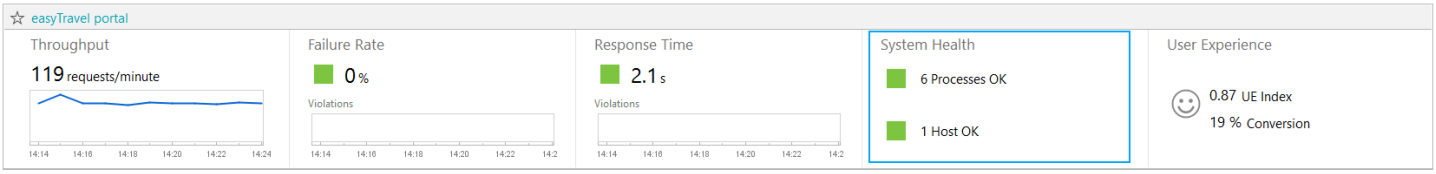


Response time chart

A violation is indicated if the median response time or the response time of the slowest 10% of requests is higher than expected for at least one Business Transaction.

System health

The *System Health* section shows the status of running processes for the application. Processes are affected if they are unhealthy or are running on unhealthy hosts.



System health indicators

The **Processes** and **Host** health indicator icons can display the following states:

- Green – OK
- Red – Unhealthy
- Affected percentage
- Grey – N/A

Affected processes are either unhealthy on their own or are running on an unhealthy host. For more information, see Host Health Monitoring. Hover over an icon to display information about affected processes and hosts. Click the link in the icon's hover popup for detailed information on the affected processes and hosts.

The following table describes the four states of Process Health. In the table, **<PROCESS_NAME>** represents the name of the target process. **X**, **Y**, and **<#>** represent the number of processes.

Icon	State	Description	Available Links (hover the mouse over the icon to see links)
	X Processes OK	No problems detected. All processes show normal behavior. Process <PROCESS_NAME> shows normal behavior.	Application Processes Web Server
	X Processes unhealthy	Health problems detected. Process <PROCESS_NAME> shows health problems. <#> Processes with health problems detected.	Application Processes • Unhealthy Application Processes • Web Servers • Unhealthy Web Servers

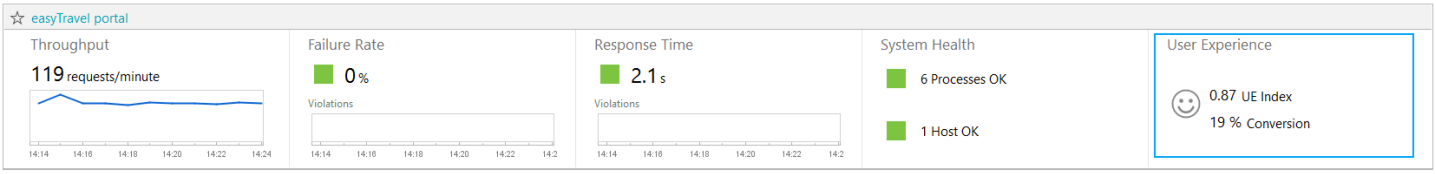
	X of Y affected	Health problems detected. • Process <PROCESS_NAME> shows health problems. • <#> Processes with health problems detected.	Application Processes • Unhealthy Application Processes • Web Servers • Unhealthy Web Servers
	N/A	Currently no data available.	None.

The following table describes the four states of Host Health. In the table, **<HOST_NAME>** represents the name of the target host. **X**, **Y**, and **<#>** represent the number of hosts.

Icon	State	Description	Available Links (hover the mouse over the icon to see links)
	X Host OK	No problems detected. • Host <HOST_NAME> shows normal behavior. • All hosts show normal behavior.	**Hosts** links to the **Infrastructure Monitoring Dashboards**.
	X Hosts unhealthy	Health problems detected. • Host <HOST_NAME> shows health problems. • <#> Hosts with health problems detected.	**Hosts** links to the **Infrastructure Monitoring Dashboards**.
	X of Y affected	Health problems detected. • Host <HOST_NAME> shows health problems. • <#> Hosts with health problems detected.	**Hosts** links to the **Infrastructure Monitoring Dashboards**.
	N/A	Currently no data available.	None.

User experience

The User Experience rating comprises the UE Index (the Average Apdex Rating) and the Average Conversion Rate. It is based on user visits during the last 6 hours.



User experience rating

The *Average Apdex Rating* is an index for application performance. The **User Experience** is rated on the following scale for the Average Apdex Rating:

- Unacceptable (0 to <0.5)
- Poor (0.5 to <0.7)
- Fair (0.7 to <0.85)
- Good (0.85 to <0.94)
- Excellent (0.94 to 1)

See User Experience Management for more information.

The *Average Conversion Rate* is the percentage of all visitors who performed any of the actions defined in the **System Profile > User Experience > Conversion** goal. In the **Application Overview Home**, you can go directly to the conversion goal configuration by clicking the cogwheel icon right to go to **Conversion** and selecting **Edit conversion goals**. You can also define a conversion goal by clicking on the cogwheel to go to **Conversion** and selecting **Add new conversion goal**.

The easyTravel Portal example shows the percentage of visitors that became paying customers in the displayed timeframe.

Application ranking

Entries in the **Application Overview Home** are ranked so they are listed in order of the most relevant applications. Favorite applications are on top. Applications with violations or health problems are ranked after the favorite applications.

An application becomes more relevant if it signals abnormalities (failure rate violation, response time violation, or unhealthy processes) or provides a higher throughput than other applications.

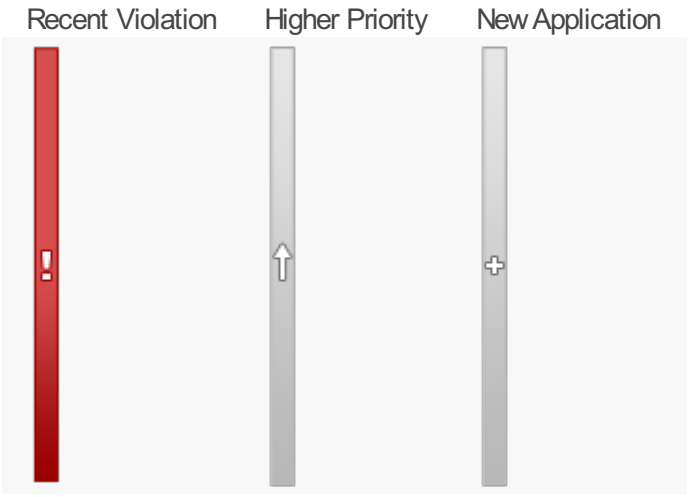
The table lists the ranking criteria by priority. If a ranking criterion provides no measurable difference, the next criterion in the priority list is considered.

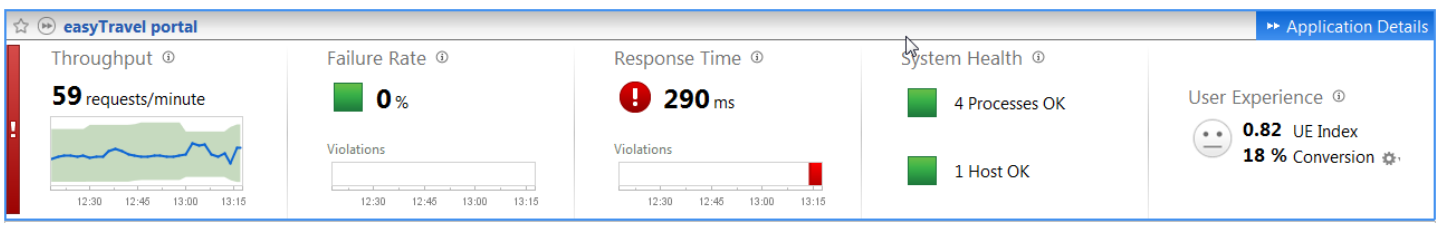
Priority	Criterion	Applications are ranked…
1	Application favorite	…higher when set as a favorite application.

2	Failure rate violation	…higher with an active failure rate baseline violation.
3	Response time violation	…higher with an active response time baseline violation.
4	Process health	…by number of unhealthy contributing processes.
5	Throughput	…by average throughput value or average throughput baseline value (if the baseline is higher than the current throughput).
6	Failure rate	…by current failure rate value.
7	Response time	…by current response time value.
8	Application name	… sorted by name, ascending.

Markers

If an Application has recent violations, got a higher priority, or is newly created, it is marked as shown:





Application with Recent Violation

Configuration

Access the **Application Overview Home** configuration through the configuration icon located on the top right. Clicking the icon displays a list of configuration options for your applications, as described in the table below.

Configuration Option	Description
Manage Applications	Manage existing and create new Application items in the **Applications** pane of the **System Profile Preferences** dialog box.
Email Notifications	Configure recipients for notification of baseline violations.

Application Details

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

General

Walkthrough

Related pages

Baseline and smart alerting explained

Application Overview dashboard

Identify problems in your application

Business Transaction details

Baseline chart

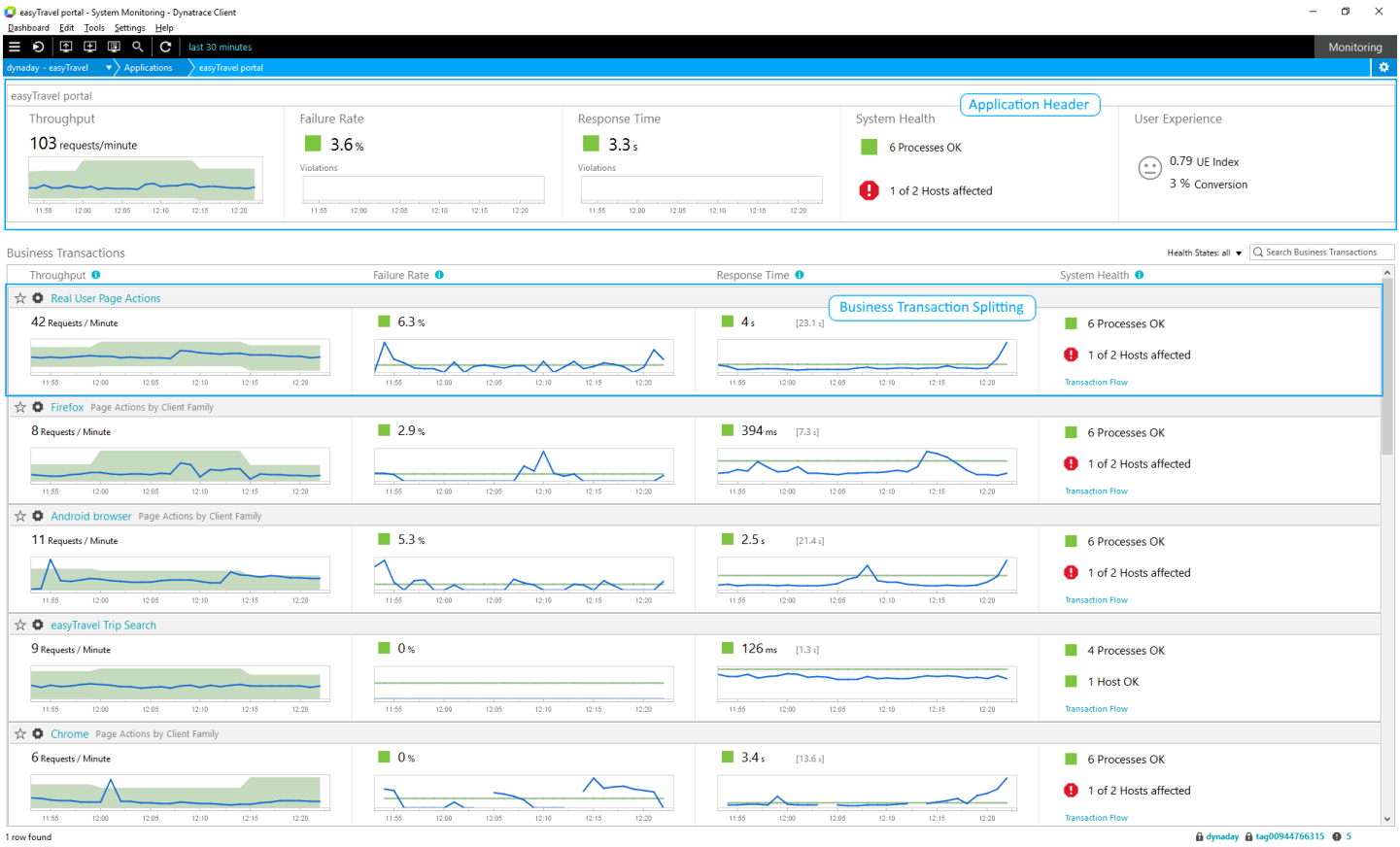
Infrastructure monitoring

System Profile - Business Transactions

General

Click an application name in the **Application Overview Home** to display the **Application Details** for the application. It shows the aggregated status of the selected application.

Listed below the Application Header are the detailed Business Transaction Splittings that constitute the application. The Business Transaction Splittings are sorted by priority. Favorite Business Transactions are always on top of the list. Next come Business Transactions with violations. See Ranking for more information.



Application Details

Use the following walkthrough to learn how to use the **Application Overview** to analyze performance problems. For more information, see Identify Problems in your Application.

Walkthrough

Application header

The application header shows the status of your application. See Application Overview Home - Application for more information.

Configuration: reset base line for this application

Use this configuration function if the data of the last week does not make a useful baseline. This may be the case, for example, if severe problems occurred in the last week, or after deployment of new versions of applications that are known to have different response times.

To reset the baseline, click the configuration cogwheel icon and select **Reset Base lines for this Application**, then click **OK** in the confirmation message. The baselines for all Business Transactions of the application will be reset and a new baseline will be built from new data.

> **Warning**
>
> Resetting the baseline cannot be undone. Building a new baseline may take several hours, depending on the throughput.

Business Transaction splittings

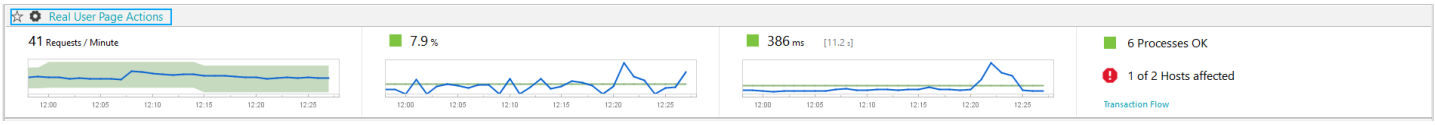
By default, the Business Transaction **Web Page Requests** is monitored. This Business Transaction is split by URI parts and application.

To give you the best picture of how your applications and processes are doing and to alert you to problems as quickly as possible, Application Details are sorted by failure rate, response time, and by either process time or throughput, depending on which is more significant.

If you want to configure which Business Transactions to monitor, see Select Monitored Business Transactions.

Business Transaction splitting and Business Transaction

The figure below shows the URI splitting **orange.jsf** and the Business Transaction **(Web Page Requests)**.



Business Transaction splitting

For Business Transaction Splittings, there may also be a placeholder, **<...>**. This placeholder is used for all paths of the Business Transaction that do not constitute a splitting by themselves. Usually, these URIs do not meet the default condition of having a frequency of at least 5%.

Click the Business Transaction Splitting name to open the Business Transaction Details dashlet.

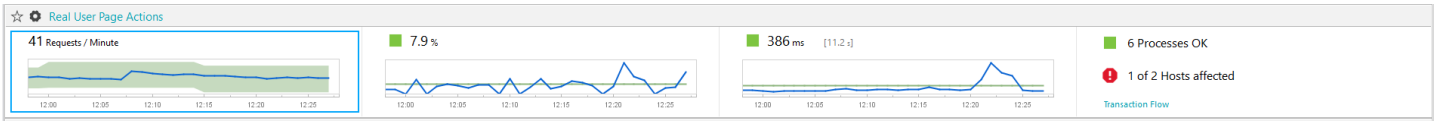
When the Business Transaction does not receive any data, the whole section is disabled and no further drilldowns can be performed.

You can use the star button on the left side of the header to define favorite Business Transaction Splittings and Business Transactions. Favorite Business Transaction Splittings are listed on top of the list. See Ranking.

Favorites are stored by user. Different users within the AppMon deployment may have different sets of favorites.

Throughput

The **Requests/Minute** chart shows the throughput count (blue) and the expected range (green), based on the data of the last week. This is the actual throughput for this Business Transaction in the timeframe.



Throughput

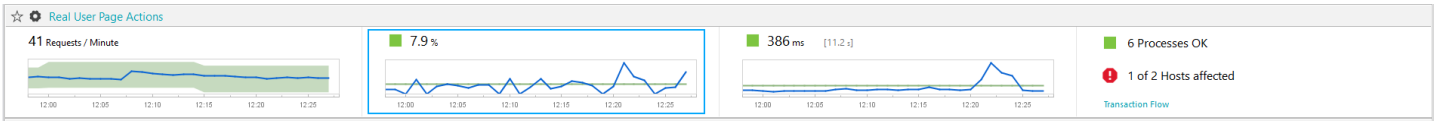
Throughput influences the failure rate and response time measurements: if the throughput is higher, the statistical spread for the failure rate and response time is smaller. See Application Overview Home - Throughput for more information.

Click the Throughput section to open the Business Transaction Details dashlet, focused on the throughput for its main chart.

Failure rate

The **Failure Rate** chart shows the failure rate trend (blue), baseline (green) and violations (red area). A violation is indicated if this Business Transaction currently produces more failures than expected, based on the data of the last week.

The number above the chart is the actual failure rate for this Business Transaction in the last 2 minutes. If there has been no activity in the last 2 minutes, a dash (**–%**) is shown.



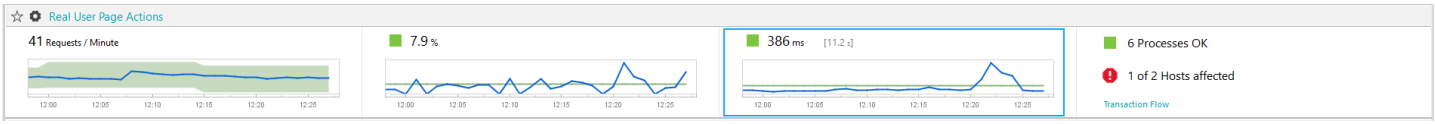
Failure rate

Click the **Failure Rate** section to open the Business Transaction Details dashlet, focused on the Failure Rate for its main chart.

See Baseline and Smart Alerting Explained - Failure rate for more information on how Failure Rate is calculated.

Response times

The **Response Times** chart shows the median response time and response time of the slowest 10%.



Response times

The median response time is the **typical** response time (50% of requests have a slower response time and 50% have an equal or faster response time).

The response time of the slowest 10% represents the outliers (10% of requests have a slower response time and 90% have an equal or faster response time).

The numbers above the chart are the actual response times for this Business Transaction in the last 2 minutes. If there has been no activity in the last 2 minutes, a dash (**– ms**) is shown.

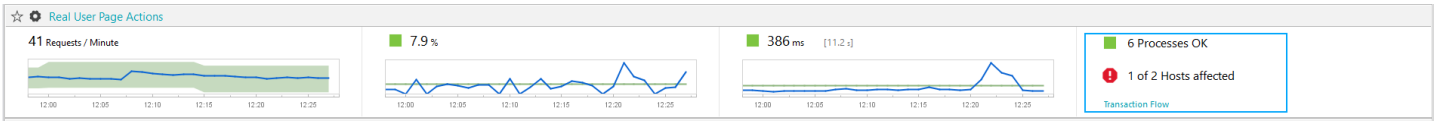
A violation is indicated if either the **median** or the **slowest 10%** response time for this Business Transaction is higher than expected.

Click the **Response Time** section to open the Business Transaction Detailsdashlet, focused on the Response Time for its main chart.

See Baseline and Smart Alerting Explained - Response Time for more information on how response time is calculated.

System health

The System Health section lists the health status of the hosts and processes that are used by this Business Transaction.



System health

Processes are affected if they are unhealthy or running on unhealthy hosts.

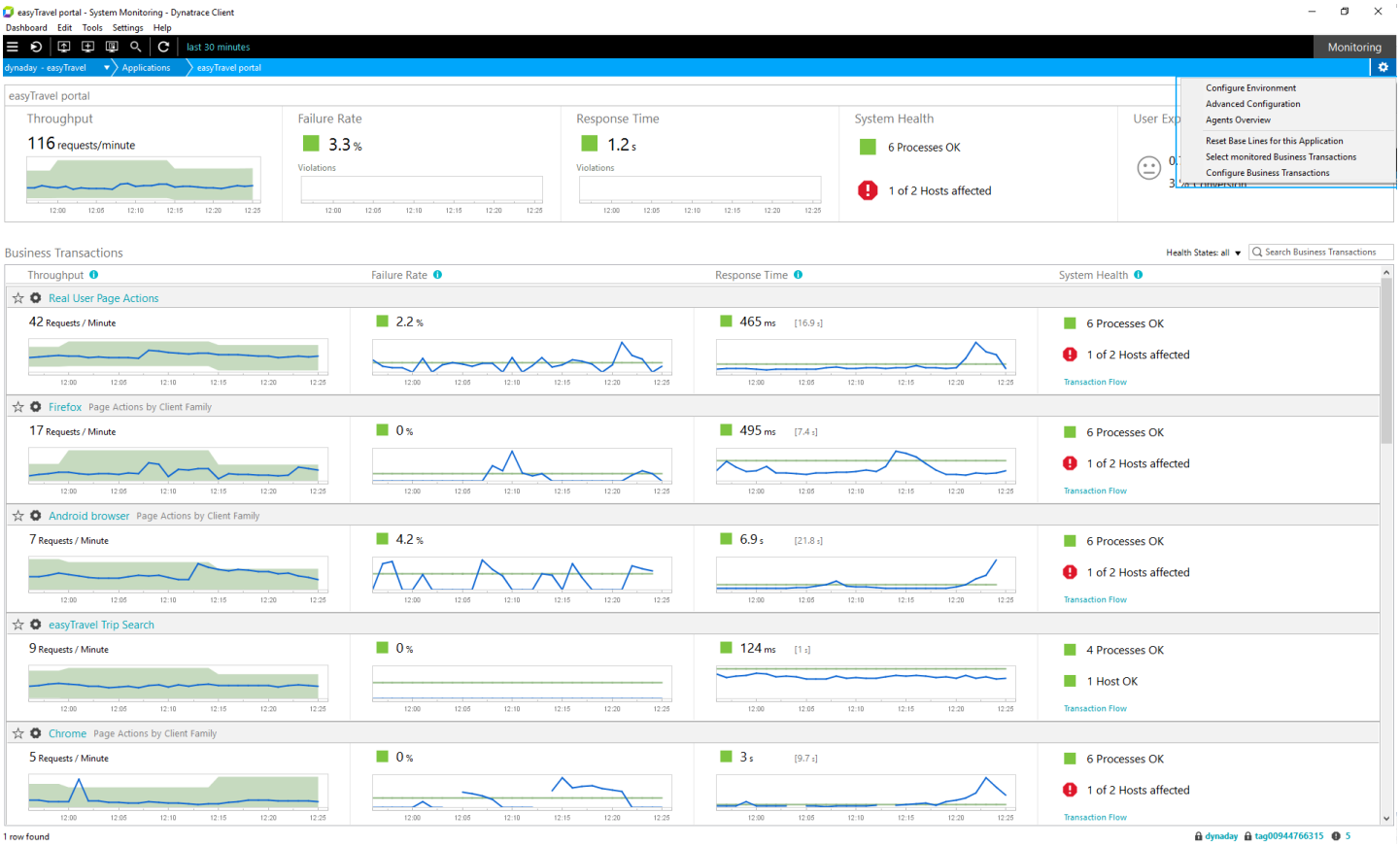
You can also access the Transaction Flow for a single Business Transaction.

See Application Overview Home - System Health and Identify Problems in your Application - Analyze Transaction Flow and Host Health for more information.

Configuration: select monitored Business Transactions

> **Note**
>
> This functionality is only available for users with administrative rights.

To include Business Transactions in the Application Monitoring or exclude them, click the cog wheel icon in the Application Overview Home Header or next to the header of the Business Transactions list and select **Select monitored Business Transactions**.

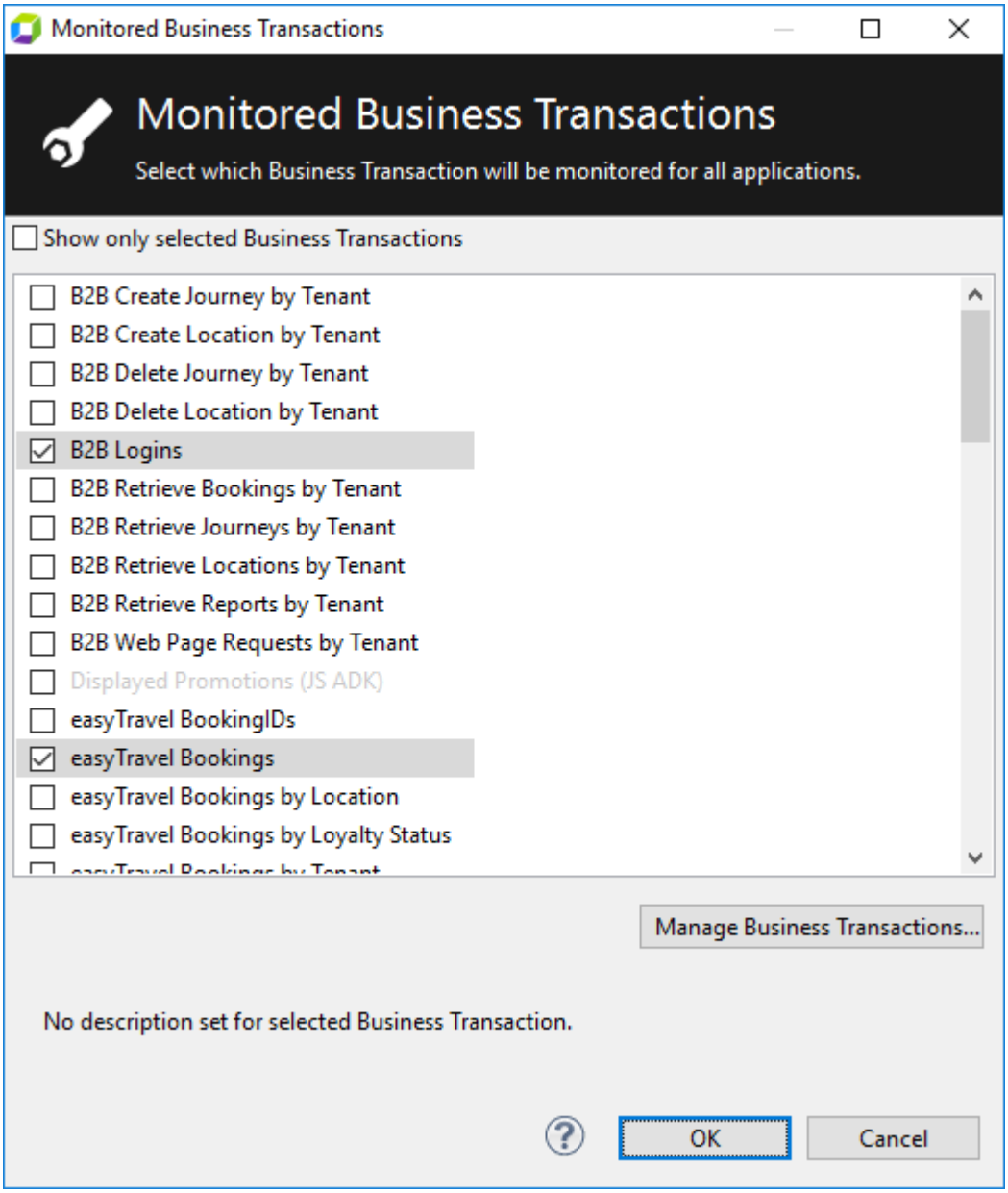


Configuration Icon in the Business Transaction Details

The dialog box lists all available Business Transactions of the target System Profile. Business Transactions can be monitored only if they are split by Application and not more than one additional Splitting and are **not** split by Agent. For example, in the figure above, the Business Transaction is split into **easyTravel portal** and **Web Page Requests**. Visit-based Business Transactions are not available for monitoring.

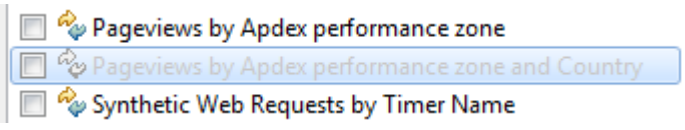
Select the check box to select a Business Transaction for monitoring; clear the check box to exclude it from monitoring.

All applications within the target System Profile are affected by the changes: selected Business Transactions are monitored for all applications of the target System Profile.



Selecting Business Transactions to monitor

A Business Transaction that is dimmed as shown in the following figure is not available for Application Overview Monitoring.

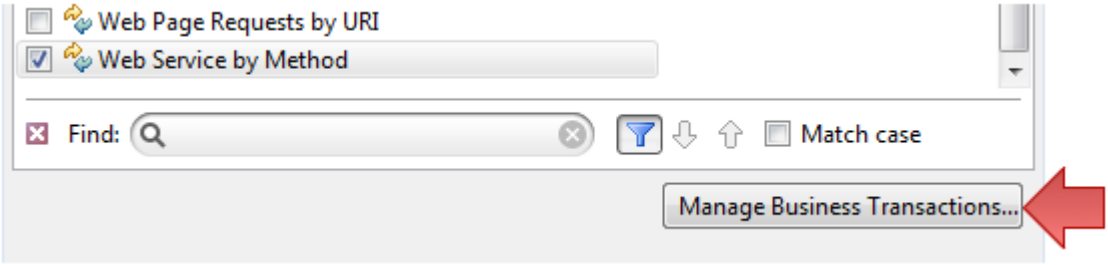


Unavailable Business Transaction

There are three possible reasons why it is unavailable:

- It is not split by Application at all.
- It is split by Application and **more than** one other measure.
- It is split by Agent

If it is not split by Application, you have to add the Application Splitting to the Business Transaction to make it available. If it is split by Application and more than one other measure, you have to remove the additional Splittings of the Business Transaction. If it is split by Agent you have to remove the Agent splitting. To make either change, open the **Business Transaction Editor** in the System Profile by clicking the **Manage Business Transactions** button in the **Monitored Business Transactions** dialog box.



Manage Business Transactions button

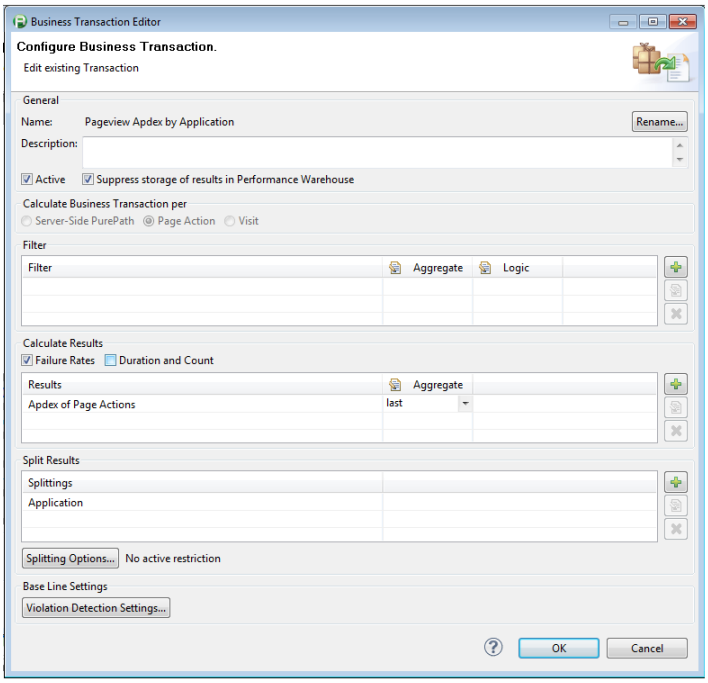
See System Profile - Business Transactions for more information.

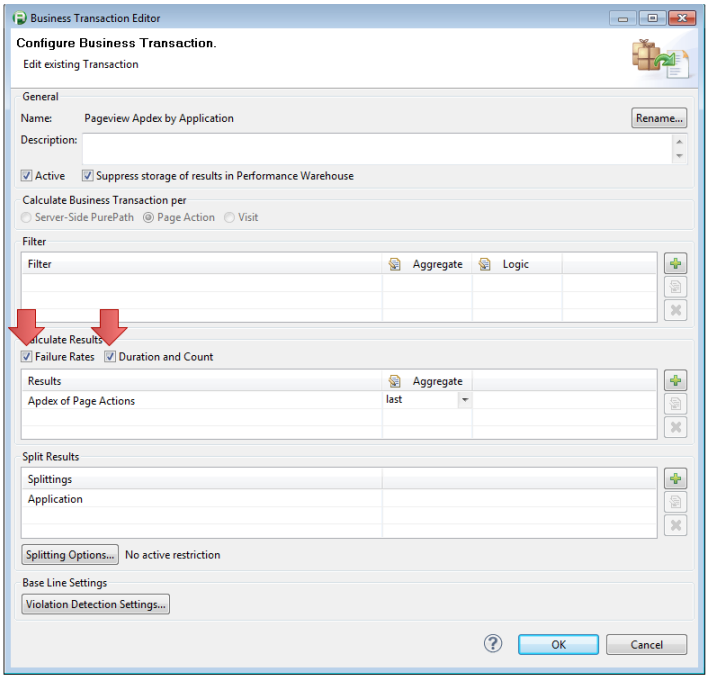
Note

If you do not want changes to the Business Transaction to affect other applications, copy the Business Transaction and make your changes to the copy.

When you select a Business Transaction for the Application Overview, the check boxes for Failure Rates and Duration and Count are selected automatically. These selections are needed for Application Monitoring.

If either the Failure Rates and Duration and Count check box is unselected, the Business Transaction is removed from the Application Overview. To include it again, select the Business Transaction again in the Monitored Business Transactions dialog box.





Selecting required settings

Ranking

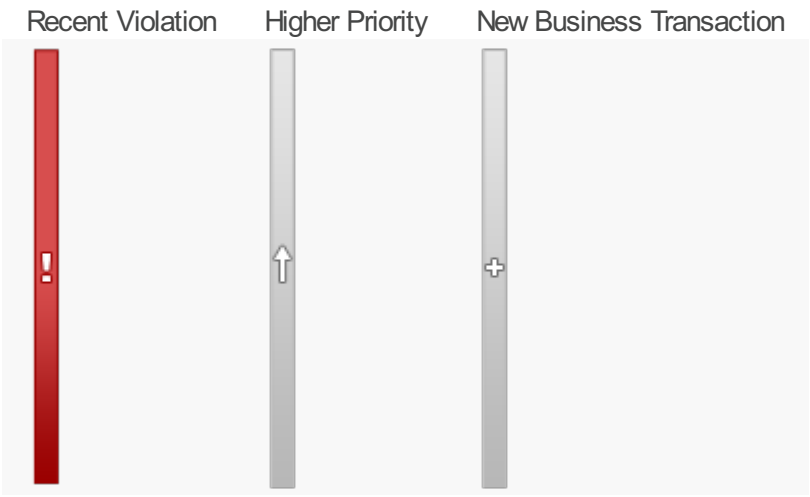
Business Transaction Splittings are listed in Application Details in order of relevance. A Business Transaction Splitting becomes more relevant if it signals abnormalities (failure rate violation, response time violation, or unhealthy processes) or has a higher throughput than other applications. Your favorite Business Transaction Splittings are listed first, and then the other splittings are listed according to the criteria in the following table. If a ranking criterion provides no measurable difference, the next criterion is considered.

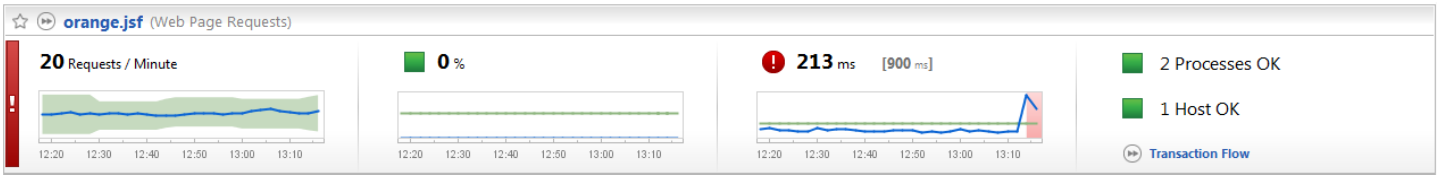
Priority	Criterion	Description
1	Splitting favorite	Favorite splittings are listed first.
2	Failure rate violation	Splittings with an active failure rate baseline violation are ranked higher.
3	Response time violation	Splittings with an active response time baseline violation are ranked higher.
4	Process health	Splittings are ranked by the number of unhealthy contributing processes.

5	Throughput	Splittings are ranked by average throughput value, or by the average throughput baseline value if it is available and is higher than the average throughput value.
6	Failure rate	Splittings are ranked by the current failure rate value.
7	Response time	Splittings are ranked by the current response time value.
8	Splitting Name	Splittings with equal rankings are sorted by name, ascending.

Markers

If a Business Transaction has recent violations, gets a higher priority, or is newly created it is marked as shown:





Application with recent violation

Business Transaction details

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Business Transaction splitting header

Configurations

Related pages

Identify problems in your application

Baseline and smart alerting explained

Application Overview dashboard

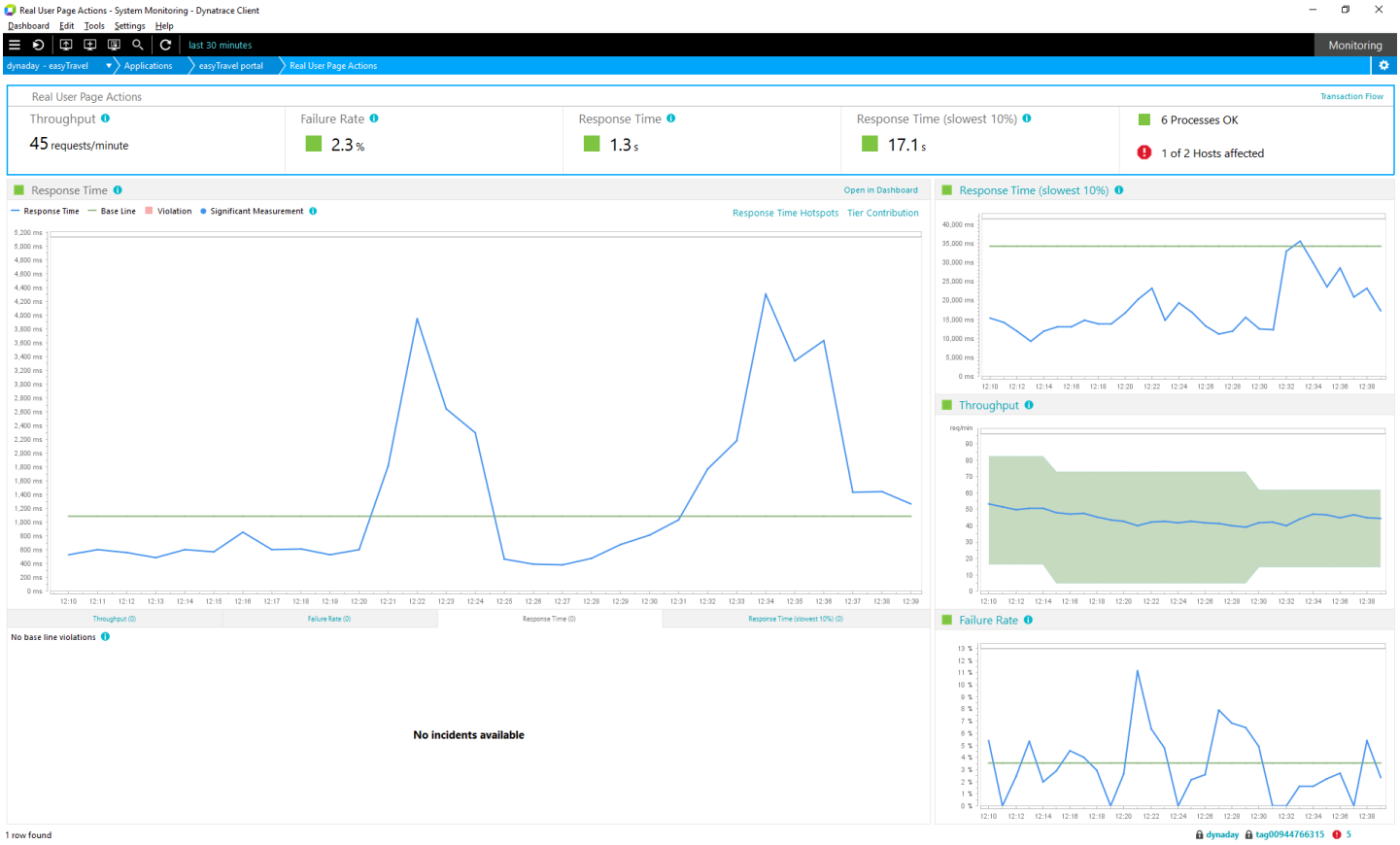
Application Details

Baseline chart

Infrastructure monitoring

This view shows the detailed charts for the selected Business Transaction Splitting. For each chart, there is also a list of violations in the timeframe.

There are specific drilldown options for failure rate and response times.



Business Transaction details

See Identify Problems in your Application - Identify the Transaction Problem for more information.

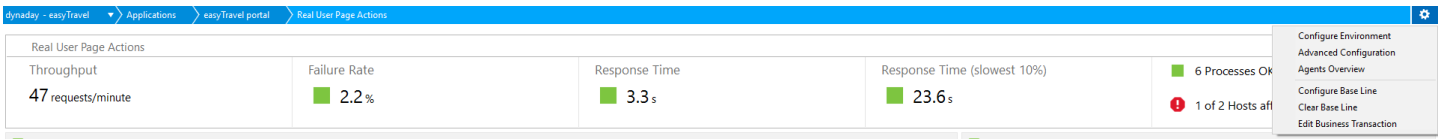
Business Transaction splitting header

The Business Transaction splitting header shows the status of the Business Transaction Splitting. See Application Overview - Business Transaction Splittings for more information.

Configurations

The configuration icon provides a menu with two actions:

- Configure Base Line
- Clear Base Line



Configuration Menu

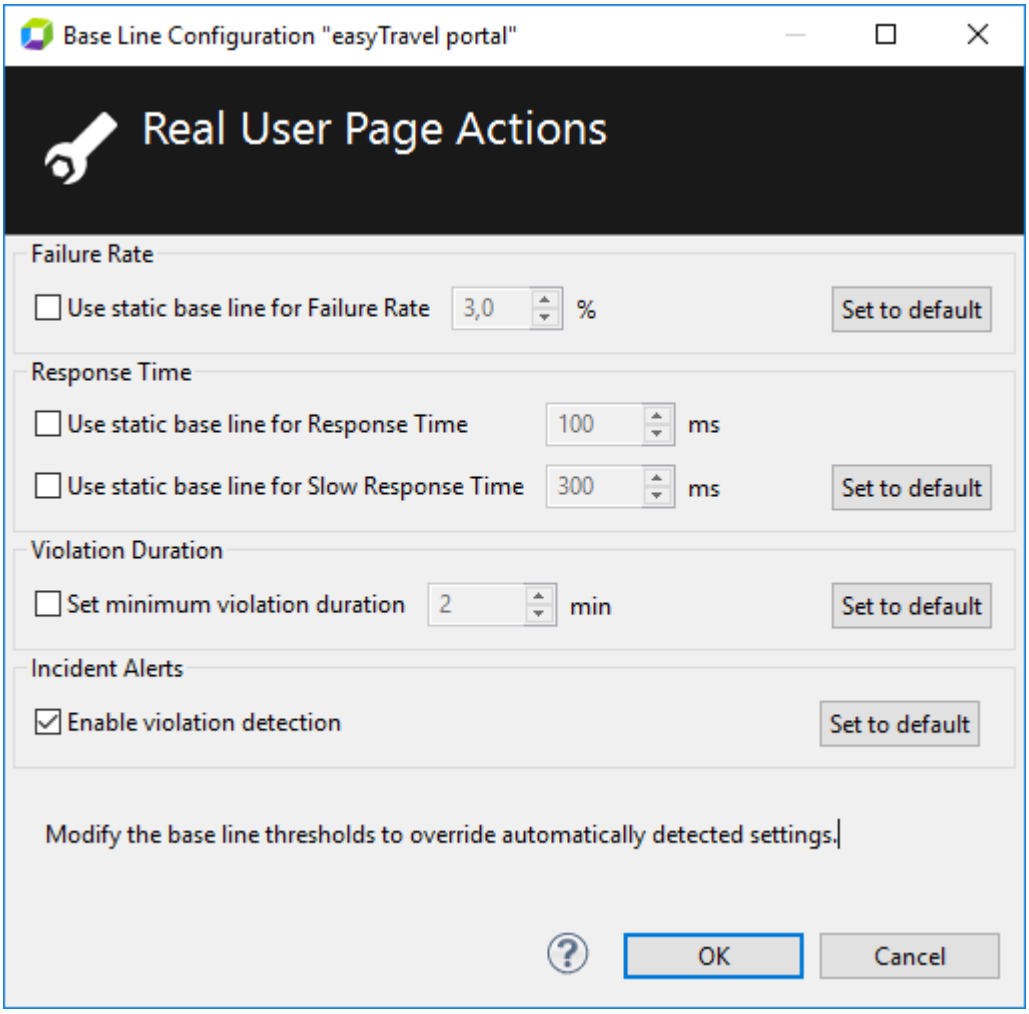
Configure base line

Use the Base Line Configuration dialog box to adjust the thresholds for failure rate and response times. Values greater than these settings trigger a baseline violation incident.

You can also open this dialog box through the System Profile Preferences.

1. Select **Business Transactions**.
2. Select a business transaction and click **Edit** to display the **Business Transaction Editor**.
3. Click **Violation Detections Settings** (you may need to scroll down to see the button).

Base Line Configuration



Base line configuration

Modifying the baseline thresholds overwrites the automatically detected settings.

You can also adjust the minimum violation duration. There is also an option to ignore a splitting by disabling the violation detection for this Business Transaction splitting. If you disable violation detection, the Business Transactions violation indicators become unavailable, as shown.



Unavailable violation indicators

Clear base line

When you select and confirm the selection, the baselines for the selected Business Transaction are reset and the data is lost.

Resetting the baseline is useful when an application is redeployed and a different behavior than before is expected.
Depending on the throughput, building a new baseline for the target Business Transaction may take an hour or longer.

Charts and violation lists

Click the Business Transaction name to open a dashlet that contains charts for Throughput, Failure Rate, Response Time, and Response Time (slowest 10%). Click a chart name to display it as the main chart. The baseline violations list displays below the main chart.

The following figure illustrates the main chart and violations list.

Main chart

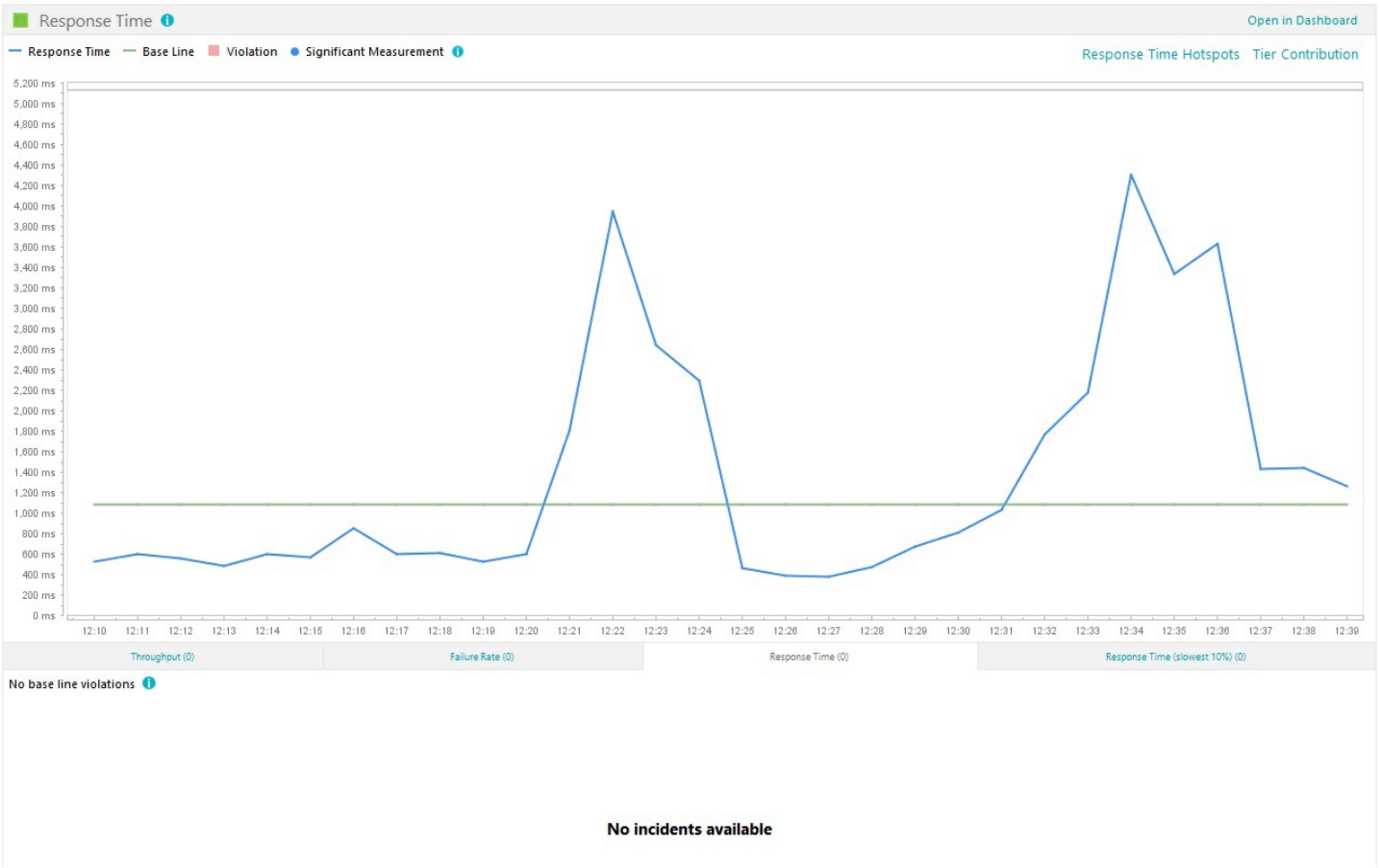
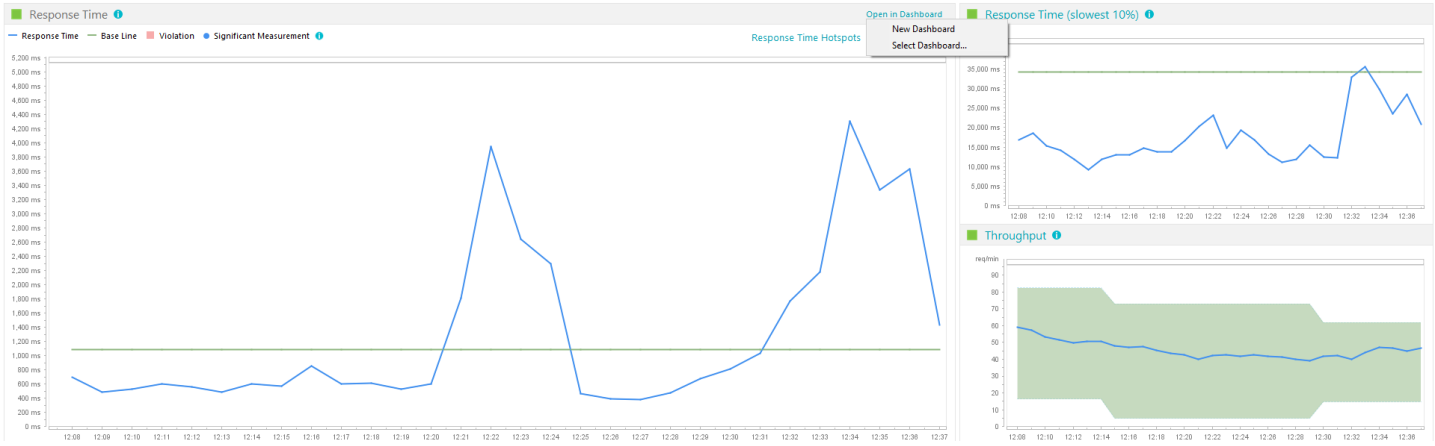


Chart features

Hover over a violation area to display a popup window with details and drilldown links for the violation.

Click **Open** in Dashboard to move the currently selected main chart into any dashboard. See Baseline Chart for details.



Open in dashboard commands

See Baseline and Smart Alerting Explained to find out more about how baselines and violations work.

Throughput chart

The Throughput chart shows the throughput count as a line graph (blue) against the expected range (green), based on the data from the last week. No violations as listed for throughput.

Hover over the data line to display a tooltip with the data for the selected point.

Failure rate chart

The Failure Rate chart shows the failure rate trend (blue), baseline (green), and violations (red areas). The baseline shows the failure rate behavior based on the data from the last week.

Click the **Errors** link to drill down to the Errors dashlet. Click **Tier Contribution** to drill down to the Transaction Flow in Failures visualization mode. These drilldowns are available for the timeframe or specifically for the violation duration.

Response time

The median response time is the time where 50% of requests have a slower response time and 50% have an equal or faster response time. This value is considered the typical response time. The chart shows the response time trend (blue), baseline (green), and violations (red areas). The baseline shows the response time behavior based on the data from the last week.

Click **Response Time Hotspots** to drill down to the Hotspots dashboard. Click **Tier Contribution** to drill down to the Transaction Flow in Response Time visualization mode. These drilldowns are available for the timeframe or specifically for the violation duration.

Response time (slowest 10 percent)

The response time of the slowest 10% denotes the 90th percentile: 10% of requests have a slower response time and 90% have a faster or equal response time. These transactions are the outliers.

The chart shows the response time trend (blue), baseline (green), and violations (red areas). The baseline reflects the response time behavior based on the data from the last week.

Click **Response Time Hotspots** to drill down to the Hotspots dashboard. Click **Tier Contribution** to drill down to the Transaction Flow in Response Time visualization mode. These drilldowns are available for the timeframe or specifically for the violation duration.

Baseline chart

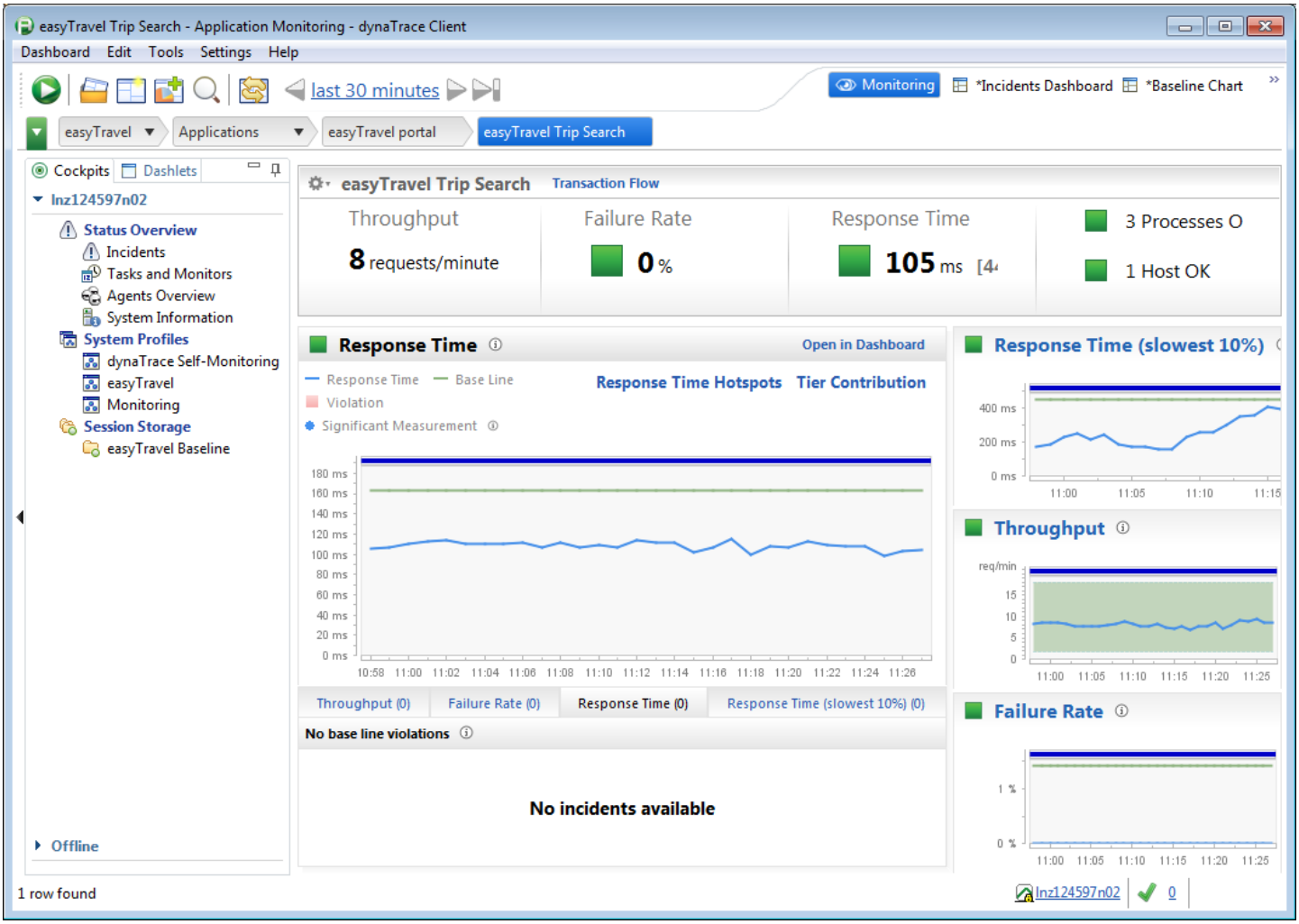
Looking for the 6.5 version of this page? Open in 6.5 documentation

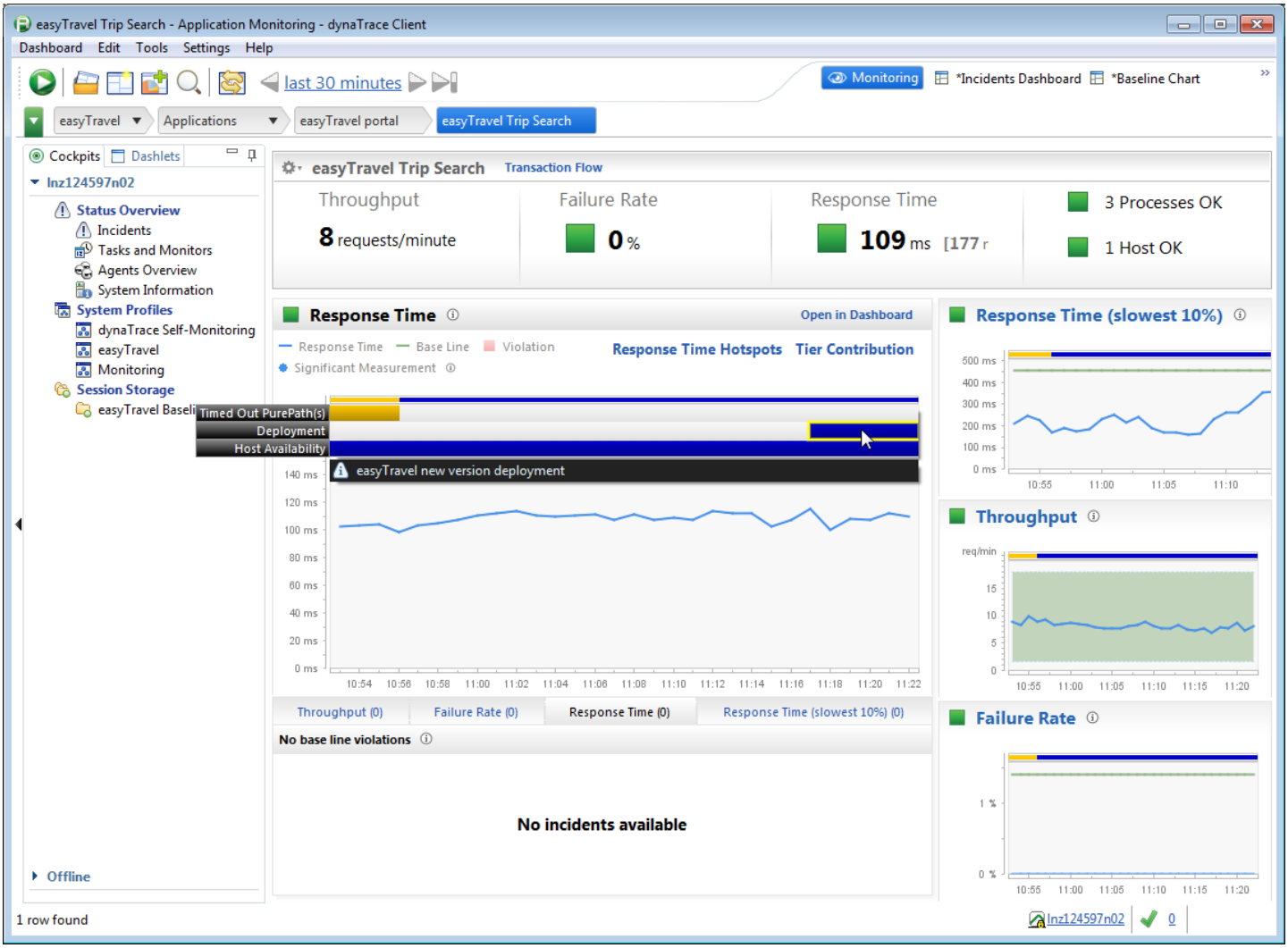
Related pages

Business Transaction details

Incidents dashlet

Charting

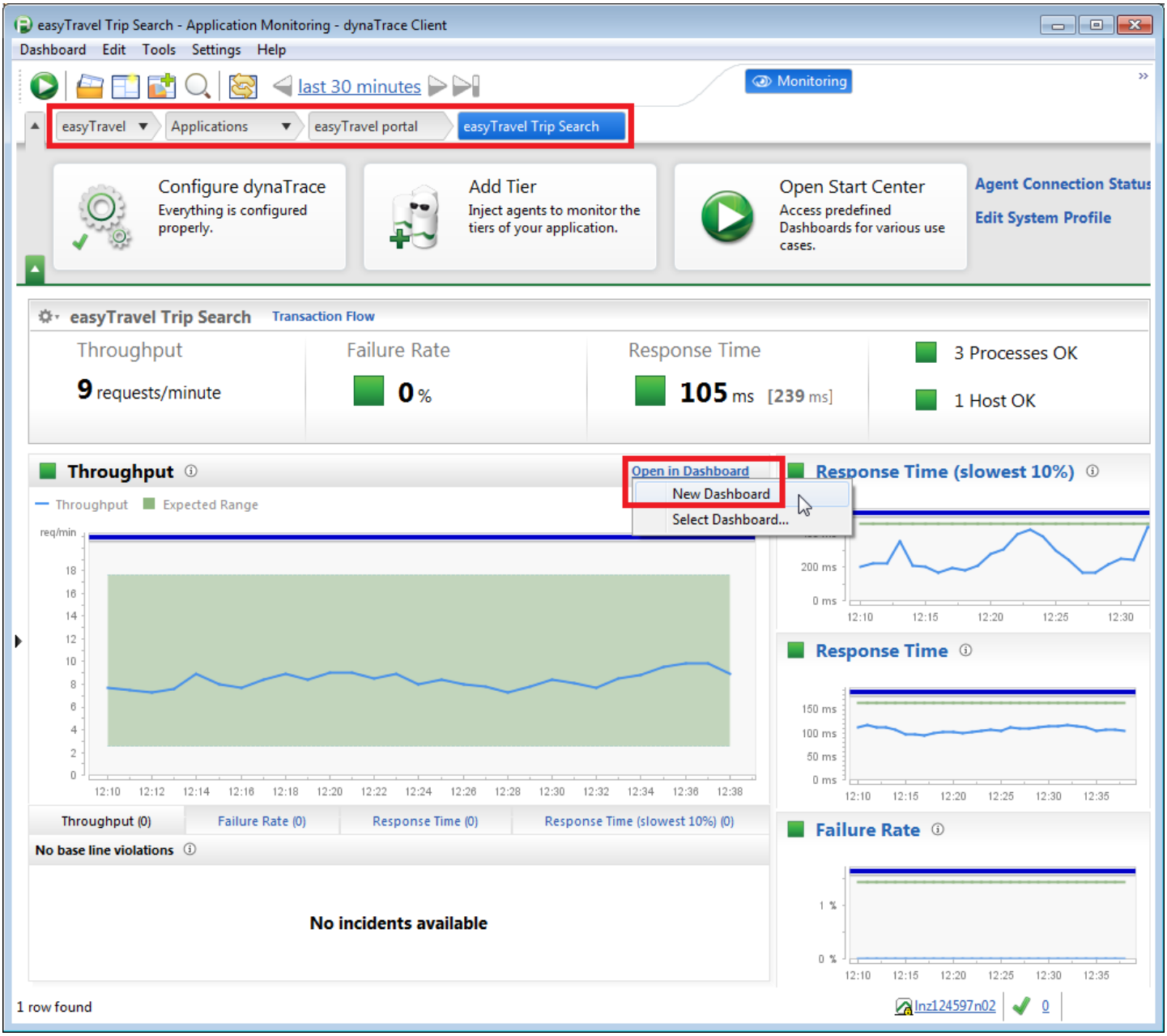


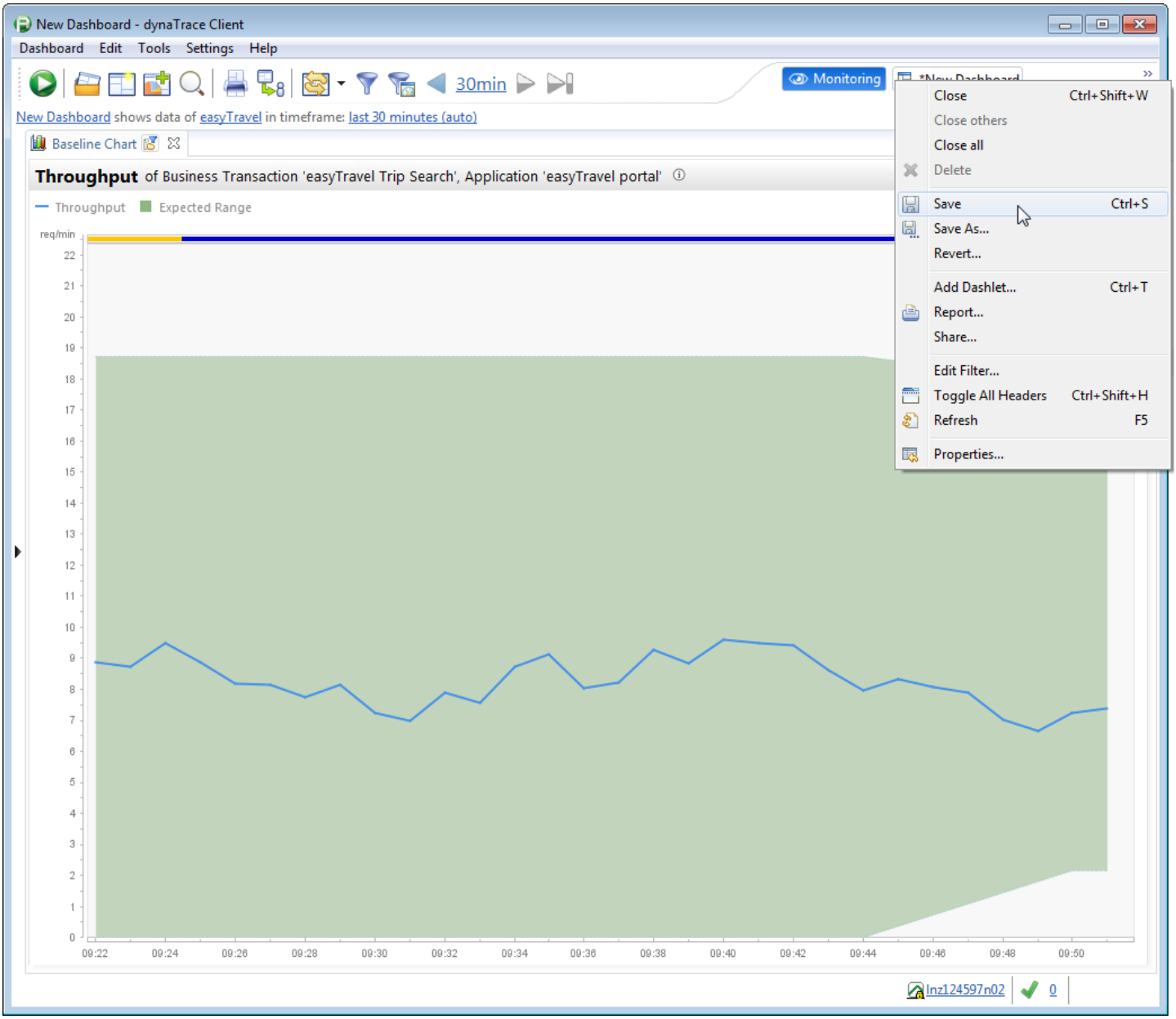


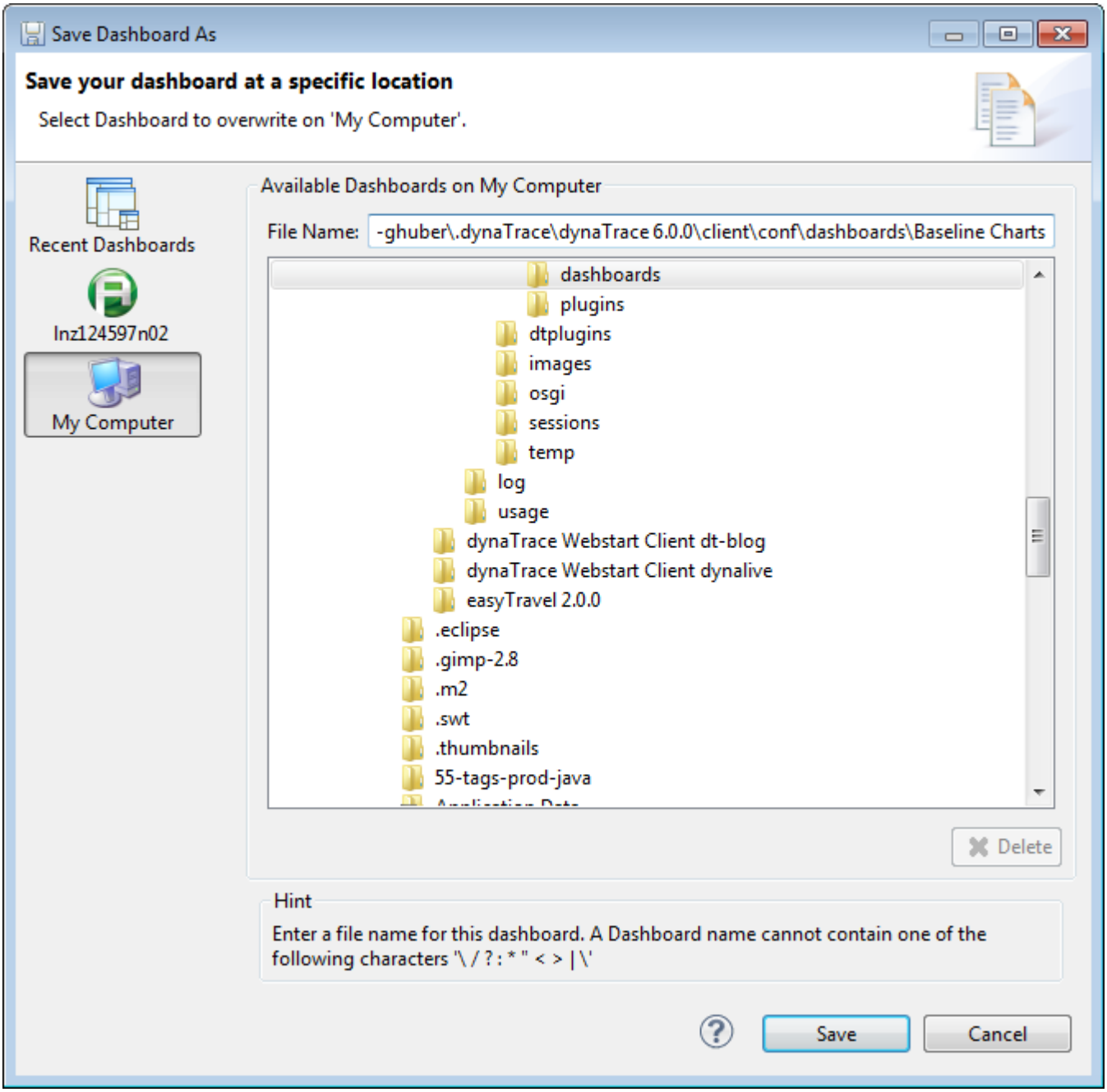
Open in dashboard link / menu

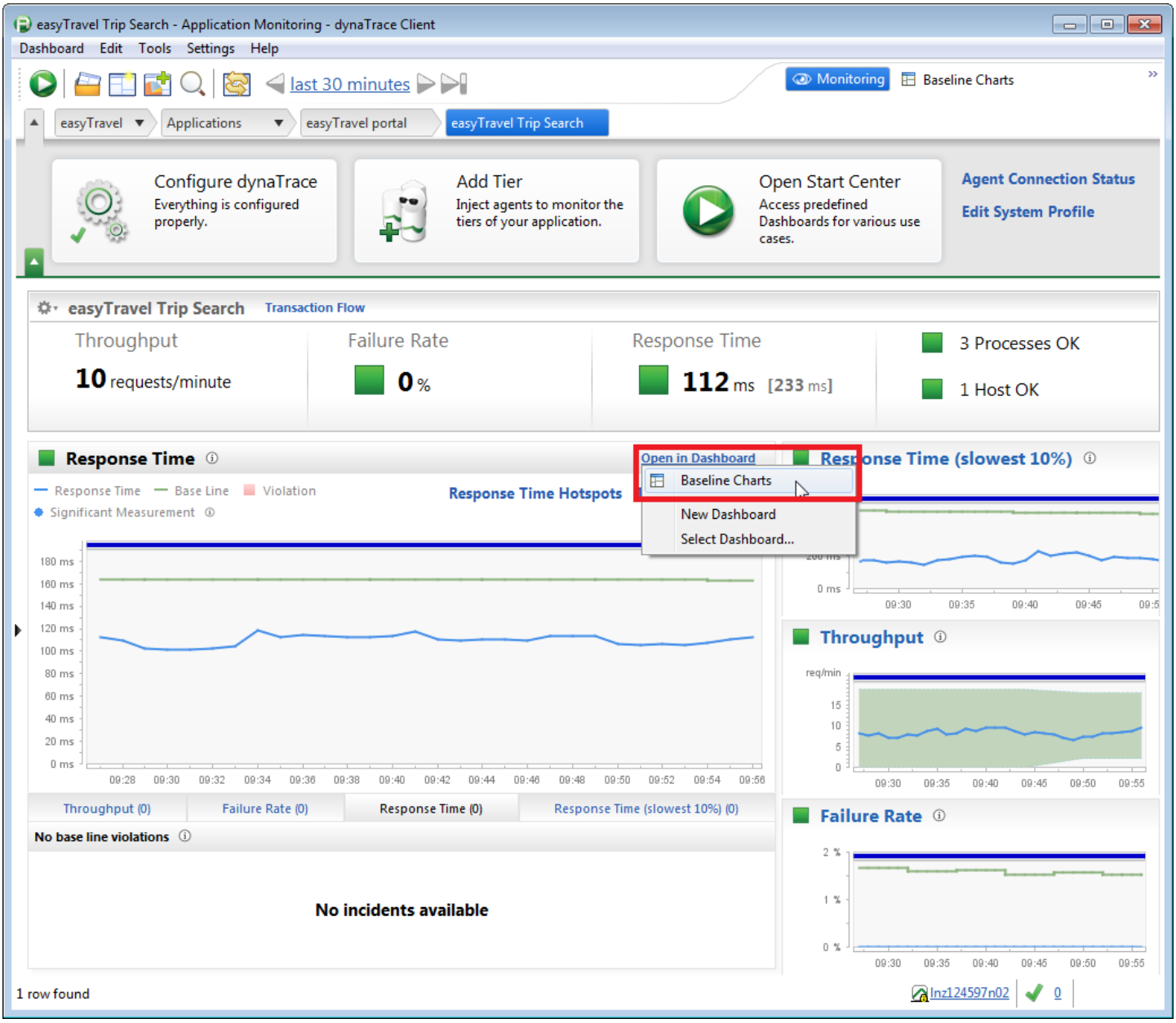
The Open in Dashboard menu (looks like a link) lets you open the selected Business Transaction Details dashlet in a new dashboard or in a saved one and thus create a custom dashboard. Optionally arrange the dashboard's dashlets into the desired layout.

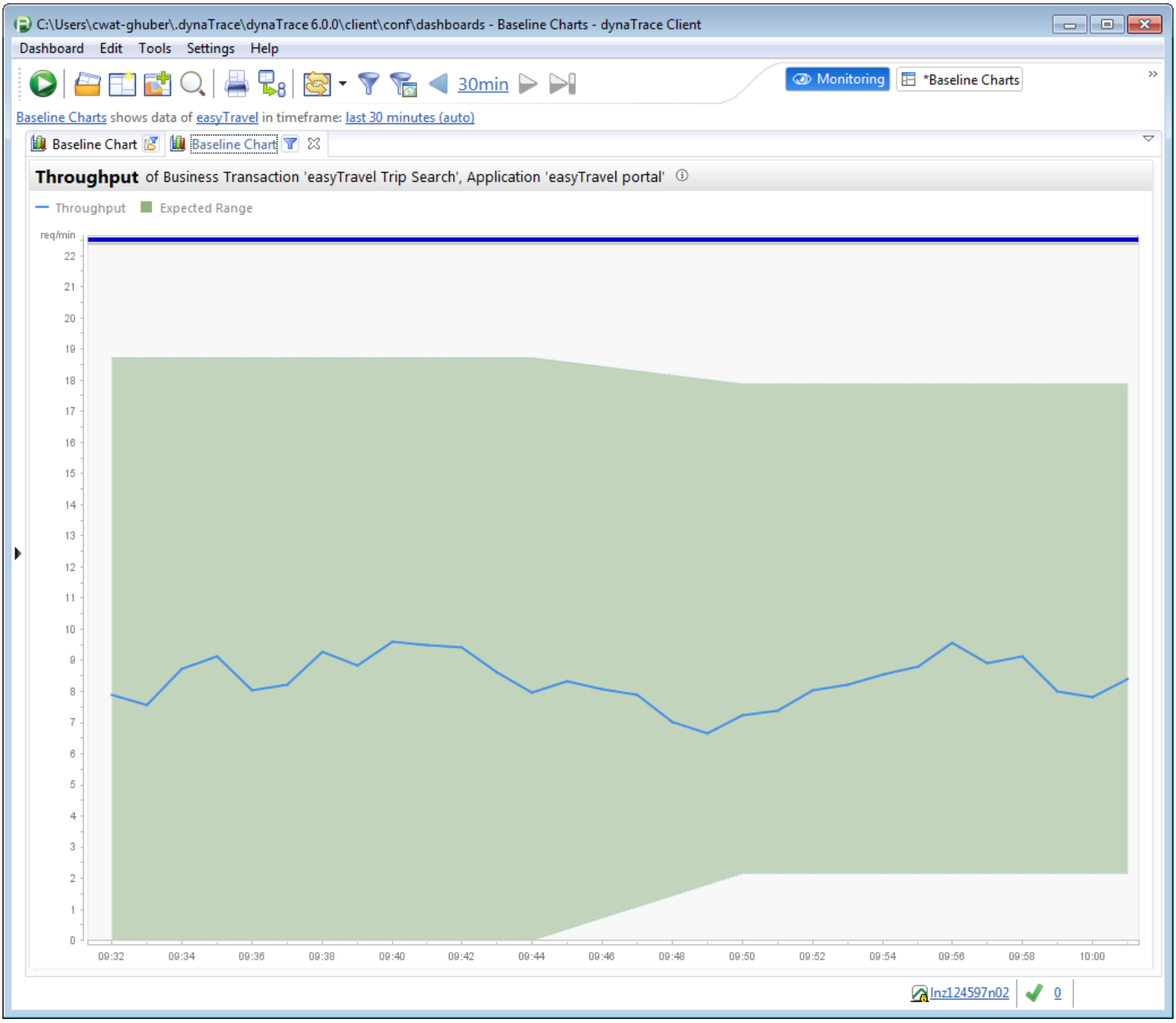
The following screenshots show how to open the Business Transaction Details Throughput and Response Time charts in new dashlets and arrange them on top of each other in a new dashboard called Baseline Charts.

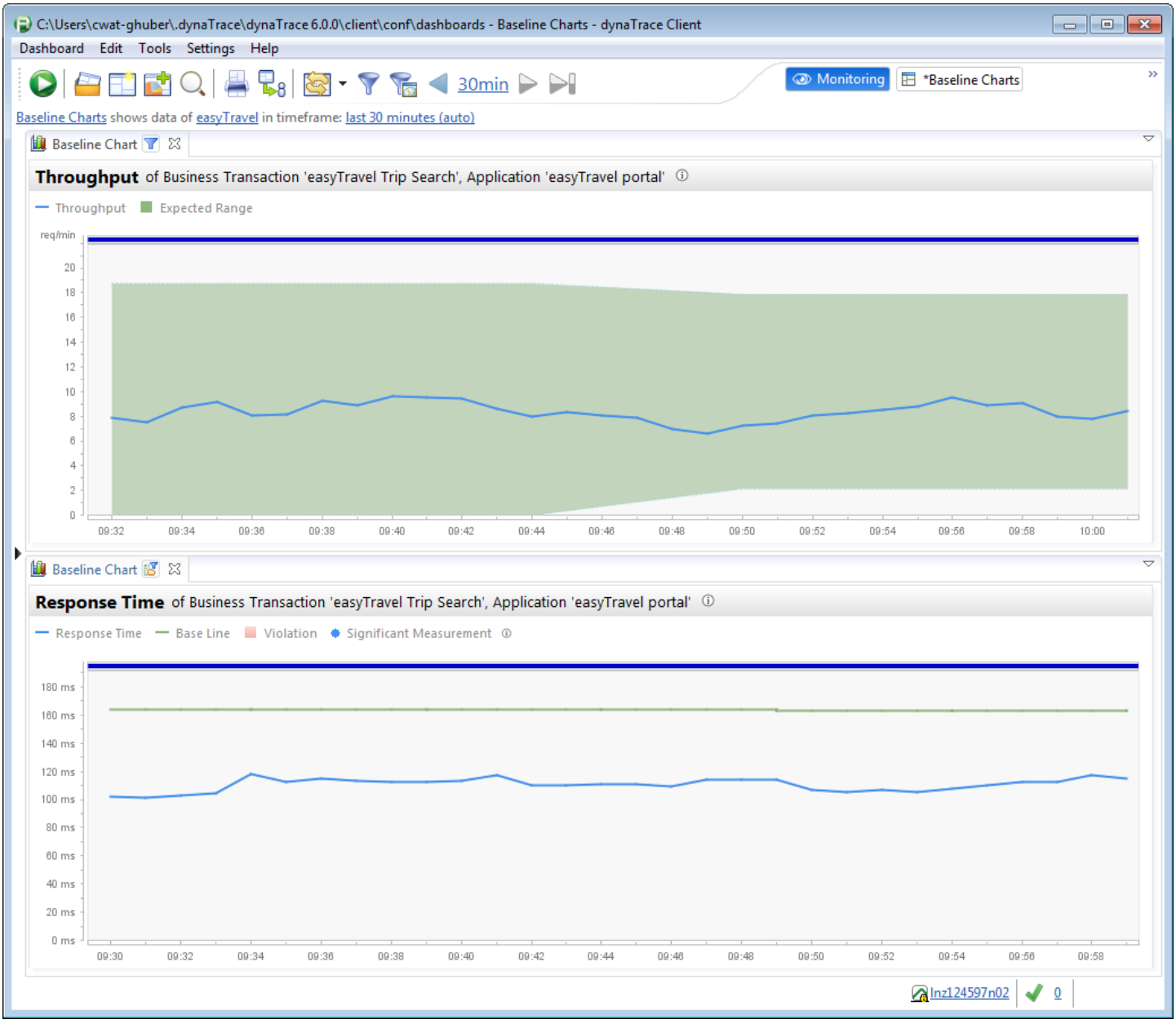

easyTravel Trip Search - Application Monitoring - dynaTrace Client
Dashboard Edit Tools Settings Help
last 30 minutes
Monitoring
easyTravel
Applications
easyTravel portal
easyTravel Trip Search
Configure dynaTrace
Everything is configured properly.
Add Tier
Inject agents to monitor the tiers of your application.
Open Start Center
Access predefined Dashboards for various use cases.
Agent Connection Status
Edit System Profile
easyTravel Trip Search
Transaction Flow
Throughput
9 requests/minute
Failure Rate
0 %
Response Time
105 ms [239 ms]
3 Processes OK
1 Host OK
Throughput
Open in Dashboard
New Dashboard
Select Dashboard...
Throughput
Expected Range
req/min
Response Time (slowest 10%)
Response Time
Failure Rate
Throughput (0)
Failure Rate (0)
Response Time (0)
Response Time (slowest 10%) (0)
No base line violations
No incidents available
1 row found
lnz124597n02
0


New Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
New Dashboard shows data of easyTravel in timeframe: last 30 minutes (auto)
Baseline Chart
Throughput of Business Transaction 'easyTravel Trip Search', Application 'easyTravel portal'
Throughput
Expected Range
req/min
Close Ctrl+Shift+W
Close others
Close all
Delete
Save Ctrl+S
Save As...
Revert...
Add Dashlet... Ctrl+T
Report...
Share...
Edit Filter...
Toggle All Headers Ctrl+Shift+H
Refresh F5
Properties...
lnz124597n02
0


Save Dashboard As
Save your dashboard at a specific location
Select Dashboard to overwrite on 'My Computer'.
Recent Dashboards
lnz124597n02
My Computer
Available Dashboards on My Computer
File Name: -ghuber\.dynaTrace\dynaTrace 6.0.0\client\conf\dashboards\Baseline Charts
dashboards
plugins
dtplugins
images
osgi
sessions
temp
log
usage
dynaTrace Webstart Client dt-blog
dynaTrace Webstart Client dynalive
easyTravel 2.0.0
.eclipse
.gimp-2.8
.m2
.swt
.thumbnails
55-tags-prod-java
Delete
Hint
Enter a file name for this dashboard. A Dashboard name cannot contain one of the following characters '\ / ? : * " < > | \'
Save
Cancel


easyTravel Trip Search - Application Monitoring - dynaTrace Client
Dashboard Edit Tools Settings Help
last 30 minutes
Monitoring
Baseline Charts
easyTravel
Applications
easyTravel portal
easyTravel Trip Search
Configure dynaTrace
Everything is configured properly.
Add Tier
Inject agents to monitor the tiers of your application.
Open Start Center
Access predefined Dashboards for various use cases.
Agent Connection Status
Edit System Profile
easyTravel Trip Search
Transaction Flow
Throughput
10 requests/minute
Failure Rate
0 %
Response Time
112 ms [233 ms]
3 Processes OK
1 Host OK
Response Time
Response Time Base Line Violation
Significant Measurement
Response Time Hotspots
Open in Dashboard
Baseline Charts
New Dashboard
Select Dashboard...
Response Time (slowest 10%)
Throughput
req/min
Failure Rate
Throughput (0)
Failure Rate (0)
Response Time (0)
Response Time (slowest 10%) (0)
No base line violations
No incidents available
1 row found
lnz124597n02
0

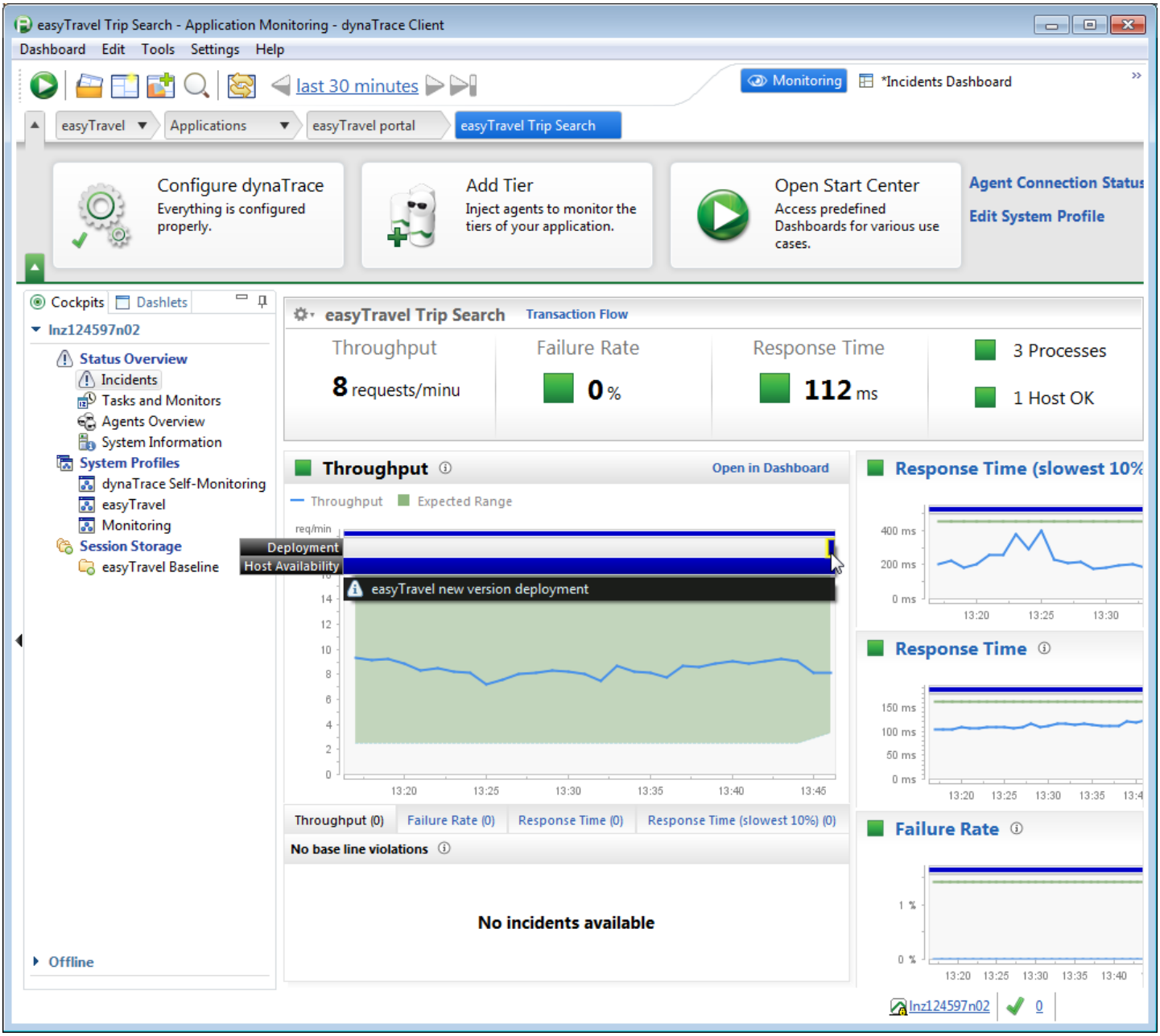

C:\Users\cwat-ghuber\.dynaTrace\dynaTrace 6.0.0\client\conf\dashboards - Baseline Charts - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*Baseline Charts
Baseline Charts shows data of easyTravel in timeframe: last 30 minutes (auto)
Baseline Chart
Baseline Chart
Throughput of Business Transaction 'easyTravel Trip Search', Application 'easyTravel portal'
Throughput
Expected Range
req/min
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
09:32
09:34
09:36
09:38
09:40
09:42
09:44
09:46
09:48
09:50
09:52
09:54
09:56
09:58
10:00
lnz124597n02
0



Note

A custom timeframe filter is set for the newly added dashlet. You might want to adapt it according to your needs.

As default the **Baseline Chart** shows the new informational incident rule *Deployment* in a heat field. For further information about heat fields see Incidents Dashlet and Chart Dashlet.


easyTravel Trip Search - Application Monitoring - dynaTrace Client
Dashboard Edit Tools Settings Help
last 30 minutes
Monitoring
*Incidents Dashboard
easyTravel
Applications
easyTravel portal
easyTravel Trip Search
Configure dynaTrace
Everything is configured properly.
Add Tier
Inject agents to monitor the tiers of your application.
Open Start Center
Access predefined Dashboards for various use cases.
Agent Connection Status
Edit System Profile
Cockpits
Dashlets
lnz124597n02
Status Overview
Incidents
Tasks and Monitors
Agents Overview
System Information
System Profiles
dynaTrace Self-Monitoring
easyTravel
Monitoring
Session Storage
easyTravel Baseline
Offline
easyTravel Trip Search
Transaction Flow
Throughput
8 requests/minu
Failure Rate
0 %
Response Time
112 ms
3 Processes
1 Host OK
Throughput
Open in Dashboard
Throughput
Expected Range
req/min
Deployment
Host Availability
easyTravel new version deployment
Response Time (slowest 10%
Response Time
Failure Rate
Throughput (0)
Failure Rate (0)
Response Time (0)
Response Time (slowest 10%) (0)
No base line violations
No incidents available
lnz124597n02
0

End User Experience dashboard

Extended World Map Dashboard

Infrastructure monitoring dashboards

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Infrastructure overview - global / overall

Infrastructure overview - bound to System Profile

Working with views of the infrastructure in the AppMon Client

Set up alerts

Related pages

Infrastructure

Infrastructure monitoring

Application Overview dashboard

Host health monitoring

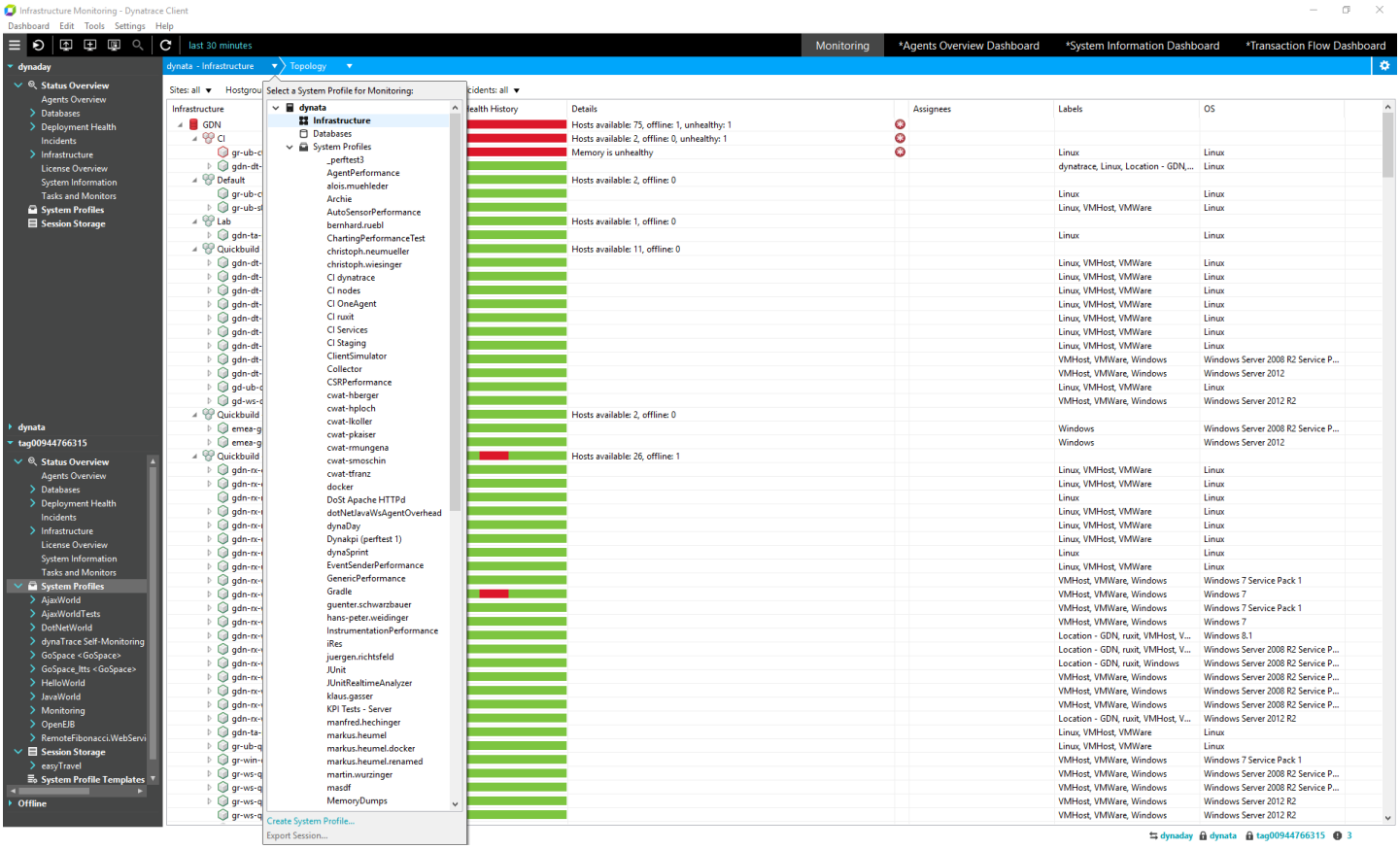
This page describes approaches to infrastructure monitoring using the following:

- Infrastructure Overview dashboard in the AppMon AppMon Client and its view modes (topology; host lists by name or label and hosts shown as tiles).
- web UI Operations dashboards showing hosts and processes and infrastructure and application alerts.

Infrastructure overview - global / overall

This global view (over all System Profiles) is currently only available in the AppMon Client and not in the web dashboards.
Select **Infrastructure** from the breadcrumbs popup to see the full picture. On the second breadcrumb select the view that suits your purpose best.

The following shows a filtered, System-Profile-bound Infrastructure Overview.



Infrastructure Overview Dashboard (global / overall view) **> Topology view**

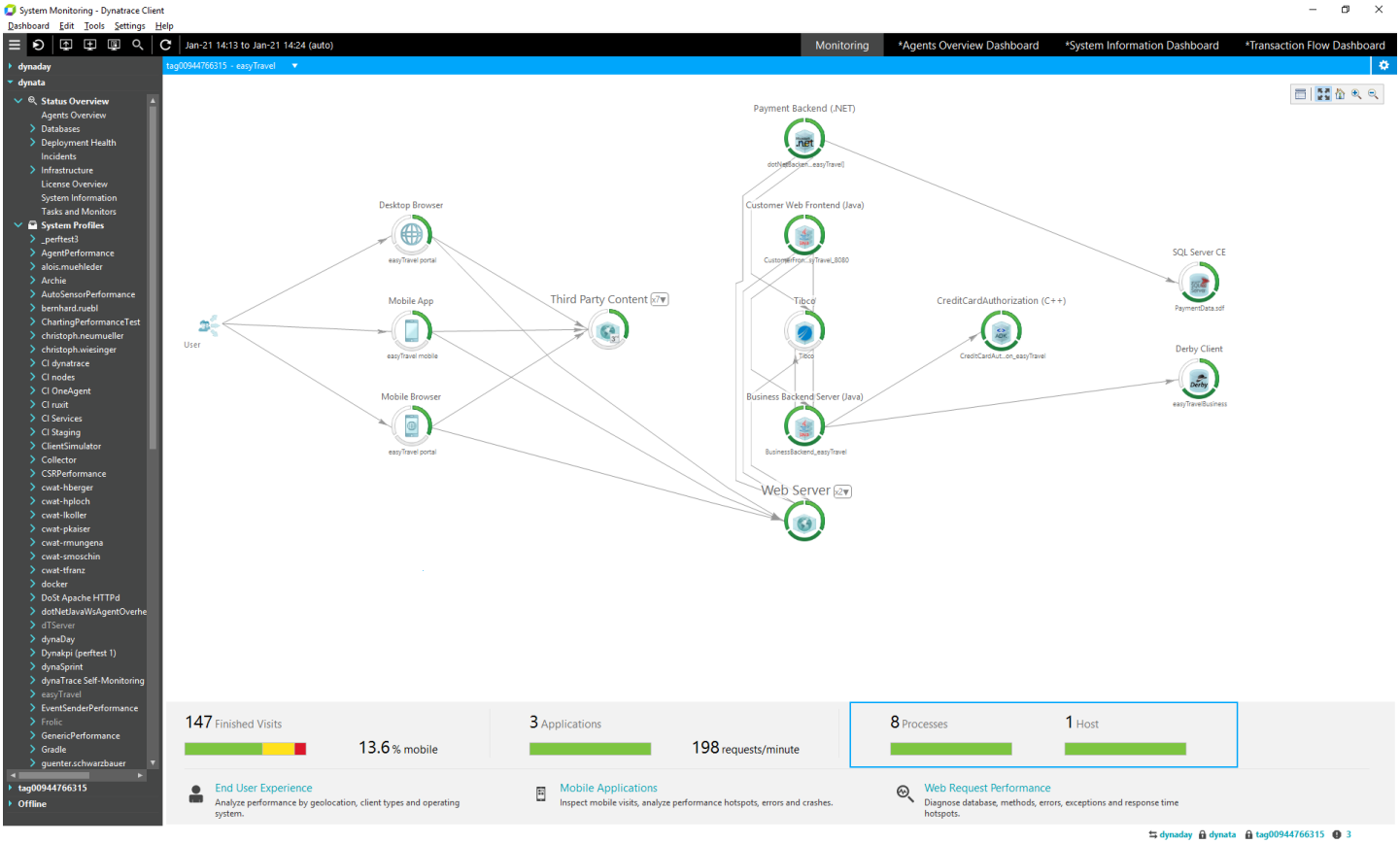
Infrastructure overview - bound to System Profile

Web UI Operations Dashboards

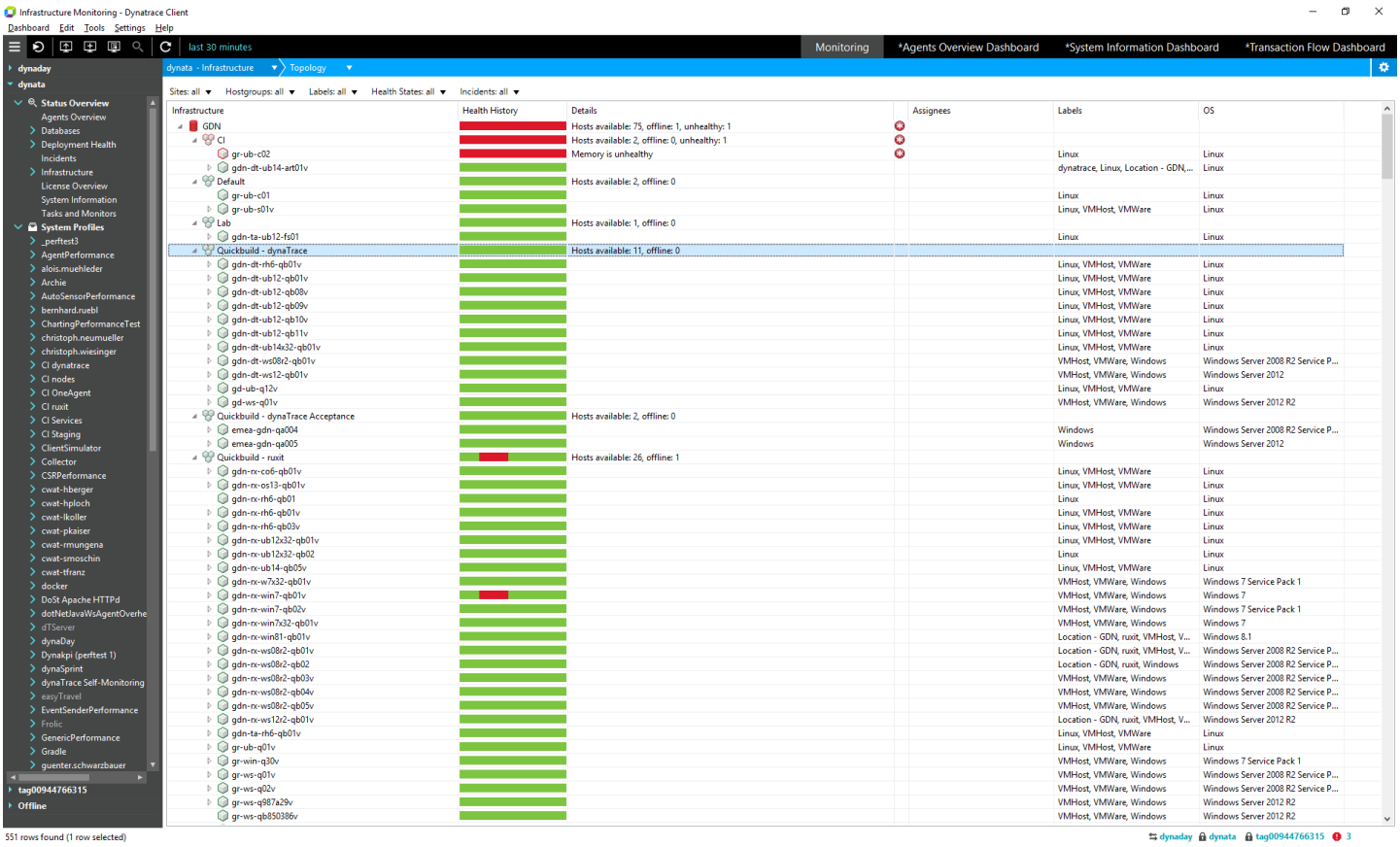
The Web UI provides a general infrastructure overview filtered by the selected system profile to quickly identify problems related to hosts, processes and databases. Additionally host, process and database overviews allow for additional filtering and enable you to drill down to a detail view of each host, process or database for in depth analysis. The various operations views are accessible via the side panel under the operations section.

AppMon Client infrastructure overview filtered to System Profile

In the AppMon Client please note the difference between the global view when you select Infrastructure from the breadcrumbs – and the filtered view that stems from a drill-down from a System Profile - (Application) Monitoring Overview.



easyTravel System Profile - (Application) Monitoring Overview - click Processes & Hosts to drill down



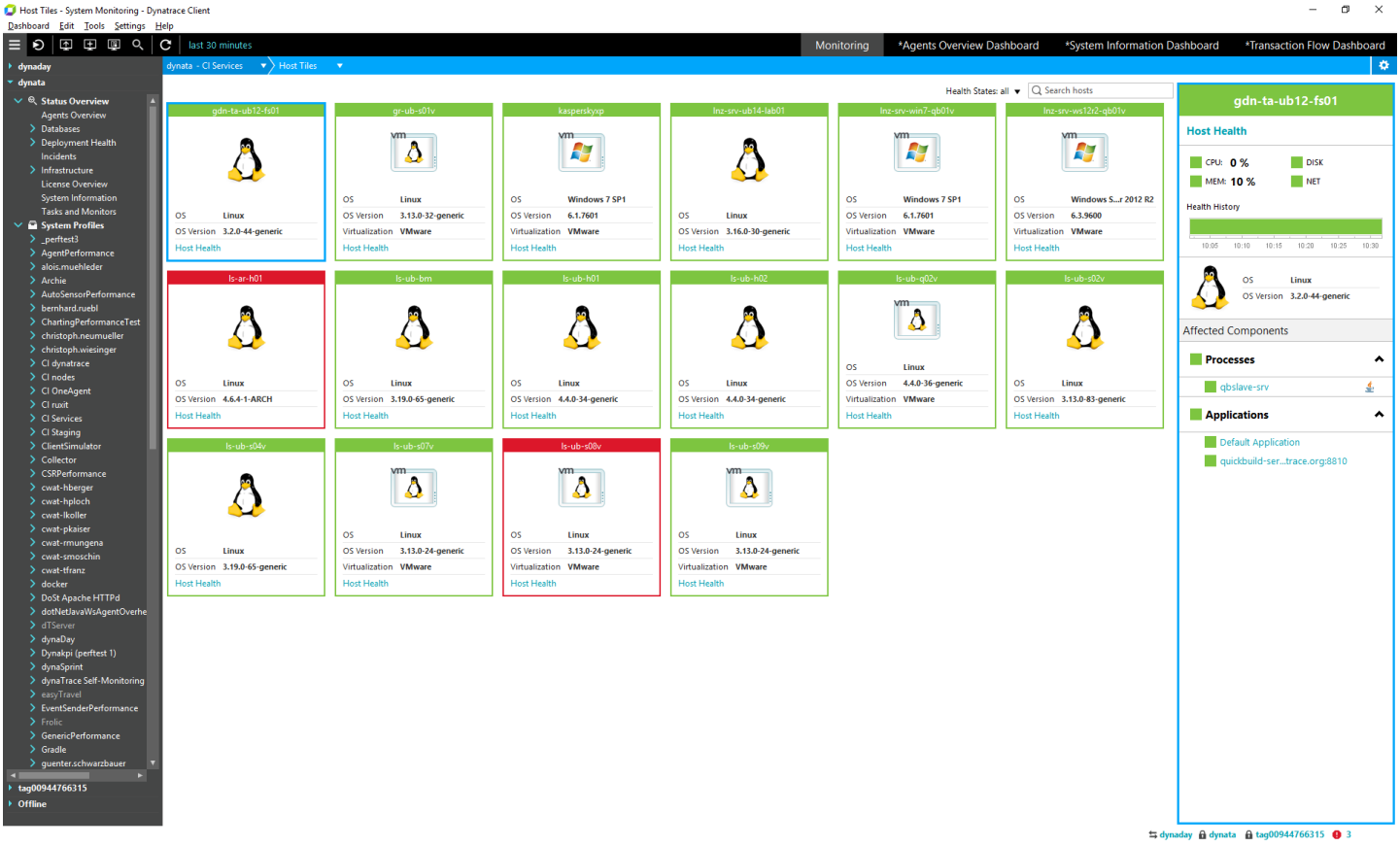
Topology view after a drill down from Application Monitoring Overview, only showing hosts related to easyTravel applications

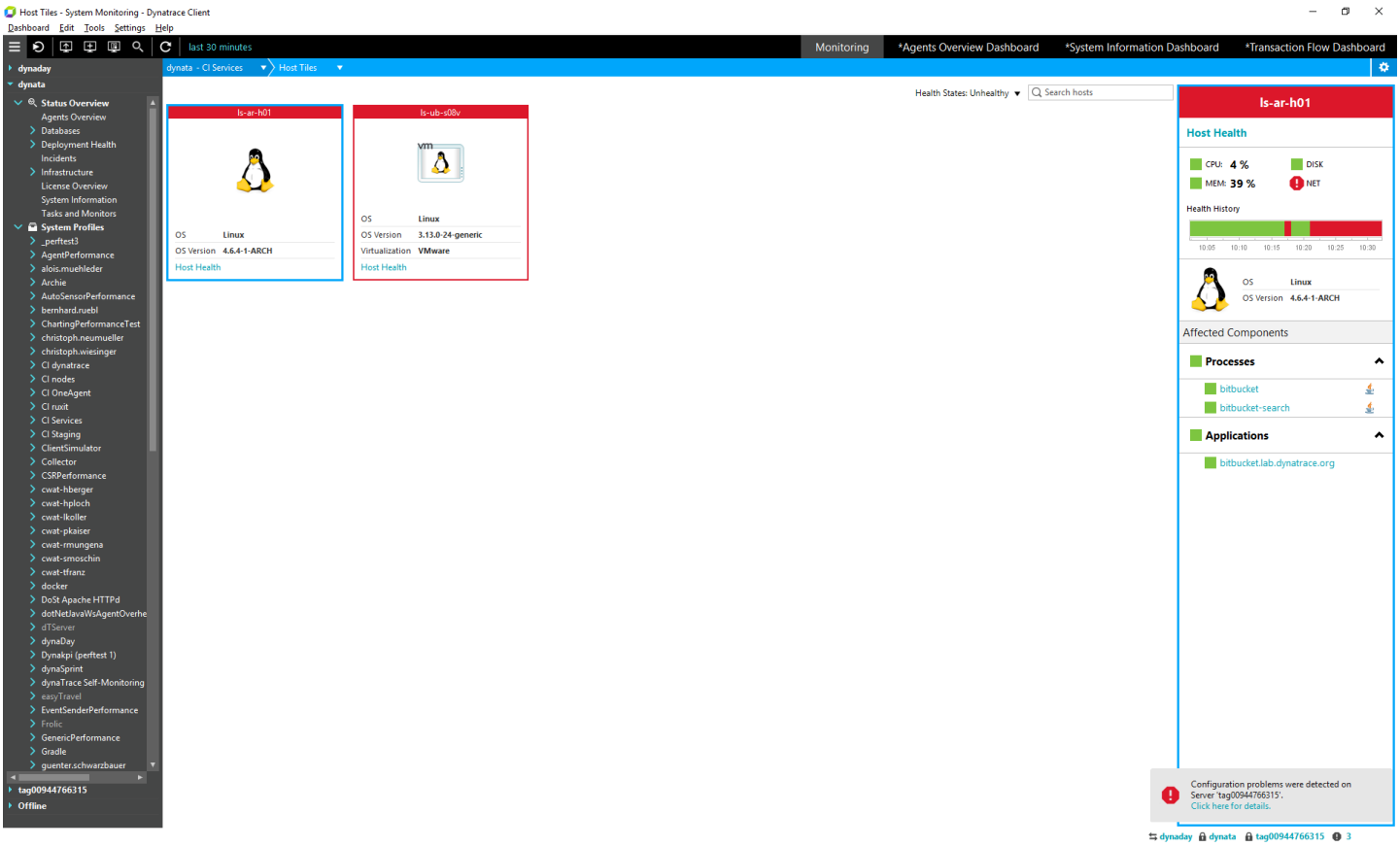
Working with views of the infrastructure in the AppMon Client

The features of the views should be pretty much self-explanatory. In doubt, hover to get additional info. If hovering isn't available (on a tablet) bravely tap / touch to explore.

Working with views - host tiles

If the Host Tiles view gets too crowded with the several thousand hosts that can be monitored at once with one AppMon Server, please switch to a different view and use its new grouping and filter features.

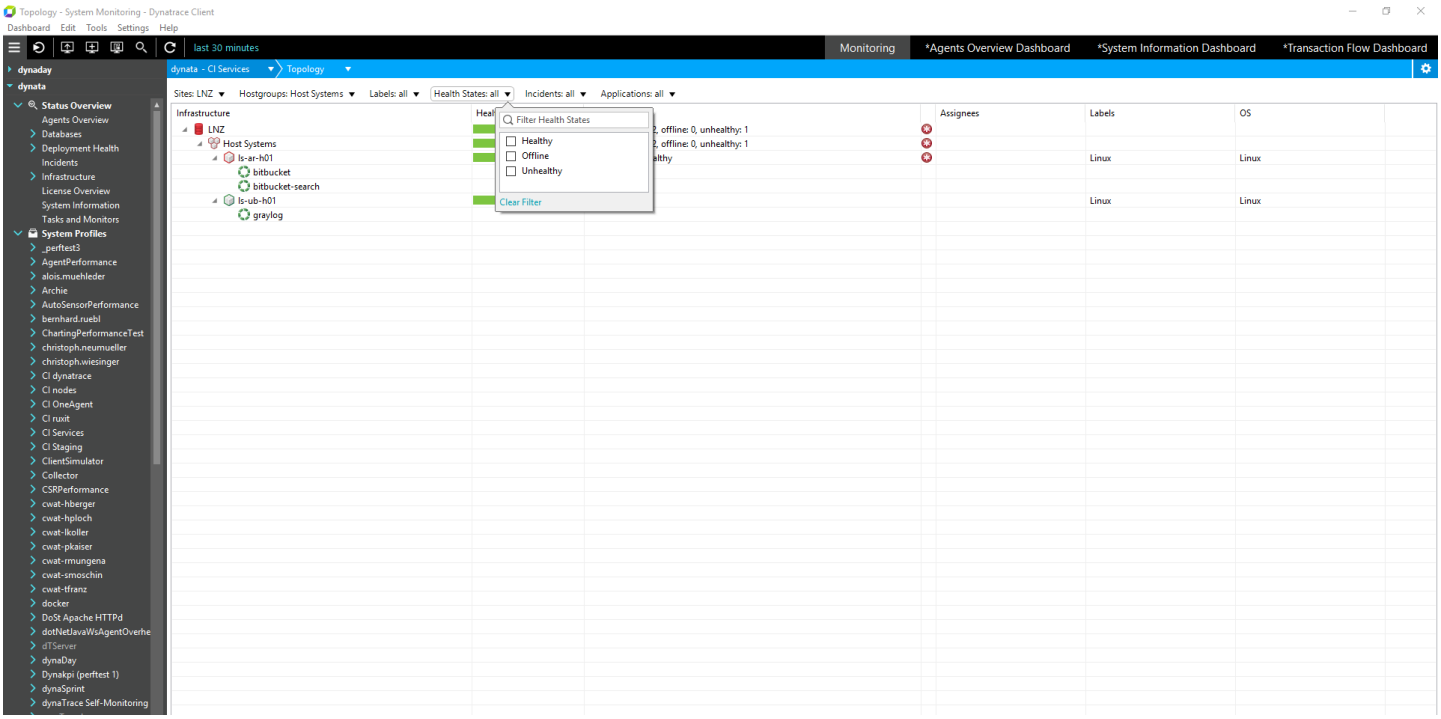




Working with views - topology, host / process list, hosts by label

Please be sure to utilize the sites, hostgroups, health states filters on top of the selected view, corresponding to the items that you set up in **Settings > Dynatrace Server > Infrastructure**.

Filtering data



The show data can be filtered via selecting different filter options.

Filter options within a group are connected with a logical **or**. Different filter groups are connected with a logical **and**. Filters are applied to hosts except for the process list. Application data does not necessarily reflect the selected timeframe but rather the current status.

Example

If Site=”site”, Hostgroup=”Windows” and Hostgroup=”Linux” are selected as filters they are grouped as followed:

*Site=”site” **AND** (Hostgroup=”Windows” **OR** Hostgroup=”Linux”) means that only hosts with Site=”site” and either having Hostgroup=”Windows” or Hostgroup=”Linux” are shown.

Set up alerts

In the AppMon Client, choose **Settings > Dynatrace Server > Infrastructure > Alerts** horizontal tab to set up notifications with great flexibility. See Infrastructure for more information.

Mobile Applications dashboard

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Performance

Analytics

User experience

Event tracking

This dashboard shows UEM metrics for mobile applications.



Performance

The average response time of mobile actions for the selected timeframe is charted. Additionally maximum and minimum response times during same period dispaly.

Analytics

The chart shows number of mobile visits (blue), errors in mobile actions (orange) and application crashes (red). The numbers are also summarized left to the chart.

User experience

This section gives you an overview about the user experience of your mobile visits. See User Experience Management for details about how user experience index is calculated.

Event tracking

The average number of actions a visit triggers in your mobile application is visualized in this chart. Please check the click path in the Visits Dashlet dashlet to get a better understanding about how your application is used.

Transaction Flow dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Transaction Flow icons

View modes

Zooming

Transaction Flow zooming actions

Differences in zooming view modes

Health visualization

Values visible in Transaction Flow

Work with Transaction Flow

Related pages

External API services

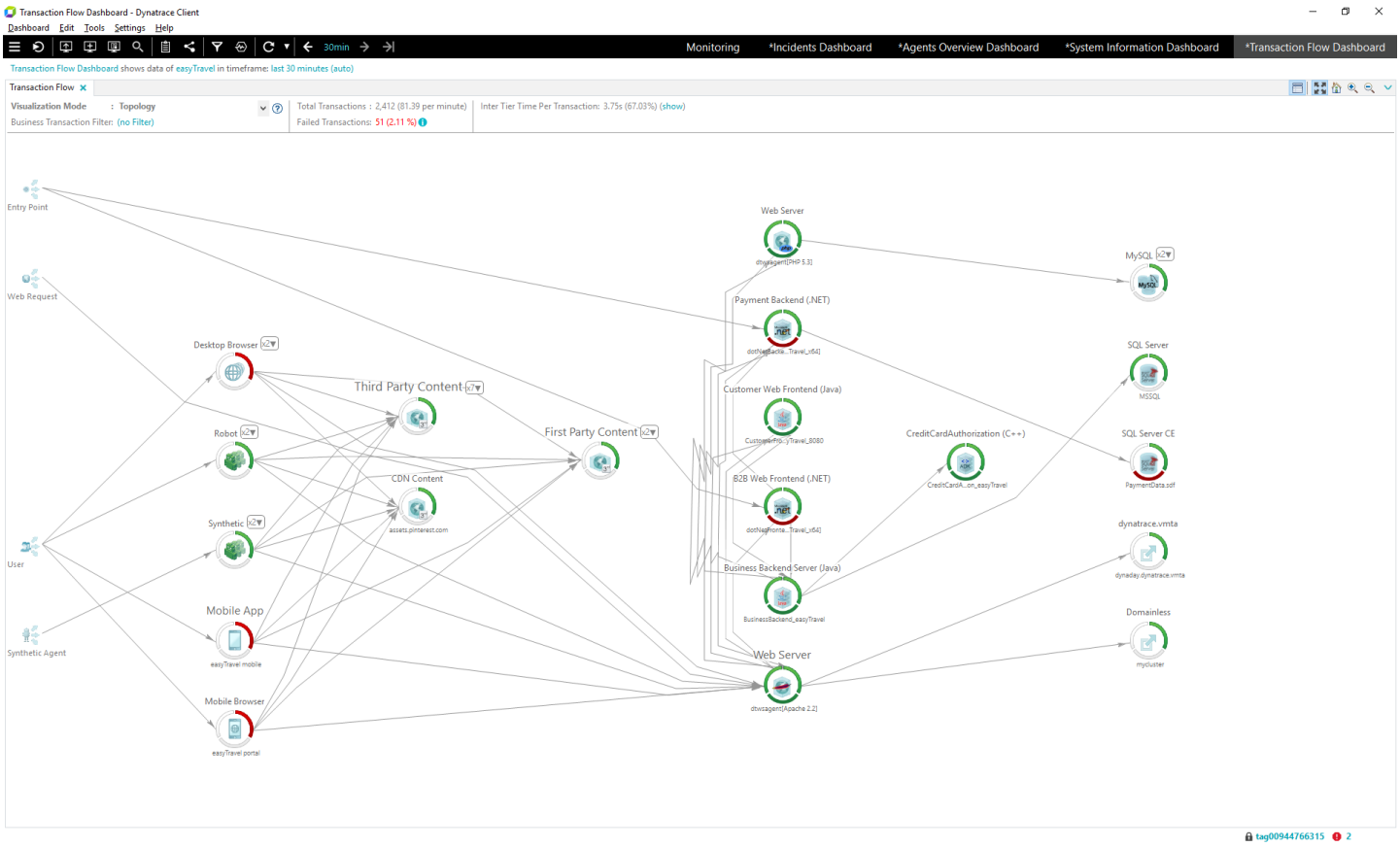
Observed tiers

Follow your transactions with Transaction Flow

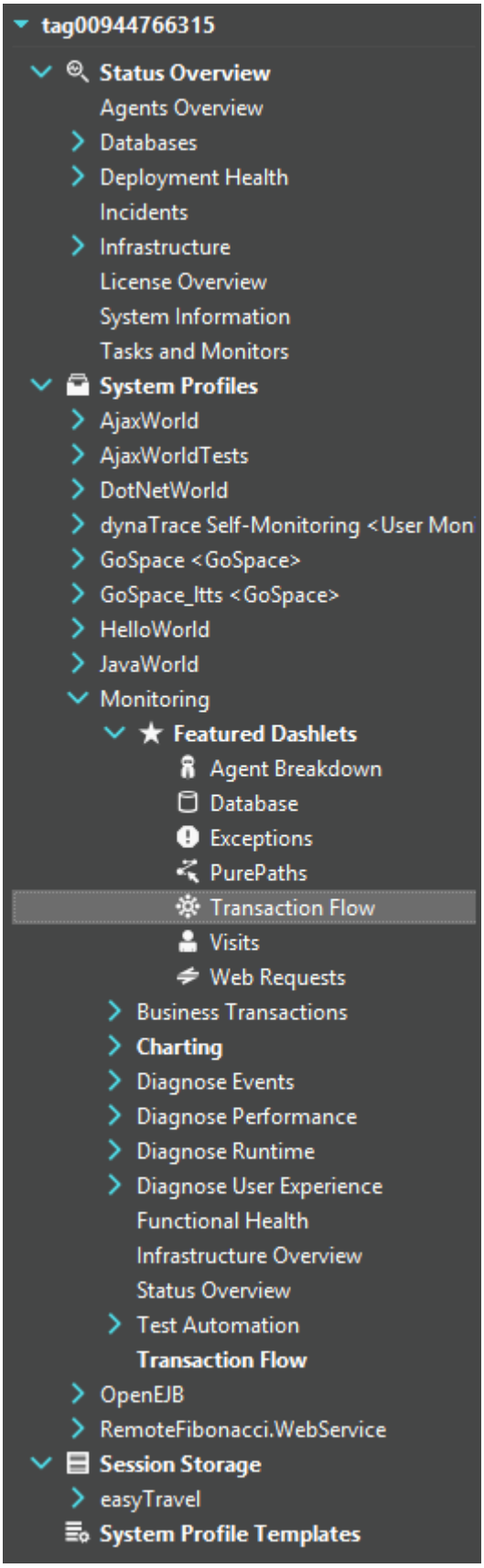
Watch and learn

Analyzing Transaction Flow in AppMon Client

To open the **Transaction Flow** dashlet, expand your System Profile in the Cockpit and double-click the **Transaction Flow** entry. An overview of your environment appears. This includes the Agent Groups from which you receive events, as seen above.



The **Transaction Flow** dashlet supports different visualization modes that provide different views and detail levels. The Topology View mode is the default mode. Use this to analyze the basic operation. Use the Transaction Response Time view to analyze transactions.



The dashlet header contains information about the current settings and data:

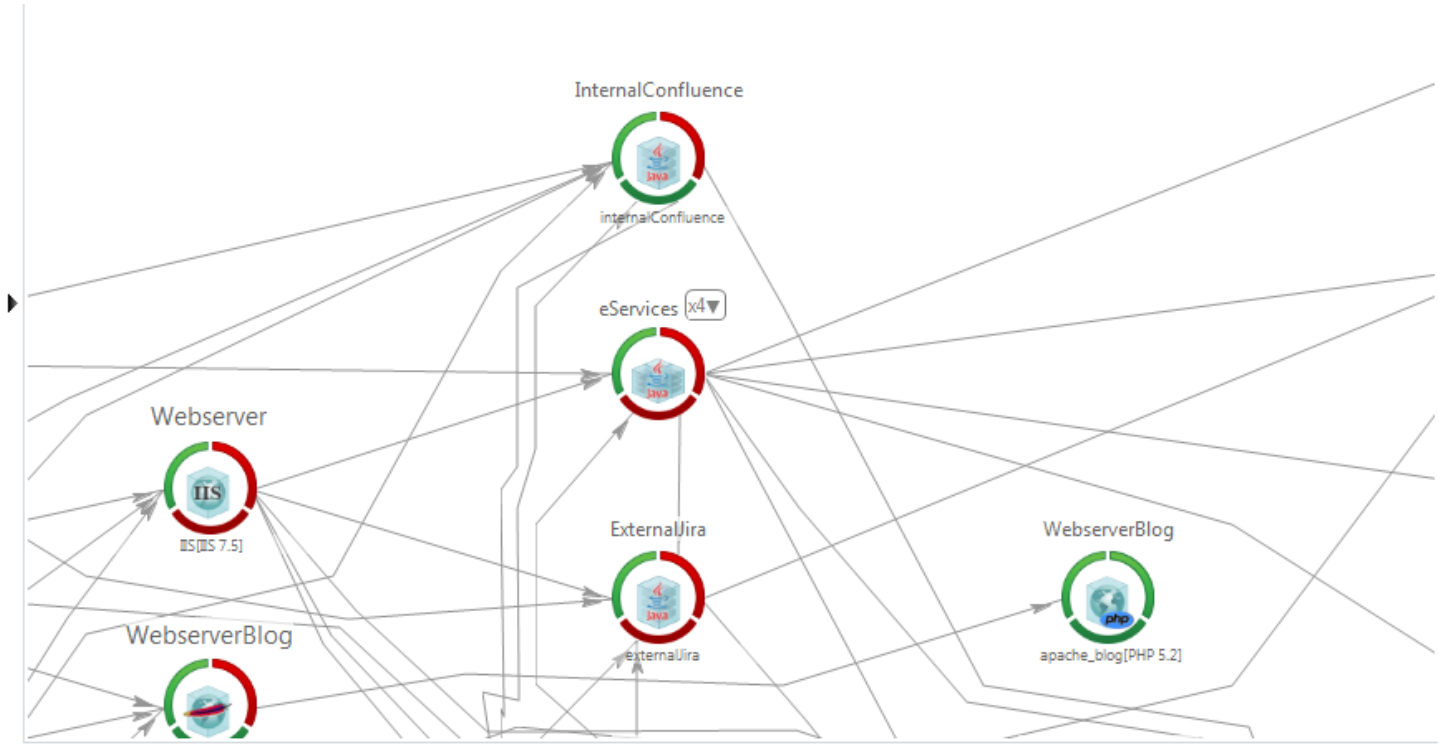


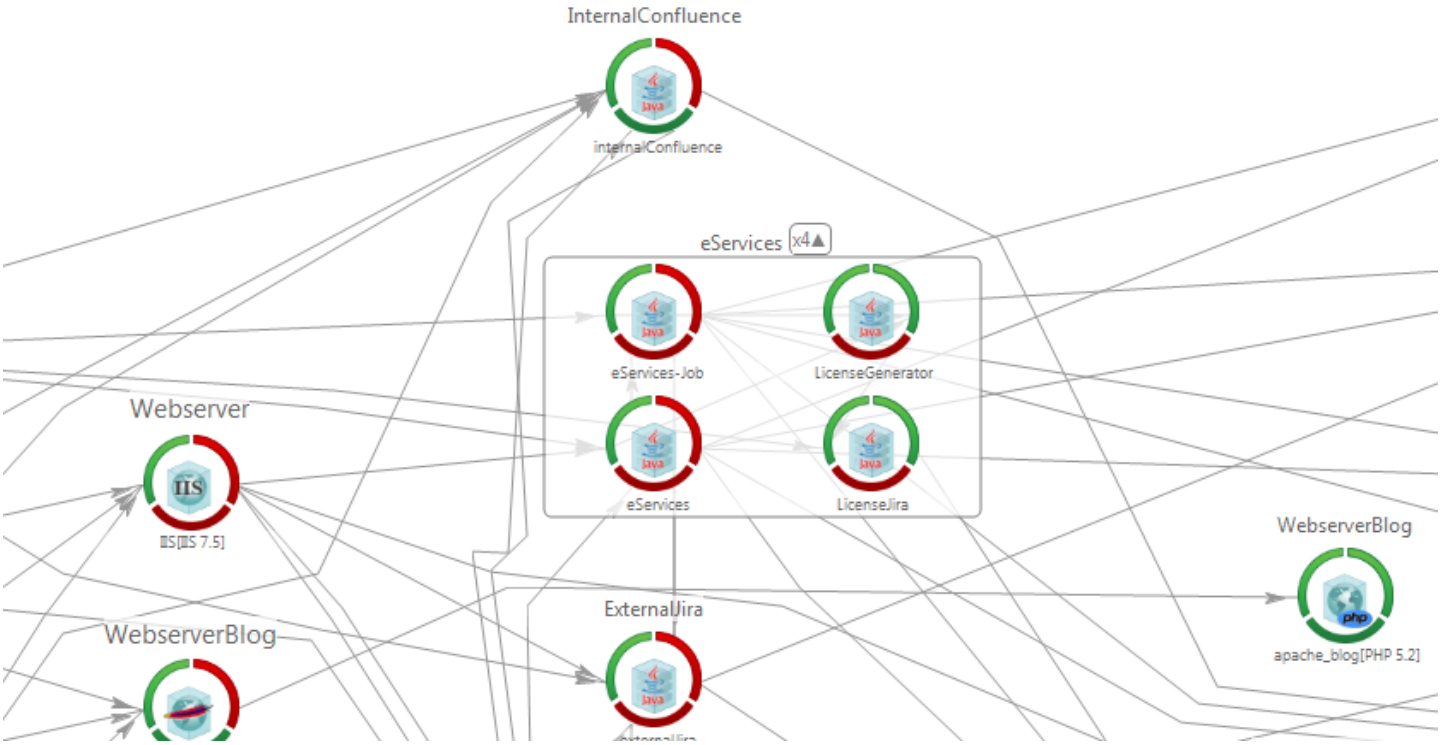
Visualization Mode:	Topology or Transaction Response Time view
Business Transaction Filter:	Displays data based on a certain business transaction
Total Transactions:	Number of transactions and average number of transactions per minute
Failed Transactions:	Number and percent of failed transactions in relation to the total number of transactions
Inter Tier Time/Transaction:	Average latency time for transactions in milliseconds and total transaction execution time percentage

The diagram consists of the following components:

- One or more entry points (generic, web request, synthetic request, browser, or user) AppMon:
- Collapsible grouping frames for defined Agent Groups
- One or more database pools
- Inter-tier connectors or communication channels * Blue - Incoming
  - Green: Outgoing calls of currently select Agents
  - Purple: A link shared between two selected Agents on multi-select

AppMon groups Agents by Agent Group. The System Profile specifies the Agent Group. If your system landscape consists of many Agents, the collapsed view only shows the Agent Groups, and not each single Agent. To switch between the collapsed and expanded views, use the context menu. To access the context menu, double-click an Agent Group or click the symbol in the upper right corner of the Agent Group. If an Agent Group contains only one Agent, the group is always expanded and you cannot collapse it. In order to prevent the Transaction Flow from exploding an upper limit of 100 Agents per Agent Group was introduced. Agent Groups that contain more than 100 Agents cannot be expanded.



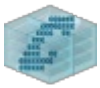


The Transaction Flow diagram layout is organized around Agent Groups and not, necessarily, physical hosts. You can use a single Agent Group, but we recommend that you configure your System Profiles with multiple Agent Groups that fit your landscape. This helps with management and configuration changes and creates meaningful diagrams.

Transaction Flow icons

Legend for Transaction Flow icons

Icons whose meanings may not be obvious:


CICS group


CICS Transaction Gateway (z/OS)


MariaDB


MariaDB group


Messaging


Messaging group


WebSphere Message Broker

Overlays whose meanings may not be obvious:


Amazon overlay


Azure overlay


Cassandra overlay


HBase overlay


MongoDB overlay


-Virtual Box overlay


-Xen overlay


Virtual machine overlay

Icons, grouped:


Entry point


-User


-Synthetic Agent


-Browser


-Web


Web server group


Web server


-Apache



-IIS


App group


-Browser



-.NET



-Java



-JavaScript


-Mobile native



-ADK


App



-.NET



-Java



-JavaScript


Browser app


-Firefox


-IE


Cassandra



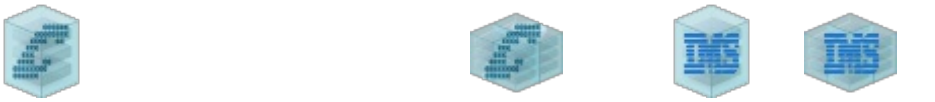
-PHP



DB group -DB2 group -Derby group - Informix group MariaDB group SAP MaxDB group - mySQL group - Oracle group - PostgreSQL group -SQL Serve group



DB -DB2 -Derby - Informix MariaDB SAP MaxDB - mySQL - Oracle - PostgreSQL -SQL Serve



CICS Transaction Gateway (z/OS) CICS group IMS IMS group



Web request Web request group 3rd Party 3rd Party group WebSphere Message Broker WebSphere Message Broker group



Cloud overlay Amazon EC Amazon EC group Azure Azure group SQL Azure SQL Azure group

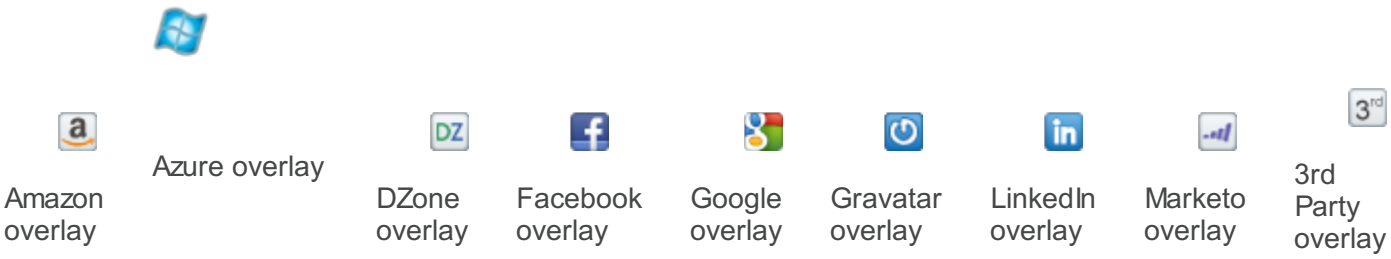
Overlays, grouped:



Virtual machine overlay -Hyper-V verlay -Virtual Box overlay -vmWare overlay -Xen overlay



Container overlay -Docker overlay -LXC overlay



View modes

Currently the Transaction Flow visualization supports two view modes:

- Topology view: Use this view mode for a general overview of your application and the flow of transactions. It shows the rate of calls and the protocols between tiers, as well as the inter-tier time.
- Transaction Response Time view: Use this view mode to analyze individual transaction types. It shows the response time per transaction, protocol contribution, and detailed call statistics per transaction or passing transaction.

Topology view

The Topology view identifies which application components communicate with each other, how many calls they make, the call rate, protocols, and the latency impact. This information is based on all the PurePaths in the current live-session, per default, and unfiltered.
The diagram shows the call rate on communication channels between Agents and the inter-tier latency time, if you enable it in the header.

To get detailed information for each component in the diagram, hover over the component and use the context menu. The tool tips contain a subset of the details available via the context menu and provide quick access to the most relevant data. Component values are bold face in the Transaction Flow visualization.

Transaction response time view

The Transaction Response Time view analyzes single transactions and their flow through the application landscape. Use it to trace a single type of transaction and identify which components contribute to the execution of that transaction type. Use this information to identify bottlenecks and resources used.
The diagram shows the number of transactions that pass through a communication channel, and the average calls per transaction on that channel. It also shows the inter-tier latency time, if you enable this.

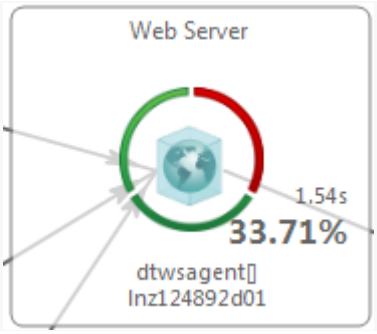
Zooming

You can use zoom in reduced and normal view mode via the mouse wheel and the dashlet actions.

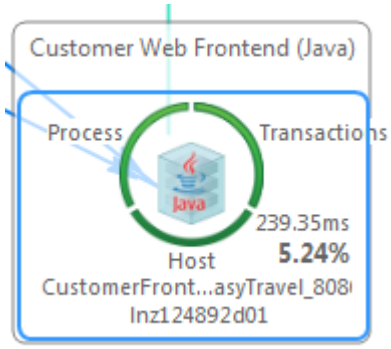
Transaction Flow zooming actions



Differences in zooming view modes

Topic	Normal view mode	Reduced view mode
Contents		
Hotspot	The Normal view mode visualizes hotspots via different font sizes to the response time contribution.	The low-resolution mode visualizes hotspots via different font sizes to the Agent Group name.
Visualized Data	Shows detailed information about Agent, Agent Groups, and their connection.	Does not show detailed information about the Agent, the Agent Group, and their connection.

Health visualization



Every Agent and Agent Group has its own health status. AppMon visualizes this via a segmented health circle that contains the following:

- Process

- Host and transaction health
- Three states:
  - Red: The host or process is unhealthy, or transactions are failing.
  - Green: Everything is okay.
  - Grey: No transactions are available, or the host or process is unknown.

Values visible in Transaction Flow

This section discusses the values in the Transaction Flow. You can't see all the detail values in both the Topology and Transaction Response Time views. The values in the Agent or communication link that is currently selected are in bold face.

The following terms are found throughout the Transaction Response Time view:

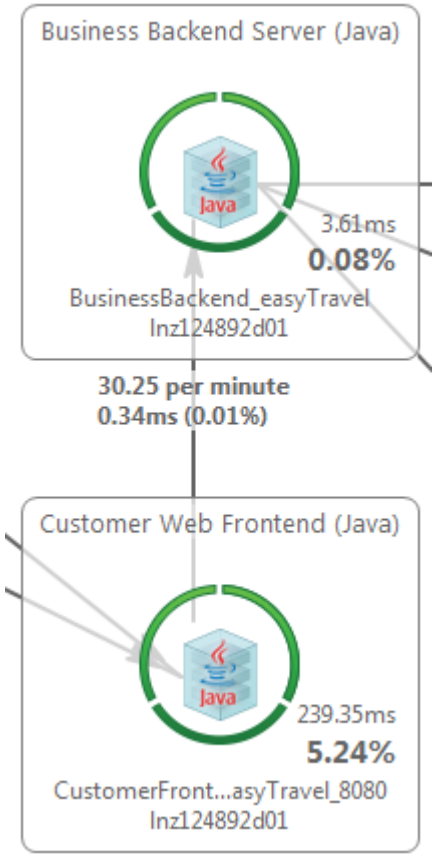
- **Per Transaction**: The value calculated is based on the overall transactions to which the dashlet filter applies. The dashlet header displays the number of total transactions.
- **Per Passing Transaction**: The value calculated is based on the transactions between two Agents or Agent Groups.

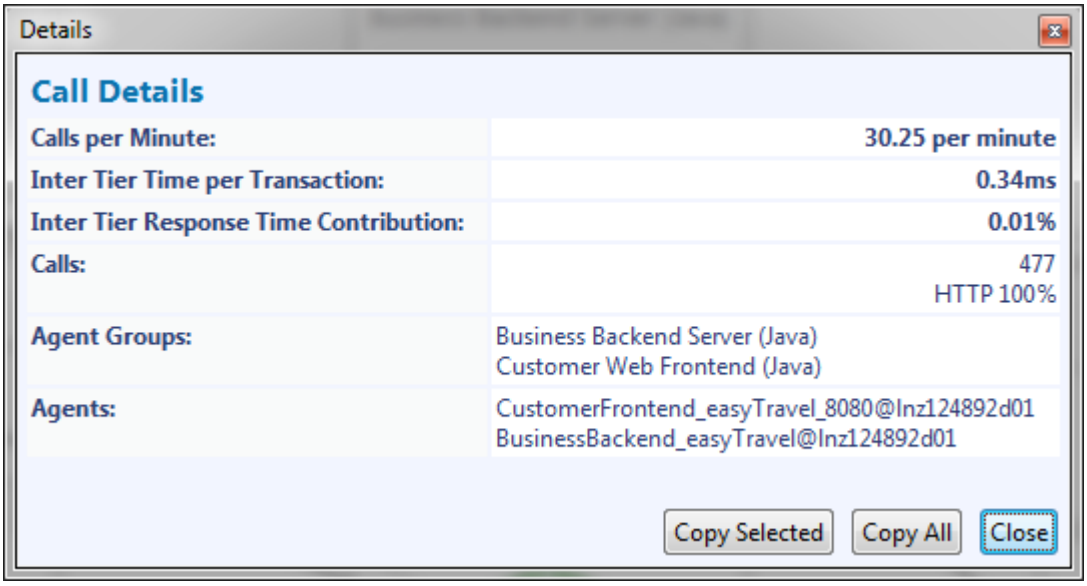
Inter-tier communication channels

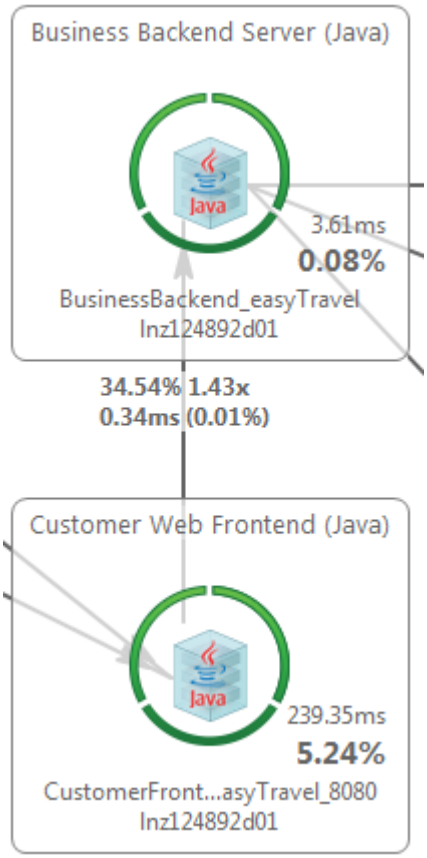
Inter-tier communication is the communication between two AppMon Agents, typically the different tiers of your application. Inter-tier time is the time (network latency, non-instrumented tiers, and queuing) spent between two Agents: One Agent's last PurePath measure point and another Agent's first measure point.

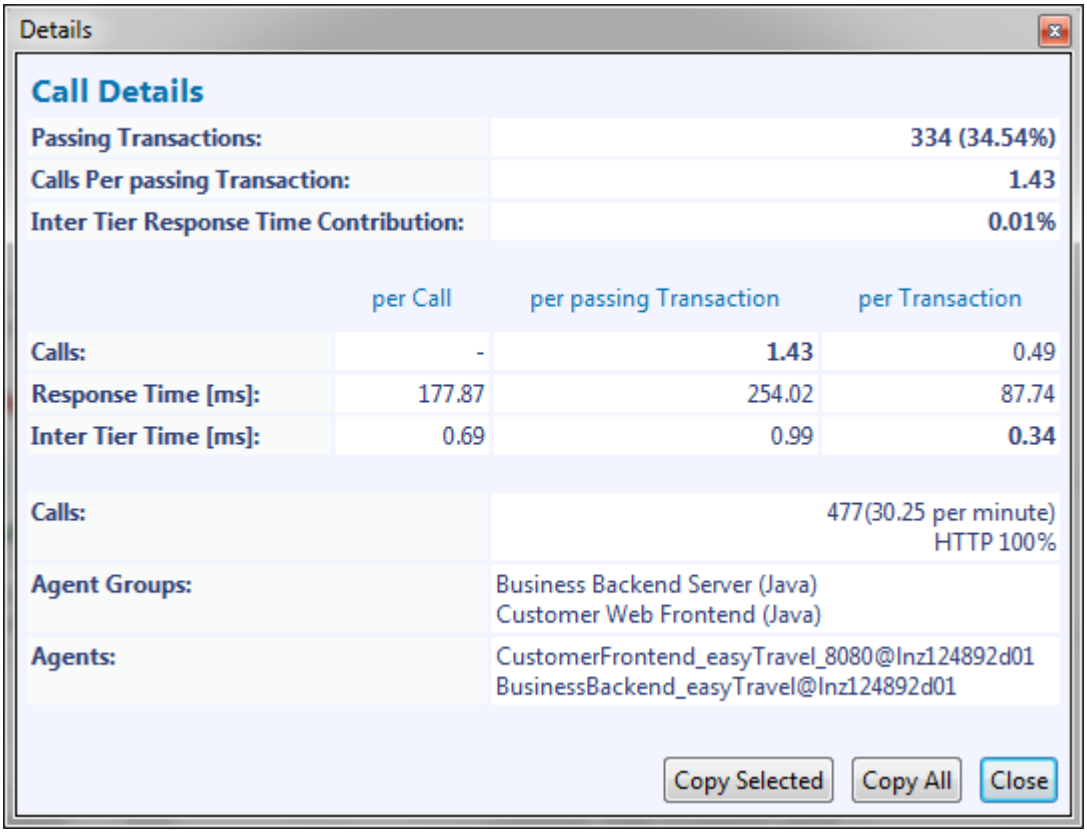
Call Details	Description
Passing Transactions	Number and percentage of transactions that passed through the communication channel in relation to all transactions.
Calls per Passing Transaction	Average number of calls invoked per passing transaction.
Inter-Tier Response Time Contribution	Inter-tier time impact on the overall transaction response time during the specified time frame.
Calls	Average number of calls per passing transaction, per overall transaction.
Response Time	Average response time of a communication channel's end point per call, per passing transaction, per overall transaction.

Inter-Tier Time	Average inter-tier time per call, per passing transaction, per overall transaction.
Calls	Total number of calls and call per minute, sent over a communication channel, and the percentage of different protocol contributions.
Agent Groups	Agent Groups involved in this communication.
Agents	Agents involved in this communication.




Details
Call Details
Calls per Minute:
30.25 per minute
Inter Tier Time per Transaction:
0.34ms
Inter Tier Response Time Contribution:
0.01%
Calls:
477
HTTP 100%
Agent Groups:
Business Backend Server (Java)
Customer Web Frontend (Java)
Agents:
CustomerFrontend_easyTravel_8080@lnz124892d01
BusinessBackend_easyTravel@lnz124892d01
Copy Selected
Copy All
Close

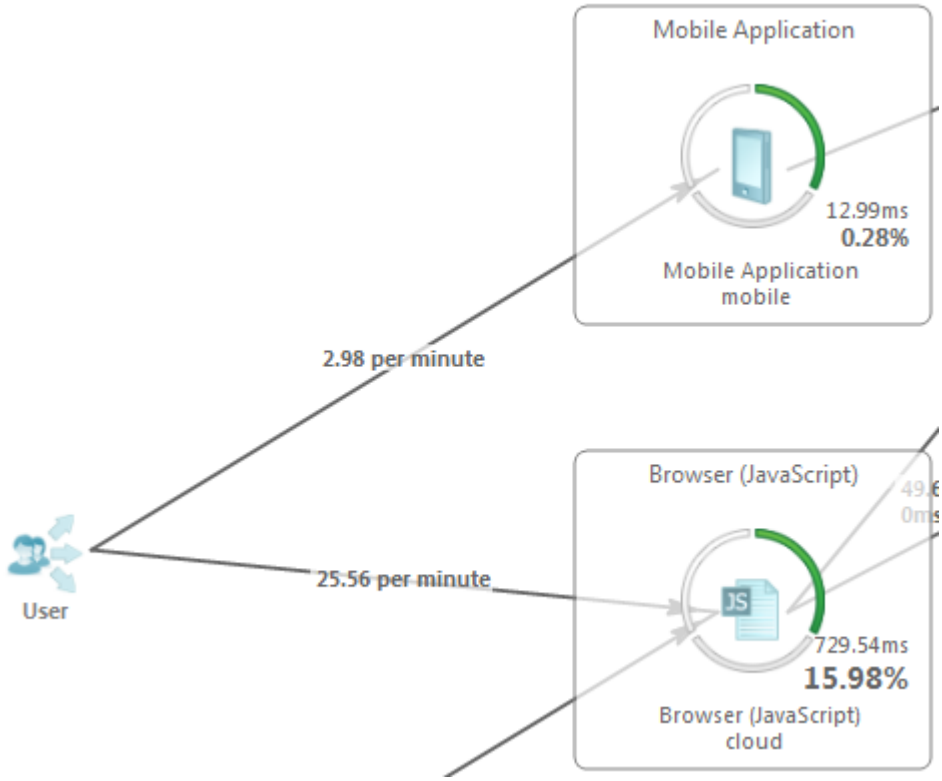

Business Backend Server (Java)
Java
3.61ms
0.08%
BusinessBackend_easyTravel
lnz124892d01
34.54% 1.43x
0.34ms (0.01%)
Customer Web Frontend (Java)
Java
239.35ms
5.24%
CustomerFront...asyTravel_8080
lnz124892d01

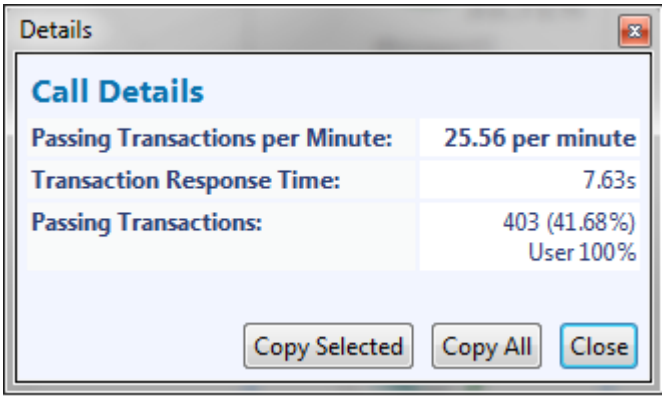


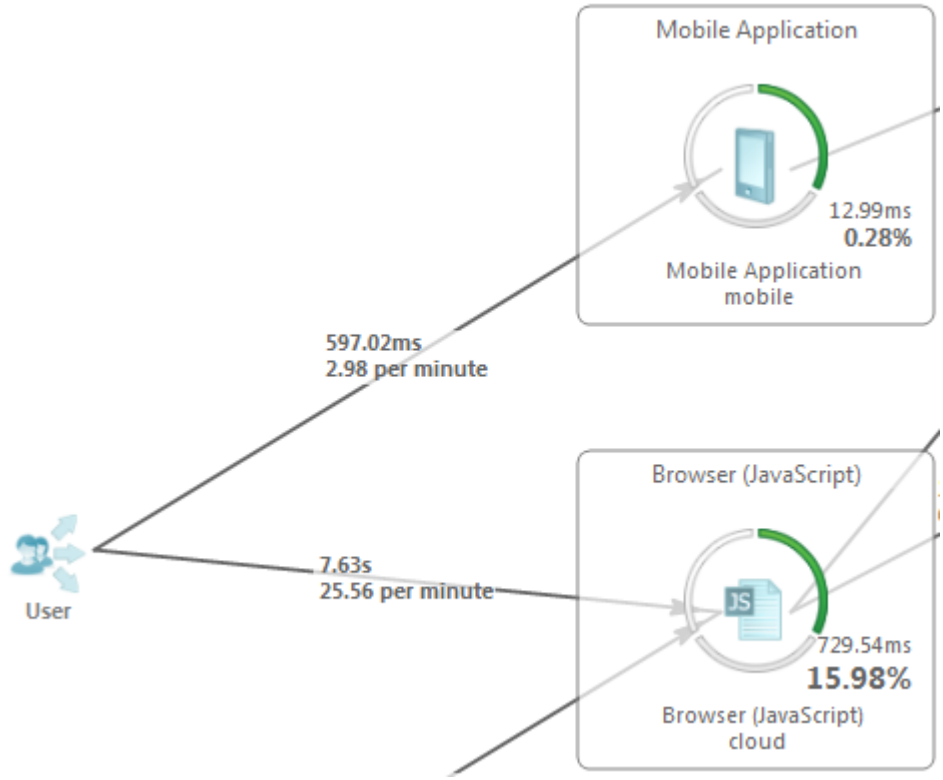
Entry communication channels

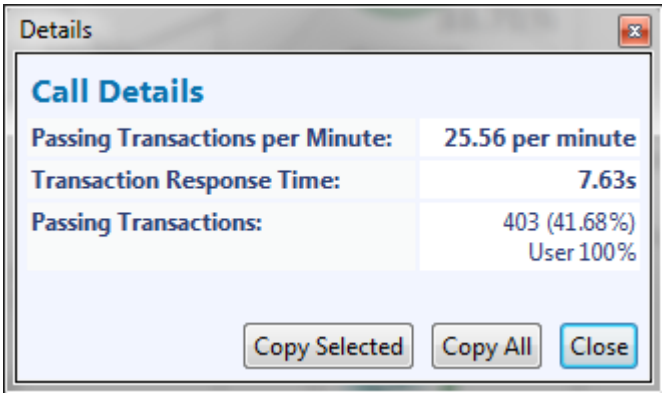
Entry communication channels typically mark a PurePath entry point.

Call Details	Description
Passing Transactions/min	Number of transactions invoked on this entry point per minute during the specified time frame.
Transaction Response Time	Average transaction response time or average server response time, if browser diagnostics is not used.
Passing Transactions	Total number of invoked transactions during the specified time frame or in the live session.


Mobile Application
12.99ms
0.28%
Mobile Application
mobile
2.98 per minute
Browser (JavaScript)
JS
729.54ms
15.98%
Browser (JavaScript)
cloud
25.56 per minute
User


Details
Call Details
Passing Transactions per Minute:
25.56 per minute
Transaction Response Time:
7.63s
Passing Transactions:
403 (41.68%)
User 100%
Copy Selected
Copy All
Close



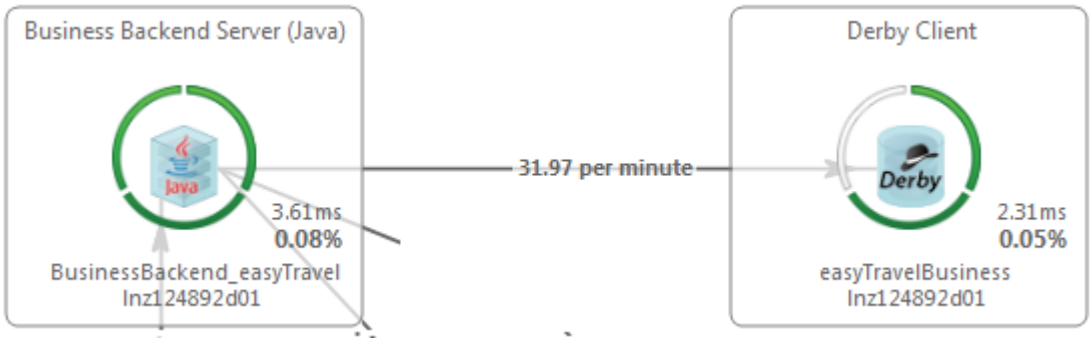


Database communication channels

Database communication channels represent database calls made by your applications.

Database Call Details	Description
Executions / min	Number of executions per minute.
Acquisition Time / Transaction	Average time per transaction for a connection acquisition.
Passing Transactions	Number of transactions that pass into the database and the percentage in relation to all transactions.

Executions per Passing Transaction	Average number of executions per transactions for this database.
Executions	Number of executed database queries.
Response Time	Average time for connection acquisition and query execution.
Acquisition Time	Time for connection acquisition
Executions	Total amount of executions and the percentage of different protocol contributions.
Instances	Database name / SID and database host
Type	Database vendor / type
Agent Groups	Agent Groups calling the database
Agents	Agents calling the database

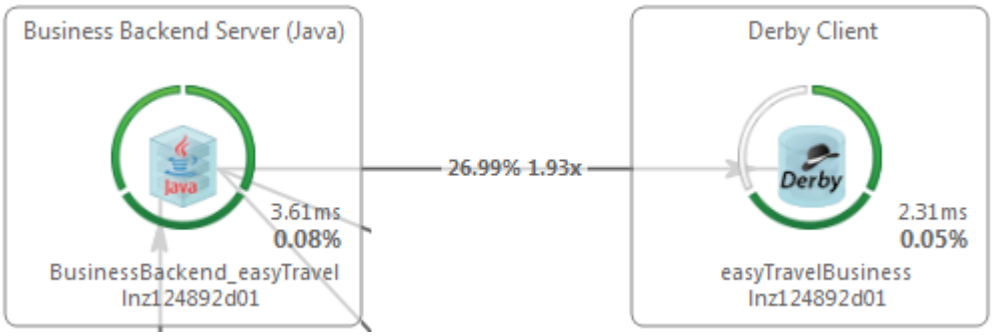


Details

Database Call Details

Executions per Minute:	**31.97 per minute**
Acquisition Time per Transaction:	0.03ms
Executions:	504 JDBC 100%
Instances:	easyTravelBusiness@lnz124892d01
Type:	Derby Client
Agent Groups:	Business Backend Server (Java) Derby Client
Agents:	easyTravelBusiness@lnz124892d01 BusinessBackend_easyTravel@lnz124892d01

Copy Selected Copy All Close



Details

Database Call Details

Passing Transactions:	**261 (26.99%)**
Executions per passing Transaction:	**1.93**

	per Call	per passing Transaction	per Transaction
Executions:	-	**1.93**	0.52
Response Time [ms]:	2.24	4.33	1.17
Acquisition Time [ms]:	0.07	0.13	0.03

Executions:	504(31.97 per minute) JDBC 100%
Instances:	easyTravelBusiness@lnz124892d01
Type:	Derby Client
Agent Groups:	Business Backend Server (Java) Derby Client
Agents:	easyTravelBusiness@lnz124892d01 BusinessBackend_easyTravel@lnz124892d01

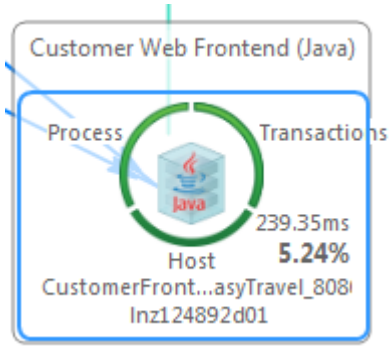
Copy Selected Copy All Close

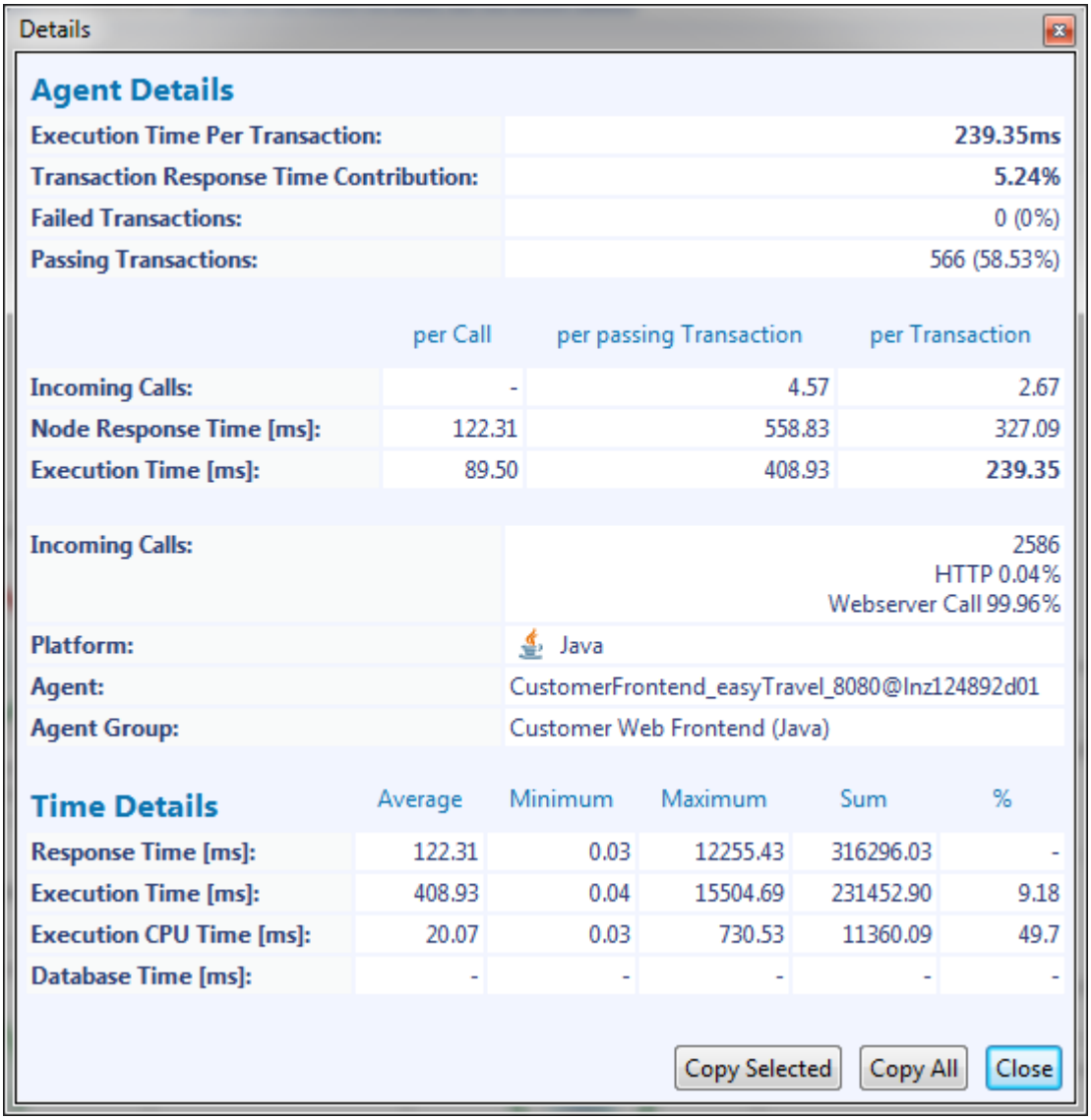
Agents

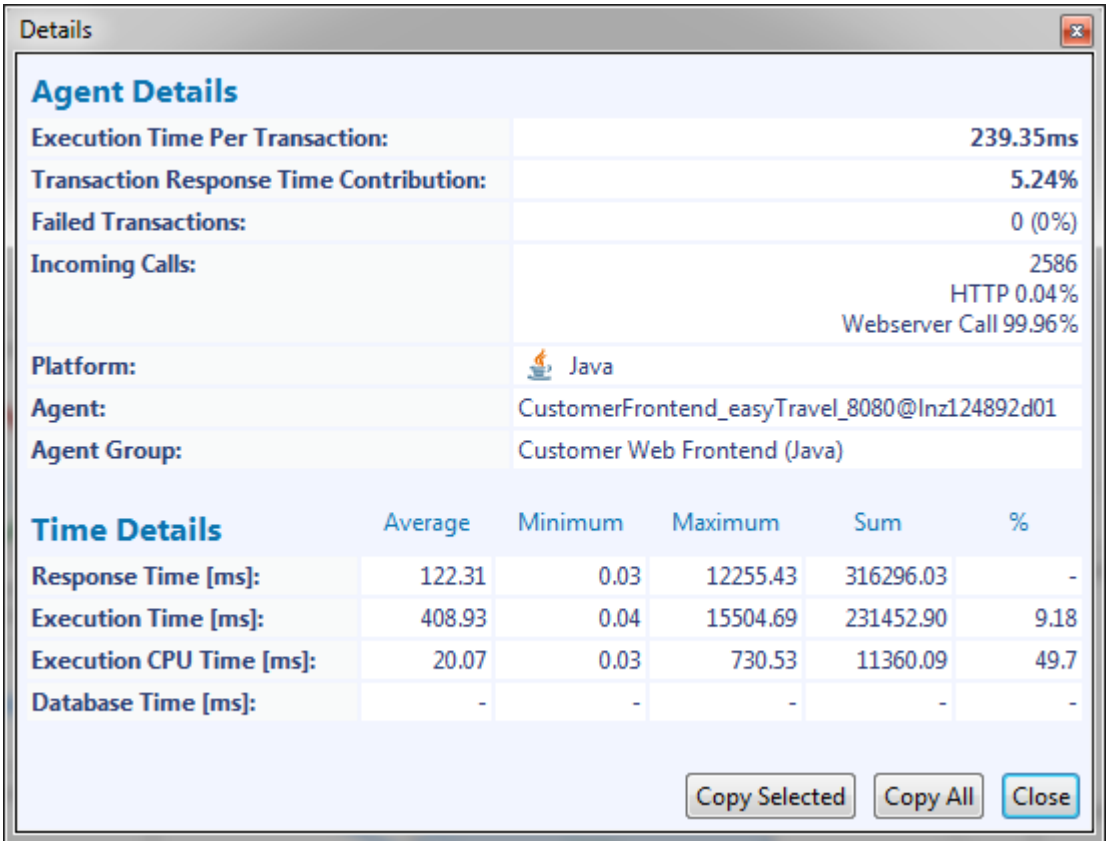
Each connected Agent that contributes data to the current session, time frame, or filter, has an icon that contains the Agent name, host name, and technology. AppMon groups Agents by grouping frames, according to the System Profile configuration. The grouping frames details are similar to the Agent, but show aggregated values.

Agent Details	Description
Execution Time / Transaction	Average execution time spent on this Agent or Agent Group per overall transaction.
Transaction Response Time Contribution	Percentage of time this Agent contributes to the overall average response time.
Failed Transactions	Number of failed transactions and the percentage in relation to the total number of transactions.
Passing Transactions	Number of transactions that passed this Agent.
Incoming Calls	The average number of calls per passing transaction and per overall transaction.
Node Response Time	Average response time of this Agent per call, per passing transaction, and per overall transaction.
Execution Time	Average execution time on this Agent per call, per passing transaction, and per overall transaction.
Incoming Calls	The total amount of calls made on this Agent and the percentage of different protocol contributions.
Platform	Technology on which the Agent runs.
Agent	Full Agent name (all Agents in this group on Agent Group details).
Agent Group	Agent Group to which the Agent belongs (name of Agent Group on Agent Group details).
Time Details	
Response Time	Overview of the min, max, average, sum, and percentage of the response time for all calls of an Agent or Agent Group.

Execution Time	Overview of the min, max, average, sum, and percentage of the execution time for all calls of an Agent or Agent Group.
Execution CPU Time	Overview of the min, max, average, sum, and percentage of the CPU (execution) time for all calls of an Agent or Agent Group.
Database Time	Overview of the min, max, average, sum, and percentage for the database time of all calls of an Agent or Agent Group.







Database

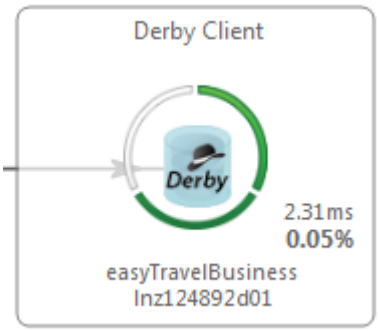
Executed database statements display in one single aggregated view via the database component. This view shows the number of executed database calls and their execution time.

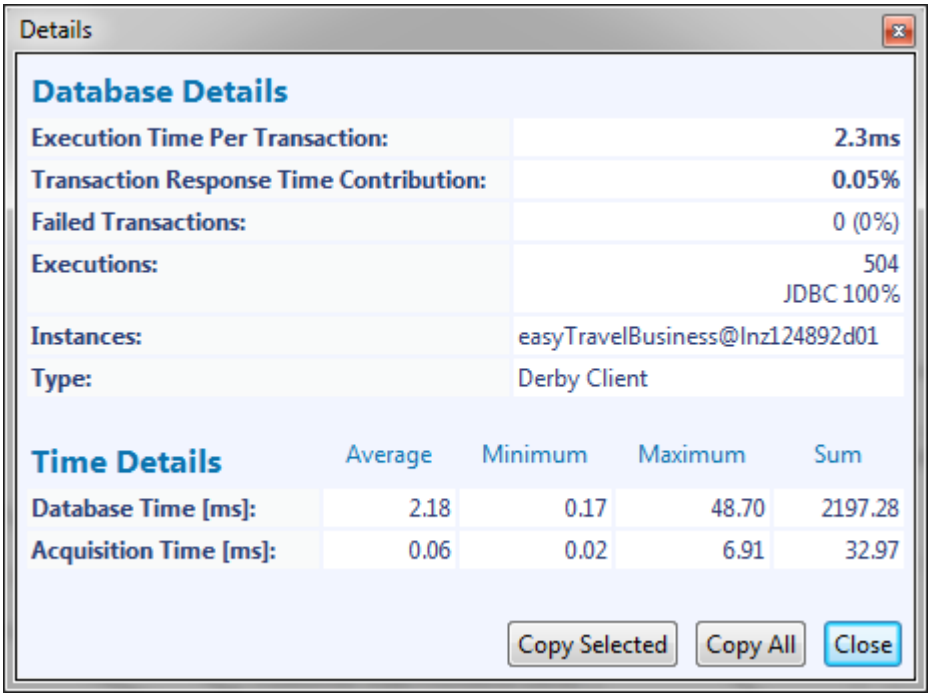
Database Details	Description
Response Time / Transaction	Average time per transaction for connection acquisition and query execution.
Transaction Response Time Contribution	Percentage of time this database contributes to the overall average response time.
Failed Transactions	Number of failed transactions and their percentage in relation to the total number of transactions.
Passing Transactions	Number of transactions that passed this Agent.
Executions	Number of executed database queries.
Database Response Time	Average time of connection acquisition and query execution.

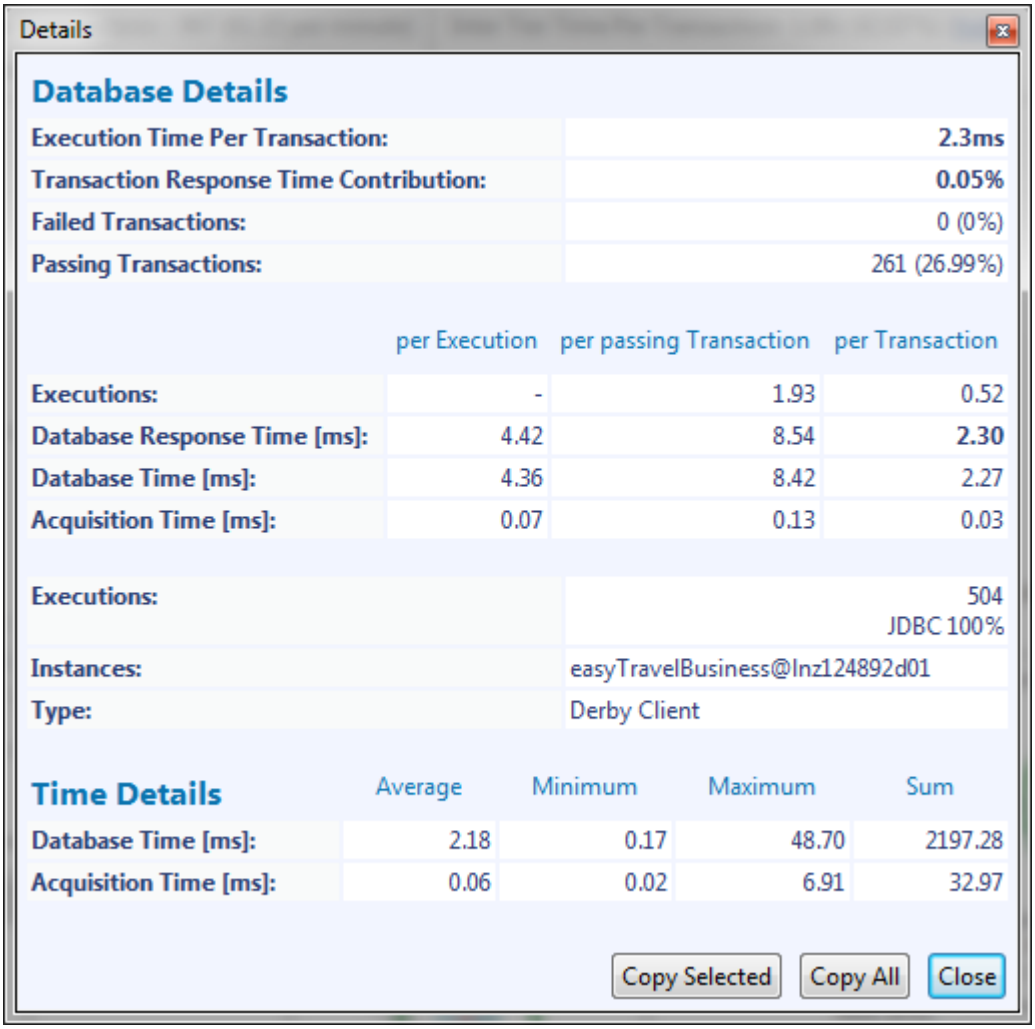
Database Time	Time of query execution.
Acquisition Time	Time of connection acquisition.
Executions	Number of database queries executed.
Instance	Database name or SID and database host.
Type	Database vendor or type.

Time Details

Database Time	Overview of the min, max, average, and sum of database execution time.
Acquisition Time	Overview of the min, max, average, and sum of database connection acquisition time.

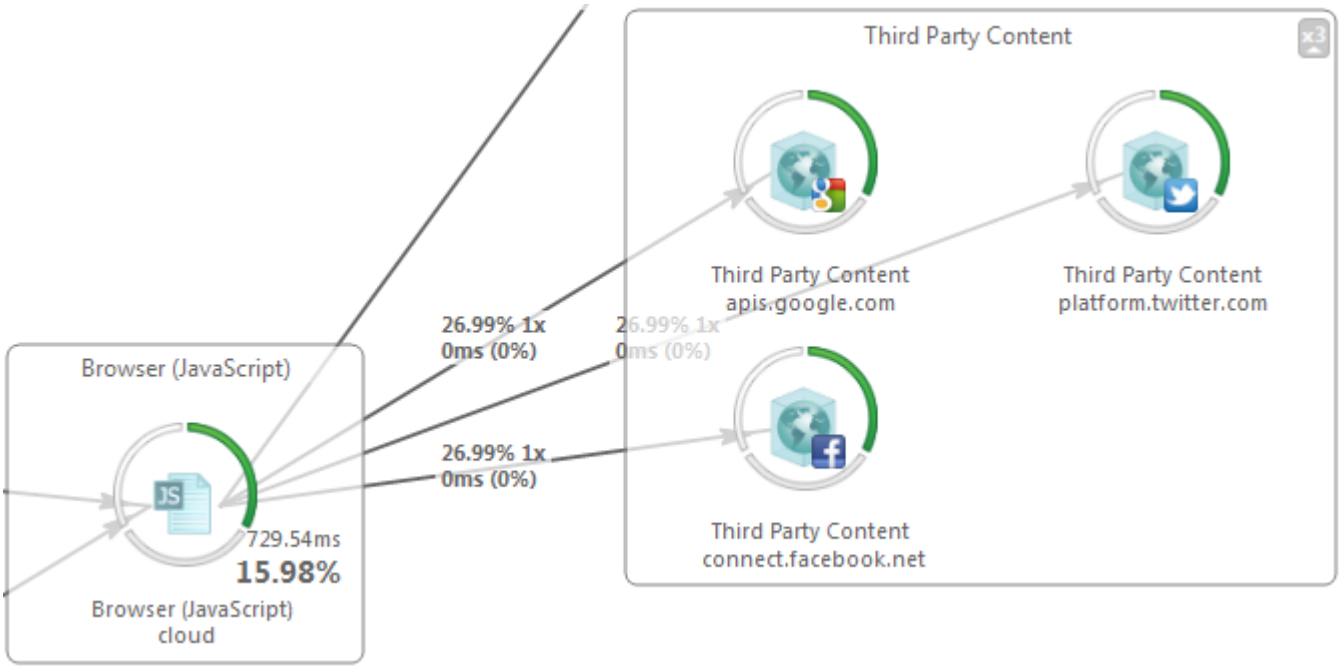






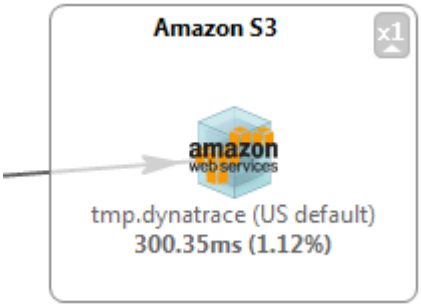
Third party calls

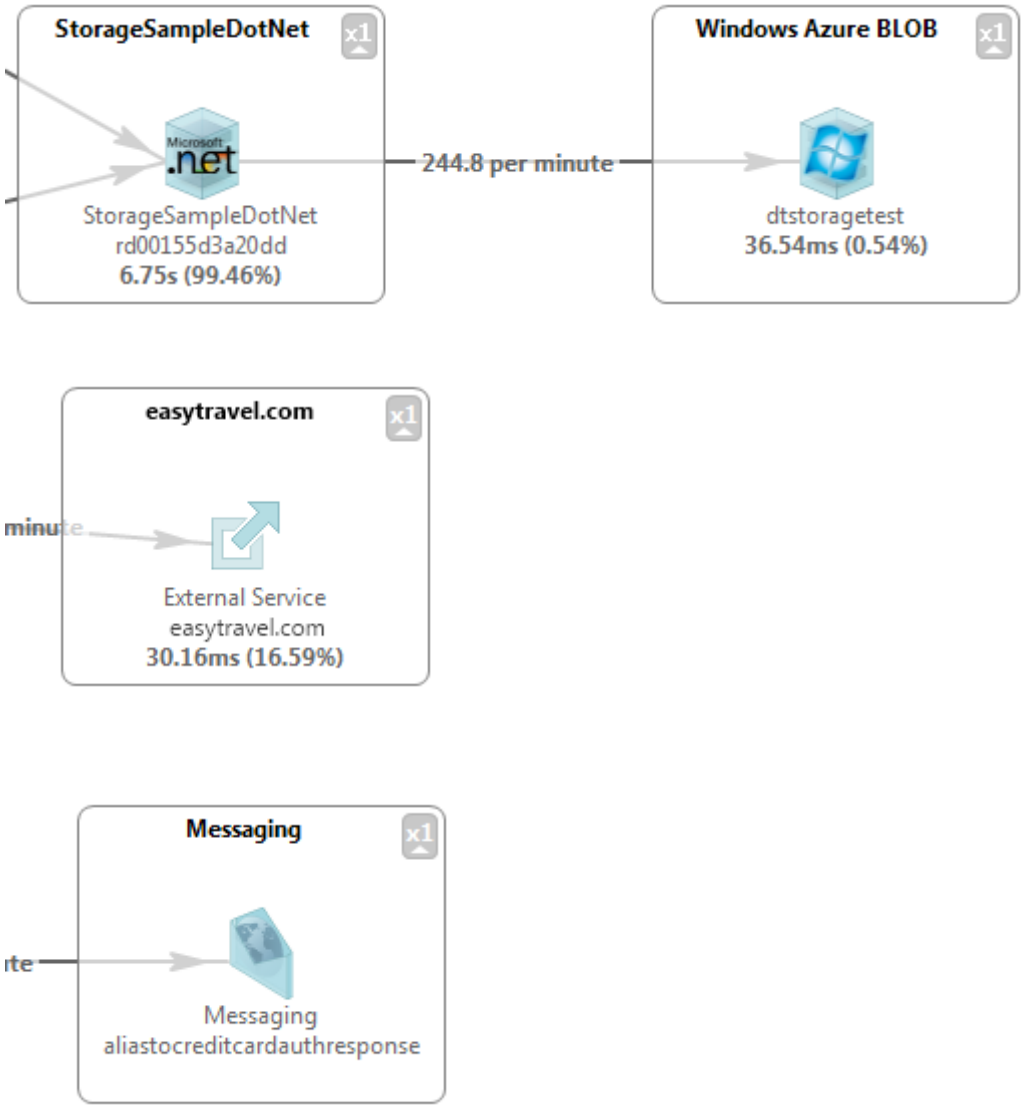
In installations that use User Experience Monitoring, AppMon visualizes user actions with third party content via Third Party nodes. The third party calls value is the average response time per call. AppMon does not apply these times to overall response time calculation, because most of the third party calls run asynchronously.



External services

AppMon detects and visualizes server side transactions, like external service calls, in the Transaction Flow dashlet. Amazon Web services, Microsoft Azure Cloud hosted services and other non-instrumented processes are visualized to the right of the Transaction Flow. External messaging calls are part of the external call detection and are also visualized.

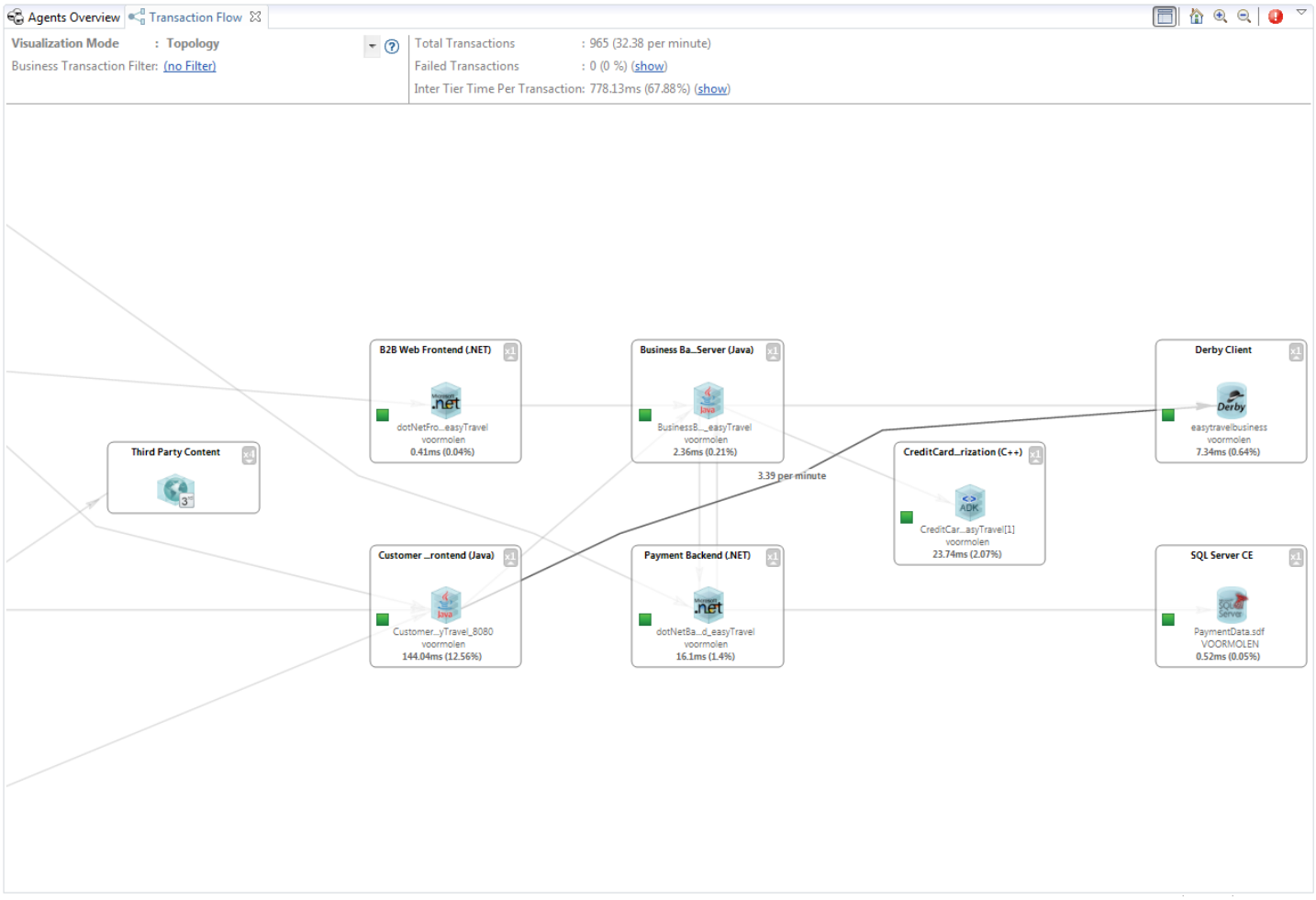


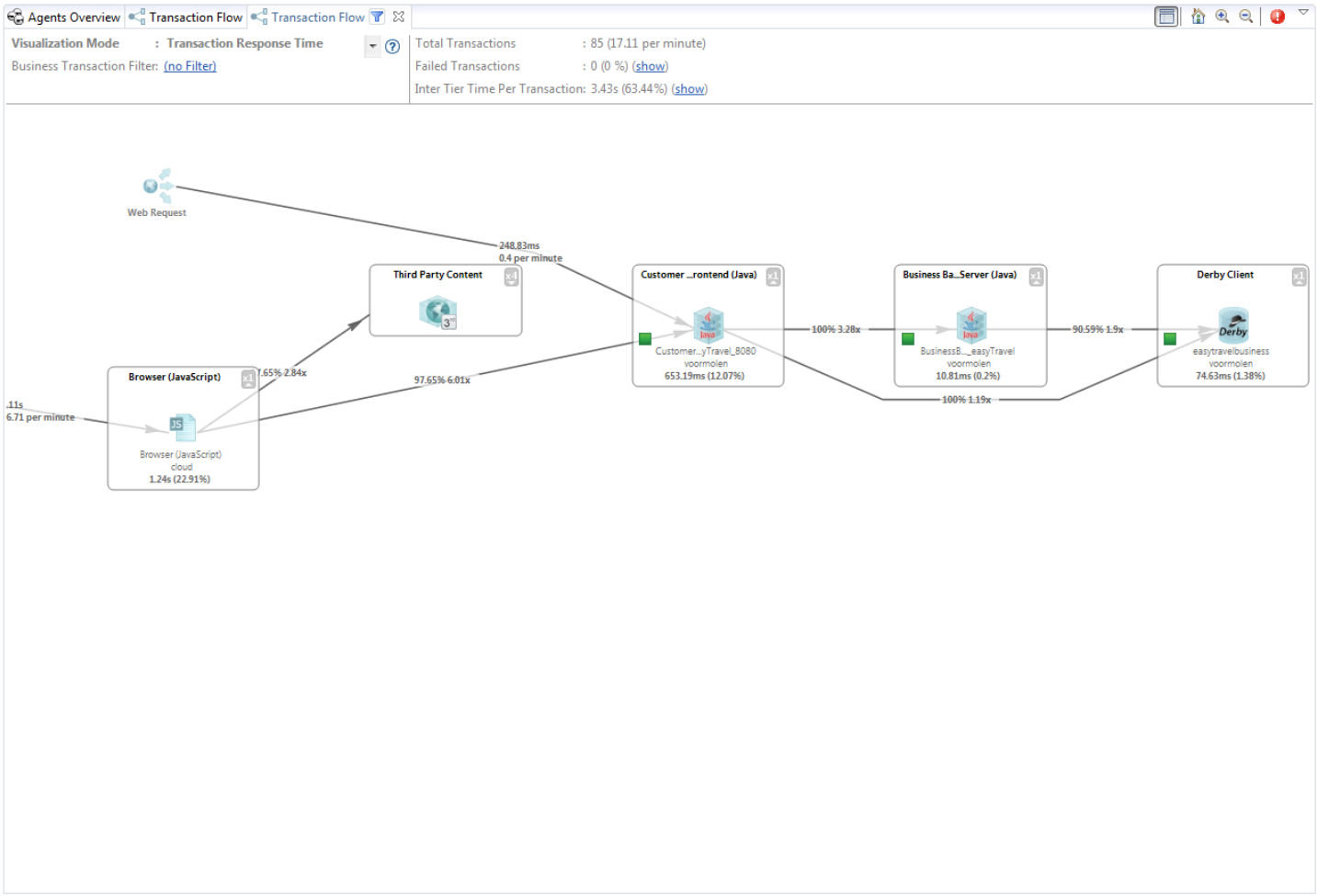


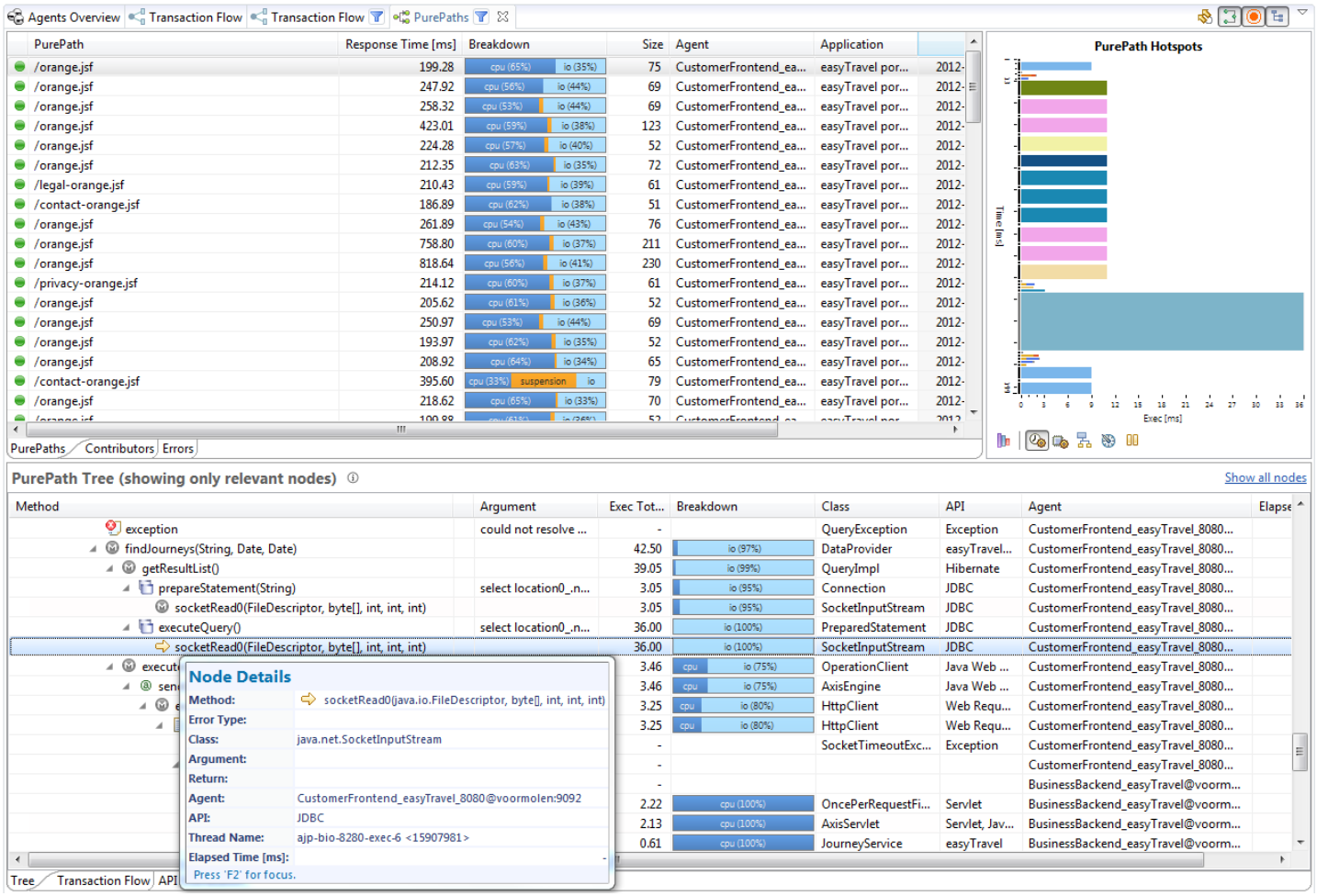
AppMon groups external service calls by top-level domain names (or cloud service types, if applicable) and groups of such services are named according to their hosts. External service call detection is performed via the .NET and Java Web Request Sensors.

Work with Transaction Flow

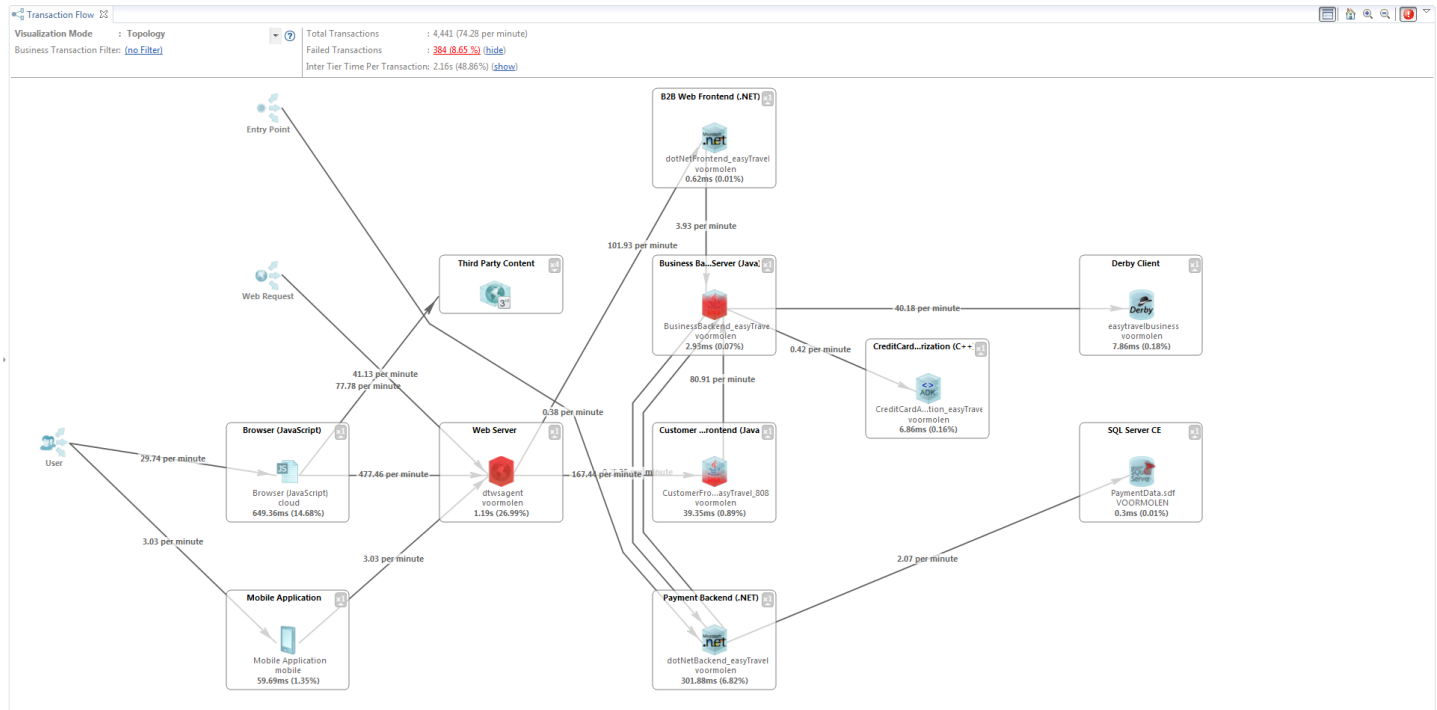
Hover over Agents, Agent Groups, or communication links, to see drilldowns for root cause analysis. The example below displays an architectural problem: A call directly from the front end to the database. Use the drilldown from the communication link between the front end and database to go to the two affected transactions and analyze them.


Agents Overview
Transaction Flow
Visualization Mode : Topology
Business Transaction Filter: (no Filter)
Total Transactions : 965 (32.38 per minute)
Failed Transactions : 0 (0 %) (show)
Inter Tier Time Per Transaction: 778.13ms (67.88%) (show)
Third Party Content
B2B Web Frontend (.NET)
dotNetFro...easyTravel
voormolen
0.41ms (0.04%)
Business Ba...Server (Java)
BusinessB..._easyTravel
voormolen
2.36ms (0.21%)
Derby Client
easytravelbusiness
voormolen
7.34ms (0.64%)
3.39 per minute
CreditCard...rization (C++)
CreditCar...asyTravel[1]
voormolen
23.74ms (2.07%)
Customer ...rontend (Java)
Customer...yTravel_8080
voormolen
144.04ms (12.56%)
Payment Backend (.NET)
dotNetBa...d_easyTravel
voormolen
16.1ms (1.4%)
SQL Server CE
PaymentData.sdf
VOORMOLEN
0.52ms (0.05%)


Agents Overview
Transaction Flow
Transaction Flow
Visualization Mode : Transaction Response Time
Business Transaction Filter: (no Filter)
Total Transactions : 85 (17.11 per minute)
Failed Transactions : 0 (0 %) (show)
Inter Tier Time Per Transaction: 3.43s (63.44%) (show)
Web Request
248.83ms
0.4 per minute
Third Party Content
Customer ...rontend (Java)
Business Ba...Server (Java)
Derby Client
100% 3.28x
90.59% 1.9x
100% 1.19x
97.65% 6.01x
7.65% 2.84x
.11s
6.71 per minute
Browser (JavaScript)
Browser (JavaScript)
cloud
1.24s (22.91%)
Customer...yTravel_8080
voormolen
653.19ms (12.07%)
BusinessB..._easyTravel
voormolen
10.81ms (0.2%)
easytravelbusiness
voormolen
74.63ms (1.38%)



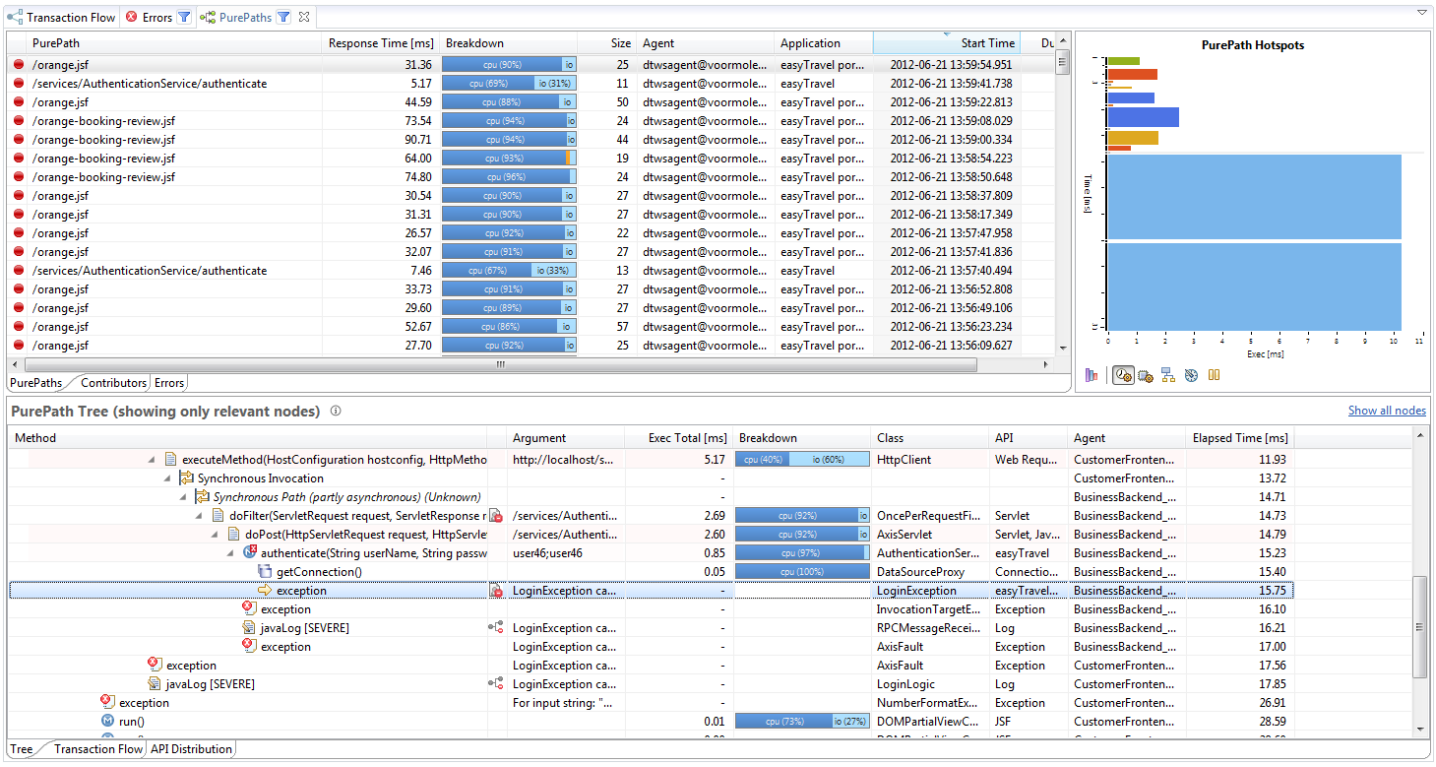
The Transaction Flow legend shows that 12 failed transactions and highlights the Agents and nodes in red. The Agent's details show the number of failed transactions and the direct drilldown to errors on this node. The **Errors** dashlet shows errors and drilldowns for analysis.



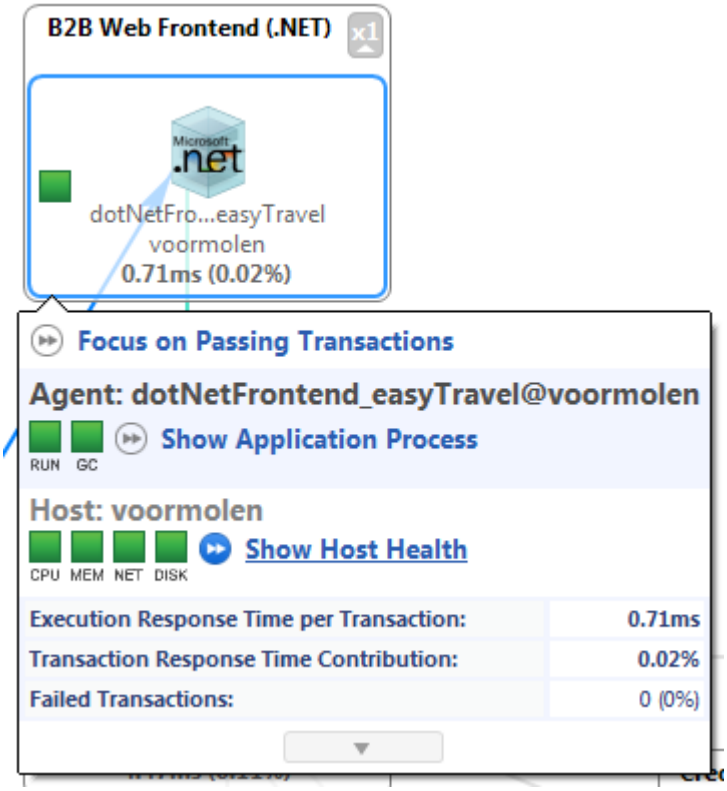
Transaction Flow | Errors

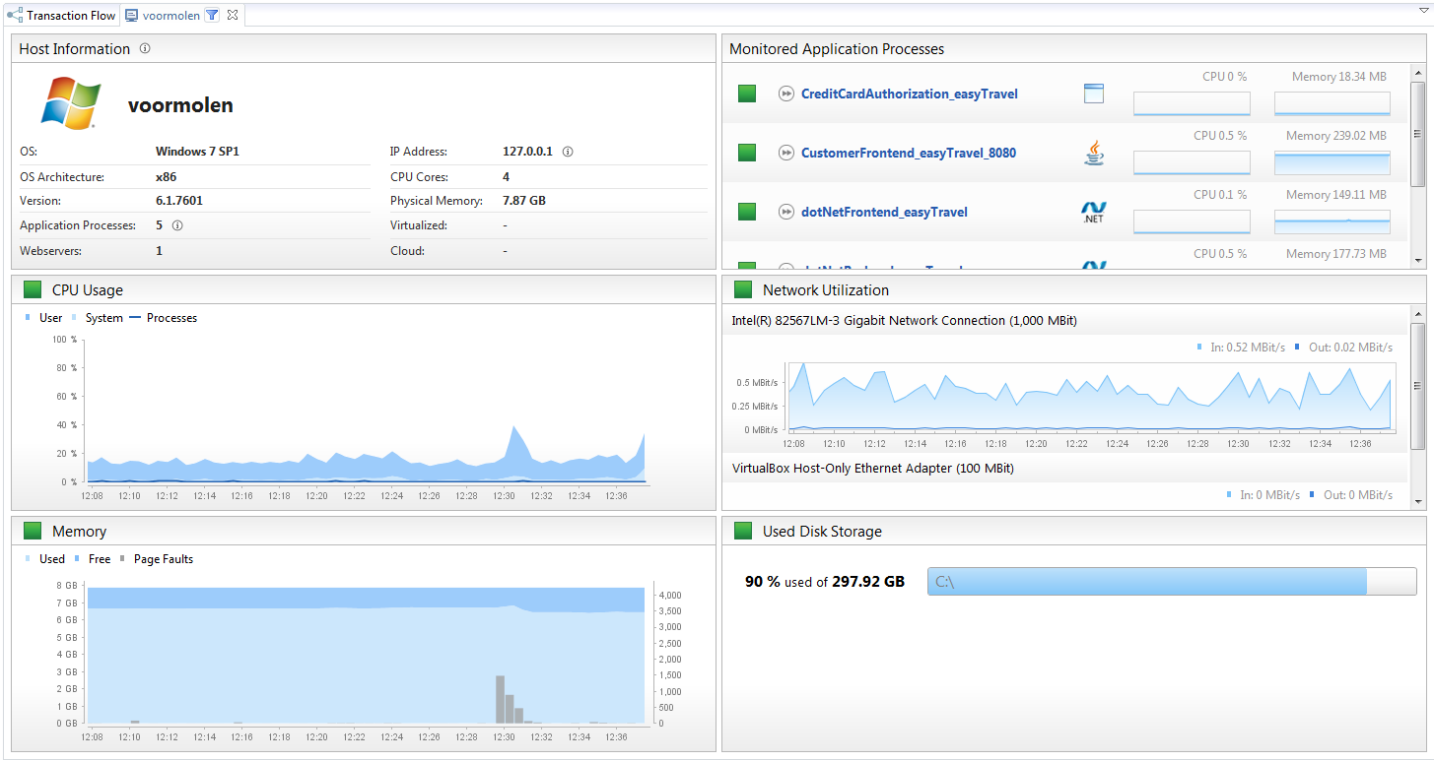
Error Rule	Transactions	Error count	Failing Transactions	Failing Page Actions
Exceptions	298	0	0	298
easyTravel Exceptions	298	-	-	298
Database Exceptions	0	0	-	-
Unhandled Exceptions	0	-	0	-
Unhandled Inter-Tier Exceptions	0	0	-	-
Log Messages	298	298	0	0
Important Loggings	298	298	-	-
HTTP response codes	298	0	0	298
Page Action fails on 500 HTTP Response	298	-	-	298
HTTP 4xx Response on Transaction Entry	0	-	0	-
HTTP 4xx Response (internal)	0	0	-	-
HTTP 5xx Response on Transaction Entry	0	-	0	-
HTTP 4xx Inter-Tier Response	0	0	-	-
HTTP 5xx Response (internal)	0	0	-	-
HTTP 5xx Response within Transaction	0	0	-	-
Broken Incoming Links	0	0	-	-
CSS or JS Resource Not Found	0	-	-	0
Error Page Redirect	0	0	-	-
HTTP 5xx Inter-Tier Response	0	0	-	-
Important HTTP 4xx Response on Transac	0	-	0	-
Important HTTP 4xx Response within Trai	0	0	-	-
Browser Errors	0	0	0	0
JavaScript SyntaxError	0	0	-	-

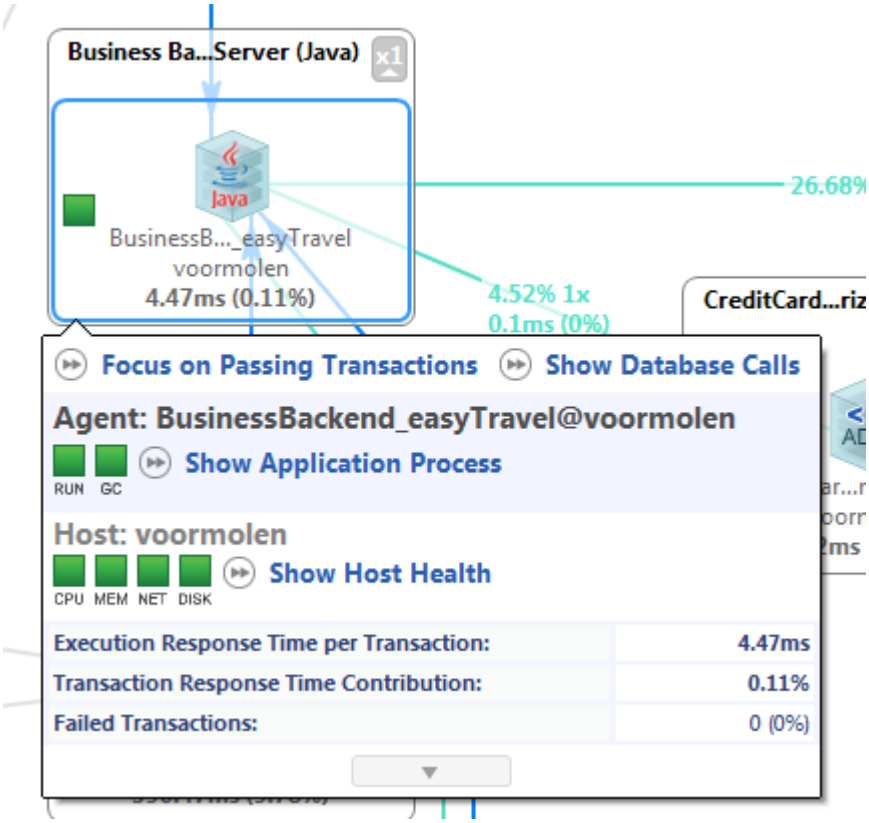
Error Rule	Transactions	Count	Exception Class	Message
easyTravel Exceptions	298	298	LoginException	LoginException caused by LoginProblems ...



Drill down to monitor the health of your applications or to show which database calls are made from a selected node.






Business Ba...Server (Java)
x1
Java
BusinessB..._easyTravel
voormolen
4.47ms (0.11%)
26.68%
4.52% 1x
0.1ms (0%)
CreditCard...riz
Focus on Passing Transactions
Show Database Calls
Agent: BusinessBackend_easyTravel@voormolen
Show Application Process
RUN GC
Host: voormolen
Show Host Health
CPU MEM NET DISK
Execution Response Time per Transaction: 4.47ms
Transaction Response Time Contribution: 0.11%
Failed Transactions: 0 (0%)

Observed tiers

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Configure observed tier rules

Observed tier appearance

Related pages

Transaction Flow dashlet

External API services

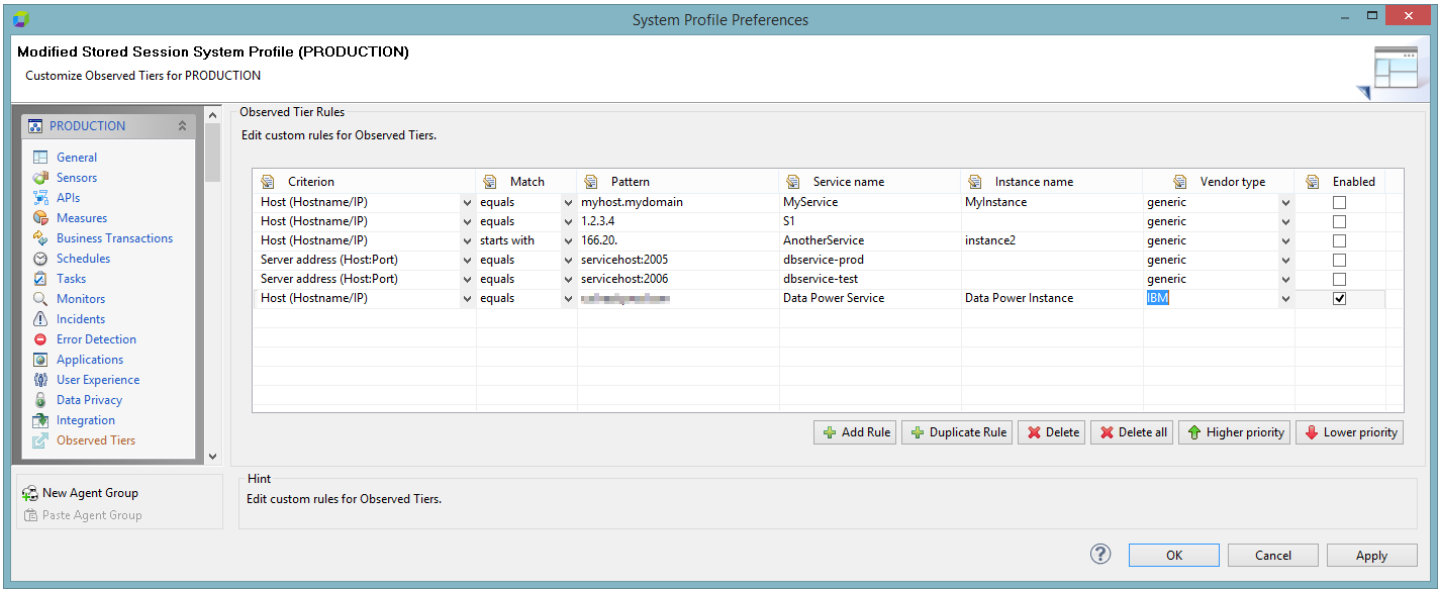
You can define **Observed Tiers** in the Transaction Flow to expose custom tiers such as DataPower, TIBCO EMS, and load balancer, that would not be visible when no agents are deployed.

Configure observed tier rules

Configure the rules that define Observed Tiers in the system profile.

From the Transaction Flow or Entry Screen, select **Configure Observed Tiers** from the context menu.

Observed Tiers can be added or removed after the PurePath data has been captured. Compared to using External API Services, this reduces performance overhead while monitoring the application.



The Observed Tier Rules Section of the System Profile Preferences Dialog Box

The following columns define an Observed Tier:

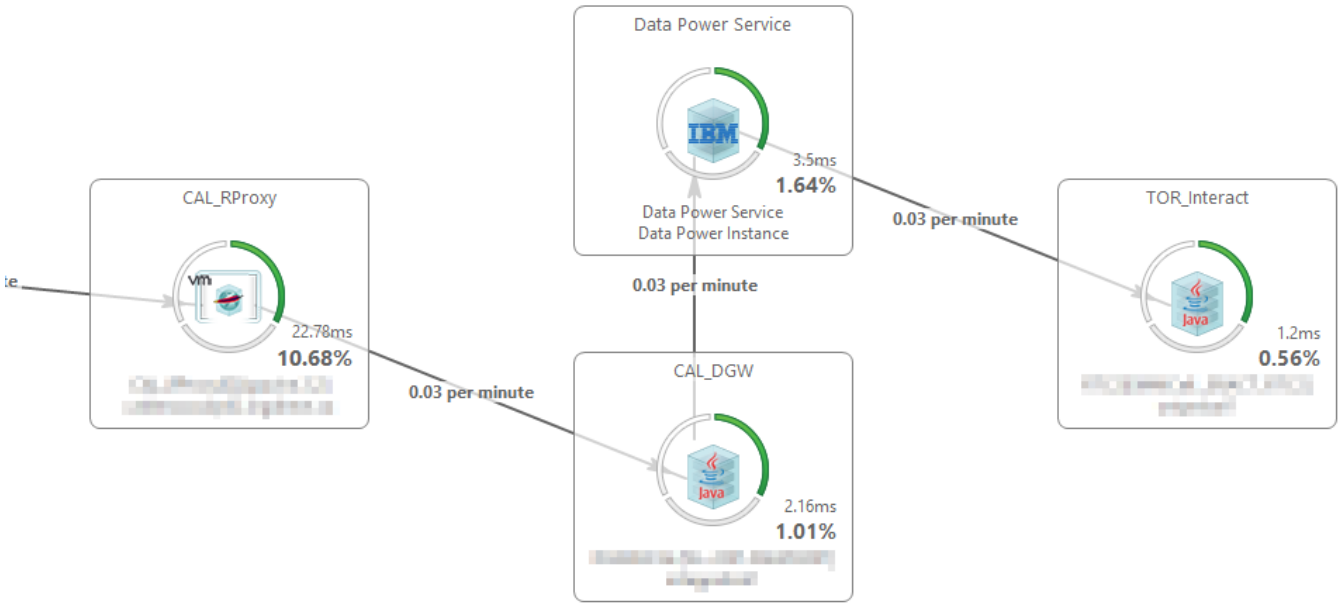
- **Criterion**: The Observed Tier must match against one of the following selections:
  - **Host (Hostname/IP)**: Matches a hostname or IP address.
  - **Server address (Host:Port)**: Matches a host:port string.
  - **Destination (Message)**: Matches a JMS Topic/Queue name.
- **Match**: Select the desired match type. Valid values are equals, starts, ends, or contains.
- **Pattern**: Enter the value to match against the selected criterion.

The following columns define how the Observed Tier appears in the Transaction Flow:

- **Service name**: Enter a string for the service represented by this Observed Tier. Multiple Observed Tiers with the same service name are aggregated into a single expandable group node in the Transaction Flow.
- **Instance name**: Enter a string for a specific instance of the service to allow more fine-grained classification in the Transaction Flow. If several Observed tiers have the same service name but different instance names, expanding that service name's node in the Transaction Flow will reveal the individual instance names as child nodes.
- **Vendor type**: Select a supported vendor for the Observed Tier from the list. This setting controls the icon that represents this service in the Transaction Flow.
- **Enabled**: Set by default. Observed Tiers do not appear in the Transaction Flow unless they are enabled. You can define several Observed Tier rules and then enable or disable them in different combinations to best match your diagnostic needs.

Observed tier appearance

The time booked to an Observed Tier is the sum of all self-times (such as node time without the node's children's times) of all nodes that contribute to that Observed Tier, which is the client-side web request method node's self-time.



Data Power Service (Data Power Instance) showing up as Observed Tier

External API services

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Configuration in Method Rule dialog box

Business Transactions

PurePath dashlet

Transaction Flow

Leaf nodes

You can specify any Java / .NET / PHP method as an invocation of an **external API service**.

Configuration in Method Rule dialog box

To configure a method as an external API service call, you first need a method rule for that method. The configuration is done directly in **System Profile Preferences > Sensors > Add / Edit Method Rule** dialog box. Just select the checkbox on the bottom (shown in screenshot 1) and all subsequent method invocations matching this method rule is marked as external API service calls.

External API service calls can have optional service and instance names that are also configurable in the method rule configuration dialog box:

- The **service name** is a user-chosen identifier string that will be the caption of all collected calls in the transaction flow.
- The **instance name** is also a user-chosen identifier string, but with some special features. It can be filled with captured arguments and return values, the method name and the service name.

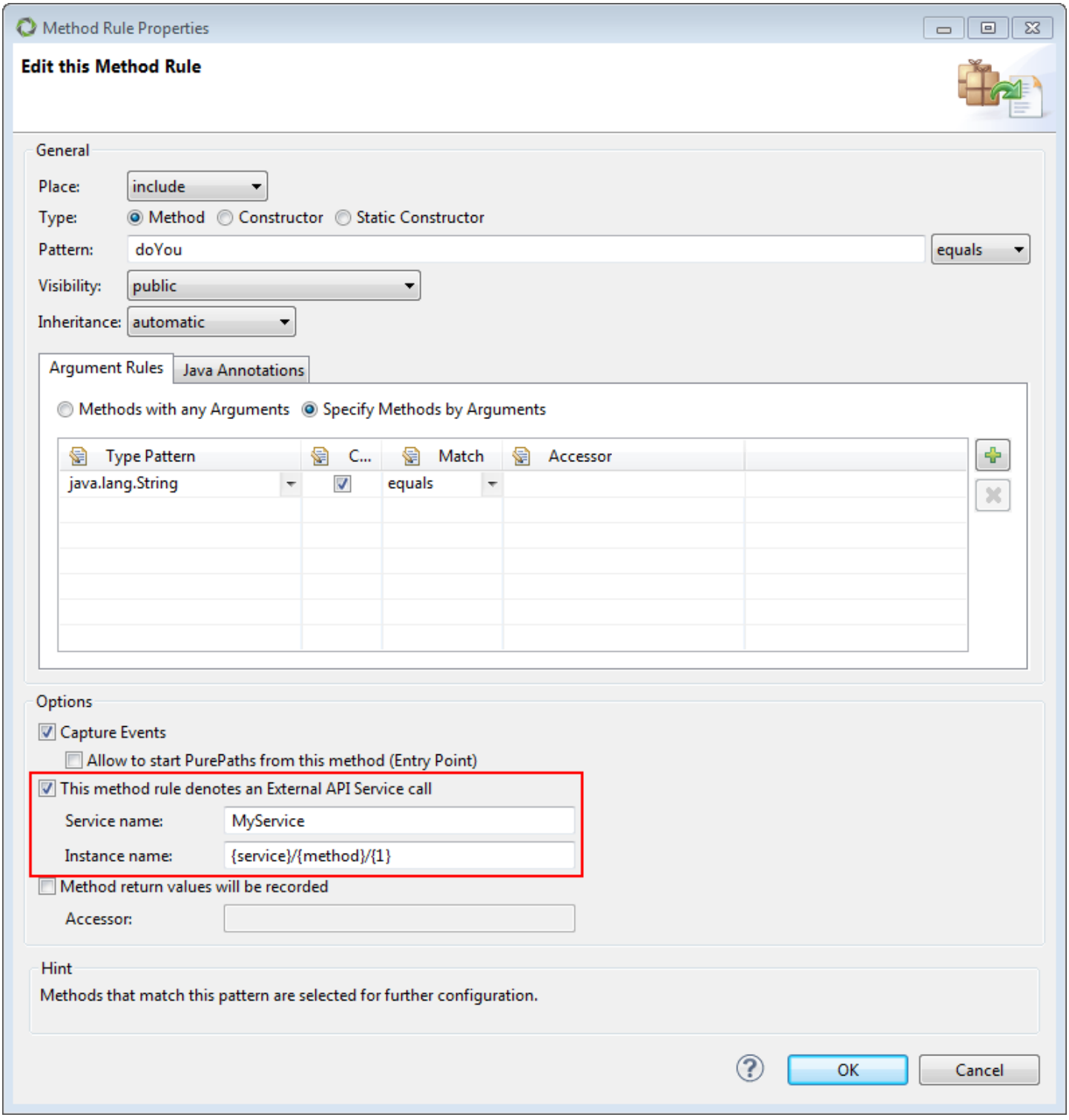
Instance name syntax

This addresses use-cases where a call is made to some service and a username, a service name, or virtually anything of interest is passed as a **method argument**. To enable this feature, the user has to enable **argument capturing** for all arguments of interest, and/or **return value capturing**, if the return value is of interest, respectively.

In the instance name pattern string each argument can be referred to with placeholders of the syntax `{1}, {2}`… where the index inside braces is the 1-based argument index, `{0}` (zero) being reserved for the method return value.

The instance name, if not provided, falls back to the method name. Furthermore, special placeholders `{method}` and `{service}` evaluates to the method name and service name, respectively. It's possible to quote strings in single quotes, then they are not be evaluated with the logic described here.

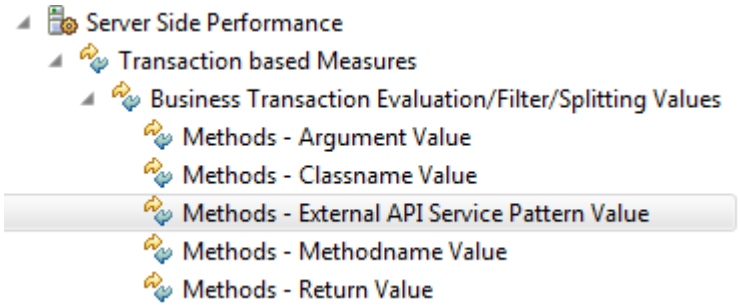
The method rule configuration dialog box issues a warning, if an argument referred to by the pattern has capturing not enabled. It issues an error (and thus prevent the user from closing the dialog box), if the pattern has a syntax error (unclosed braces or quotations) or refers to non-existing arguments.



Screenshot 1: Method Rule configuration dialog box

Business Transactions

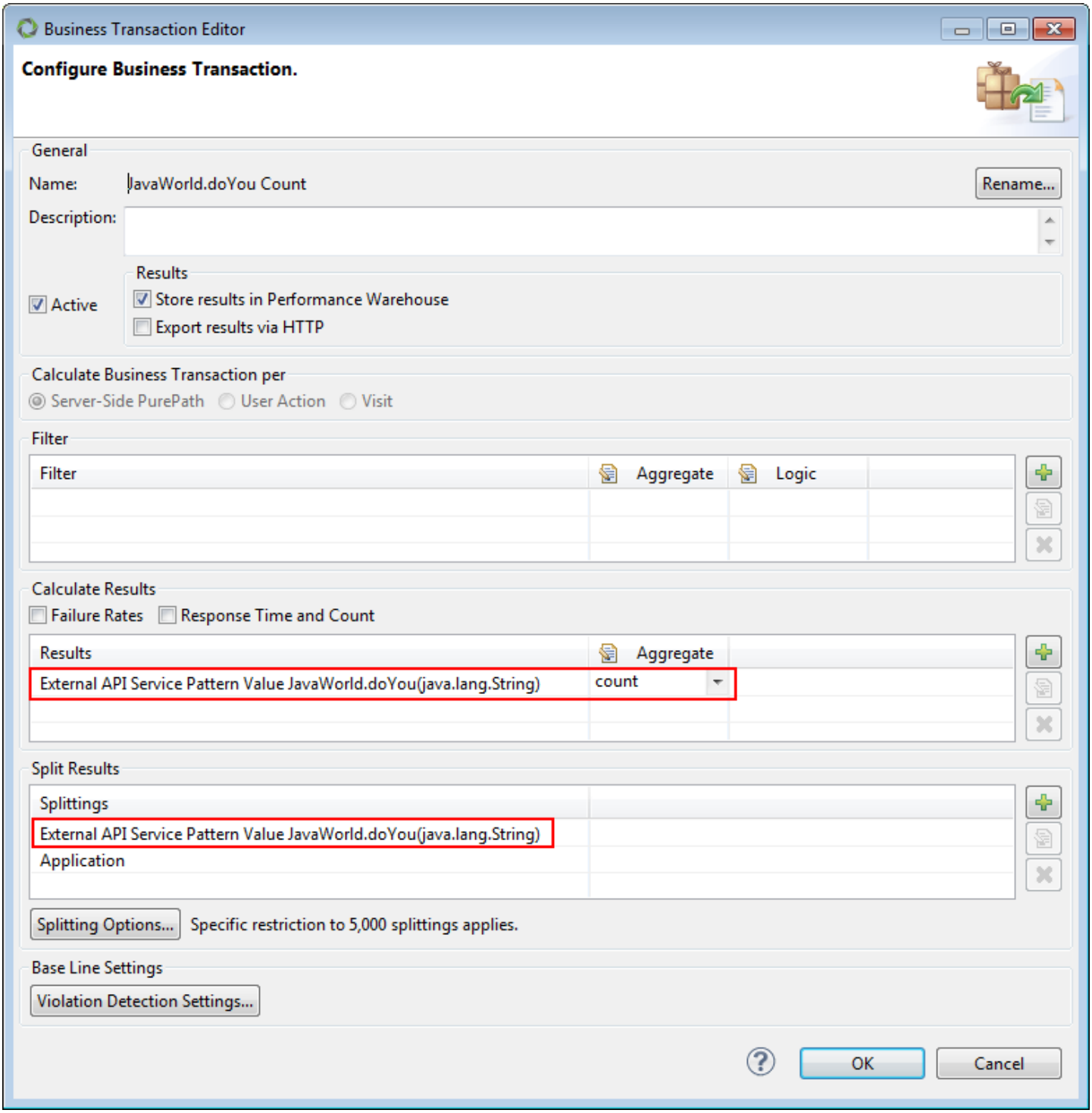
To collect data of external API service calls, a method metric called **External API Service Pattern Value** is available for use as splitting and result measure for business transactions. It evaluates to the formatted pattern value of all calls that match the signature of the metric.

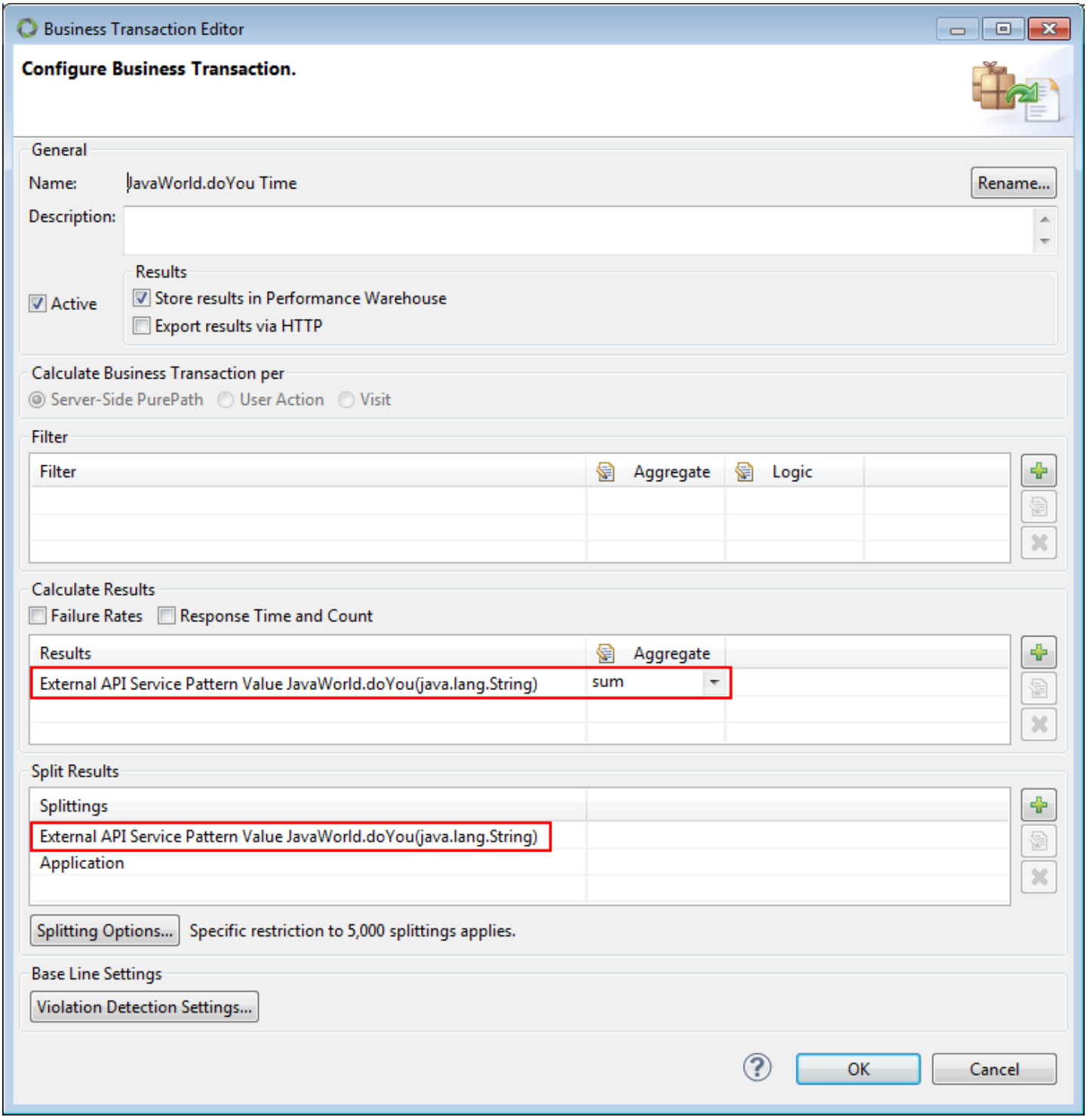


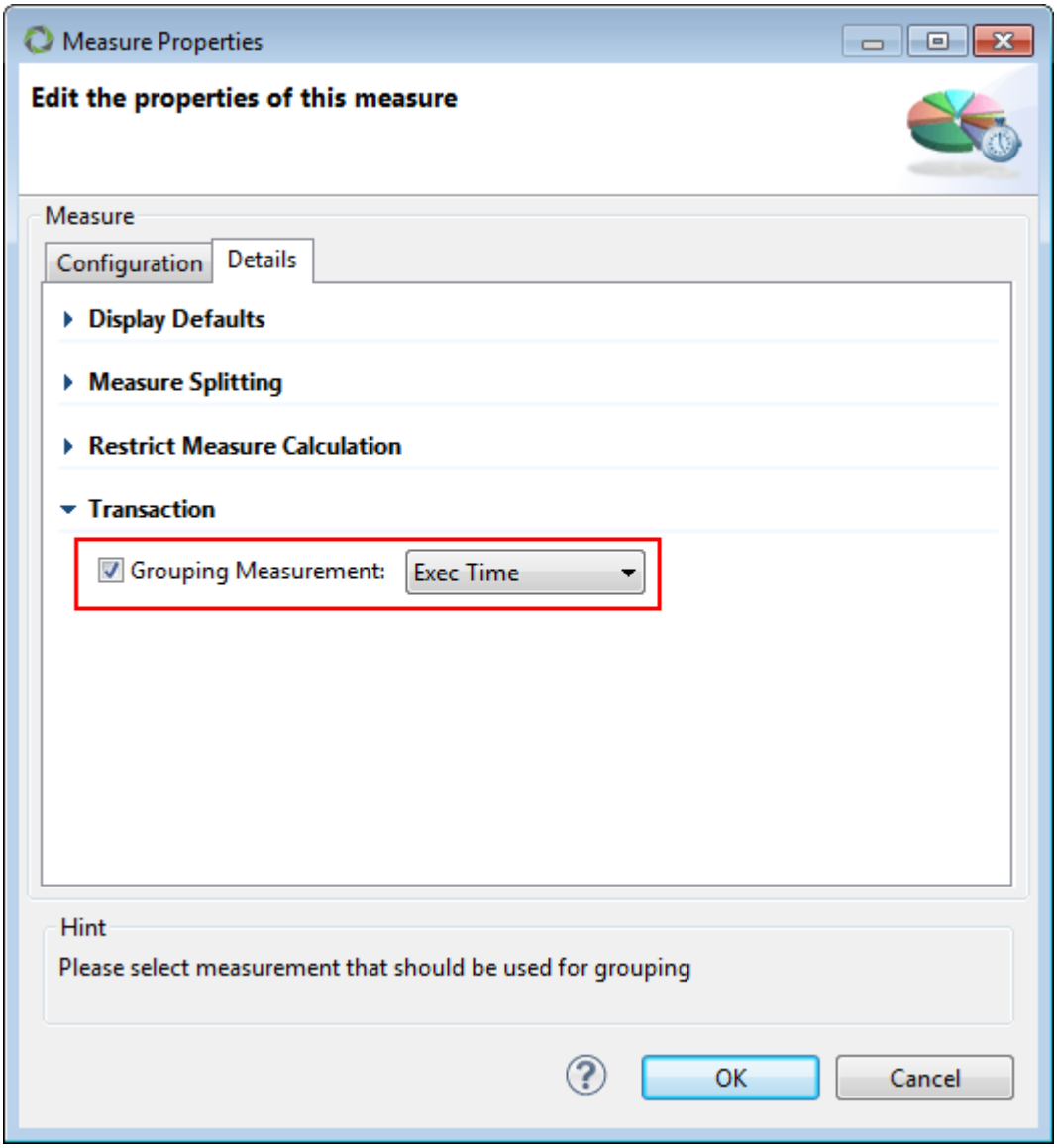
Screenshot 2: external API service pattern value method metric for Business Transactions

It's possible to configure business transactions that split per formatted pattern value of external API service calls, however, to have correct counts such as method invocation counts and times such as method execution duration, the business transaction must be carefully configured as follows:

- If both Count and Time are of interest, the user has to create one BT for each respectively.
- The Count business transaction must have its Calculate Results aggregate set to Count (see screenshot 3).
- The Time business transaction must have it's Calculate Results aggregate set to Sum (see screenshot 4).
- Both business transactions must use the same measure (created from subscribing the **External API Service Pattern Value** metric) as Calculate Results and Splitting measure. This must be the same measure instance per reference. Do not create it twice.
- The underlying External API Service Pattern Value measure must be configured to have its Grouping Measurement set to Exec Time (see in screenshot 5).


Business Transaction Editor
Configure Business Transaction.
General
Name: JavaWorld.doYou Count
Rename...
Description:
Active
Results
Store results in Performance Warehouse
Export results via HTTP
Calculate Business Transaction per
Server-Side PurePath
User Action
Visit
Filter
Filter
Aggregate
Logic
Calculate Results
Failure Rates
Response Time and Count
Results
Aggregate
External API Service Pattern Value JavaWorld.doYou(java.lang.String)
count
Split Results
Splittings
External API Service Pattern Value JavaWorld.doYou(java.lang.String)
Application
Splitting Options...
Specific restriction to 5,000 splittings applies.
Base Line Settings
Violation Detection Settings...
OK
Cancel


Business Transaction Editor
Configure Business Transaction.
General
Name: JavaWorld.doYou Time
Rename...
Description:
Active
Results
Store results in Performance Warehouse
Export results via HTTP
Calculate Business Transaction per
Server-Side PurePath
User Action
Visit
Filter
Filter
Aggregate
Logic
Calculate Results
Failure Rates
Response Time and Count
Results
Aggregate
External API Service Pattern Value JavaWorld.doYou(java.lang.String)
sum
Split Results
Splittings
External API Service Pattern Value JavaWorld.doYou(java.lang.String)
Application
Splitting Options...
Specific restriction to 5,000 splittings applies.
Base Line Settings
Violation Detection Settings...
OK
Cancel



The following screenshots show business transaction results split per pattern value and have method execution time and count series, respectively.

Name	Result Measures	Application	Splittings	Min	Max	Avg	Sum	Time [ms] Sum
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you8	111.00	111.00	111.00	111.00	55.31
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you4	111.00	111.00	111.00	111.00	55.31
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you3	111.00	111.00	111.00	111.00	55.31
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you9	111.00	111.00	111.00	111.00	55.31
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you5	111.00	111.00	111.00	111.00	55.31
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you7	111.00	111.00	111.00	111.00	55.31
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you2	111.00	111.00	111.00	111.00	55.31
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you6	111.00	111.00	111.00	111.00	55.31
JavaWorld.doYou Count	External API Service Patter...	Default Application	MyService/doYou/you1	111.00	111.00	111.00	111.00	55.31

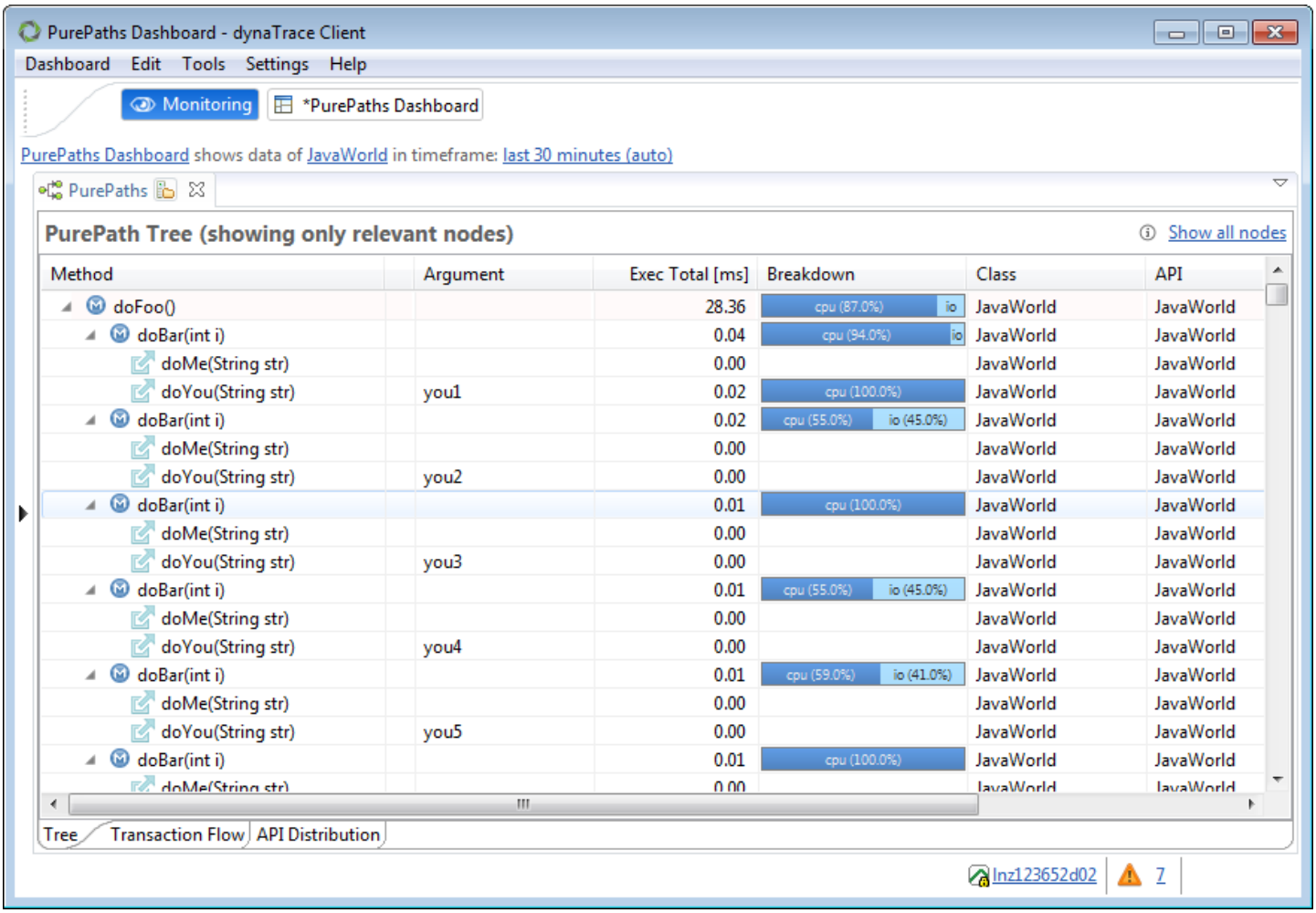
Screenshot 6: Business Transaction with Method Execution - Count series

Name	Result Measures	Application	Splittings	Min	Max	Avg	Sum	Time [ms] Sum
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you9	0.46	0.46	0.46	0.46	55.31
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you5	0.81	0.81	0.81	0.81	55.31
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you6	10.61	10.61	10.61	10.61	55.31
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you2	1.18	1.18	1.18	1.18	55.31
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you3	0.94	0.94	0.94	0.94	55.31
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you4	4.71	4.71	4.71	4.71	55.31
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you1	1.63	1.63	1.63	1.63	55.31
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you8	0.45	0.45	0.45	0.45	55.31
JavaWorld.doYou Time	External API Service Patter...	Default Application	MyService/doYou/you7	0.42	0.42	0.42	0.42	55.31

Screenshot 7: Business Transaction with Method Execution - Time series

PurePath dashlet

In the PurePath dashlet, all external API service calls are marked with a special icon that looks the same as generic external services in transaction flow. Furthermore, the details dialog box shows the configured service and instance name, as well as the formatted instance name, with all placeholders replaced, as shown in transaction flow.

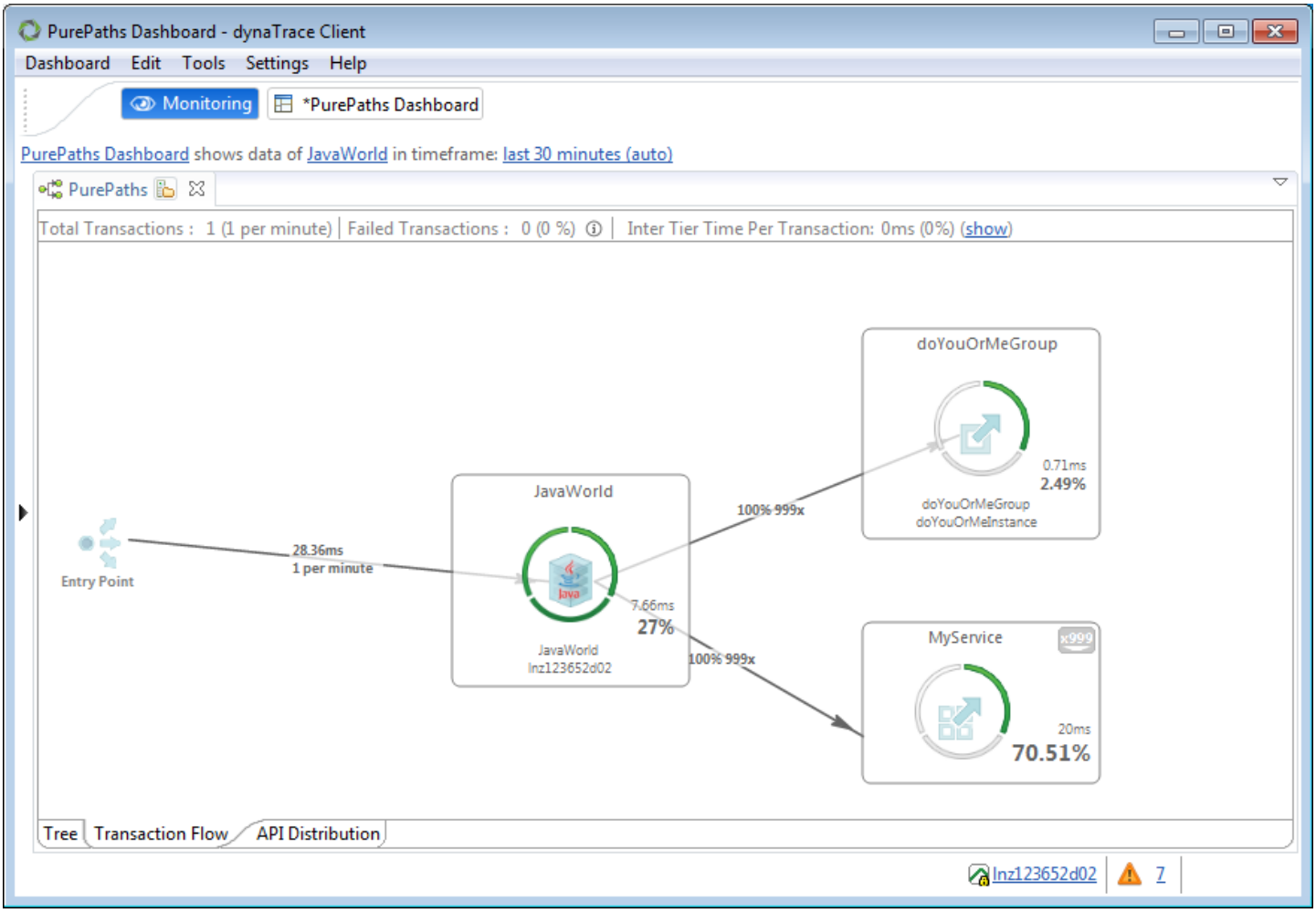


Screenshot 8: PurePath view of external API service calls

The easiest way to configure a business transaction for a specific method is to select the method node in PurePath dashlet, open the context menu and select “New Business Transaction…”. This will create a business transaction upon a method rule that is also newly created for the selected method node.

Transaction Flow

If an application contains external API service calls, those are shown as external service calls in the transaction flow – as usual. The tier groups are named after the service name (falling back to external API service if not specified in the method rule). The individual tiers (nodes) are named after the corresponding formatted instance names (falling back to the method names if not specified in the method rule).
Please note that the transaction flow will not allow to expand groups that contain 100+ nodes to avoid unreadable dashboards. This can easily happen if a captured argument is e.g. simply an increasing number. The instance name pattern (especially the contributing arguments) must be chosen with care.



Screenshot 9: Transaction Flow view of external API service calls

Leaf nodes

Only leaf nodes can be marked as external API service calls. That is, if a rule applies to a non-leaf node (node with child nodes), that node will not be marked as such. However, if the child nodes are exception, allocation or logging nodes, this rule does not apply.

Functional Health dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Selected time frame

Other time frame

Error HotSpots

Available drilldowns

Related pages

System Profile - Error Detection

User Actions Health dashlet

Use the **Functional Health** dashlet to analyze the transaction load and failure rate compared to another time frame.



Functional Health Dashlet

Selected time frame

The upper part of the dashlet displays the overall failure rate of transactions and the number of failed transactions. The chart at the right displays transaction load and failure rate. Click on this to drill down to the Errors dashlet.

The pie chart at the left shows the total transaction failure rate.

Other time frame

Use the middle chart to compare current functional health with a previous state.

Error HotSpots

The bottom part of the dashlet shows the error hotspots by application or by detected errors. The numbers refer to the selected time frame of the dashlet. Click on this to go to the Errors dashlet.

Available drilldowns

If you hover over a chart bar, the mouse cursor changes to indicate that you can drill down to the Errors dashlet. Click to open the Errors dashlet with a time frame filter applied for the selected period.

The links available in the error hotspots table will also open an errors dashlet either focusing on the selected application or selecting the corresponding error entry.

Performance specific dashlets

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Agent Breakdown dashlet

API Breakdown dashlet

Business Transaction Hotspots dashlet

Business Transactions dashlet

Components dashlet

Database dashlet

Database Hotspots dashlet

Entry Points dashlet

Memory Allocations dashlet

Messaging dashlet

Method Hotspots dashlet

Methods dashlet

Mobile Workload dashlet

Naming Services dashlet

PurePath Comparison dashlet

PurePaths dashlet

Remoting dashlet

Response Time Hotspots dashlet

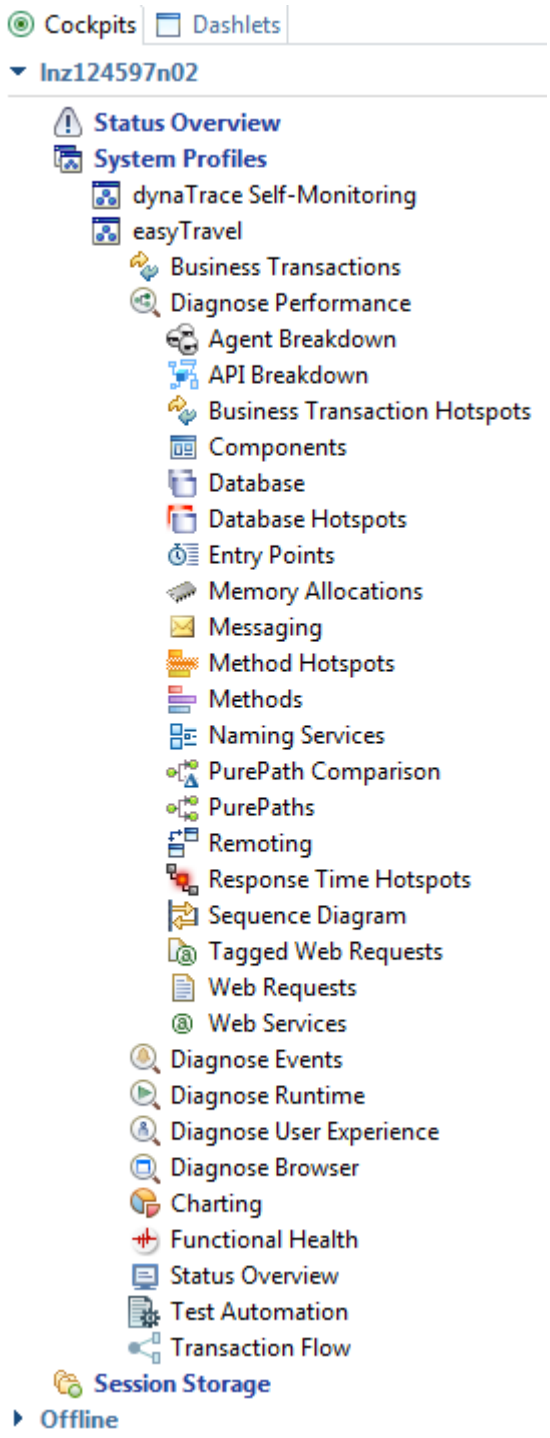
Sequence Diagram dashlet

Tagged Web Requests dashlet

Web Requests dashlet

Web Services dashlet

AppMon offers multiple ways to analyze PurePaths. The **Diagnose Performance** node of a System Profile in the Cockpit lists the available performance-specific dashlets. For details about a dashlet, click the link below.



Dashlet	Description and Content

Agent Breakdown Dashlet	Analyzes the distribution of PurePaths over Agents and the system under diagnosis (SUD; the application where the Agent is inserted).
API Breakdown Dashlet	Provides a view of accumulated performance data, broken down into different APIs.
Business Transaction Hotspots Dashlet	Provides an overview of business transaction data of various time frames.
Components Dashlet	Aggregates method calls in the context of software components (for example, Enterprise Java Beans).
Database Dashlet	Provides an overview of executed SQL statements, databases, and connection pools.
Database Hotspots Dashlet	Helps to identify database statements that contribute the most to the overall execution time.
Entry Points Dashlet	Shows an aggregated representation of PurePaths that start at the same entry point.
Memory Allocations Dashlet	Identifies methods within PurePaths that create instances of selected classes, based on Memory Sensor Rules.
Messaging Dashlet	Displays all sent or received messages from messaging services (for example, JMS).
Method Hotspots Dashlet	Helps to identify methods that contribute most to the overall execution time.
Methods Dashlet	Displays method calls aggregated by name, package, or class, including references to callers and callees.
Naming Services Dashlet	Displays all calls to naming service interfaces (for example, JNDI).
PurePath Comparison Dashlet	Offers an in-depth structural comparison of PurePaths.
PurePaths Dashlet	Provides the central representation of PurePaths, including an aggregated view of contributors and API distribution, and displays a graphical visualization.
Remoting Dashlet	Displays remoting metrics across horizontal and vertical application layers.
Response Time Hotspots Dashlet	Helps to find application hot spots within tiers and APIs, based on response times.

Sequence Diagram Dashlet	Visualizes selected PurePaths as a UML sequence diagram.
Tagged Web Requests Dashlet	Lists web requests originating from integrated load testing systems.
Web Requests Dashlet	Displays web requests, for example servlets, JSPs, or ASP.NET pages.
Web Services Dashlet	Provides an overview of all web service calls.

Methods dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Context menu options

Related pages

Performance specific dashlets

Overview

The **Methods** dashlet provides information about the methods included in captured PurePaths Dashlet, including methods that called or were called by a selected item.

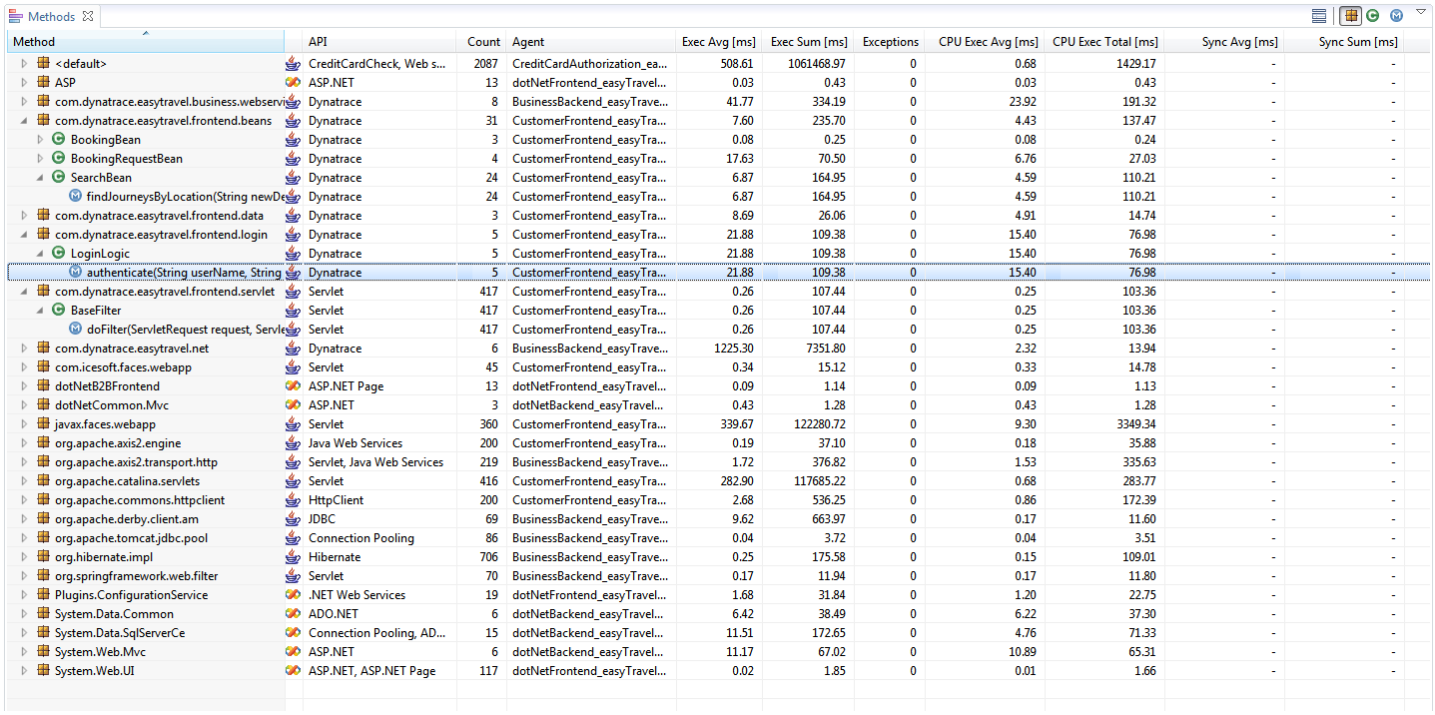
By default, the dashlet only lists the methods, in a tree that shows the packages and classes containing the methods. Click the first icon

in the dashlet toolbar to show the **Caller** and **Callee** panes. Double-click an item in the methods tree to display the caller and/or callee in these panes.

Use the grouping icons on the toolbar

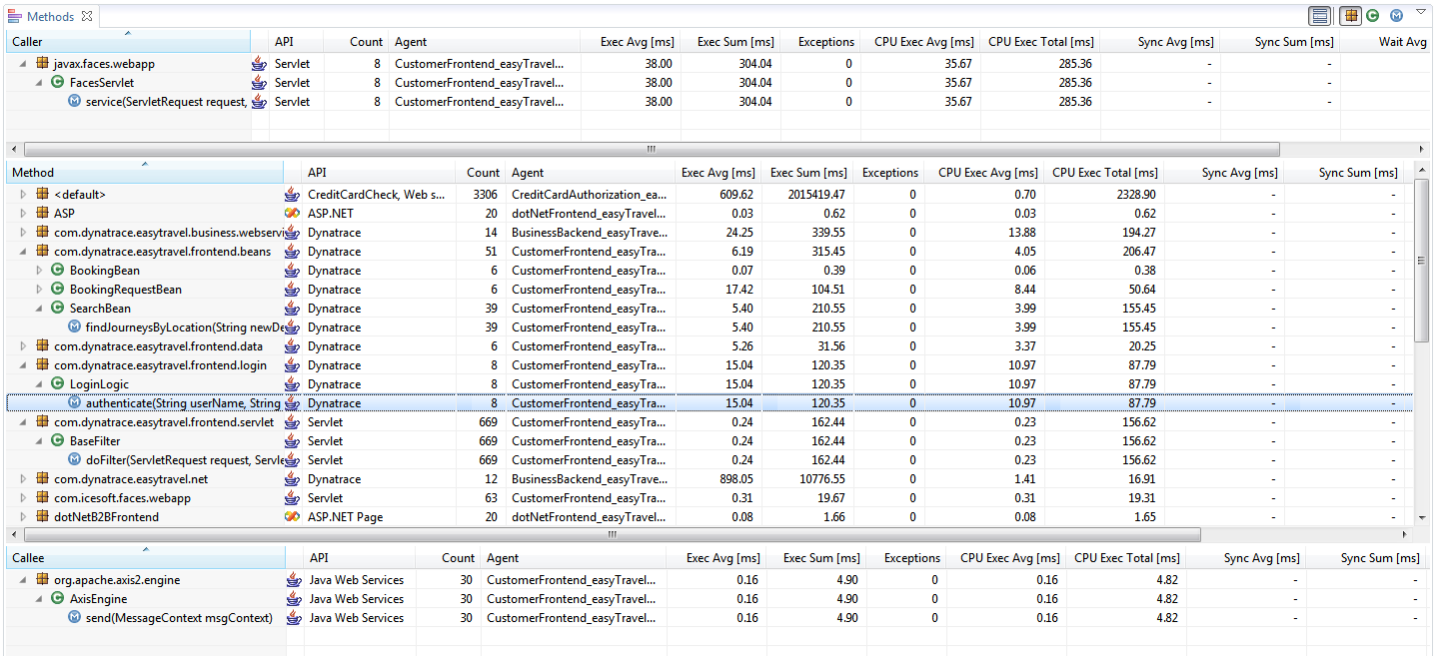
to select whether to group the methods by packages or by classes, or to show the methods in a flat list. The default view is grouping by package. If grouping by package or class is selected, use

to open packages and classes and to navigate to the methods.



Methods

Method	API	Count	Agent	Exec Avg [ms]	Exec Sum [ms]	Exceptions	CPU Exec Avg [ms]	CPU Exec Total [ms]	Sync Avg [ms]	Sync Sum [ms]
<default>	CreditCardCheck, Web s...	2087	CreditCardAuthorization_ea...	508.61	1061468.97	0	0.68	1429.17	-	-
ASP	ASP.NET	13	dotNetFrontend_easyTravel...	0.03	0.43	0	0.03	0.43	-	-
com.dynatrace.easytravel.business.webservi	Dynatrace	8	BusinessBackend_easyTrave...	41.77	334.19	0	23.92	191.32	-	-
com.dynatrace.easytravel.frontend.beans	Dynatrace	31	CustomerFrontend_easyTra...	7.60	235.70	0	4.43	137.47	-	-
BookingBean	Dynatrace	3	CustomerFrontend_easyTra...	0.08	0.25	0	0.08	0.24	-	-
BookingRequestBean	Dynatrace	4	CustomerFrontend_easyTra...	17.63	70.50	0	6.76	27.03	-	-
SearchBean	Dynatrace	24	CustomerFrontend_easyTra...	6.87	164.95	0	4.59	110.21	-	-
findJourneysByLocation(String newDe	Dynatrace	24	CustomerFrontend_easyTra...	6.87	164.95	0	4.59	110.21	-	-
com.dynatrace.easytravel.frontend.data	Dynatrace	3	CustomerFrontend_easyTra...	8.69	26.06	0	4.91	14.74	-	-
com.dynatrace.easytravel.frontend.login	Dynatrace	5	CustomerFrontend_easyTra...	21.88	109.38	0	15.40	76.98	-	-
LoginLogic	Dynatrace	5	CustomerFrontend_easyTra...	21.88	109.38	0	15.40	76.98	-	-
authenticate(String userName, String	Dynatrace	5	CustomerFrontend_easyTra...	21.88	109.38	0	15.40	76.98	-	-
com.dynatrace.easytravel.frontend.servlet	Servlet	417	CustomerFrontend_easyTra...	0.26	107.44	0	0.25	103.36	-	-
BaseFilter	Servlet	417	CustomerFrontend_easyTra...	0.26	107.44	0	0.25	103.36	-	-
doFilter(ServletRequest request, Servle	Servlet	417	CustomerFrontend_easyTra...	0.26	107.44	0	0.25	103.36	-	-
com.dynatrace.easytravel.net	Dynatrace	6	BusinessBackend_easyTrave...	1225.30	7351.80	0	2.32	13.94	-	-
com.icesoft.faces.webapp	Servlet	45	CustomerFrontend_easyTra...	0.34	15.12	0	0.33	14.78	-	-
dotNetB2BFrontend	ASP.NET Page	13	dotNetFrontend_easyTravel...	0.09	1.14	0	0.09	1.13	-	-
dotNetCommon.Mvc	ASP.NET	3	dotNetBackend_easyTravel...	0.43	1.28	0	0.43	1.28	-	-
javax.faces.webapp	Servlet	360	CustomerFrontend_easyTra...	339.67	122280.72	0	9.30	3349.34	-	-
org.apache.axis2.engine	Java Web Services	200	CustomerFrontend_easyTra...	0.19	37.10	0	0.18	35.88	-	-
org.apache.axis2.transport.http	Servlet, Java Web Services	219	BusinessBackend_easyTrave...	1.72	376.82	0	1.53	335.63	-	-
org.apache.catalina.servlets	Servlet	416	CustomerFrontend_easyTra...	282.90	117685.22	0	0.68	283.77	-	-
org.apache.commons.httpclient	HttpClient	200	CustomerFrontend_easyTra...	2.68	536.25	0	0.86	172.39	-	-
org.apache.derby.client.am	JDBC	69	BusinessBackend_easyTrave...	9.62	663.97	0	0.17	11.60	-	-
org.apache.tomcat.jdbc.pool	Connection Pooling	86	BusinessBackend_easyTrave...	0.04	3.72	0	0.04	3.51	-	-
org.hibernate.impl	Hibernate	706	BusinessBackend_easyTrave...	0.25	175.58	0	0.15	109.01	-	-
org.springframework.web.filter	Servlet	70	BusinessBackend_easyTrave...	0.17	11.94	0	0.17	11.80	-	-
Plugins.ConfigurationService	.NET Web Services	19	dotNetFrontend_easyTravel...	1.68	31.84	0	1.20	22.75	-	-
System.Data.Common	ADO.NET	6	dotNetBackend_easyTravel...	6.42	38.49	0	6.22	37.30	-	-
System.Data.SqlServerCe	Connection Pooling, AD...	15	dotNetBackend_easyTravel...	11.51	172.65	0	4.76	71.33	-	-
System.Web.Mvc	ASP.NET	6	dotNetBackend_easyTravel...	11.17	67.02	0	10.89	65.31	-	-
System.Web.UI	ASP.NET, ASP.NET Page	117	dotNetFrontend_easyTravel...	0.02	1.85	0	0.01	1.66	-	-



Methods

Caller	API	Count	Agent	Exec Avg [ms]	Exec Sum [ms]	Exceptions	CPU Exec Avg [ms]	CPU Exec Total [ms]	Sync Avg [ms]	Sync Sum [ms]	Wait Avg
javax.faces.webapp	Servlet	8	CustomerFrontend_easyTravel...	38.00	304.04	0	35.67	285.36	-	-	
FacesServlet	Servlet	8	CustomerFrontend_easyTravel...	38.00	304.04	0	35.67	285.36	-	-	
service(ServletRequest request,	Servlet	8	CustomerFrontend_easyTravel...	38.00	304.04	0	35.67	285.36	-	-	

Method	API	Count	Agent	Exec Avg [ms]	Exec Sum [ms]	Exceptions	CPU Exec Avg [ms]	CPU Exec Total [ms]	Sync Avg [ms]	Sync Sum [ms]
<default>	CreditCardCheck, Web s...	3306	CreditCardAuthorization_ea...	609.62	2015419.47	0	0.70	2328.90	-	-
ASP	ASP.NET	20	dotNetFrontend_easyTravel...	0.03	0.62	0	0.03	0.62	-	-
com.dynatrace.easytravel.business.webservi	Dynatrace	14	BusinessBackend_easyTrave...	24.25	339.55	0	13.88	194.27	-	-
com.dynatrace.easytravel.frontend.beans	Dynatrace	51	CustomerFrontend_easyTra...	6.19	315.45	0	4.05	206.47	-	-
BookingBean	Dynatrace	6	CustomerFrontend_easyTra...	0.07	0.39	0	0.06	0.38	-	-
BookingRequestBean	Dynatrace	6	CustomerFrontend_easyTra...	17.42	104.51	0	8.44	50.64	-	-
SearchBean	Dynatrace	39	CustomerFrontend_easyTra...	5.40	210.55	0	3.99	155.45	-	-
findJourneysByLocation(String newDe	Dynatrace	39	CustomerFrontend_easyTra...	5.40	210.55	0	3.99	155.45	-	-
com.dynatrace.easytravel.frontend.data	Dynatrace	6	CustomerFrontend_easyTra...	5.26	31.56	0	3.37	20.25	-	-
com.dynatrace.easytravel.frontend.login	Dynatrace	8	CustomerFrontend_easyTra...	15.04	120.35	0	10.97	87.79	-	-
LoginLogic	Dynatrace	8	CustomerFrontend_easyTra...	15.04	120.35	0	10.97	87.79	-	-
authenticate(String userName, String	Dynatrace	8	CustomerFrontend_easyTra...	15.04	120.35	0	10.97	87.79	-	-
com.dynatrace.easytravel.frontend.servlet	Servlet	669	CustomerFrontend_easyTra...	0.24	162.44	0	0.23	156.62	-	-
BaseFilter	Servlet	669	CustomerFrontend_easyTra...	0.24	162.44	0	0.23	156.62	-	-
doFilter(ServletRequest request, Servle	Servlet	669	CustomerFrontend_easyTra...	0.24	162.44	0	0.23	156.62	-	-
com.dynatrace.easytravel.net	Dynatrace	12	BusinessBackend_easyTrave...	898.05	10776.55	0	1.41	16.91	-	-
com.icesoft.faces.webapp	Servlet	63	CustomerFrontend_easyTra...	0.31	19.67	0	0.31	19.31	-	-
dotNetB2BFrontend	ASP.NET Page	20	dotNetFrontend_easyTravel...	0.08	1.66	0	0.08	1.65	-	-

Callee	API	Count	Agent	Exec Avg [ms]	Exec Sum [ms]	Exceptions	CPU Exec Avg [ms]	CPU Exec Total [ms]	Sync Avg [ms]	Sync Sum [ms]
org.apache.axis2.engine	Java Web Services	30	CustomerFrontend_easyTravel...	0.16	4.90	0	0.16	4.82	-	-
AxisEngine	Java Web Services	30	CustomerFrontend_easyTravel...	0.16	4.90	0	0.16	4.82	-	-
send(MessageContext msgContext)	Java Web Services	30	CustomerFrontend_easyTravel...	0.16	4.90	0	0.16	4.82	-	-

Metrics

Note

The **Method Hotspots** dashlet displays data from Auto Sensors, but the **Methods** dashlet does not.

The **Methods** table provides the following information:

- **Method**: A string representing the method name, including argument types.
- **Platform**: This unlabeled column displays an icon representing the software platform on which this method was executed: Java or .NET.
- **API**: The API identifier with which this method call is associated.
- **Class**: The full-qualified class name of this method.
- **Count**: The number of times this method has been executed.
- **Agent**: The Agent that recorded this method call.
- **Exceptions**: The number of exceptions raised in this method.
- **Exec Avg [ms]**: The method's average execution time, measured as its response time minus the response time of all called (and instrumented) methods.
- **Exec Min [ms]**: The method's minimum execution time, measured as its response time minus the response time of all called (and instrumented) methods.
- **Exec Max [ms]**: The method's maximum execution time, measured as its response time minus the response time of all called (and instrumented) methods.
- **Exec Sum [ms]**: The method's total execution time, measured as its response time minus the response time of all called (and instrumented) methods.
- **Exceptions**: The number of times the method was exited because of an exception event.
- **CPU Exec Avg [ms]**: The average CPU time for execution within this method only.
- **CPU Exec Min [ms]**: The minimum CPU time for execution within this method only.
- **CPU Exec Max [ms]**: The maximum CPU time for execution within this method only.
- **CPU Exec Sum [ms]**: The total CPU time for execution within this method only.
- **Sync Avg [ms]**: The average synchronization time within this method only.
- **Sync Min [ms]**: The minimum synchronization time within this method only.
- **Sync Max [ms]**: The maximum synchronization time within this method only.
- **Sync Sum [ms]**: The total synchronization time within this method only.
- **Wait Avg [ms]**: The average waiting time within this method only.
- **Wait Min [ms]**: The minimum waiting time within this method only.
- **Wait Max [ms]**: The maximum waiting time within this method only.
- **Wait Sum [ms]**: The total waiting time within this method only.

By default, not all metrics are included in the table. To select which columns to display, right-click a column head and select or clear the column checkboxes as needed.

Context menu options

Right-click the dashlet to display a context menu that includes these options:

- **New Business Transaction**: Open the **Business Transaction Editor** to create a new Business Transaction based on measures related to the selected item. This option is only available during real time sessions.
- **Create Measures**: Open the **Measure Configuration** dialog box to add measures to a subscription. This option is only available during real time sessions.
- **Source Lookup**: See Source Lookup Feature for a detailed description.
- **Add Sensor Rules** (only available on live sessions):
  - **Refine with Callee Methods**: All methods that are called by the selected methods are retrieved and can be used as input for new Inclusion Sensor rules.
  - **Exclude Selected Methods**: All selected methods can be used as input for new Exclusion Sensor rules.
  - **Include Selected Methods**: All selected methods can be used as input for new Inclusion Sensor rules.

Agent Breakdown dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

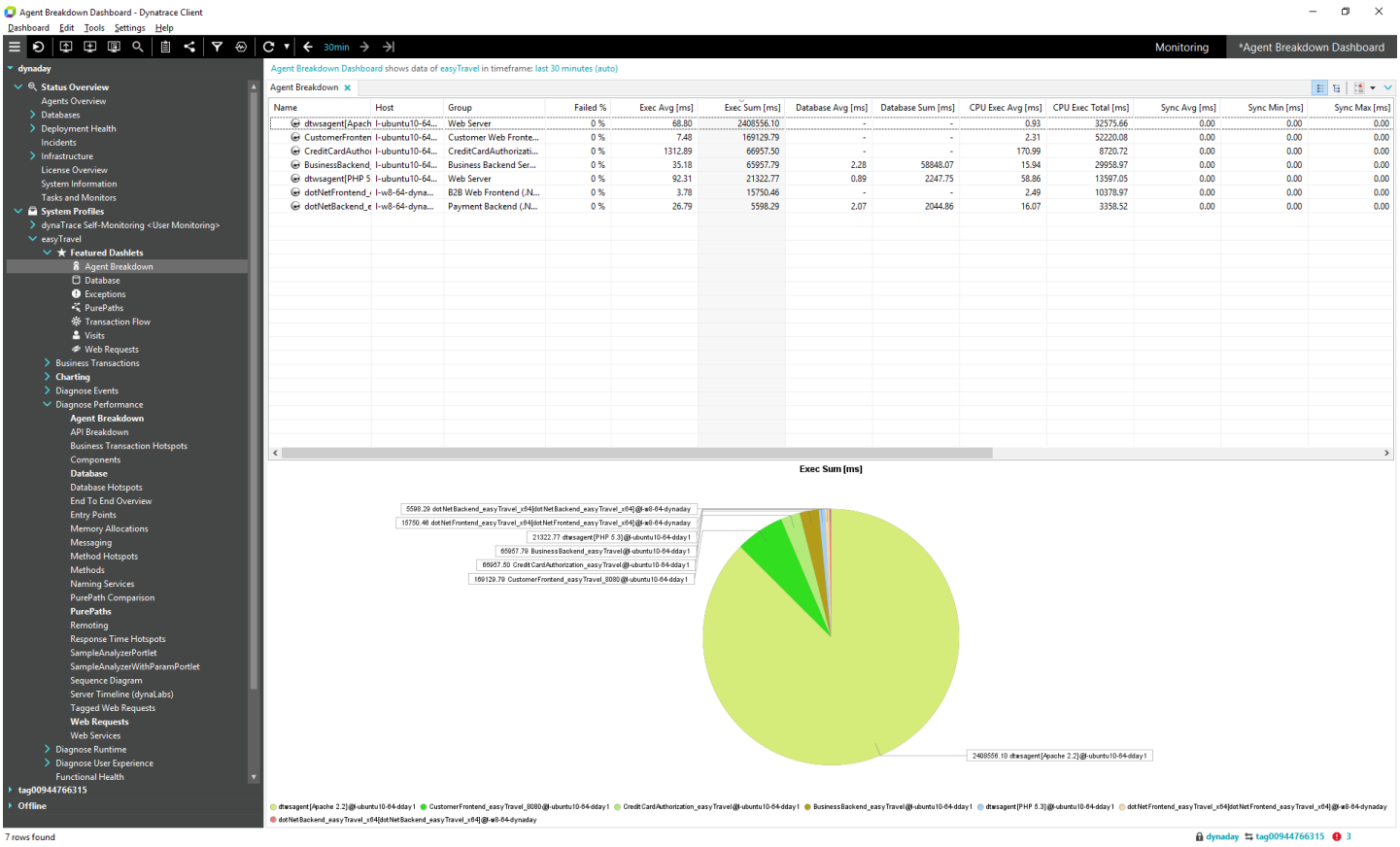
Related pages

Performance specific dashlets

Overview

The **Agent Breakdown** dashlet provides a view of accumulated performance data, broken down into different Agents or Agent Groups. This information helps to isolate performance issues at the Agent level. Each line of the table represents timing information from an Agent.

The pie chart in the bottom part of the dashlet displays the data from the table column that is used for sorting. To select a column, click the column header. The pie chart is only available for columns that contain timing data.



Agent Breakdown dashlet

Use the toolbar icons at the top right of the dashlet to select the structure mode:

- **Flat**: Every line in the table represents one Agent and its timing values. This is the default.
- **By Agent Group**: The Agent Groups are listed in the table, and Agents are grouped under their Agent Group. An Agent Group row shows the accumulated timing data for the whole Agent Group. When the Agent Group is collapsed, the pie chart represents the totals for the Agent Group. When the Agent Group is expanded, the chart contains the data for every single Agent of the Agent Group.

Metrics

The **Agent Breakdown** dashlet provides average (**Avg**), minimum, maximum, and total values for key performance metrics:

- **Exec**: Execution time spent in the target application of the AppMon Agent.
- **Database**: Time spent executing SQL statements in the target application of the AppMon Agent .
- **CPU Exec** : CPU time spent in the target application of the AppMon Agent.
- **Sync**: Time spent in the target application waiting for monitors.

- **Wait**: Time spent in the target application waiting for other threads.
- **Suspension**: Time the target application of the AppMon Agent was suspended.

A more detailed description of each column is available in the **Columns** tab of the **Dashlet Properties** dialog box. In this tab, you can select which columns to show or hide. To display the **Dashlet Properties** dialog box, right-click the dashlet and select **Dashlet Properties** from the context menu.

Business Transaction Hotspots dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Usage

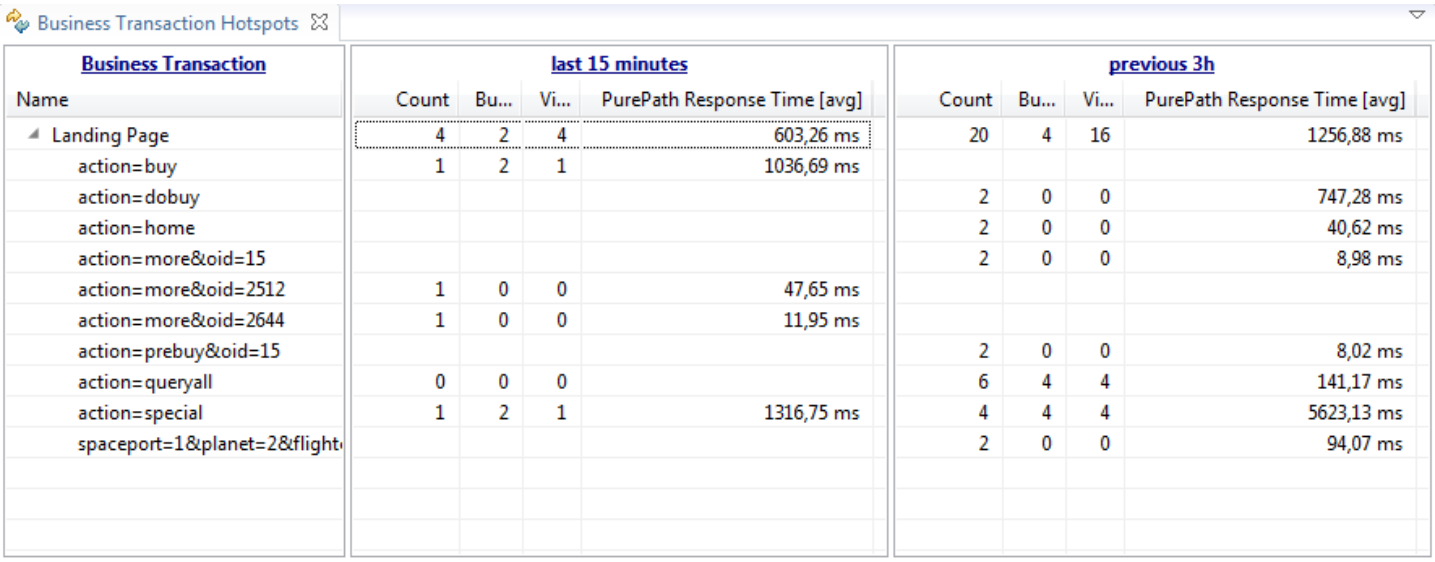
Related pages

Performance specific dashlets

Overview

The **Business Transaction Hotspots** dashlet displays historical and live information about business transactions.

The leftmost table displays the name of the business transaction. Dynamic groups of business transactions are displayed as nodes in a tree structure. The table to the right of the business transactions list displays the data collected during a specified time frames. By default, the dashlet contains one data table. You can configure the dashlet to display up to four tables. As the following figure shows, the tables display the same columns, to compare the data from different time frames.



Business Transaction Hotspots dashlet

Usage

Context menu actions

Right-click any table in the **Business Transaction Hotspots** dashlet to use these context-specific actions:

- **Add Business Transaction**: Select Business Transactions to add to the dashlet.
- **Configure Time Frames**: Define one or more time frames for the data display.
- **Configure Columns**: Define the data aggregations to display for the Business Transactions in the table.
- **Drill Down**: Display corresponding data in the dashlet selected from the cascading menu. Drilldowns from this dashlet set a Business Transaction related to the incidents and the time frame of the selected table. For example, if you have two data tables, one displaying data from this week and one from last week, drilling down from the first data table sets the time-frame in the target dashlet to this week, but drilling down from the second sets the time frame to last week.
- **Show in Dashboard**: Display the dashlet, with the current configurations, in the dashboard selected through the cascading menu.
- **Add Snapshot to Dashboard**: Display the dashlet in the selected dashboard with the time frame automatically set to **last 30 minutes**.

- **Show Measures in Chart**: Open a Chart dashlet that charts the measures visible in the table. The series are subscribed in the new chart based on which columns are visible in the **Business Transaction Hotspots** dashlet:
  - **Count**: The measure with the highest count among the result measures of the business transaction.
  - **Violations**: All violations measures that have result measures of the selected Business Transaction as their source measure.
  - **Any other result measure**: A series with each displayed aggregation type.
- **Edit Business Transaction**: Open the **Business Transaction Editor** for the selected Business Transaction.
- **Delete**: Remove the selected Business Transaction from the dashlet.

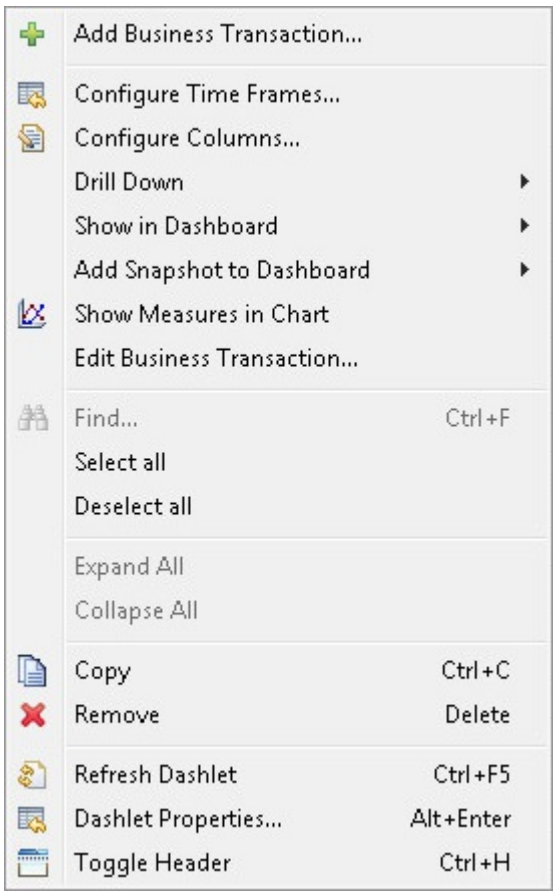


Table Context Menu

Add Business Transactions

To add new Business Transactions to the dashlet:

1. Click the **Business Transaction** link at the top of the leftmost table or select **Add Business Transaction** from the table's context menu. The **Business Transaction Picker** dialog box appears.
2. Select the Business Transaction. If necessary, select the splitting for the transaction.
   If there is no violation measure available for the Business Transaction, a warning displays at the top of the dialog box. In this case, the column **Violations per Business Transaction[%]** does not show any data for the selected Business Transaction. You can create a violation measure. See System Profile - Measures for more information.
3. Click Add. The dialog box remains open so you can select more Business Transactions.
4. When you are done, click Close.

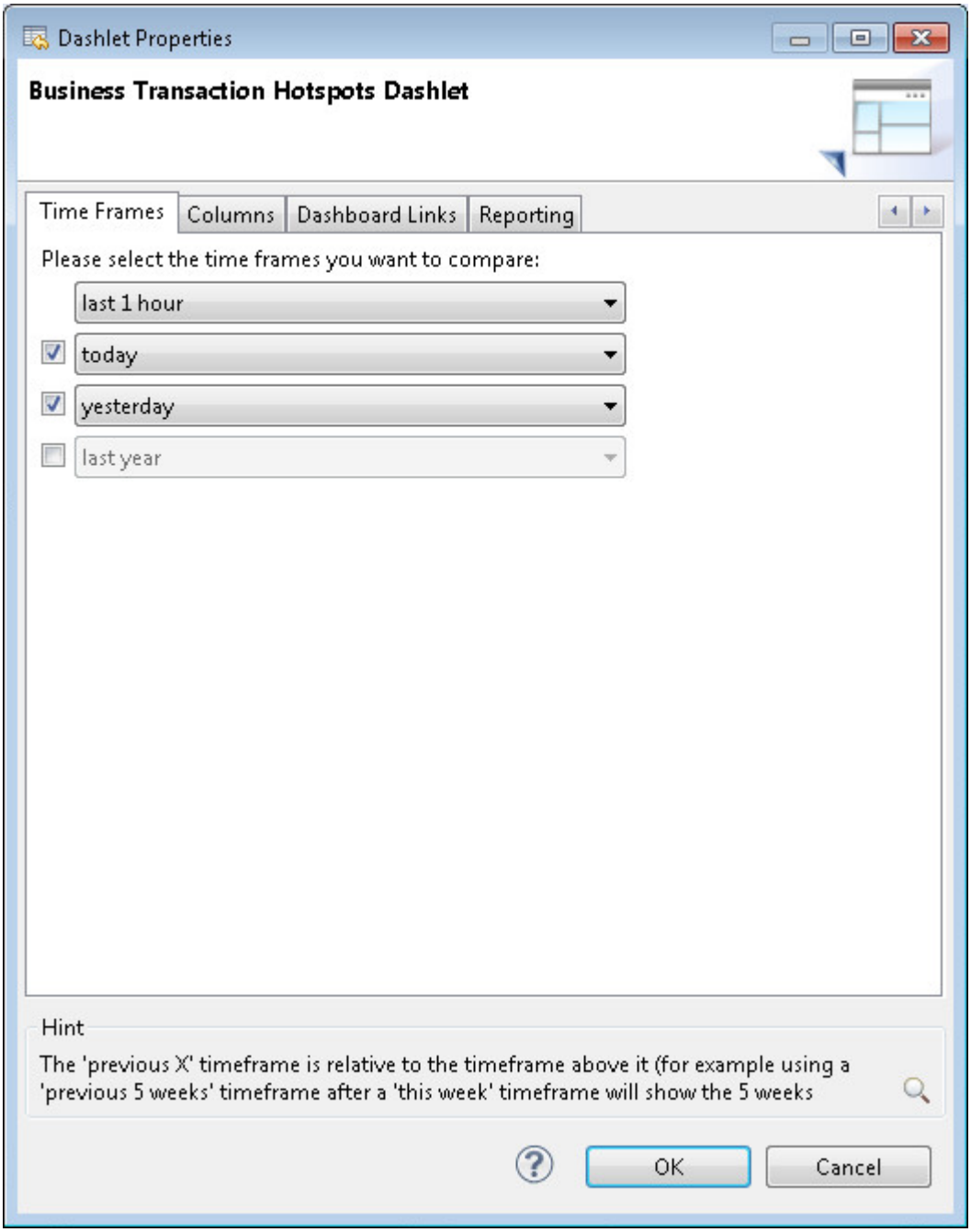
Add or remove tables

To add or remove tables in the dashlet:

1. Right-click the dashlet and select **Dashlet Properties** from the context menu.
2. In the **Dashlet Properties** dialog box, select **Time Frames**.
3. Select the checkboxes to add the tables, and then select the time frame for each table.

Manage time frames

The selected time frame is listed in each table above the column headers. To change the time frame for a table, you can click the link (for example, Auto) and select the new time frame from the popup list. This list, however, does not include all the configurable time frames. To further refine the time frame or to configure multiple tables, either click the time frame link or right-click the dashlet, then select Configure Time Frames to display the Time Frames tab.



Time frames tab

The **Auto** default setting is available only for the first table. The list for each table includes time frames ranging from the last 10 seconds through the last 365 days. Note that **last <number> hours** is not the same as **today**, **this week**, etc. For example, the **last 7 days** is calculated relative to the date and time (*hh:mm:ss*) you select the time range, but **this week** is midnight on the previous Monday through the current date and time.

The list also includes these configurable time frames:

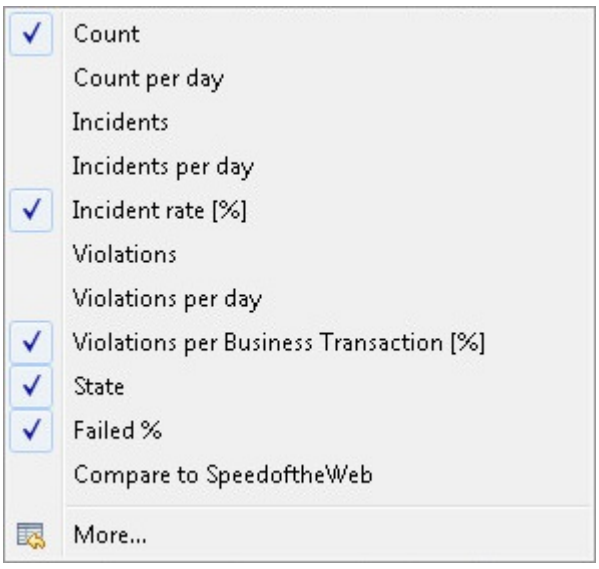
- **last X**: Select both the number and the time unit (**seconds** through **months**). The **Shift by** setting adjusts the time earlier. For example, if the time is 8:49 when you select **Last 1 hour** and **Shift by 5 minutes**, data will be displayed for the last hour as of 7:44, not 7:49. This is a relative time frame, which changes as time passes.
- **custom**: Define a start and end date and time for the time frame. This time frame is absolute; it will not change as time passes.
- **previous X**: This time frame is only available for the second, third, and forth table because it is relative to the time frame for the preceding table. For example, today is Friday and the first time frame is **today**. If you choose to display the previous 3 days, then the second table displays data from Tuesday 00:00 until Friday 0:00.

> **Note**
>
> Hover over the time frame link in a table to see the current start and end dates and times for the time frame.

Choose columns to display

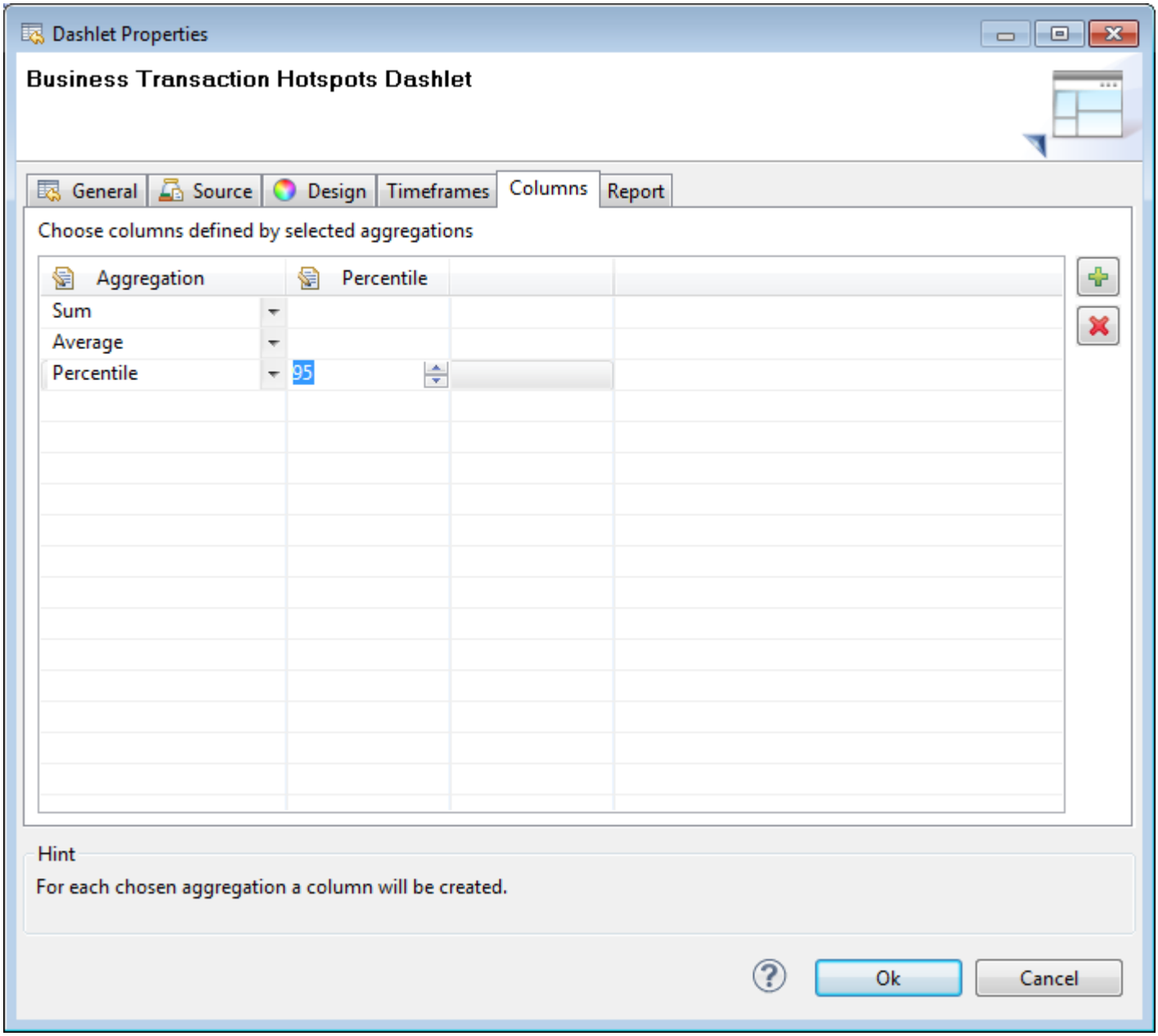
To choose which columns to display, right-click the table header in one of the data tables to display a context menu in which you can add or remove columns in the table.

The following figure shows the default column selections. When you add a Business Transaction to the dashlet, columns are automatically added to the table to display relevant statistics, and these columns are also added to the context menu.



Column Context Menu

By default, the average aggregation column is displayed for most measures. To add different aggregations, select **More** from the columns context menu or **Configure Columns** from the table's context menu to display the **Columns** tab. In this tab, you can change or add aggregations. The following figure shows a **Percentile** aggregation added to the default **Sum** and **Average** aggregations. Note that **Sum** is available by default, but the **Sum** column is not included by default in the table.



Aggregations

To add a new aggregation, click the **Add** button and select the aggregation type. If you select the aggregation type **Percentile**, select or enter the percentile.

When you change an existing aggregation type, the corresponding table column is changed. When you add an aggregation type, a column is added to the table to display that aggregation for the measures specific to the Business Transactions in the table.

Business Transactions dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Business Transaction metrics

Reanalyze all PurePaths

Related pages

Performance specific dashlets

How to use Business Transactions

Usage of Sessions

Watch and learn

Business Transaction Management Webinar

Overview

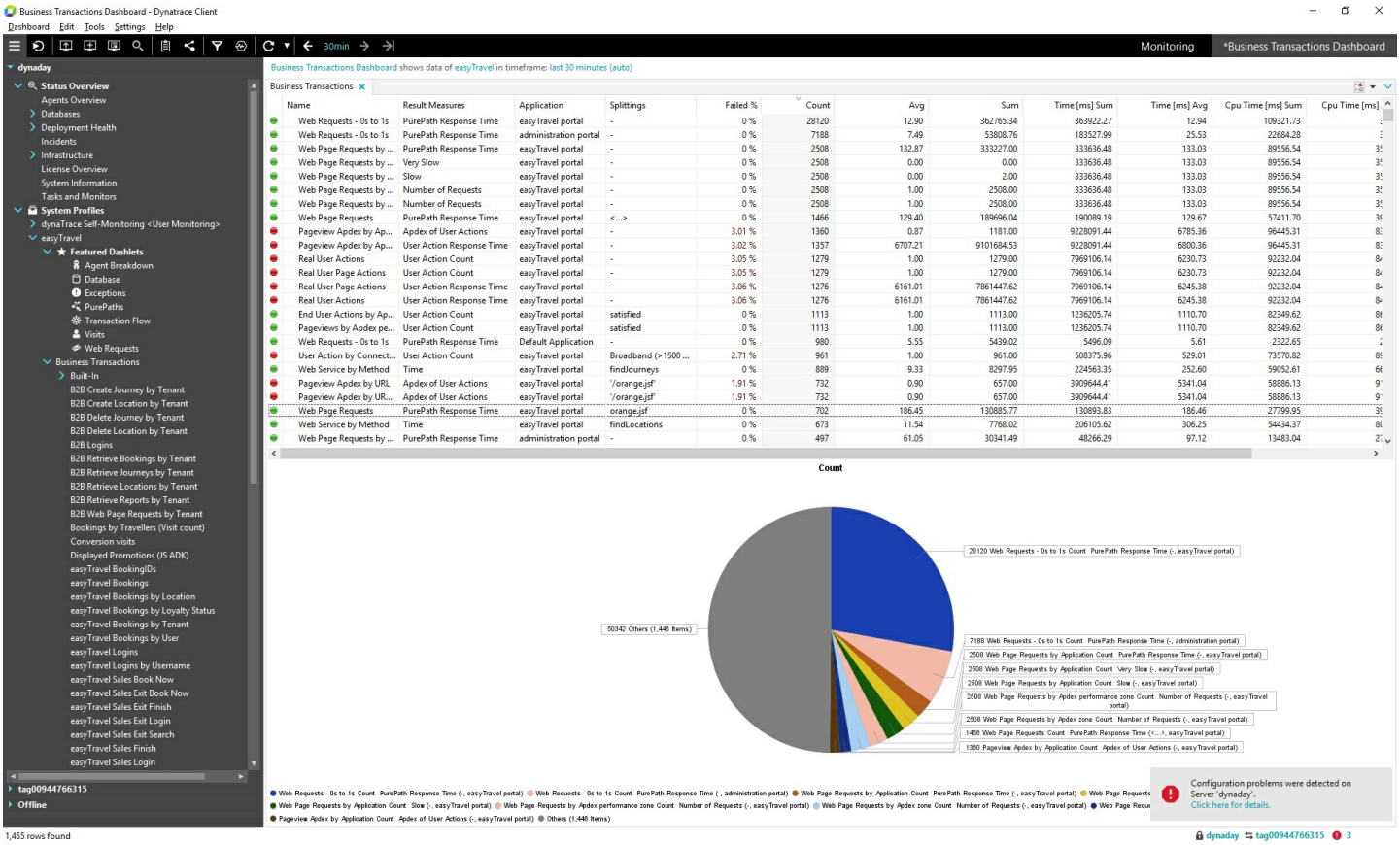
The **Business Transactions** dashlet displays all actions performed on the system under diagnosis that match a defined Business Transaction.

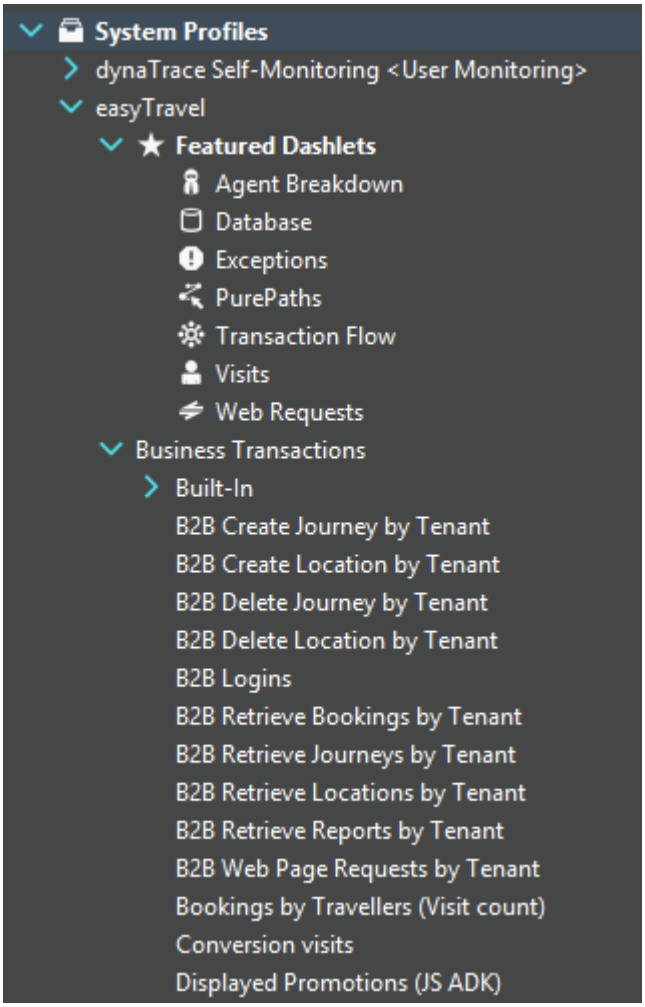
You can open the dashlet from the **Cockpit**:

- Double-click the parent **Business Transactions** node under a System Profile. All Business Transactions for the System Profile are displayed in the dashlet. This view is useful to check whether all the configured Business Transactions deliver data.

- Expand the **Business Transactions** and double-click the **Built-In** node to show all out-of-the-box Business Transactions in the dashlet.
- Double-click a Business Transaction to open a **Business Transactions** dashlet that displays data for the selected transaction.

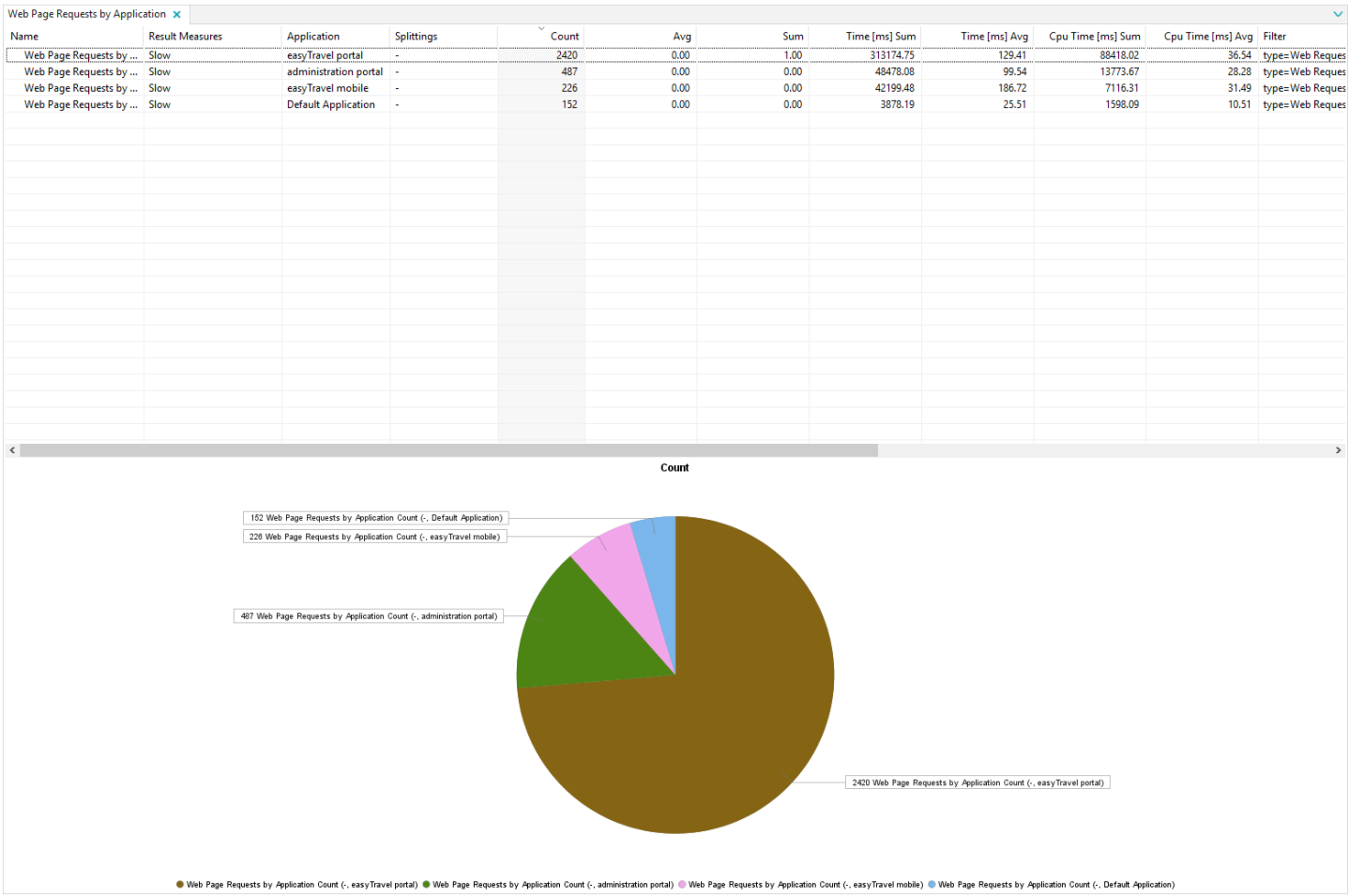
The example **Business Transactions** dashlet below shows the dashlet populated by double-clicking the parent **Business Transactions** node.





The pie chart in the dashlet visualizes the values of the currently selected column of the **Business Transactions** table. The figure shows the dashlet for the Business Transaction **Web Page Requests by URI**, with a chart for the **Count** values.

Example Business Transactions Dashlet



Example Business Transactions dashlet

Business Transaction metrics

The following columns can be displayed in the dashlet to provide metrics and other information for the Business Transactions.

- **State**: The error state of the Business Transaction, indicated by an icon. For example, a green circle indicates no errors have occurred.
- **Name**: Name of the Business Transaction.
- **Result Measures**: Measure(s) on which the Business Transaction is based. A table row is generated for each measure; a Business Transaction may be listed multiple times in the **Name** column, once for each of its measures.
- **Application**: The application for which the Business Transaction is defined.
- **Splittings**: Values received from grouping measures. Each occurrence of a different value generates a table row.
- **Failed %**: The percentage of transactions with errors.

- **Count**: Number of occurrences of the Business Transaction. For grouped Business Transactions, the number of occurrences of a specific group.
- **Min**: Minimum received value of measures used for the evaluate function of a Business Transaction. The default is 1.0 for grouping measures and 0.0 for filter measures.
- **Max**: Maximum received value of measures used for the evaluate function of a Business Transaction. The default is 1.0 for grouping measures and 0.0 for filter measures.
- **Avg**: Average of all received values of measures used for the evaluate function of a Business Transaction. The default is 1.0 for grouping measures and 0.0 for filter measures.
- **Sum**: Sum of all received values of measures used for the evaluate function of a Business Transaction. The default is the cardinality of a group for grouping measures, and 0.0 for filter measures.
- **Time [ms] Sum**: Total execution time for the PurePaths containing the Business Transaction.
- **Time [ms] Avg**: Average duration of all PurePaths containing the Business Transaction.
- **CPU Time [ms] Sum**: Total CPU time of all PurePaths containing the Business Transaction.
- **CPU Time [ms] Avg**: Average CPU time for all PurePaths containing the Business Transaction.
- **Filter**: The filter settings for filter transactions; blank for other transactions.
- **Time [ms] Min**: Execution time for the PurePath with the shortest duration.
- **Time [ms] Max**: Execution time for the PurePath with the longest duration.
- **CPU Time [ms] Min**: CPU time for the PurePath with the shortest duration.
- **CPU Time [ms] Max**: CPU time for the PurePath with the longest duration.

Not all columns are displayed by default. To select which columns to display, right-click a column head to display a context menu that lists the default columns. Select or clear the check boxes as needed. To add non-default columns to the table, click **More**.

Reanalyze all PurePaths

The **Reanalyze all PurePaths** command on the context menu for the **Business Transaction** dashlet analyzes the PurePaths and tags them with the Business Transaction, but it does not persist any measures back to the stored session.

You can right-click the stored session or offline session and select **Process Configuration Changes** to store changes in the session. This command only updates PurePath values in the Session files, system-monitoring values in the Performance Warehouse. For more information, see Usage of Sessions.

Components dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Context menu options

Sensor Configuration

Tuning options

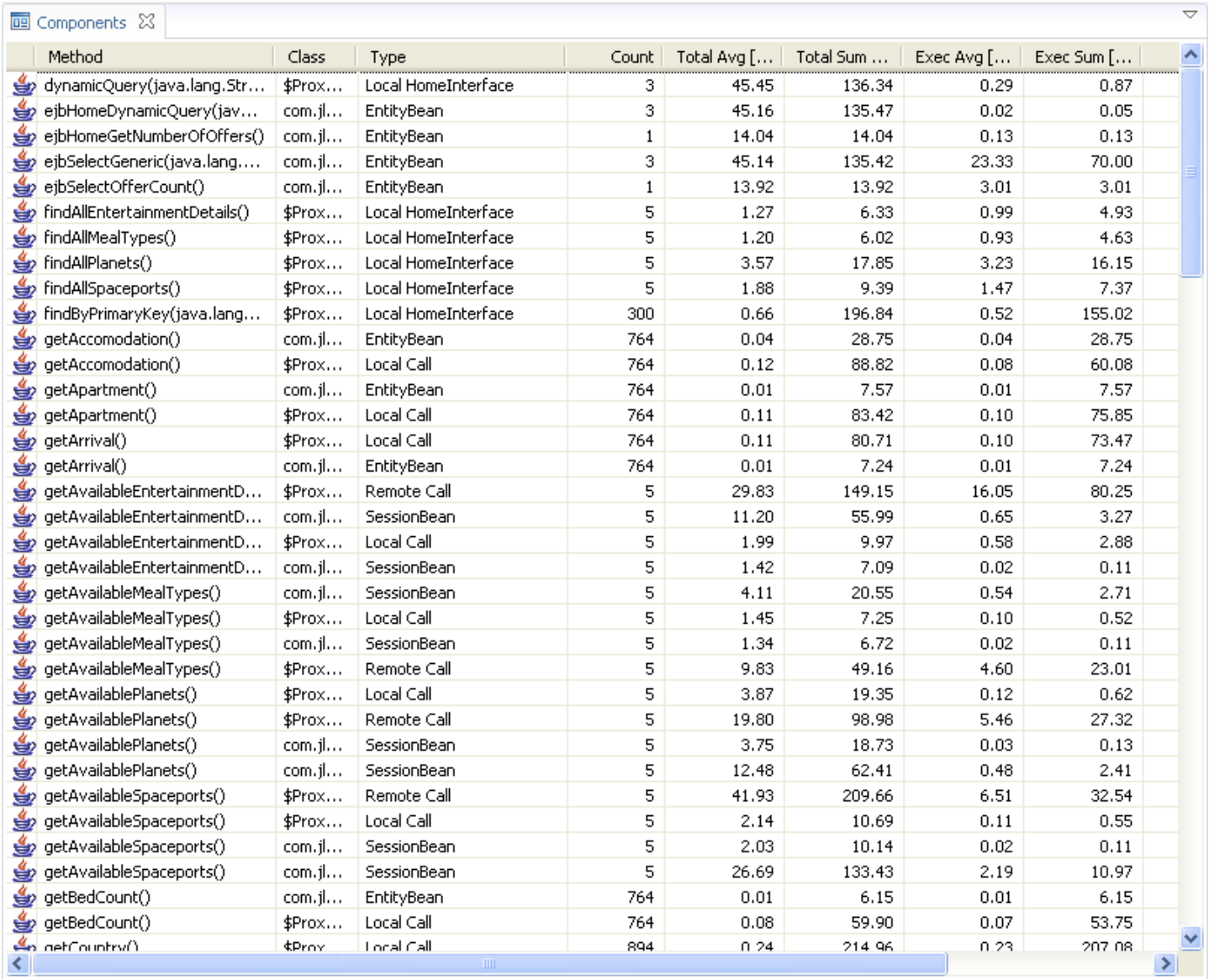
Related pages

Performance specific dashlets

Overview

To diagnose and fine-tune component management services like Enterprise Java Beans, AppMon uses dynamic bytecode instrumentation to capture performance metrics, including Java remoting metrics, across horizontal and vertical application layers. A central AppMon Server correlates the resulting measurements to PurePaths Dashlet.

The **Components** dashlet displays the collected metrics.



Method	Class	Type	Count	Total Avg [...	Total Sum ...	Exec Avg [...	Exec Sum [...
dynamicQuery(java.lang.Str...	$Prox...	Local HomeInterface	3	45.45	136.34	0.29	0.87
ejbHomeDynamicQuery(jav...	com.jl...	EntityBean	3	45.16	135.47	0.02	0.05
ejbHomeGetNumberOfOffers()	com.jl...	EntityBean	1	14.04	14.04	0.13	0.13
ejbSelectGeneric(java.lang....	com.jl...	EntityBean	3	45.14	135.42	23.33	70.00
ejbSelectOfferCount()	com.jl...	EntityBean	1	13.92	13.92	3.01	3.01
findAllEntertainmentDetails()	$Prox...	Local HomeInterface	5	1.27	6.33	0.99	4.93
findAllMealTypes()	$Prox...	Local HomeInterface	5	1.20	6.02	0.93	4.63
findAllPlanets()	$Prox...	Local HomeInterface	5	3.57	17.85	3.23	16.15
findAllSpaceports()	$Prox...	Local HomeInterface	5	1.88	9.39	1.47	7.37
findByPrimaryKey(java.lang...	$Prox...	Local HomeInterface	300	0.66	196.84	0.52	155.02
getAccomodation()	com.jl...	EntityBean	764	0.04	28.75	0.04	28.75
getAccomodation()	$Prox...	Local Call	764	0.12	88.82	0.08	60.08
getApartment()	com.jl...	EntityBean	764	0.01	7.57	0.01	7.57
getApartment()	$Prox...	Local Call	764	0.11	83.42	0.10	75.85
getArrival()	$Prox...	Local Call	764	0.11	80.71	0.10	73.47
getArrival()	com.jl...	EntityBean	764	0.01	7.24	0.01	7.24
getAvailableEntertainmentD...	$Prox...	Remote Call	5	29.83	149.15	16.05	80.25
getAvailableEntertainmentD...	com.jl...	SessionBean	5	11.20	55.99	0.65	3.27
getAvailableEntertainmentD...	$Prox...	Local Call	5	1.99	9.97	0.58	2.88
getAvailableEntertainmentD...	com.jl...	SessionBean	5	1.42	7.09	0.02	0.11
getAvailableMealTypes()	com.jl...	SessionBean	5	4.11	20.55	0.54	2.71
getAvailableMealTypes()	$Prox...	Local Call	5	1.45	7.25	0.10	0.52
getAvailableMealTypes()	com.jl...	SessionBean	5	1.34	6.72	0.02	0.11
getAvailableMealTypes()	$Prox...	Remote Call	5	9.83	49.16	4.60	23.01
getAvailablePlanets()	$Prox...	Local Call	5	3.87	19.35	0.12	0.62
getAvailablePlanets()	$Prox...	Remote Call	5	19.80	98.98	5.46	27.32
getAvailablePlanets()	com.jl...	SessionBean	5	3.75	18.73	0.03	0.13
getAvailablePlanets()	com.jl...	SessionBean	5	12.48	62.41	0.48	2.41
getAvailableSpaceports()	$Prox...	Remote Call	5	41.93	209.66	6.51	32.54
getAvailableSpaceports()	$Prox...	Local Call	5	2.14	10.69	0.11	0.55
getAvailableSpaceports()	com.jl...	SessionBean	5	2.03	10.14	0.02	0.11
getAvailableSpaceports()	com.jl...	SessionBean	5	26.69	133.43	2.19	10.97
getBedCount()	com.jl...	EntityBean	764	0.01	6.15	0.01	6.15
getBedCount()	$Prox...	Local Call	764	0.08	59.90	0.07	53.75
getCountry()	$Prox...	Local Call	894	0.24	214.96	0.23	207.08

Components dashlet

Metrics

The dashlet displays the following information:

- **Platform**: The software platform (Java or .NET) for the component.
- **Method**: The name of the called method.
- **Class**: The name of the class that implements the component interface.

- **Type**: The component type on which the method was called. Currently supported component types are:
  - Local Call: Local call to an Entity or SessionBean.
  - Remote Call: Remote call to an Entity or SessionBean.
  - HomeInterface: Call to a HomeInterface.
  - Local HomeInterface: Call to a local HomeInterface.
  - SessionBean: Server-side execution of a SessionBean method.
  - MessageDrivenBean: Server-side execution of a MessageDrivenBean method.
- **Count**: The number of times the particular component method has been called or executed.
- **Total Avg [ms]**: The average time, in milliseconds, for all executions of this component method.
- **Total Min [ms]**: The minimum time, in milliseconds, for an execution of this component method.
- **Total Max [ms]**: The maximum time, in milliseconds, for an execution of this component method.
- **Total Sum [ms]**: The accumulated time, in milliseconds, for all executions of this component method.
- **Exec Avg [ms] –** The average time, in milliseconds, for all executions of this component method, excluding time spent in nested method calls.
- **Exec Min [ms]**: The minimum time, in milliseconds, for an execution of this component method, excluding time spent in nested method calls.
- **Exec Max [ms]**: The maximum time, in milliseconds, for an execution of this component method, excluding time spent in nested method calls.
- **Exec Sum [ms]**: The accumulated time, in milliseconds, for all executions of this component method, excluding time spent in nested method calls.
- **CPU Avg [ms]**: The average CPU time, in milliseconds, for all executions of this component method.
- **CPU Min [ms]**: The minimal CPU time, in milliseconds, for an execution of this component method.
- **CPU Max [ms]**: The maximum CPU time, in milliseconds, for an execution of this component method.
- **CPU Sum [ms]**: The accumulated CPU time, in milliseconds, for all executions of this component method.

Context menu options

- **Subscribe Measures**: Open the **Measure Picker** dialog box to add measures to a subscription. This option is only available during realtime sessions.
- **New Business Transaction**: Open the **Business Transaction** dialog box to create a new Business Transaction based on the selection related measures. This option is only available during Realtime sessions.
- **Source Lookup**: Please see Source Lookup Feature for a detailed description.

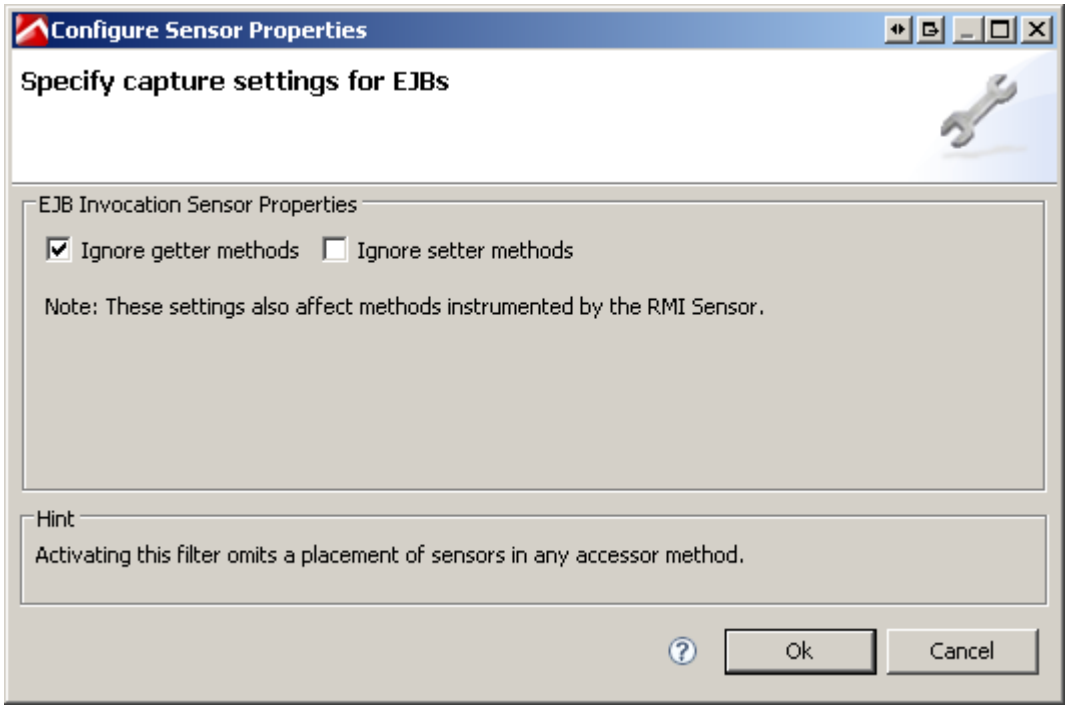
- **Add Sensor Rules** (only available on live sessions)
  - Refine with Callee Methods: All Methods that are called by the Selected Methods are retrieved and can be used as input for new Inclusion Sensor Rules.
  - Exclude Selected Methods: All Selected Methods can be used as input for new Exclusion Sensor Rules.
  - Include Selected Methods: All Selected Methods can be used as input for new Inclusion Sensor Rules.

Sensor Configuration

Use the **Configure Sensor Properties** dialog box for the EJB Invocation Sensor to specify whether to instrument plain setter and getter methods of components. By default, these methods are instrumented. Select the options if you do not want them instrumented. Methods instrumented by the RMI Sensor are also affected by these settings, because EJB invocations are often related to RMI calls.

To access the dialog box:

1. In the **System Profile Preferences** dialog box, expand an Agent Group in which the EJB Invocation Sensor Pack is placed, and select **Sensor Configuration**.
2. Click **Properties** for the EJB Invocation Sensor Pack.



Specify capture settings for EJBs

Tuning options

Follow these guidelines to tune performance based on the metrics described above.

- Use local calls. When caller and callee are located in the same JVM, access EJBs via local calls to bypass stubs and skeletons (actually the whole RMI), and security checks.
- Use value objects. To reduce network overhead, avoid single remote calls by using the Value Object Pattern.
- Use stateless session beans.
- Remove unneeded stateful session beans. Removing these objects avoids a passive state for these beans, and the attendant disk operations.
- Cache EJB references. To avoid a JNDI lookup for every request, cache EJB references in servlets.
- Cache home interfaces. Because repeated lookups to a home interface can be expensive, cache references to EJBHomes in the `init()` methods of servlets.
- Cache EJB resources. Use `setSessionContext()` or `ejbCreate()` to cache bean resources. This is an example of using bean lifecycle methods to perform application actions only once where possible. Remember to release acquired resources via the `ejbRemove()` method.
- Explicitly call `remove()`. Allow stateful session EJBs to be removed from the container cache by explicitly calling of the `remove()` method in the client.
- Tune the entity EJB´s pool size. Entity beans use both the EJB pool and cache settings. Tune the entity EJB´s pool size to minimize the creation and destruction of beans. Populating the pool with a non-zero steady size is useful for getting better response for initial requests.
- Cache bean-specific resources. Use the `setEntityContext()` method to cache bean-specific resources. Release them using the `unSetEntityContext()` method.
- Load related data efficiently for container-managed relationships (CMRs).
- Configure read-only entity beans for read-only operations.
- Use container-managed transactions. These transactions are preferred for consistency, and provide better performance.
- Don’t encompass user input time. To avoid resources being held unnecessarily for long periods, a transaction should not encompass user input or user think time.
- Identify non-transactional methods. Declare non-transactional methods of session EJBs with `NotSupported` or `Never` transaction attributes. These attributes can be found in the `ejb-jar.xml` deployment descriptor file. Transactions should span the minimum time possible, because they lock database rows.
- For very large transaction chains, use the transaction attribute `TX_REQUIRED`. To ensure EJB methods in a call chain, use the same transaction.
- Use the lowest-cost database locking that is consistent with any transaction. Commit the data after the transaction completes rather than after each method call.

- Use XA-capable data sources only when two or more data sources are going to be involved in a transaction.

Database dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Logical views

History drilldown

Sensor Configuration

Analyze bind values

Dashlet display options

Connection pool information

Tuning guidelines

Related pages

Performance specific dashlets

Overview

The **Database** dashlet (opened from the **Cockpits** sidebar) or the **Database** drill-down from the **Transaction Flow** of the **Monitoring Overview** provide an overview and in-depth information for database connection pool usage and SQL statement executions.

Logical views

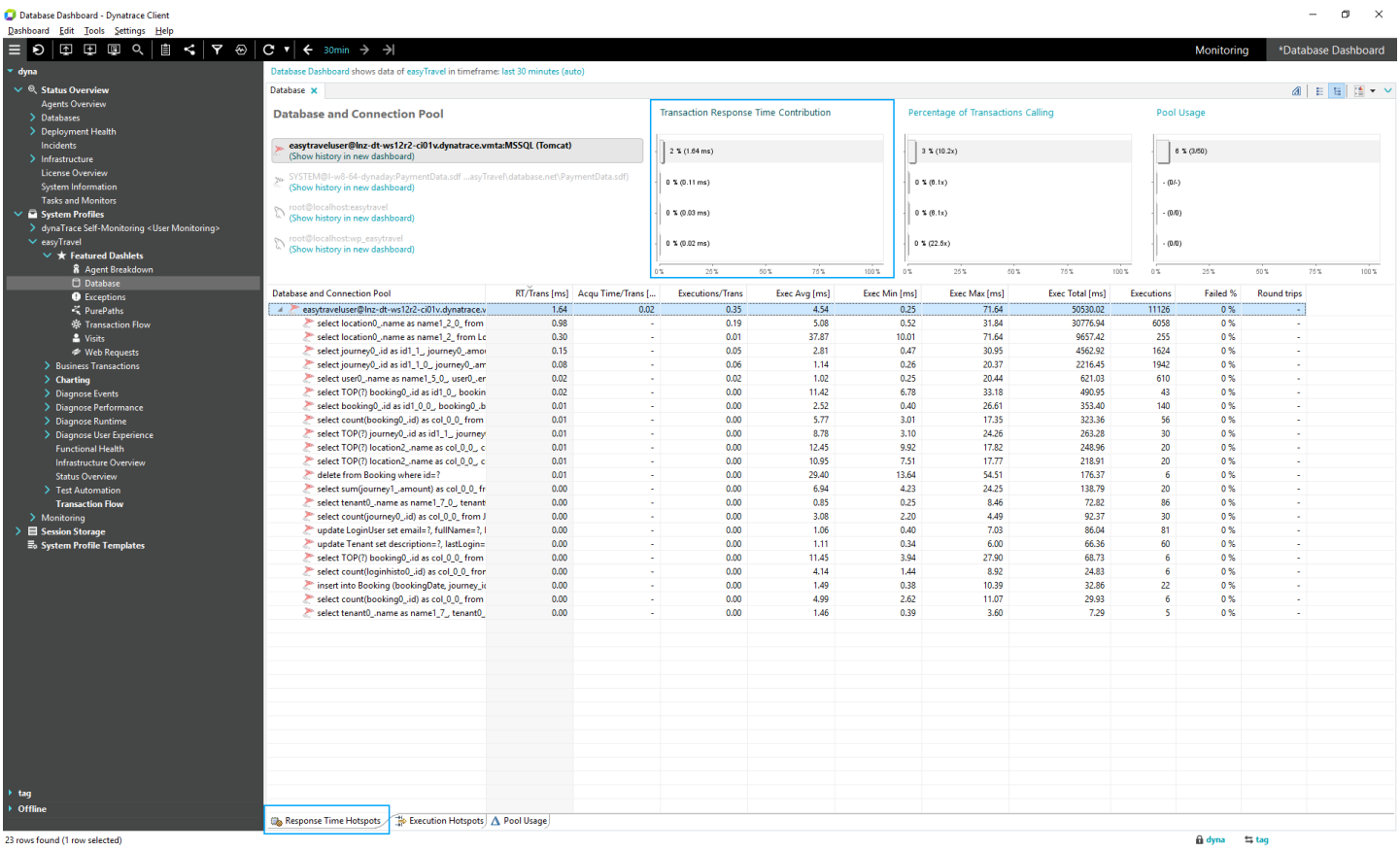
The dashlet is divided into three logical views accessible either by clicking on a chart header label at the top of the dashlet, or a table tab at the bottom:

- **Response Time Hotspots** tab – Transaction Response Time Construction chart.
- **Execution Hotspots** tab – **Percentage of Transactions Calling** chart.
- **Pool Usage** tab – **Pool Usage** chart.

Each tab includes a table that provides statistics for the executions. You can select which statistics to display by right-clicking any column head to display the context menu and selecting or clearing the columns.

Response time HotSpots tab

The **Transaction Response Time Contribution** chart visualizes the contribution of database transactions to the average transaction response time. For example, for the transactions charted in the figure, connection acquisition and query execution result in 1.64ms response time per transaction, and database time contributes on average 2% to transaction time.



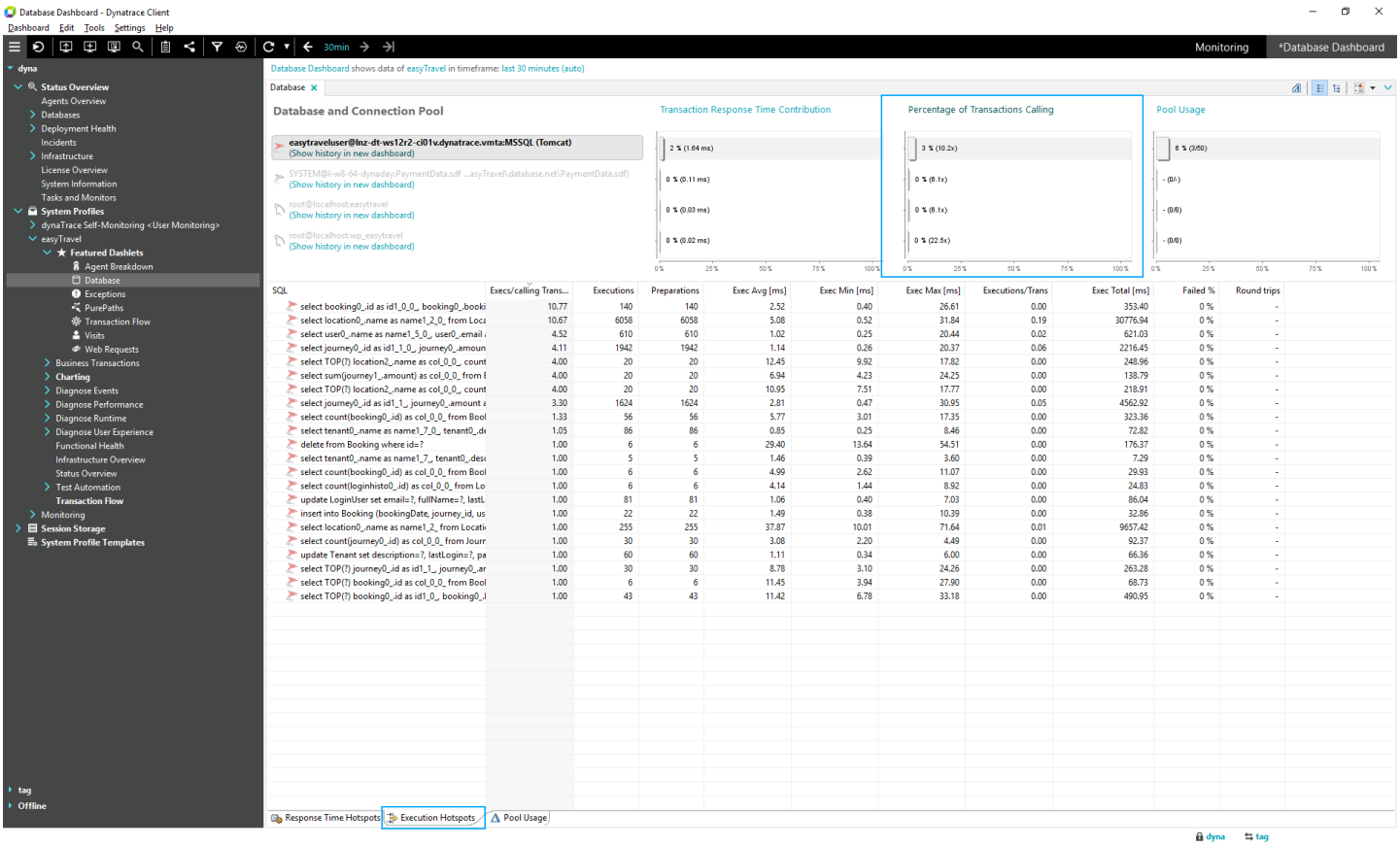
Response Time Hotspots Tab

The table below the chart displays statistics for each database and connection pool transaction. The following columns are displayed by default.

Column	Description
Database and Connection Pool	Database connection, identified by the connection properties and connection pool name.
Bind Values	The bind values for the SQL statement. This column appears if the analysis of bind values is enabled as described below.
RT/Trans	Average response time per transaction, including times for connection acquisition, query preparation, and execution.
Acqu Time/Trans	Connection acquisition time per transaction.
Executions/Trans	Number of query executions per transaction.
Exec Avg	Average execution time of the transactions involved in the execution.
Exec Min	Minimum execution time of the transactions involved in the execution.
Exec Max	Maximum execution time of the transactions involved in the execution.
Exec Total	Total execution time for all transactions.
Executions	Total number of executions.
Failed %	Percentage of failing database calls.
Round Trips	Number of round trips made to the database (call to and response from the database) during execution.

Execution HotSpots tab

The **Percentage of Transactions Calling** chart visualizes the percentage of transactions that are calling each statement or connection.
In the following figure, 3% of transactions are associated with the first connection pool.



Execution HotSpots tab

The table below the chart displays statistics for the SQL execution. The following columns are displayed by default.

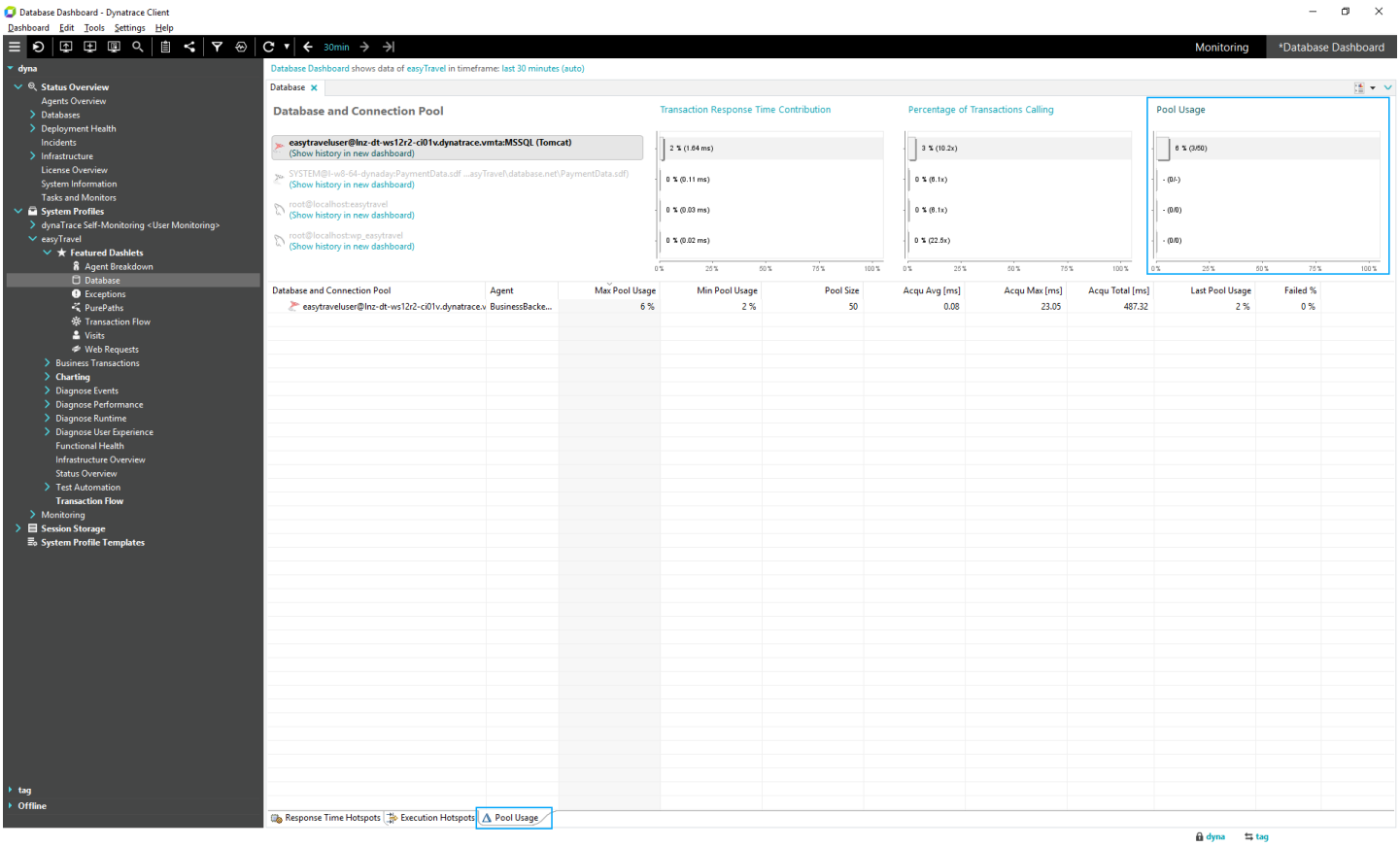
Column	Description
SQL	The SQL statement.
Bind Values	The bind values for the SQL statement. This column appears if the analysis of bind values is enabled as described below.
Execs/calling Transaction	Number of query executions per transaction that is calling a statement or connection.
Executions	Total number of executions.
Preparations	Number of `prepareStatement` calls.
Exec Avg	Average execution time of the transactions involved in the execution.

Exec Min	Minimum execution time of the transactions involved in the execution.
Exec Max	Maximum execution time of the transactions involved in the execution.
Executions/Trans	Number of query executions per transaction.
Exec Total	Total execution time for all transactions.
Failed %	Percentage of failing database calls.
Round Trips	Number of round trips made to the database (call to and response from the database) during execution.

Pool Usage tab

The pool usage chart visualizes the maximum pool usage.

In the figure below, 3 out of 50 database pool connections are in use, which is 6% usage.



Pool Usage tab

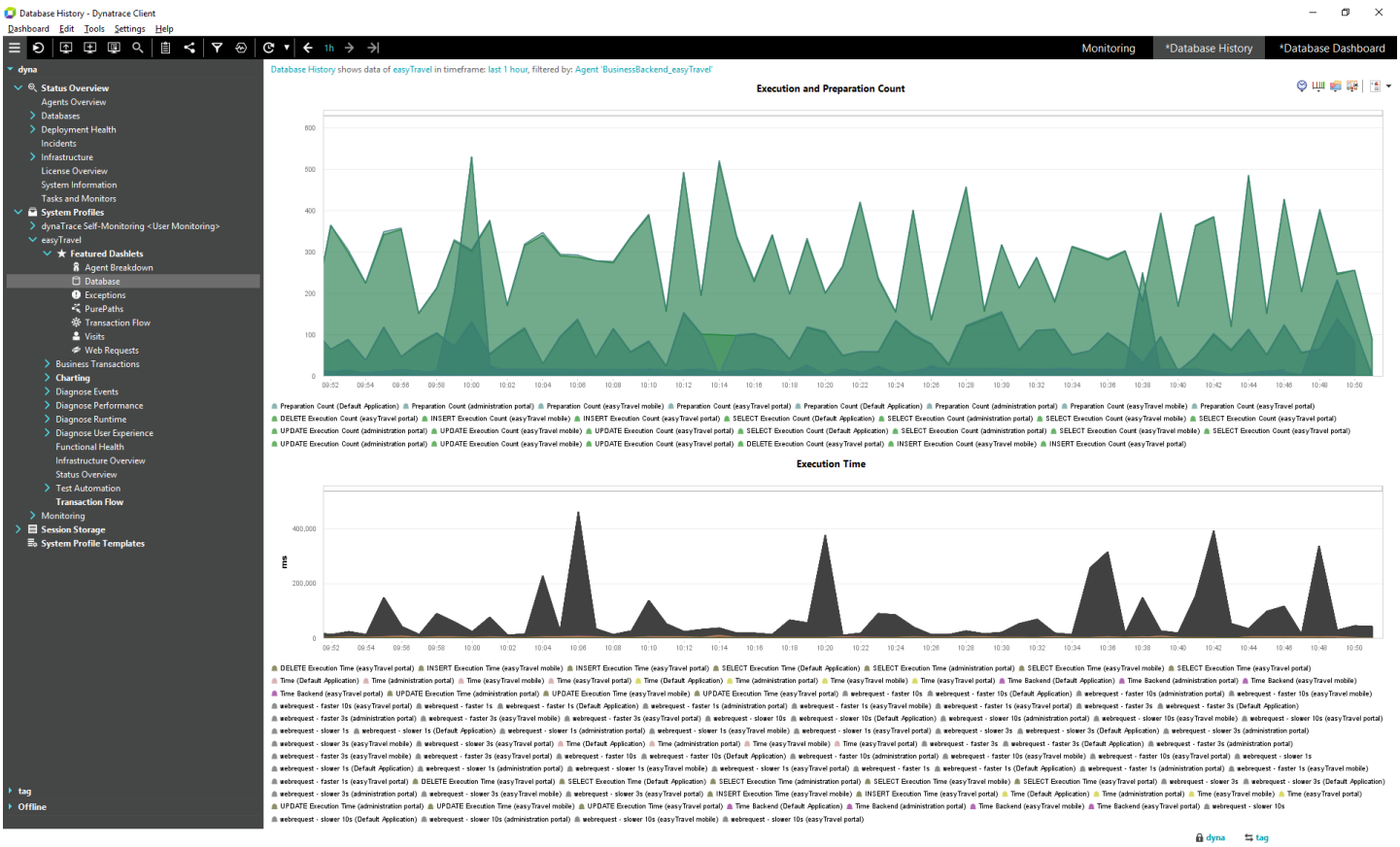
The table below the chart displays statistics for the pool usage. The following columns are displayed by default.

Column	Description
Database and Connection Pool	Database connection, identified by the connection properties and connection pool name.
Agent	The Agent name.
Max Pool Usage	Maximum pool usage.
Min Pool Usage	Minimum pool usage.
Pool Size	The configured maximum number of connections the connection pool will use.
Acqu Avg	Average connection acquisition time per `getConnection` call.

Acqu Max	Maximum connection acquisition time per `getConnection` call.
Acqu Total	Total connection acquisition time for all connections.
Last Pool Usage	Last pool usage.
Failed %	Percentage of failing database calls.

History drilldown

Each connection pool in the chart legend provides a **Show History** link. Click this link to drill down to a dashboard that displays database execution time and execution count information.



Database History dashboard

Detailed dashboards are available for the following application servers:

- WebSphere
- WebLogic
- JBoss

WebSphere

The following metrics are charted for WebSphere, as shown in the figure.

- Pool Usage
- Connection Acquisition
- Wait Time
- Thread Wait Count
- Connection Usage Time
- Prepared statement cache discard rate
- Create vs Close Count

WebLogic

The following metrics are charted for WebLogic:

- Connection Acquisition
- Pool Usage
- Delay Time
- Thread Wait Count
- Prepared statement cache discard rate
- Failures to Reconnect Count
- Created Connections
- Leaked Connections

JBoss

The following metrics are charted for JBoss:

- Pool Usage
- Connection Usage

Sensor Configuration

Use the **Configure Sensor Properties** dialog box for the JDBC Sensor or ADO.NET Sensor to enable or disable aggregation, select whether bind values should be captured, and set the maximum text size of captured SQL commands. Commands that exceed the size limit are truncated.

Analyze bind values

Bind values can be shown on / apply to the **Response Time Hotspots** and **Execution Hotspots** sub-tabs (tabs below the SQL statement list) of the **Database dashlet**, but not the Pool Usage sub-tab. Also, the more recent approach is the **Database** drill-down from the **Monitoring Overview**.

You can show bind values by:

- Selecting the **Show Bind Values** icon.
  
  in the dashlet toolbar.
- Selecting **Show Bind Values** from the dashlet's context menu.
- An alternative would be to right-click on the column-header, select **More** and check the Bind Values column to be shown.
- To show **and group by** bind values right-click in the SQL statement body and select **Group by > SQL Statement and Bind Values**.

If you see asterisks instead of certain bind values in the Bind values column, **Settings menu > Dynatrace Server… > Settings vertical tab > Confidential Strings** is turned on for that field.

Also, if transaction aggregation (same SQL statement called multiple times only listed once with the number of calls) is turned on in the (JDBC or ADO) sensor this is mutually exclusive with bind value display.

Dashlet display options

Filtering

Click an entry in the connection pool chart legend or click a bar graph to filter the table by the selected connection pool. To remove the filter, click the selected connection pool again.

Table structure modes

To select the structure mode, click the toolbar icon or right-click the dashlet and select the setting from the **Structure Mode** cascading menu:

- **Flat**: Only SQL statement records are shown in table. This mode is useful to sort and compare SQL statements across various database connection pools.
- **Connection Pool**: SQL records are grouped by their respective database connection pool. Click the small triangle to the right of a connection pool to expand it to display the SQL statements associated with it.

Connection pool information

JDBC

Application Server	Support Level
WebSphere 6 or later, WebLogic 8 or later, JBoss 3 or later	Full
Glass Fish 3.1 Open Source Edition	Basic
Interstage 8 or later, Cosminexus 8 or later	Not Required
OC4J 10g	Not Available

PMI

In some environments (like IBM WebSphere) you have to enable PMI (Performance Monitoring Infrastructure) to get JDBC pool metrics.

.NET data

Database	Support Level	DB Type	Host, Username, Password	Number of Connections Used	Pool Size
Oracle, SQL Server	Full	✔	✔	✔	Depends on the connection string.

SQL Server CE	Standard	✓	✓	Depends on the connection string.	Depends on the connection string.
ODBC, OLE DB	Basic	✓	Depends on the connection string.	Depends on the connection string.	Depends on the connection string.

Tuning guidelines

Follow these guidelines to tune database performance based on the metrics described above.

- Use a Type1 driver (JDBC-ODBC bridge) only if you do not have a driver for your database.
- For two-tier applications that communicate from a Java client to a database, use a Type2 driver, which gives better performance than a Type1 driver.
- If your system communicates from the client to the database via a middleware server, use a Type3 driver, which gives better performance than Type1 and Type2 drivers.
- Use a Type4 driver for two-tier and three-tier applications.
- Use `fetchDirection` or `fetchSize` if your driver and database support them.
- Use a connection pool to cache database connections,
- Use batch transactions.
- Use the suitable isolation level for your requirement, for example `TRANSACTION_READ_UNCOMMITED` for concurrent transaction-based applications or `TRANSACTION_NONE` for nonconcurrent transaction-based applications.
- If a statement is executed more than once, use prepared statements.
- Implement stored procedures to get a result from complex and multiple statements for a single request.
- Use the batch update facility available in `Statements`.
- Use the batch retrieval facility available in `Statements` or `ResultSet`.
- Set up the proper direction for processing rows.
- Use proper getter and setter methods.
- Close `ResultSets`, `Statements`, and `Connections` as soon as the application does not need them anymore.
- Write precise SQL queries.
- Cache read-only and read-mostly table data.
- If you retrieve a large amount of data, fetch small amounts iteratively rather than the whole amount of data at once.

Database Hotspots dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Database Hotspots table

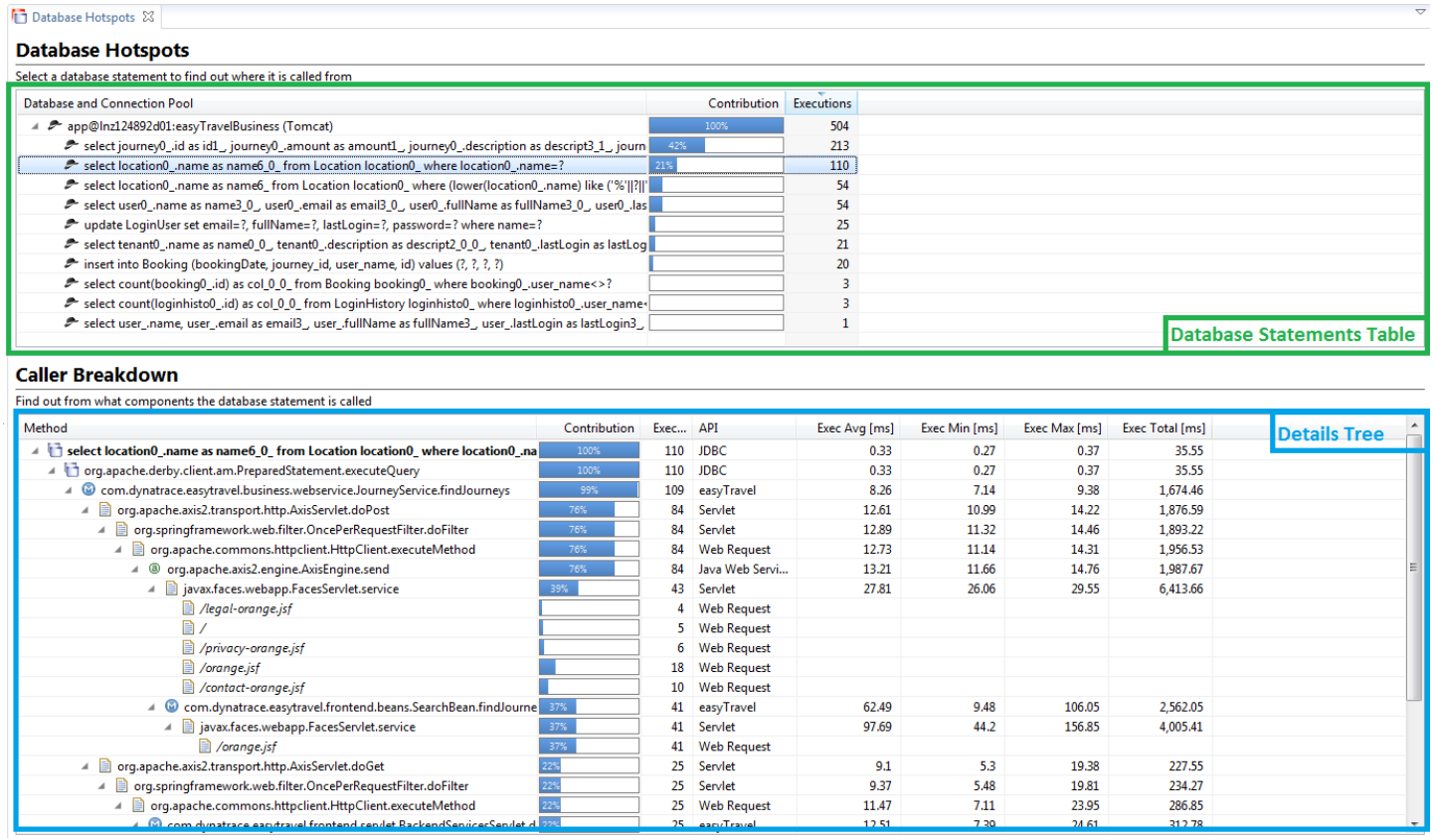
Caller breakdown tree

Related pages

Performance specific dashlets

Overview

The **Database Hotspots** dashlet provides breakdown information about database statements. You can use the dashlet to find out where a database statement is called.



Database Hotspots dashlet

The dashlet consists of the **Database Hotspots** table and the **Caller Breakdown** tree.

Database Hotspots table

The **Database Hotspots** table lists all database statement executions, grouped by their connection pool.

Column	Description
Database and Connection Pool	A tree that consists of two levels. The first level shows the connection pools and the second level lists the pool's database statement executions.
Contribution	The percentage of statement executions contributed by each connection pool and statement.
Executions	The number of statement executions.

Caller breakdown tree

Select a database statement in the **Database Hotspots** table to display a detailed caller breakdown tree of the selected statement.

The selected database statement is displayed at the top of the tree. The executers are listed in the hierarchy below the statement. Because the caller breakdown is an aggregation of many paths, it is likely that the database statement is executed by many different call paths. The tree therefore consists of branches that are weighted according to the breakdown time contribution of each tree branch.

Column	Description
Method	The selected database statement, method, or web request.
Contribution	The percentage contribution of the (sub)tree to the overall database statement execution count.
Executions	The execution count of the database statement, method, or web request.
API	The most significant API of the database statement, method, or web request.
Exec Acg	The average execution time per execution.
Exec Min	The minimum execution time per execution.
Exec Max	The maximum execution time per execution.
Exec Total	The total execution time.

Entry Points dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

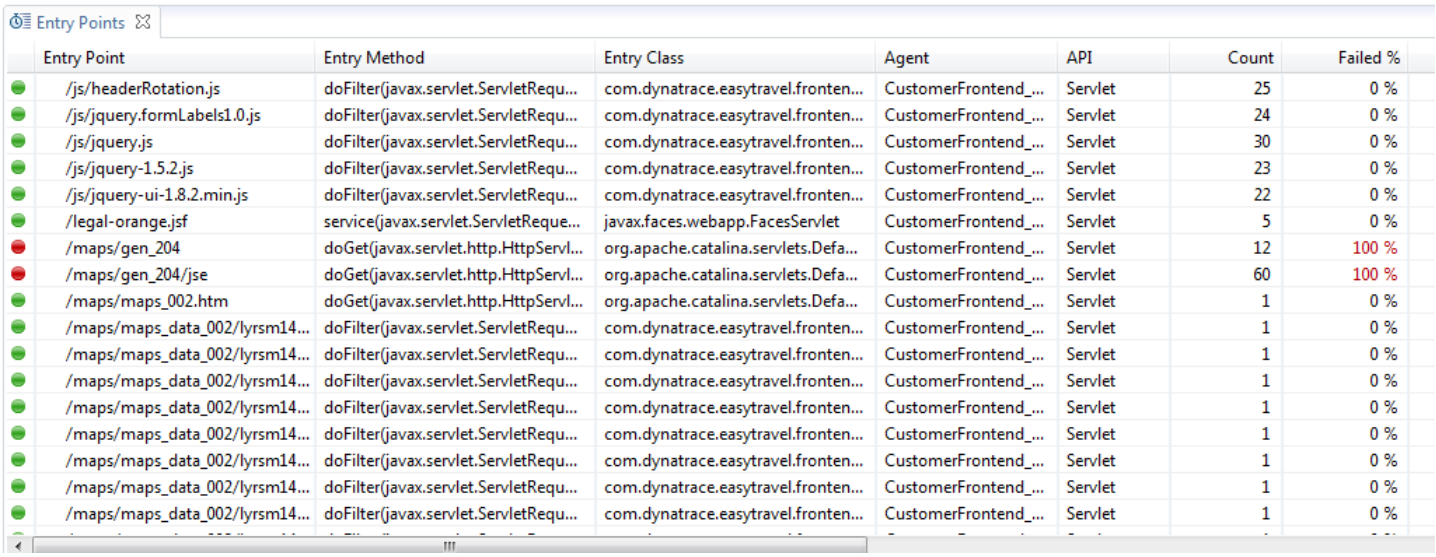
Aggregation options

Related pages

Performance specific dashlets

Overview

The **Entry Points** dashlet shows an aggregated representation of PurePaths Dashlet that started at the same entry point. It provides an overview of different types of PurePaths as well as additional statistical data such as the number of PurePaths started at a specific entry point and the average, minimum, maximum, and sum of the execution time spent by all PurePaths started at a specific entry point. You can use this information to associate a performance issue with a group of PurePaths and then to a specific PurePath instance.



Entry Points dashlet

Metrics

The table lists the following information:

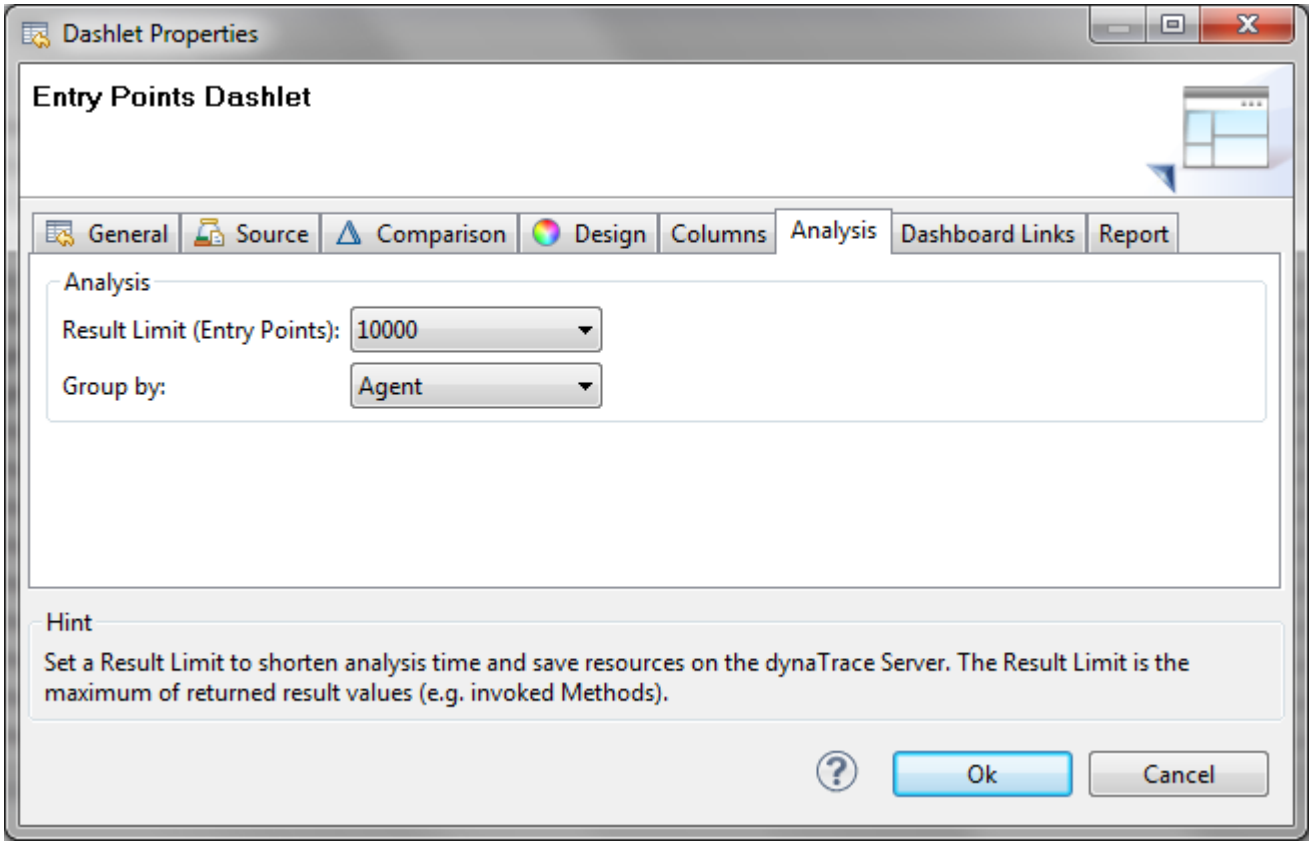
- **Error State**: The first, unlabeled column displays icons to indicate the error states of the underlying PurePaths, e.g. Successful, Failed Transaction Error.
- **Entry Point**: A string representing the PurePath entry point.
- **Entry Method**: The entry method where the PurePath starts.
- **Entry Class**: The entry class where the PurePath starts.
- **Agent**: The ID of the AppMon Agent that recorded the execution of the entry point, including the Agent name and the host name for the Agent. Only available if the entry points are grouped by Agent. See Aggregation Options, below.
- **Count**: The number times an execution of this entry point was recorded.
- **Failed %**: The percentage of failed transactions detected for this entry point.
- **Total Avg [ms]**: Average duration of the execution of all PurePaths started at this entry point.
- **Total Min [ms]**: Minimum duration of the execution of a PurePath started at this entry point.
- **Total Max [ms]**: Maximum duration of the execution of a PurePath started at this entry point.
- **Total Sum [ms]**: Accumulated duration of the execution of all PurePaths started at this entry point.
- **CPU Total Avg [ms]**: Average CPU time spent for the execution of all PurePaths started at this entry point.
- **CPU Total Min [ms]**: Minimum CPU time spent for the execution of a PurePath started at this entry point.

- **CPU Total Max [ms]**: Maximum CPU time spent for the execution of a PurePath started at this entry point.
- **CPU Total Sum [ms]**: Accumulated CPU time spent for the execution of all PurePaths started at this entry point.
- **Total Sync Avg [ms]**: Average synchronization time spent for the execution of all PurePaths started at this entry point.
- **Total Sync Min [ms]**: Minimum synchronization time spent for the execution of a PurePath started at this entry point.
- **Total Sync Max [ms]**: Maximum synchronization time spent for the execution of a PurePath started at this entry point.
- **Total Sync Sum [ms]**: Accumulated synchronization time spent for the execution of all PurePaths started at this entry point.
- **Total Wait Avg [ms]**: Average wait time spent for the execution of all PurePaths started at this entry point.
- **Total Wait Min [ms]**: Minimum wait time spent for the execution of a PurePath started at this entry point.
- **Total Wait Max [ms]**: Maximum wait time spent for the execution of a PurePath started at this entry point.
- **Total Wait Sum [ms]**: Accumulated wait time spent for the execution of all PurePaths started at this entry point.

Aggregation options

Use the **Analysis** tab in the **Dashlet Properties** dialog box to select the aggregation option:

- **Agent**: Group entry points by Agent instance IDs, which include the AppMon Agent name and host name. Recorded PurePath entries that differ by the host name of the recording AppMon Agent are added to distinct groups, providing a more fine grained aggregation.
- **Tier**: Group entry points by the AppMon Agent name, disregarding the host name. PurePath entries recorded from different AppMon Agents that have the same name are added to the same group, even if they are connected to different hosts. For example, a PurePath from `myAgent@myHost:1` are aggregated with the PurePath from `myAgent@anotherHost:2`.



Analysis tab

Memory Allocations dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Related pages

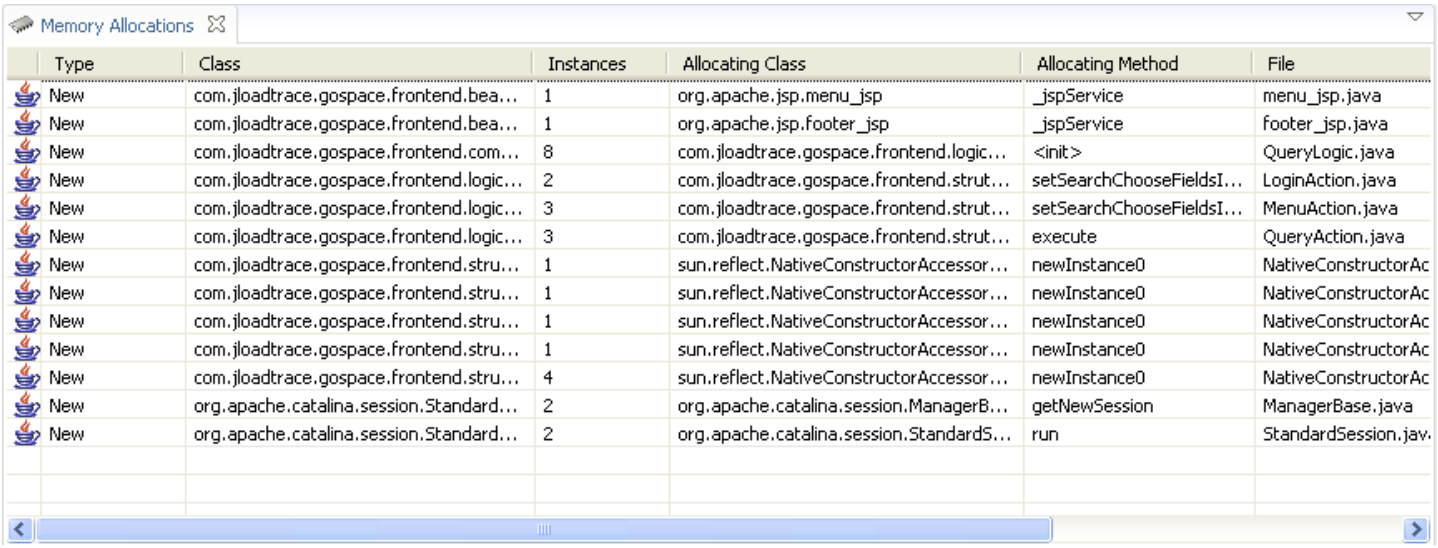
Performance specific dashlets

Overview

The **Memory Allocations** dashlet lists the memory allocation events that occurred within PurePaths. Specifically, the dashlet lists all instantiations of objects matching a Memory Sensor rule that occurred within a method call that is part of a PurePath.

Note

To get information about memory consumption independent from PurePaths, use the Total Memory Dashlet dashlet.



Type	Class	Instances	Allocating Class	Allocating Method	File
New	com.jloadtrace.gospace.frontend.bea...	1	org.apache.jsp.menu_jsp	_jspService	menu_jsp.java
New	com.jloadtrace.gospace.frontend.bea...	1	org.apache.jsp.footer_jsp	_jspService	footer_jsp.java
New	com.jloadtrace.gospace.frontend.com...	8	com.jloadtrace.gospace.frontend.logic...	<init>	QueryLogic.java
New	com.jloadtrace.gospace.frontend.logic...	2	com.jloadtrace.gospace.frontend.strut...	setSearchChooseFieldsI...	LoginAction.java
New	com.jloadtrace.gospace.frontend.logic...	3	com.jloadtrace.gospace.frontend.strut...	setSearchChooseFieldsI...	MenuAction.java
New	com.jloadtrace.gospace.frontend.logic...	3	com.jloadtrace.gospace.frontend.strut...	execute	QueryAction.java
New	com.jloadtrace.gospace.frontend.stru...	1	sun.reflect.NativeConstructorAccessor...	newInstance0	NativeConstructorAc
New	com.jloadtrace.gospace.frontend.stru...	1	sun.reflect.NativeConstructorAccessor...	newInstance0	NativeConstructorAc
New	com.jloadtrace.gospace.frontend.stru...	1	sun.reflect.NativeConstructorAccessor...	newInstance0	NativeConstructorAc
New	com.jloadtrace.gospace.frontend.stru...	1	sun.reflect.NativeConstructorAccessor...	newInstance0	NativeConstructorAc
New	com.jloadtrace.gospace.frontend.stru...	4	sun.reflect.NativeConstructorAccessor...	newInstance0	NativeConstructorAc
New	org.apache.catalina.session.Standard...	2	org.apache.catalina.session.ManagerB...	getNewSession	ManagerBase.java
New	org.apache.catalina.session.Standard...	2	org.apache.catalina.session.StandardS...	run	StandardSession.jav

Memory Allocations dashlet

Metrics

The following metrics and other information are included in the table:

- **Platform**: The first, unlabeled column indicates whether the memory allocation took place on a Java or .NET platform.
- **Type**: The object type.
- **Class**: The class of the new object instance.
- **Instances**: The number of instances of a specific class created by one allocating method of one allocating class within the perceived set of PurePaths.
- **Allocating Class**: The class of the object that generated the new object instance.
- **Allocating Method**: The method in which the object instance was created.
- **File**: The name of the source file of the allocating class, if available.
- **Line**: The line number within the source file where the allocation is performed. If allocation takes place in a native method, a dash (-) displays in this column.
- **Agent**: The AppMon Agent that registered the allocation.

Messaging dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Context menu options

Sensor Configuration

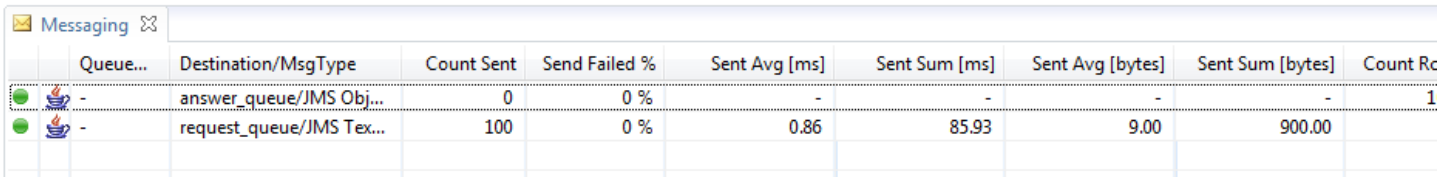
Tuning options

Related pages

Performance specific dashlets

Overview

The **Messaging** dashlet aggregates messages (JMS, MSMQ, Tibco, etc.) by destination and message type. The destination is the name of the queue or topic that a message has been sent to or received from.



Messaging dashlet

Metrics

The following metrics are listed in the Messaging dashlet:

- **Error State**: The first, unlabeled column displays icons that indicate the error state of the underlying PurePath (Successful, Failed Transaction, Error).
- **Platform**: The second, unlabeled column displays icons that indicate whether the message was transmitted on a Java or .NET platform.
- **QueueManager**: The name of the Queue Manager. Available only for WebSphere MQ and MQ Series.
- **Destination/MsgType**: The destination and message type of the sent or received message. Each row shows messages grouped by destination and message type.
- **Count Sent**: The number of times a message with the particular message type has been sent to this particular destination.
- **Sent Failed %**: The percentage of failures detected when sending a message from this destination with this message type.
- **Sent Avg [ms]**: The average time, in milliseconds, spent to send a message from this destination with this message type.
- **Sent Min [ms]**: The minimum time, in milliseconds, spent to send a message from this destination with this message type.
- **Sent Max [ms]**: The maximum time, in milliseconds, spent to send a message from this destination with this message type.
- **Sent Sum [ms]**: The accumulated time, in milliseconds, spent to send all messages from this destination with this message type.
- **Sent Avg [bytes]**: The average number of bytes required to send a message from this destination with this message type.
- **Sent Min [bytes]**: The minimum number of bytes used to send a message from this destination with this message type.
- **Sent Max [bytes]**: The maximum number of bytes used to send a message from this destination with this message type.
- **Sent Sum [bytes]**: The accumulated number of bytes used to send messages from this destination with this message type.
- **Count Rcvd**: The number of times this destination has received a message with this message type.
- **Rcvd Failed %**: The percentage of failures detected when this destination has received a message with this message type.
- **Rcvd Avg [ms]**: The average time, in milliseconds, spent to receive messages with this message type by this destination.
- **Rcvd Min [ms]**: The minimum time, in milliseconds, spent to receive a message with this message type by this destination.
- **Rcvd Max [ms]**: The maximum time, in milliseconds, spent to receive a message with this message type by this destination.

- **Rcvd Sum [ms]**: The accumulated time, in milliseconds, spent to receive messages with this message type by this destination.
- **Rcvd Avg [bytes]**: The average number of bytes used to receive a message with this message type by this destination.
- **Rcvd Min [bytes]**: The minimum number of bytes used to receive a message with this message type by this destination.
- **Rcvd Max [bytes]**: The maximum number of bytes used to receive a message with this message type by this destination.
- **Rcvd Sum [bytes]**: The accumulated number of bytes used to receive messages with this message type by this destination.
- **Count Peeked**: The number of times this destination has peeked a message with this message type.
- **Peek Failed %**: The percentage of failures detected when peeking at messages with this message type by this destination.
- **Peeked Avg [ms]**: The average time, in milliseconds, spent to peek at messages with this message type by this destination.
- **Peeked Min [ms]**: The minimum time, in milliseconds, spent to peek at a message with this message type by this destination.
- **Peeked Max [ms]**: The maximum time, in milliseconds, spent to peek at a message with this message type by this destination.
- **Peeked Sum [ms]**: The accumulated time, in milliseconds, spent to peek at messages with this message type by this destination.
- **Peeked Avg [bytes]**: The average number of bytes used to peek at a message with this message type by this destination.
- **Peeked Min [bytes]**: The minimum number of bytes used to peek at a message with this message type by this destination.
- **Peeked Max [bytes]**: The maximum number of bytes used to peek at a message with this message type by this destination.
- **Peeked Sum [bytes]**: The accumulated number of bytes used to peek at messages with this message type by this destination.

Not all of these metrics are available for all message types:

- Metrics for the size of a sent/received message arenotavailable for JMS message types `StreamMessage`, `MapMessage`, and `ObjectMessage` ; or for Tibco messages.
- Peeking of messages is only supported by MSMQ. For other message types, such as JMS and Tibco, these columns are empty.

Context menu options

- **Drill Down**: Drill down to other analysis dashlets. For example, you can drill down from an entry in the **Messaging** dashlet to the **PurePaths** dashlet to view all transactions that have interacted with a certain queue or topic.
- **Subscribe Measures**: Open the **Measure Picker** dialog box to add measures to a subscription. This option is only available for realtime sessions.
- **New Business Transaction**: Open the **Business Transaction** dialog box to create a new Business Transaction based on measure related to the selection. This option is only available for realtime sessions.
- **Source Lookup** See Source Lookup Feature for a detailed description.
- **Add Sensor Rules**
  - **Refine with Callee Methods**: All methods that are called by the selected methods are retrieved and can be used as input for new Inclusion Sensor Rules.
  - **Exclude Selected Methods**: All selected methods can be used as input for new Exclusion Sensor Rules.
  - **Include Selected Methods**: All selected methods can be used as input for new Inclusion Sensor Rules.

Sensor Configuration

See Sensor Properties for information on configuring the Messaging Sensors.

Tuning options

Use these guidelines to tune performance:

- Use `getConnection()`. JMS connections are served from a connection pool, so calling `getConnection()` on a queue connection factory is fast.
- Tune the message-driven EJB´s pool size to optimize the concurrent processing of messages.
- Cache bean-specific resources. Use the `setMesssageDrivenContext()` or `ejbCreate()` method to cache bean-specific resources, and release those resources with the `ejbRemove()` method.

- Limit use of JMS connections. When designing an application that uses JMS connections, try to use connections sparingly by either pooling them or using the same connection for multiple sessions. The JMS connection uses two threads and the sessions use one thread each. Because these threads are not taken from a pool and the resultant objects aren't pooled, you could run out of memory during periods of heavy usage. One workaround is to move `createTopicConnection` into the `init` of the JSP. Make sure you specifically close the session, because when it stays open it ties up resources.

Method Hotspots dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Prerequisites

Overview

Caller breakdown

Navigation

Related pages

Performance specific dashlets

Prerequisites

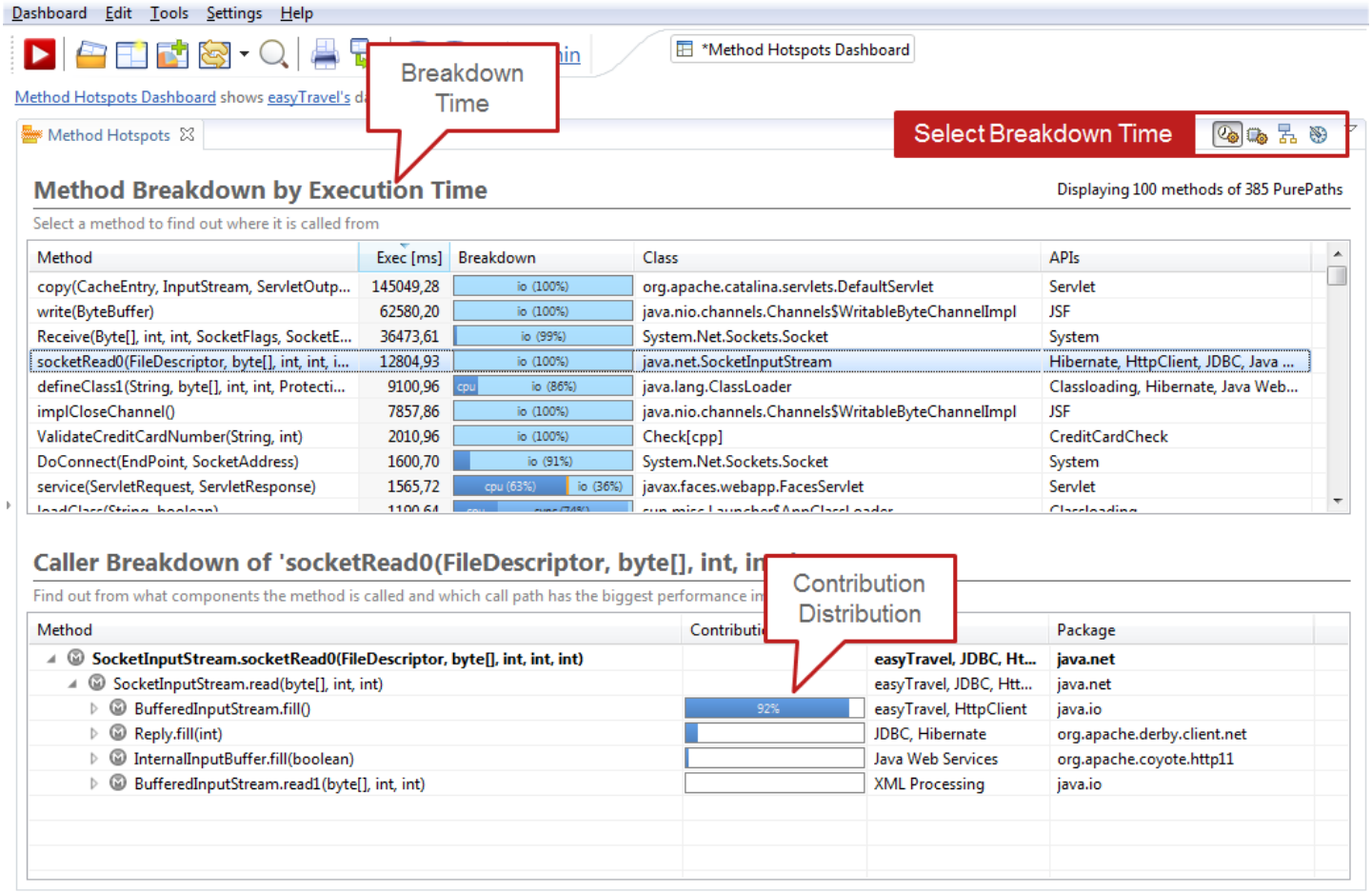
To display sync and wait times for Java and .NET applications in this dashlet, Auto Sensors must be enabled.

Overview

The **Method Hotspots** dashlet shows methods executed during the specified timeframe. The methods are aggregated according to the breakdown time selected in the dashlet toolbar:

- Execution Time
- CPU Time
- Synchronization Time
- Wait Time

The **Method Breakdown** table in the top part the dashlet lists a maximum of 100 methods beginning with the slowest method. Selecting a method displays the hierarchy of the method in the **Caller Breakdown** in the lower section of the dashlet.



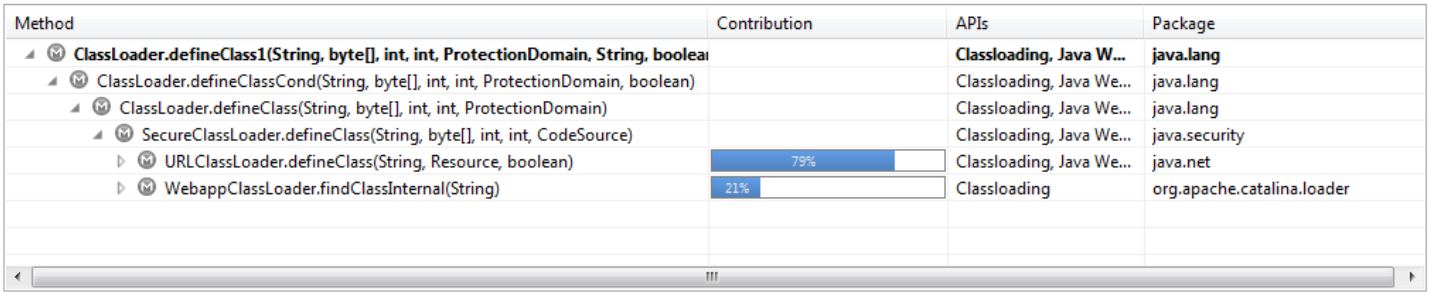
Method Hotspots Dashlet

Caller breakdown

The **Caller Breakdown**, or *Contribution Tree*, displays the detailed call aggregation of the selected method. The called method displays at the top of the tree, and the callers show below the method. Because the Contribution Tree is an aggregation of many paths, it is likely that the called method at the top is reached via many different call paths in all involved PurePaths. The Contribution Tree therefore consists of branches that are weighted according to the breakdown time contribution of each tree branch. Branches with high percentage values are typical of greater interest, because these are the caller branches that may be the likeliest sources of performance problems.

The following figure shows an aggregation of 385 PurePaths. The method **`ClassLoader.defineClass1`** is only called by `ClassLoader.defineClassCond`. Farther down the tree, there are two branches that called `SecureClassLoader.defineClass`:

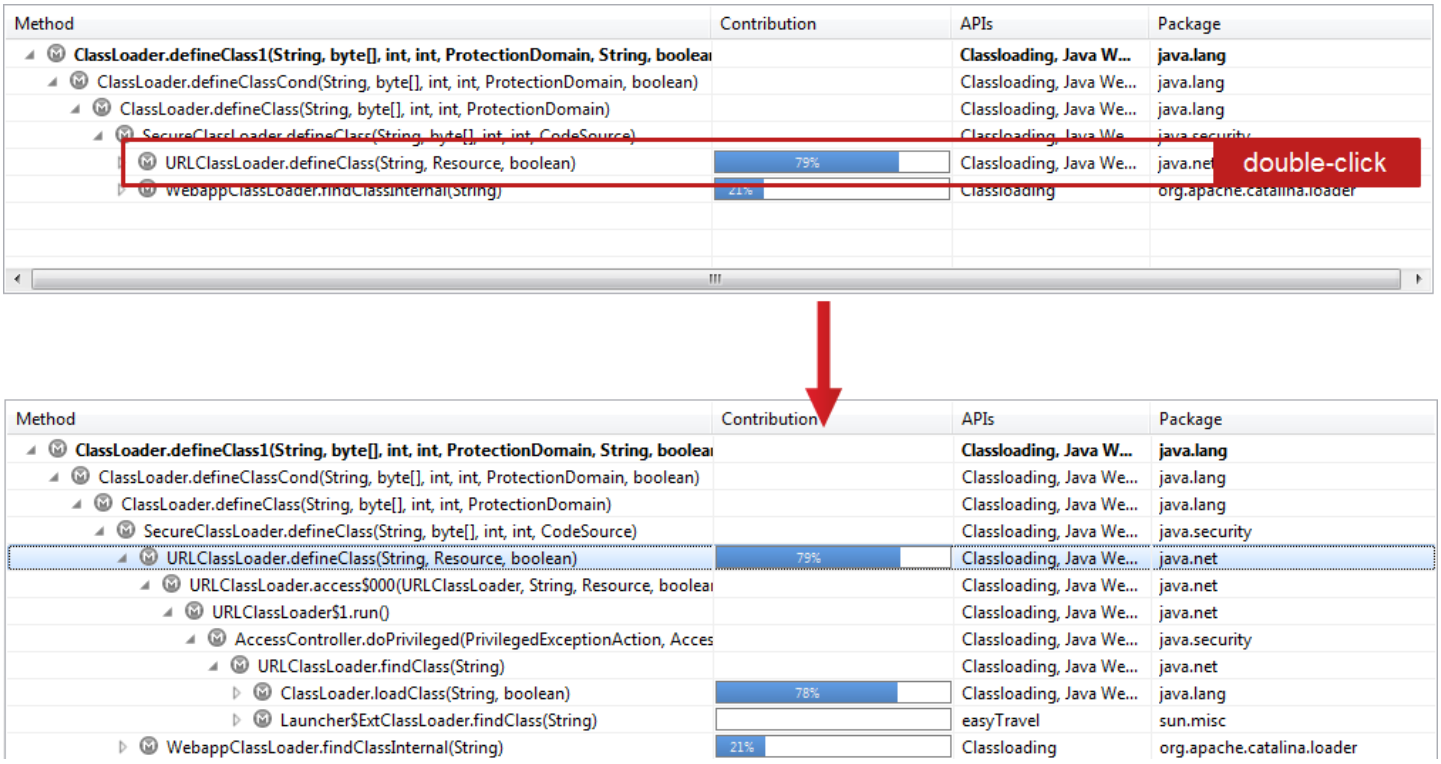
- 79% of the time was spent in **`ClassLoader.defineClass1`** when called by `URLClassLoader.defineClass`.
- 21% of the time was spent in **`ClassLoader.defineClass1`** when called by `WebappClassLoader.findClassInternal`.



Caller breakdown

Navigation

Double-clicking a node that has no contribution information expands or collapses the node. Double-clicking a node that has contribution information collapses the subtree or expands the tree below this node until either another fork with different contribution percentages is found or the tree ends.



Expanding a node

API Breakdown dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Context menu

Related pages

Performance specific dashlets

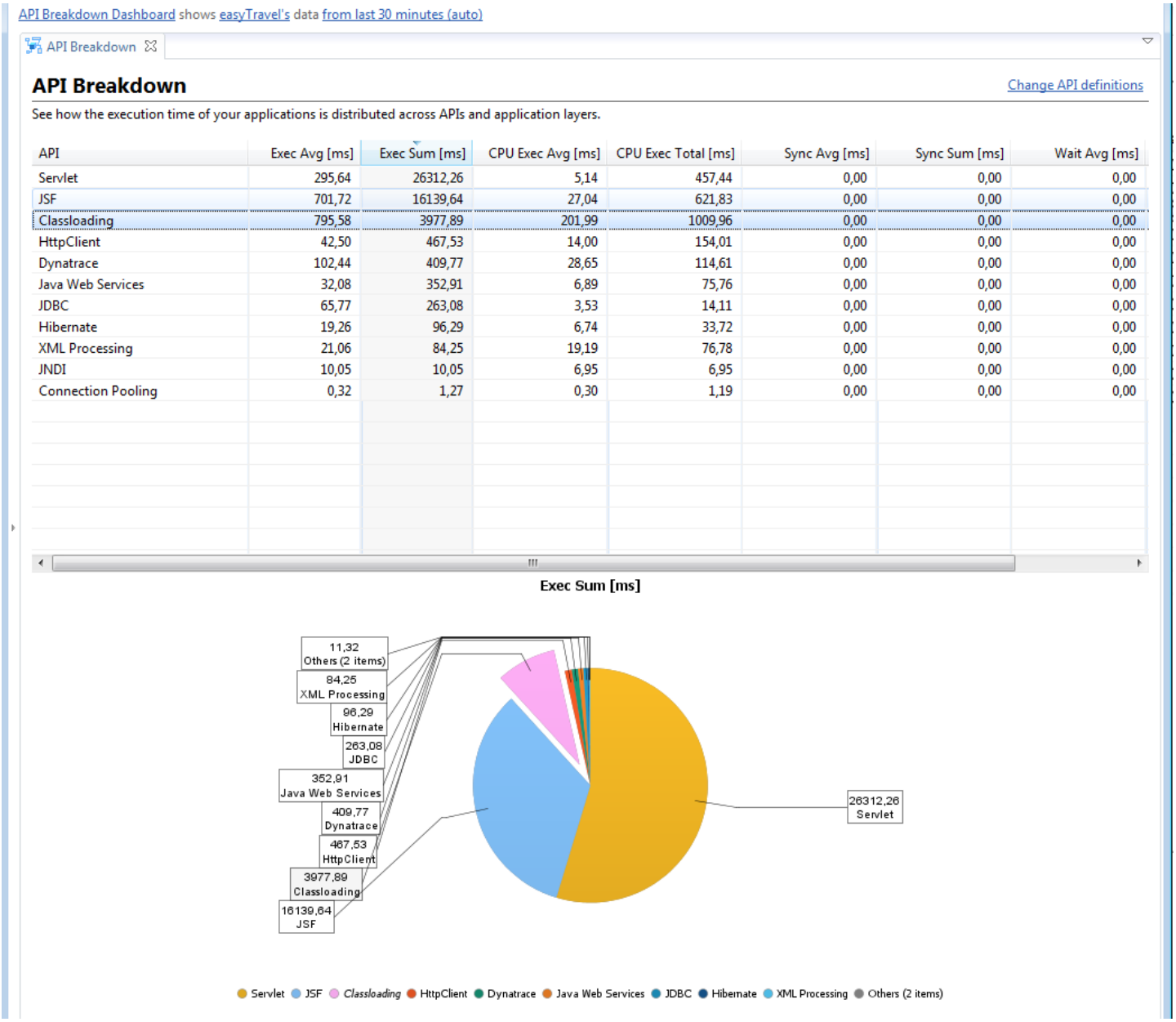
System Profile - APIs

Overview

The **API Breakdown** dashlet provides a view of accumulated performance data, broken down into different APIs. This information helps isolate performance issues at a high, API-related level. See System Profile - APIs and APIs for information about configuring the set of available APIs.

Each line of the table in the upper part of the view represents timing information for a specific API.

The pie chart in the bottom part of the view displays the data of the currently selected table column. The link **Change API definition** takes you to the dialog box where you can refine the API assignment for your application.



API Breakdown dashlet

Metrics

All times for the different metrics are without nested calls to methods of other APIs. By default, not all columns (metrics) are displayed.

To select columns to display or hide right-click on the column header and select **More ...**.

Context menu

The context menu contains these commands that are specific to the dashlet contents:

- **Drill Down**: Open a view associated with the currently selected API entry; for example, open the database view for a JDBC API entry.
- **Subscribe Measures**: Open the **Measure Picker** dialog box to add measures to a subscription. This command is only available during real-time sessions.
- **New Business Transaction**: Open the Business Transaction dialog box to create a new Business Transaction based on measures related to the selection. This command is only available during real-time sessions.

Mobile Workload dashlet

TODO

Naming Services dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

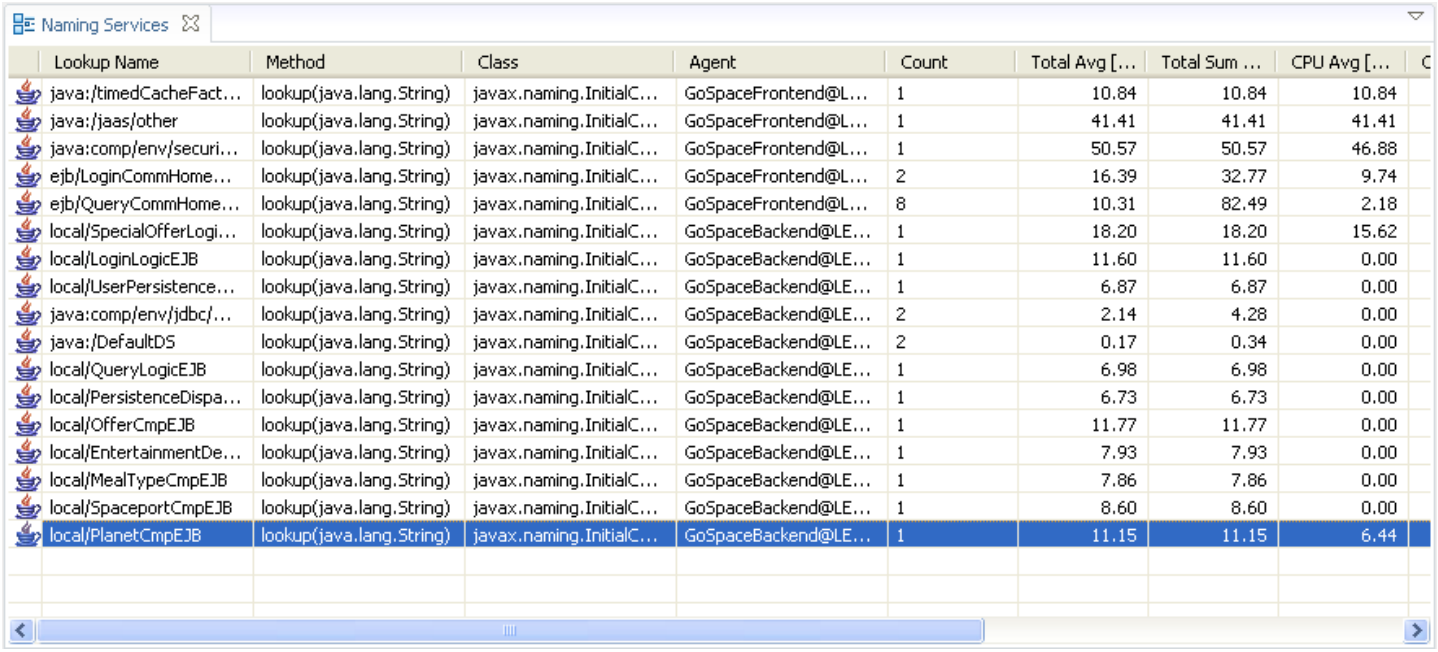
Context menu options

Related pages

Performance specific dashlets

Overview

To diagnose and tune calls to naming service interfaces, the **Naming Services** dashlet displays data from the Java Naming and Directory Interface (JNDI) API. This AP provides Java applications with a unified interface to multiple naming (Lookup Names) and directory services. JNDI is also the basis for other important APIs, including RMI, EJB, JMS, and CORBA.



Naming Services dashlet

Metrics

The **Naming Services** dashlet displays the following information:

- **Platform**: This unlabeled column displays an icon that identifies the software platform on which this particular name lookup was executed.
- **Look-up Name**: The Qualified Lookup Name of the requested object.
- **Method**: The method of the Naming Service component that was called to perform the name lookup.
- **Class**: The class of the Naming Service component that provides the method used to perform the name lookup.
- **Agent**: The name of the AppMon Agent that reported the name lookup invocation.
- **Count**: The number of times this particular Naming Service method with this particular lookup name has been called or executed.
- **Total Avg [ms]**: The average overall time, in milliseconds, spent to execute the Naming Service lookup.
- **Total Sum [ms]**: The accumulated overall time, in milliseconds, spent to execute the Naming Service lookup.
- **CPU Avg [ms]**: The average overall CPU time, in milliseconds, spent to execute the Naming Service lookup.
- **CPU Sum [ms]**: The accumulated overall CPU time, in milliseconds, spent to execute the Naming Service lookup.

- **Exec Avg [ms]**: The average overall time, in milliseconds, spent to execute the Naming Service lookup, excluding time spent in nested method calls.
- **Exec Sum [ms]**: The accumulated overall time, in milliseconds, spent to execute the Naming Service lookup, excluding time spent in nested method calls.
- **Exec CPU Avg [ms]**: The average overall CPU time, in milliseconds, spent to execute the Naming Service lookup, excluding time spent in nested method calls.
- **Exec CPU Sum [ms]**: The accumulated overall CPU time, in milliseconds, spent to execute the Naming Service lookup, excluding time spent in nested method calls.

By default, not all metrics are included in the table. To select which columns to display, right-click a column head and select or clear the column checkboxes as needed.

Context menu options

Right-click the dashlet to display a context menu that includes these options:

- **Create Measures**: Open the **Measure Configuration** dialog box to add measures to a subscription. This option is only available during realtime sessions.
- **Source Lookup**: See Source Lookup Feature for a detailed description.

PurePath Comparison dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

General usage

Comparison process

Differential PurePath metrics

Invoking the comparison from other dashlets

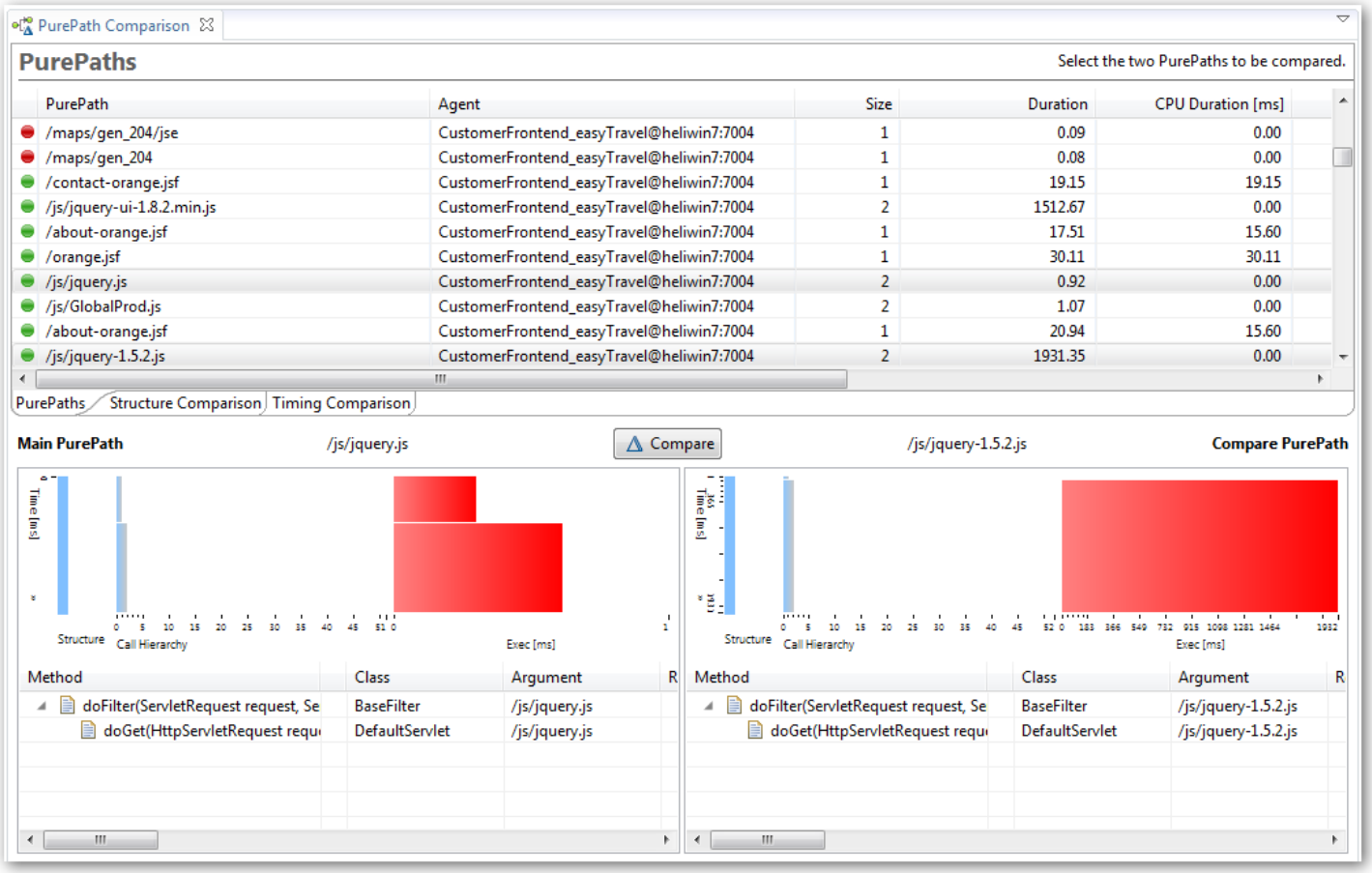
Related pages

Performance specific dashlets

Problem pattern detection

Overview

The AppMon Session Comparison feature makes it possible to compare the results of diagnostic analyses on two PurePath sessions. The information gained by these analyses, however, can hide details that are only visible in the PurePaths themselves. It therefore can be necessary to compare PurePaths on a **call-by-call** level. The *PurePath Comparison* dashlet automates and eases the structural comparison of two PurePaths.



PurePath Comparison dashlet

General usage

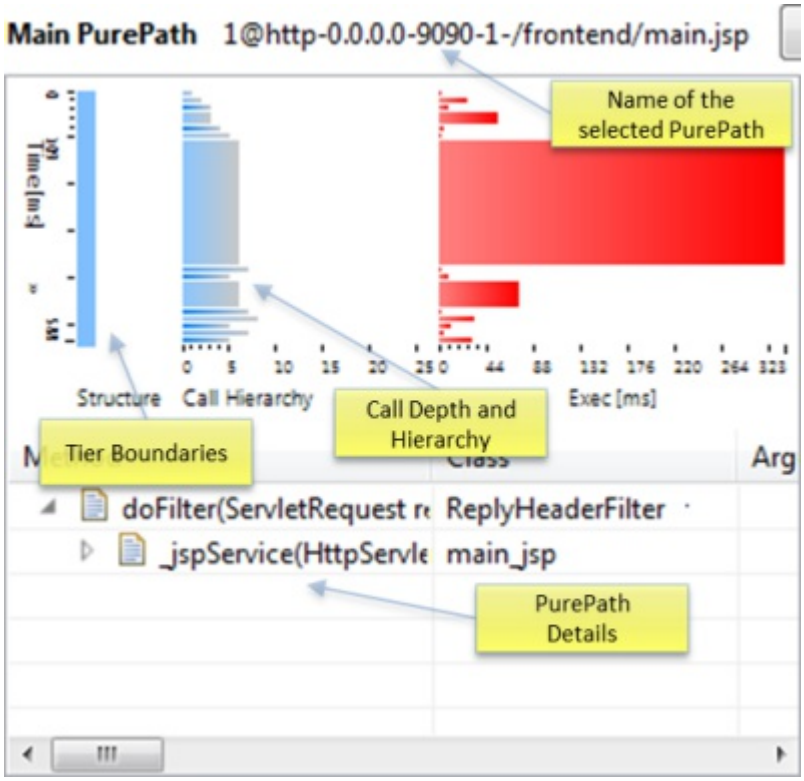
The **PurePath Comparison** dashlet can be added to a dashboard and customized like the **PurePaths** dashlet.

Hold down the Ctrl key and select any two PurePaths in the **PurePath** tab to display their details, in the order selected, in the **Main PurePath** and **Compare PurePath** panes. When just two PurePaths are available, they will be selected and compared automatically.

At the top of each pane is a modified Hotspots visualization. This visualization is extended to include the path structure and call hierarchy, two key metrics for the structural comparison of two PurePaths. Below the visualization is a table that lists the detailed metrics.

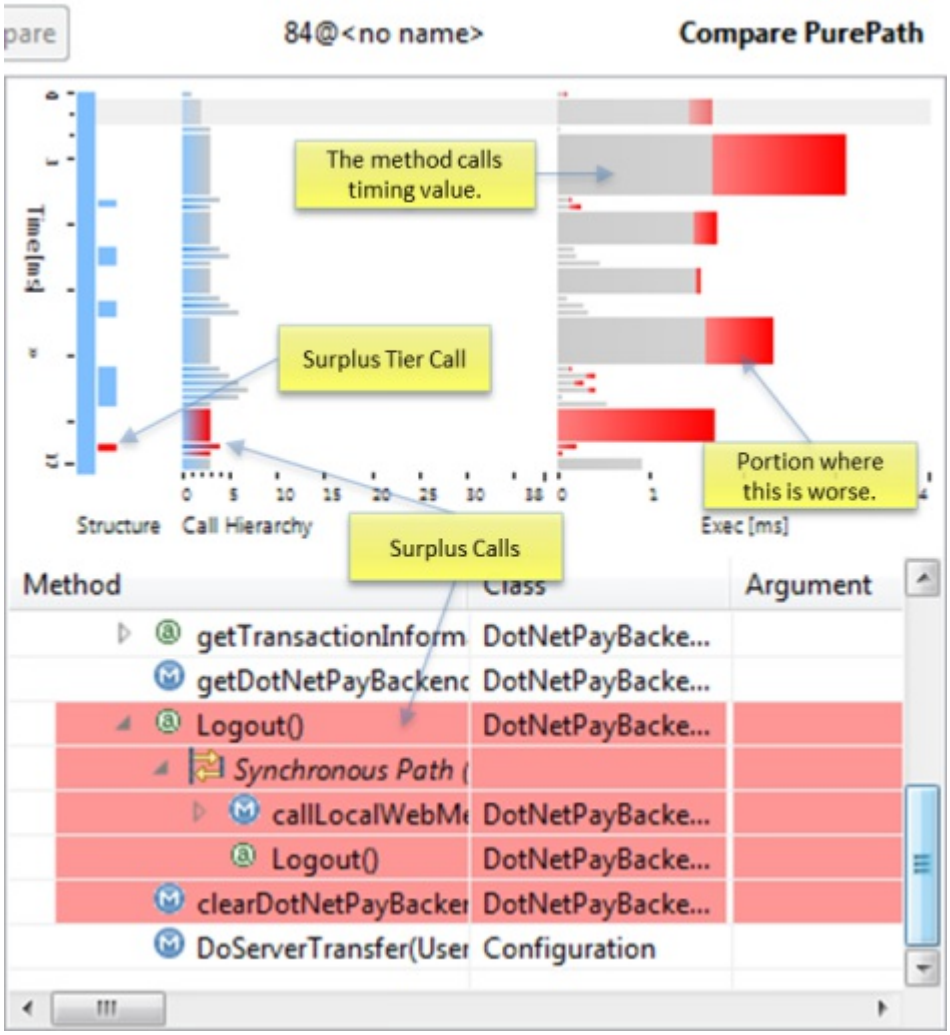
You can configure a comparison data source, to compare PurePaths from different input sources. The PurePaths from the comparison data source are displayed in a second table next to the table for the main data source. You do not have to hold down the Ctrl key to select comparison PurePaths from different data sources.

The following figure shows one of the two panes that display the PurePath details. The label at the top identifies the selected PurePath. On the left side of the HotSpots view, the **Structure** chart (in blue) visualizes the bounds of the various threads and tiers of which this PurePath is composed. The **Call Hierarchy** chart shows the depth of the contributing method calls within their invocation trace. The table at the bottom is equivalent to the tree visualization in the **PurePaths** dashlet. It lists all available metrics for the selected PurePath.



PurePath details

When two PurePaths are loaded in the comparison panes, their details line up to give a general impression of their similarity or differences. You can trigger the automated comparison by clicking the **Compare** button located between the two slots, or by switching to either the **Structure Comparison** or the **Timing Comparison** tab. As the call-by-call comparison can be time consuming, a progress dialog box will open up, offering the opportunity to cancel the task.

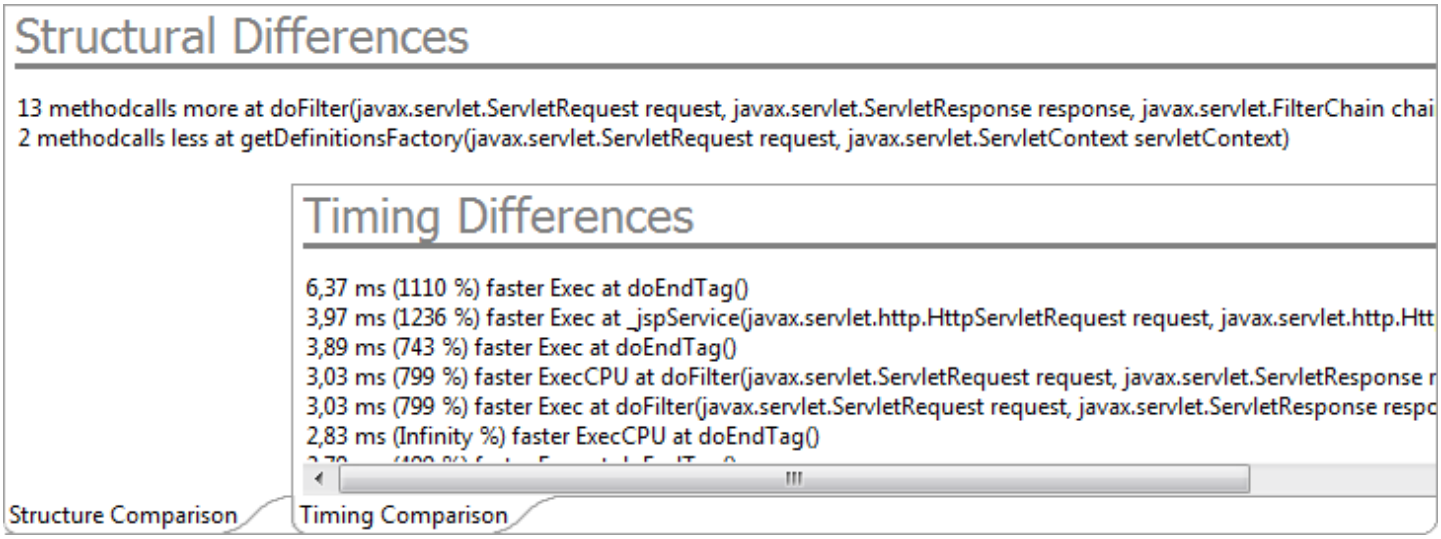


Detailed PurePath differences

After a successful comparison, the details are updated and detected differences are highlighted in red. Matching method calls are linked to each other: clicking one call automatically browses to the corresponding call. The call's metrics are extended with hints about the similarity between their values. Any drilldown that is triggered from the details yields a differential drilldown with the **Main PurePath** as the main data source and the **Compare PurePath** as the comparison data source.

The **Structure Comparison** and **Timing Comparison** tabs offer an alternative representation of the comparison results. By focusing on the detected differences, they call attention to potential points of interest. Especially when the compared PurePaths are large and also very similar, this information can reduce the effort necessary to browse the ample detail data.

Clicking an entry in **Structure Comparison** and **Timing Comparison** selects the corresponding call in the details pane. The reported modifications are always relative to the **Main PurePath**.



Structure Comparison and Timing Comparison Tabs

Comparison process

The **PurePath Comparison** dashlet executes a call-by-call comparison of two selected PurePaths to detect structural and timing differences. This implies looking at every method call of the PurePaths in question individually, instead of basing the results on aggregated data. Thus, the time necessary to compare the two PurePaths depends greatly on their size, ranging from instant for small PurePaths (fewer than 1000 nodes) up to a minute for larger PurePaths (more than 10,000 nodes). Another factor is the degree of structural similarity between the two PurePaths; identical PurePaths are processed much faster than completely different ones.

The comparison happens on the Client, with no impact on the Server other than requesting the data for the two PurePaths.

Differential PurePath metrics

When comparison results are available, the regular PurePath method call metrics are extended to reflect the difference to their counterparts. These extended metrics are only available for matching method calls. Two method calls match if their fully qualified class and method names match.



Differential Metrics

For all timing measurements such as **Total**, **Execution**, **CPU**, …) the following data displays:

`value ( percentage % )`

where:

`percentage = ( difference / other ) * 100` `difference = value – other`

For all captured strings such as Arguments, Return values, and Thread names:

`value`

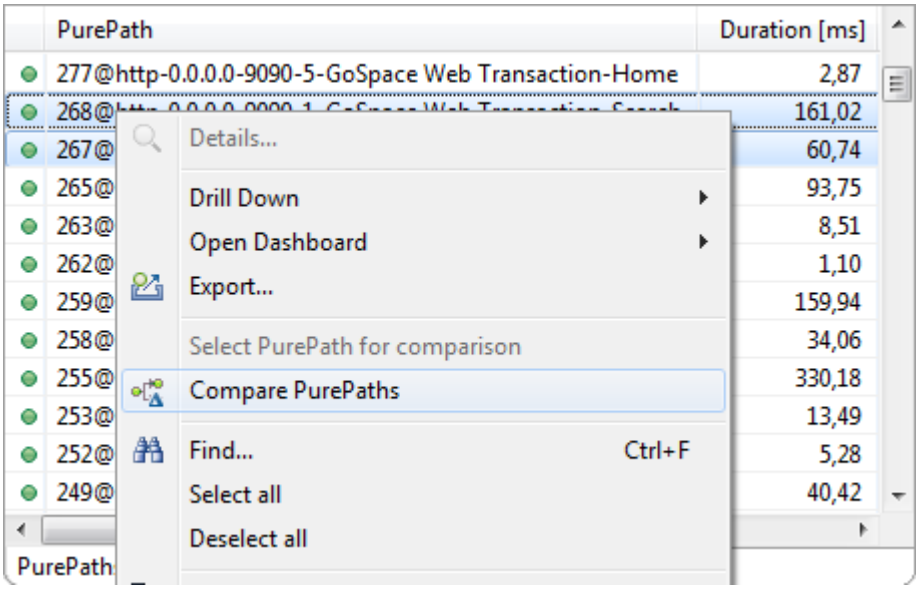
when identical, otherwise:

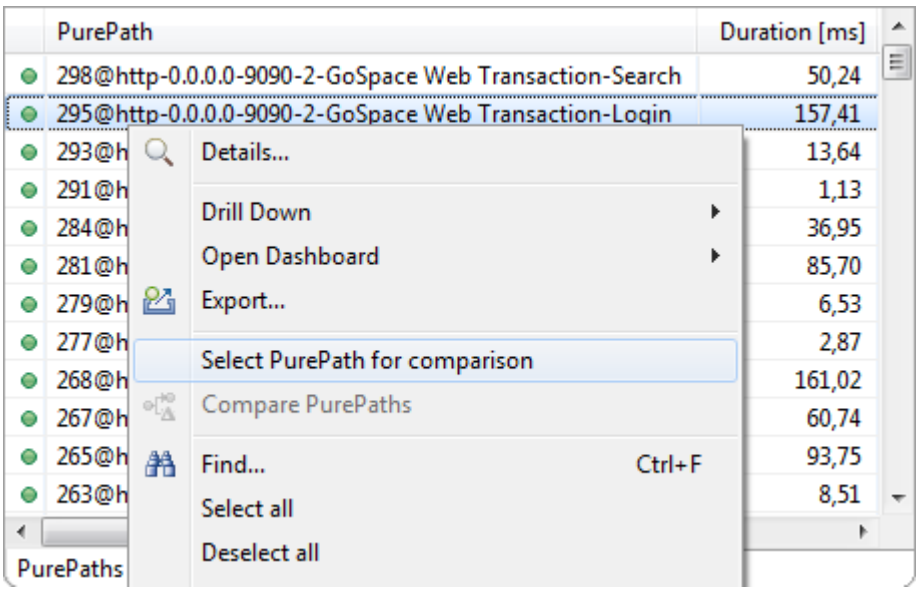
`value ( other )`

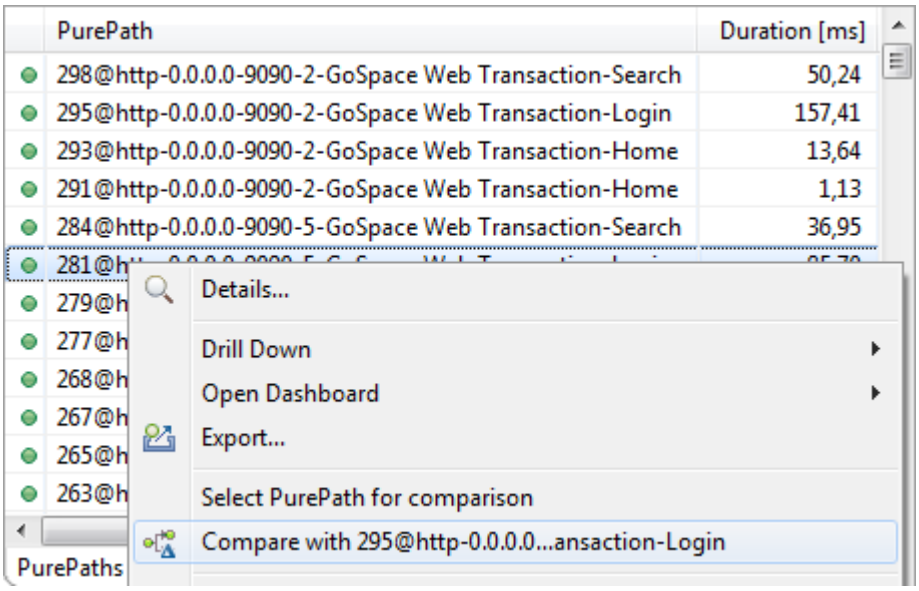
Invoking the comparison from other dashlets

Like most other diagnostic dashlets, the **PurePath Comparison** dashlet can be the target of a drilldown. When the data set has been refined to contain just two PurePaths, the comparison is triggered automatically.

In the PurePaths dashlet, two options are available on the context menu to select two PurePaths and send them to the **PurePath Comparison** dashlet.


PurePath
Duration [ms]
277@http-0.0.0.0-9090-5-GoSpace Web Transaction-Home 2,87
268@ 161,02
267@ 60,74
265@ 93,75
263@ 8,51
262@ 1,10
259@ 159,94
258@ 34,06
255@ 330,18
253@ 13,49
252@ 5,28
249@ 40,42
Details...
Drill Down
Open Dashboard
Export...
Select PurePath for comparison
Compare PurePaths
Find... Ctrl+F
Select all
Deselect all
PurePath


PurePath
Duration [ms]
298@http-0.0.0.0-9090-2-GoSpace Web Transaction-Search 50,24
295@http-0.0.0.0-9090-2-GoSpace Web Transaction-Login 157,41
293@h 13,64
291@h 1,13
284@h 36,95
281@h 85,70
279@h 6,53
277@h 2,87
268@h 161,02
267@h 60,74
265@h 93,75
263@h 8,51
Details...
Drill Down
Open Dashboard
Export...
Select PurePath for comparison
Compare PurePaths
Find... Ctrl+F
Select all
Deselect all
PurePaths


PurePath
Duration [ms]
298@http-0.0.0.0-9090-2-GoSpace Web Transaction-Search 50,24
295@http-0.0.0.0-9090-2-GoSpace Web Transaction-Login 157,41
293@http-0.0.0.0-9090-2-GoSpace Web Transaction-Home 13,64
291@http-0.0.0.0-9090-2-GoSpace Web Transaction-Home 1,13
284@http-0.0.0.0-9090-5-GoSpace Web Transaction-Search 36,95
281@h
279@h
277@h
268@h
267@h
265@h
263@h
Details...
Drill Down
Open Dashboard
Export...
Select PurePath for comparison
Compare with 295@http-0.0.0.0...ansaction-Login
PurePaths

PurePaths dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

PurePaths tab

Contributors tab

Errors tab

PurePath Hotspots pane

Tree tab

Transaction Flow tab

API Distribution tab

Related pages

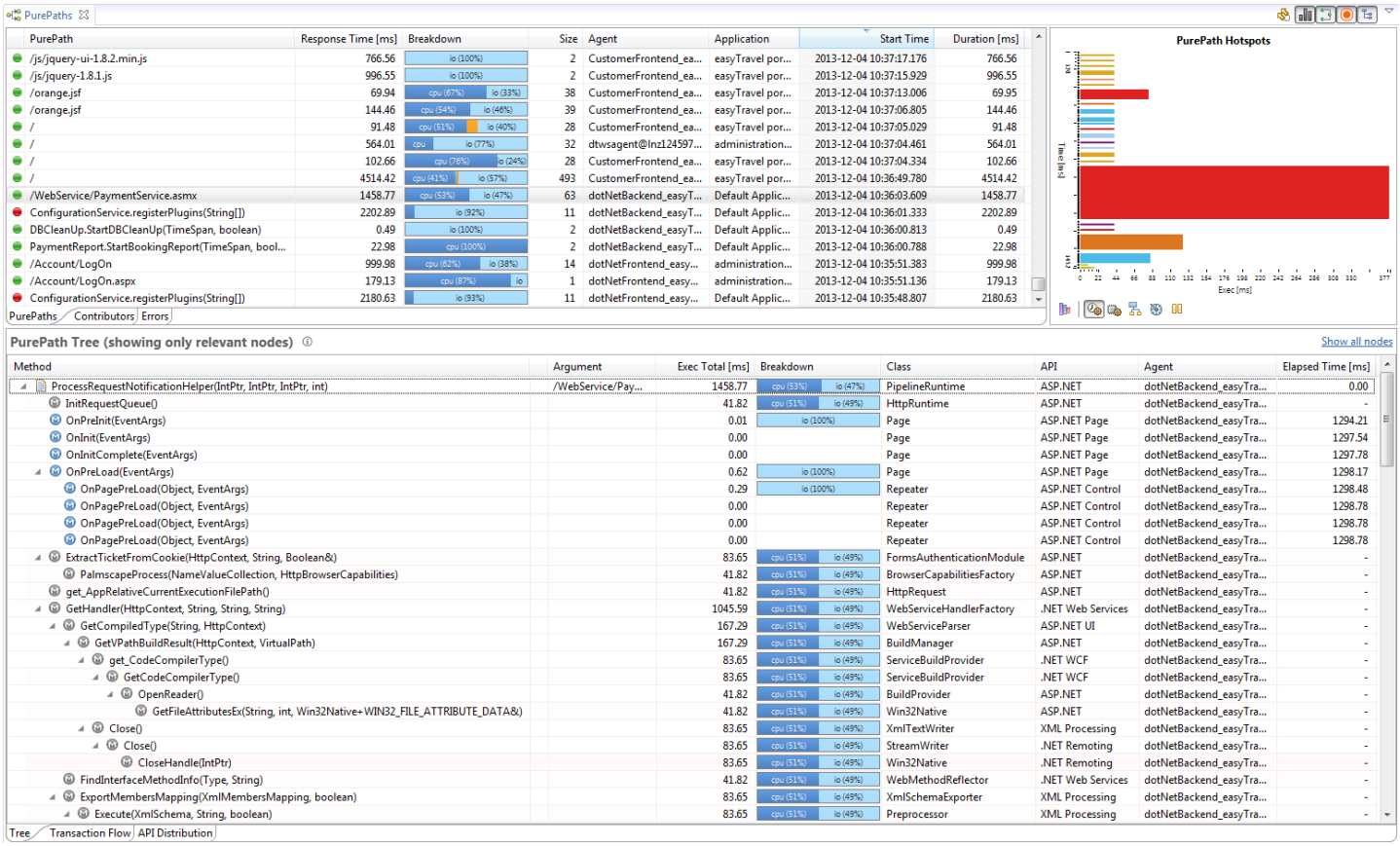
Performance specific dashlets

Problem pattern detection

Overview

The **PurePaths** dashlet is the central representation of all captured PurePaths. You can use this dashlet to identify performance-critical methods and analyze their position within the PurePath, to recognize the cause of a performance problem. Because all special diagnosis dashlets can be traced back to the **PurePaths** dashlet, it is the basis of performance analysis.

You can use this dashlet to get either a top-town or a bottom-up performance diagnosis for selected PurePaths. A top-down approach can identify certain methods as the root cause of poor performance. With a bottom-up approach, you can use the performance measurements to identify the exact method and business transaction that caused the biggest performance issues. Bottom-up analysis can also identify all the effects of a method that does not perform well.



PurePaths dashlet

The upper half of the dashlet contains these tabs:

- PurePaths
- Contributors
- Errors

The PurePath Hotspots pane is to the right of these tabs.

The lower half contains these three tabs:

- Tree
- Transaction Flow

- API Distribution

PurePaths tab

The table in the **PurePaths** tab displays the data for recorded PurePaths. It is the basis for other components of this view.

PurePath metrics

The **PurePaths** table displays the following performance metrics.

- **Error State**: The first, unlabeled column displays icons that indicate the state of the PurePath.
- **PurePath**: A string representing the PurePath identifier.
- **Response Time [ms]**: The total execution time for the PurePath in milliseconds, excluding asynchronous paths that may take longer independently.
- **Exec Sum [ms]**: The overall time spent executing any captured method within the PurePath.
- **Breakdown**: The breakdown of percentage between CPU, Sync, Wait, and I/O times.
  A high I/O time doesn't necessarily mean that the application spends its time in I/O operations, because the I/O time also contains kernel time.
- **Size**: The number of method calls within an PurePath. Only instrumented methods are included.
- **Agent**: The name of the Agent from which this PurePath starts.
- **Application**: The application to which the PurePath belongs. This is the first matched application, as defined in the System Profile.
- **Top Findings**: Shows the problem pattern category top findings for the PurePath. See Problem Pattern Detection for more information on Top Findings.
- **Start Time**: The time when this PurePath started.
- **Transactions**: The names of the Business Transactions applied to the PurePath.
- **Duration [ms]**: The time from the first method until the last method completed, including asynchronous method calls that may finish after the first method completed.
- **CPU Sum [ms]**: The overall required CPU time for executing this path. This value does not have to be equivalent to the total CPU time of the first method of this path if more than one thread is involved in the PurePath.
- **Sync Sum [ms]**: The overall required sync time (e.g. wait for monitors) for executing this path. This value is equivalent to total sync time of the first method of this path if more than one thread is involved in the PurePath. Only instrumented methods for which synchronization time recording is enabled are measured.

- **Wait Sum [ms]**: The overall required wait time (e.g. waiting for other threads or I/O) for executing this path. This value is equivalent to the total wait time of the first method of this path if more than one thread is involved in the PurePath. Only instrumented methods for which synchronization time recording is enabled are measured.
- **Suspension Sum [ms]**: The time the path was in suspension (typically garbage collector runs).
- **Agent ID**: The numerical ID of the Agent instance from which this PurePath starts.
- **Tag ID**: The continuous numerical ID of PurePaths recorded by the given Agent.
- **Thread Name**: The name of the root path's thread.
- **End Time**: The time with the PurePath ended.
- **Web Request Response Time [ms]**: The processing time required to complete a web request, including (if applicable) Servlet 3 asynchronous processing time.
- **Client IP**: The client IP address for web requests.

By default, not all metrics are included in the table. To select which columns to display, right-click a column head and select or clear the column checkboxes as needed.

PurePath states

The icons in the first column of the **PurePaths** table indicate the state of the PurePath:

-  **Complete**: The PurePath is completed. All expected events for this PurePath were received, and it is no longer waiting for new events.
-  **Asynchronous Complete**: A PurePath on which asynchronous Servlet 3 processing took place is completed.
-  **Processing**: The PurePath is currently being processed. Additional events describing this PurePath are expected. Data for PurePaths in this state is not used for other analysis dashlets than the PurePath dashlet.
-  **Processing**: The PurePath timed out while being processed.

- 
  Waiting: A completed PurePath received all expected events but is still waiting for possible events created by asynchronously executed parts of the PurePath. (See Note 1, below.) If any Agent crashes, this state might remain until the Agent is removed from the AppMon Server by a timeout (default 30 seconds), because this Agent might have been called by existing PurePaths.
- 
  Timeout: A PurePath did not receive all expected events within the timeout period. One of the following applies:
  - At least one method within the PurePath has not returned, and for the configured timeout (See Note 1, below) no new data has arrived for the PurePath. The transaction is probably stuck.
  - The PurePath is a long-running transaction that has not been finished within the Maximum PurePath Age timeout (See Note 2, below).
- 
  Path Timed Out: The PurePath did not finish in time.
- 
  Corrupted: A problem occurred while creating the PurePath. PurePaths marked as corrupted are not used for other analysis views than the PurePath view. Corrupted PurePaths are not analyzed at all: they do not provide any measurements or incidents. A PurePath can be corrupted when:
  - At least one event carrying data for this PurePath has been skipped. Check the PurePath details for the number of skipped events. Events are skipped when they are not read fast enough from the instrumented application where they were created. This can be caused by:
    - Slow network between the instrumented Application and the AppMon Collector
    - Slow network between the AppMon Collector and the AppMon Server
    - Too much instrumentation on one or more instrumented applications
  - The instrumented application was shut down while the PurePath was in the **Processing** state.
- 
  Truncated Complete: The PurePath was truncated because the number of nodes exceeded the maximum allowed nodes. Either reduce the instrumentation to get shorter PurePaths or increase the number of maximum allowed nodes.
- 
  Truncated Processing: The PurePath is still expecting events, but has already exceeded the maximum allowed nodes. Data received with incoming events is used to update existing nodes, but no new nodes are created.

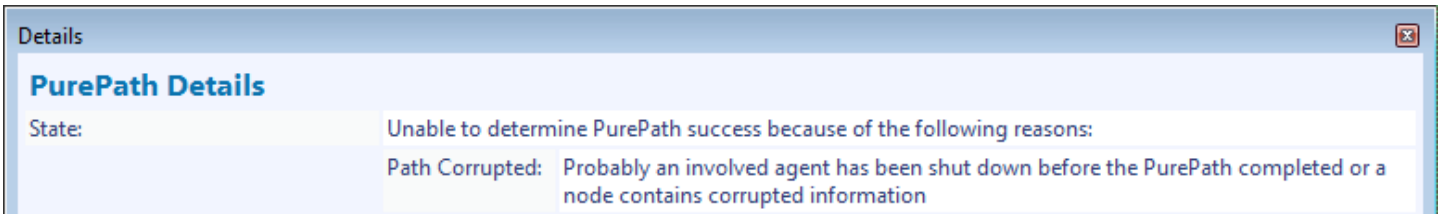
- 
  Truncated Waiting: The PurePath is waiting for new events from asynchronous calls, but has already exceeded the maximum allowed nodes. Data received with incoming events is used to update existing nodes, but no new nodes are created.
- 
  Truncated Timeout: The PurePath timed out while waiting for expected events and has exceeded the maximum allowed nodes. Data received with incoming events is used to update existing nodes, but no new nodes are created.
- 
  Incomplete: The PurePath was stored in a stored session while the PurePath was in the Processing or Waiting state. It cannot be guaranteed that the PurePath is completed. Incomplete PurePaths can only occur in stored sessions.
- 
  Truncated Incomplete: A truncated PurePath was stored in a stored session while the PurePath was in the Processing or Waiting state. It cannot be guaranteed that the dumped PurePath is completed. Incomplete PurePaths can only occur in stored sessions.
- 
  Error: The PurePath contains an error.
- 
  Path Corrupted: It is not possible to determine PurePath success because the PurePath is corrupted. Either an involved Agent may have been shut down before the PurePath completed, or a node may contain corrupted information.
- 
  Transaction Failed: The transaction failed. Review the details for more information.
- 
  Transaction Failed: The transaction failed because it timed out.

Notes

1. **PurePath completeness timeout**: The *completeness timeout* defines the timeframe for PurePaths to be finalized when additional data might still come in. The default is 600 seconds (10 minutes); that is, the timeout is triggered 10 minutes after the last event came in on that PurePath. If you have constant events coming in, the PurePath will continue until there is no event within the defined timeout period. The timeout can be configured to a maximum of 3600 seconds (1 hour). Check Agent connections and application logic before changing the timeout value: `-Dcom.dynatrace.diagnostics.completenessTimeout=<valueInSeconds`
2. **Maximum PurePath age**: In very rare cases, it might be necessary to change the default PurePath timeout, for instance for an extremely long running job of SQL statements. In `<DT_HOME>/dtserver.ini`, add `-Dcom.dynatrace.diagnostics.maximumPurePathAge=<valueInSeconds&`
   Then restart the AppMon Server to implement the changes.
   Be aware that increasing this value can result in enormous PurePath node sizes, challenging the stability of the AppMon Server. However, if DB statement aggregation (currently available for JDBC and ADO.NET) is enabled, it may shrink the node size for parameterized queries, which reduces the load.

The icons give you a quick visual clue about a PurePath´s state. To review details, either hover over the row or right-click the row and select **Details** from the context menu.

The following figure provides example details for a Path Corrupted ( ) icon.



PurePath details

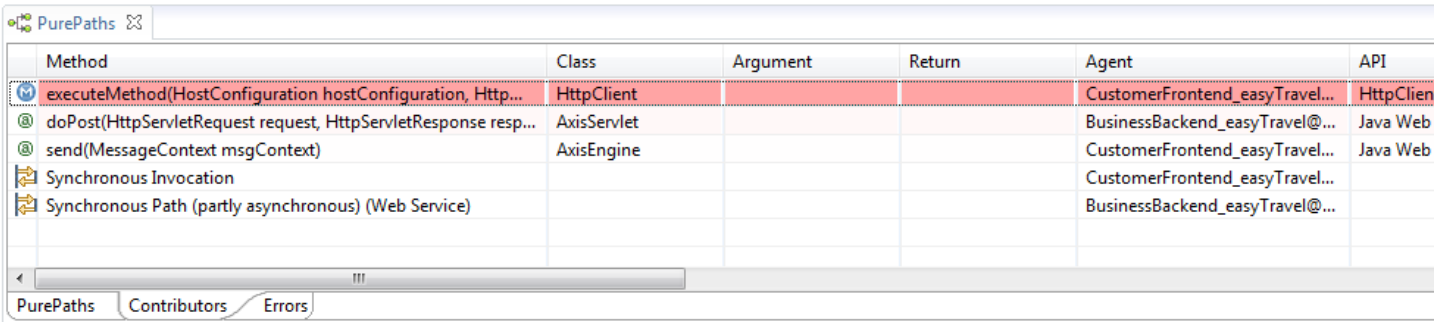
PurePaths context menu

Right-click the **PurePaths** tab to display a context menu that includes these options:

- **Drill Down**: Right-click a PurePath and select a dashlet from the cascading menu to display information for just that PurePath in the selected dashlet.
- **Show in Dashboard**: Right-click a PurePath or select multiple PurePaths, then right-click the dashlet and select from the cascading menu to open a dashboard or create a new dashboard filtered for the selected PurePath.
- **Add Snapshot to Dashboard**: Display the dashlet in the selected dashboard with the time frame automatically set to **last 30 minutes**.
- **Export**: Open the **Export Session** dialog box to export the selected PurePaths to a user-defined session file.
- **Store Session**: Select one or more PurePaths, then right-click the dashlet and select this option to open the **Store Session** dialog box to save the selected PurePaths as a new session. The session will be available in the **Online Session** tree. This option is only available for realtime sessions.
- **Result Limit**: Select the maximum number of PurePaths for the session. The default is 10,000.
- **Select PurePath for comparison**: To compare two PurePaths:
  1. Right-click the first PurePath and select **Select PurePath for comparison**.
  2. Right-click the second PurePath and select **Compare with <PurePath>**.
     The **PurePath Comparison** dashlet opens, with data from the two selected PurePaths.

Contributors tab

The table in the **Contributors** tab lists all instrumented methods of the selected PurePath. Depending on the sort column, the most runtime-critical methods are highlighted by background coloring. You can sort the table by a column's data by clicking the column head.



Contributors Tab

Contributors metrics

The **Contributors** table displays the following metrics.

- **Type**: The first, unlabeled column displays an icon that indicates the object type.
- **Method**: A string representing the method name, including argument types.
- **Class**: The fully qualified class name of the contributor method.
- **Argument**: The argument value of the method call if arguments have been captured.
- **Return**: The return value of the method call.
- **Agent**: The Agent instance that captured the method call.
- **API**: The API identifier with which the method call is associated.
- Exec Total [ms] – The overall duration of the method call from method entry until method exit.
- **Exec [ms]**: The time that was used for execution of this method only, excluding time consumed in nested methods calls.
- **Exec %**: The execution time of the method call as a percentage of the total execution time of the PurePath.
- **CPU Total [ms]**: The total CPU time required for executing the method call from method entry to method exit.
- **CPU Exec [ms]**: The CPU time that was used for execution of this method only, excluding CPU time consumed in nested methods calls.
- **CPU %**: The CPU execution time of this method call as a percentage of the total CPU execution time of the PurePath.
- **Sync Total [ms]**: The total sync time required for executing the method call from method entry to method exit.
- **Sync [ms]**: The sync time that was used for execution of this method only, excluding sync time consumed in nested method calls.
- **Sync %**: The sync time of this method call as a percentage of the total sync time of the PurePath.
- **Wait Total [ms]**: The total wait time for executing the method call from method entry to method exit.
- **Wait [ms]**: The wait time that was used for execution of this method only, excluding wait time consumed in nested methods calls.
- **Wait %**: The wait time of this method call as a percentage of the total wait time of the PurePath.
- **Thread Name**: The name of the thread that was responsible for executing the method call.
- **Elapsed Time [ms]**: Time elapsed between the start of the PurePath and entry of this method.
- **PHP Error Type**: The PHP error type of an error occurred.

The information

icon in a column header indicates that the column contains performance-critical information. If one of these columns is used for sorting, the intensity of the background color indicates the severity state of the dedicated value in relation to all other values contained in this column.

Example

A transaction involves these objects, with these durations:

`doGet`: 100 ms

- `doSomethingAsync1` – 550 ms
- `doSomethingAsync2` – 500 ms

The **response time** is 100 ms, because `doGet` finishes after 100ms. That is, the transaction actually delivers the response back to the end user in 100 ms.

Assuming that the two asynchronous paths are started at the same time, the **duration** is 650 ms — the sum of `doGet` and the longer of the asynchronous paths — because the PurePath is completed.

The execution sum is the sum of the durations of all three objects. Overall, it took 1150 ms to execute the transaction, even though some of it was executed in parallel.

Contributors context menu

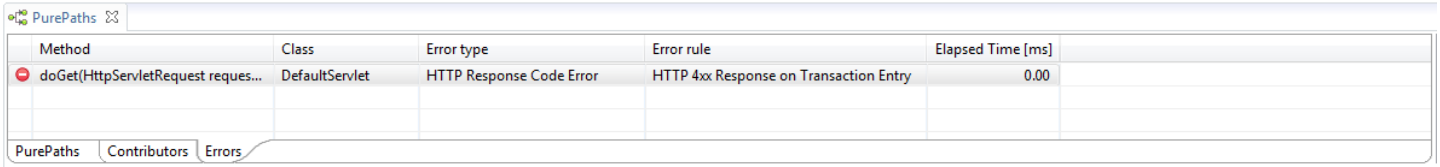
Right-click the **Contributors** tab to display a context menu that includes these options, depending on the selected object:

- **Drill Down**: Right-click a PurePath and select a dashlet from the cascading menu to display information for just that PurePath in the selected dashlet.
- **Show in Dashboard**: Right-click a PurePath or select multiple PurePaths, then right-click the dashlet and select from the cascading menu to open a dashboard or create a new dashboard filtered for the selected PurePath.
- **Add Snapshot to Dashboard**: Display the dashlet in the selected dashboard with the time frame automatically set to **last 30 minutes**.
- **New Business Transaction**: Open the **Business Transaction Editor** to create a new Business Transaction based on the selection-related measures. This option is only available during realtime sessions.
- **Create Measures**: Open the **Measure Configuration** dialog box to add measures to a subscription. This option is only available during realtime sessions.
- **Source Lookup**: See Source Lookup Feature for details.

- **Add Sensor Rules** (only available on live sessions)
  - **Refine with Callee Methods**: All Methods that are called by the Selected Methods are retrieved and can be used as input for new Inclusion Sensor Rules.
  - **Exclude Selected Methods**: All Selected Methods can be used as input for new Exclusion Sensor Rules.
  - **Include Selected Methods**: All Selected Methods can be used as input for new Inclusion Sensor Rules.
- **Define API**: Open the **APIs** pane of the **System Profile Preferences** to assign an API to the System Profile.

Errors tab

The **Errors** tab shows the errors that were detected on the selected path.

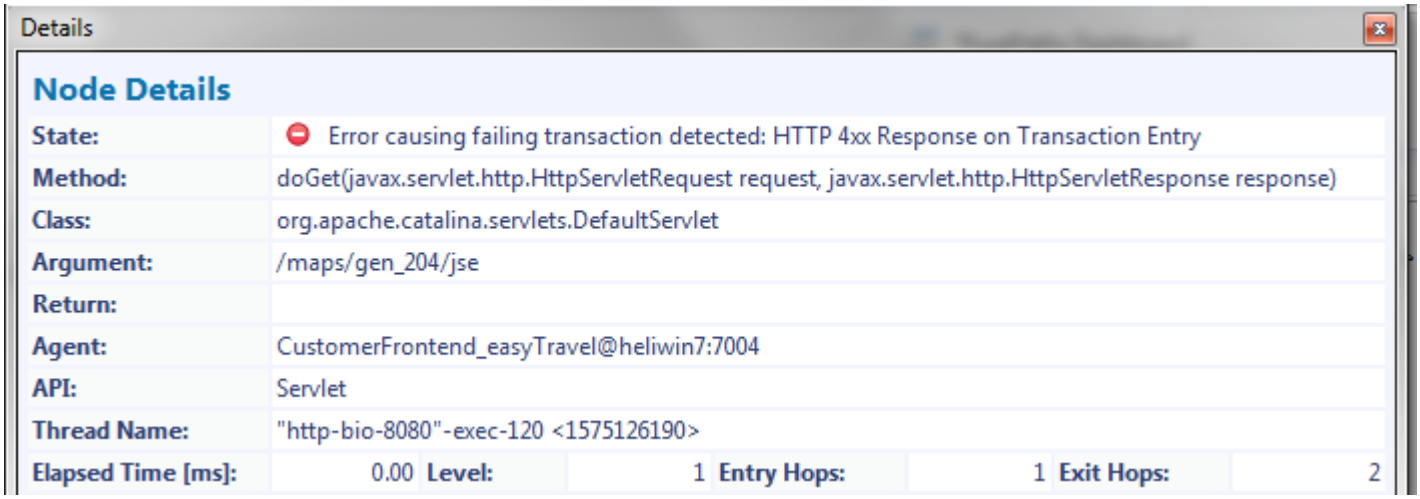


Errors tab

Error metrics

- **Method**: The method caused the error.
- **Class**: The class containing the method that caused the error.
- **Error Type**: The classification of the error: HTTP Response Code Error, Exception Error, PurePath Duration Error, Java Script Error, User Defined Error.
- **Error Rule**: The configured Error Rule that triggered the detection of the error.
- **Elapsed Time [ms]**: Time elapsed between the start of the enclosing PurePath and entry into the method that caused the error.

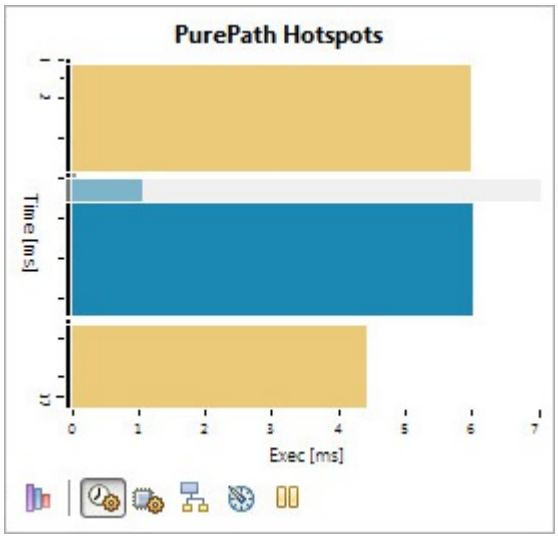
Additional information is available in the **Details** window or the **Errors Dashlet** dashlet.



PurePath error details

PurePath Hotspots pane

The **PurePath Hot Spots** pane gives a visual overview of the currently selected PurePath. It displays a duration metric for the natural tree order of the PurePath contributors. Because there usually are more contributors than space available to render them, minor contributors are aggregated. Click a hot spot to select the corresponding calls in the **Tree** tab, and vice versa.



PurePath Hotspots pane

The vertical axis defines the execution time. The horizontal axis is the metric selected in the toolbar below the chart:

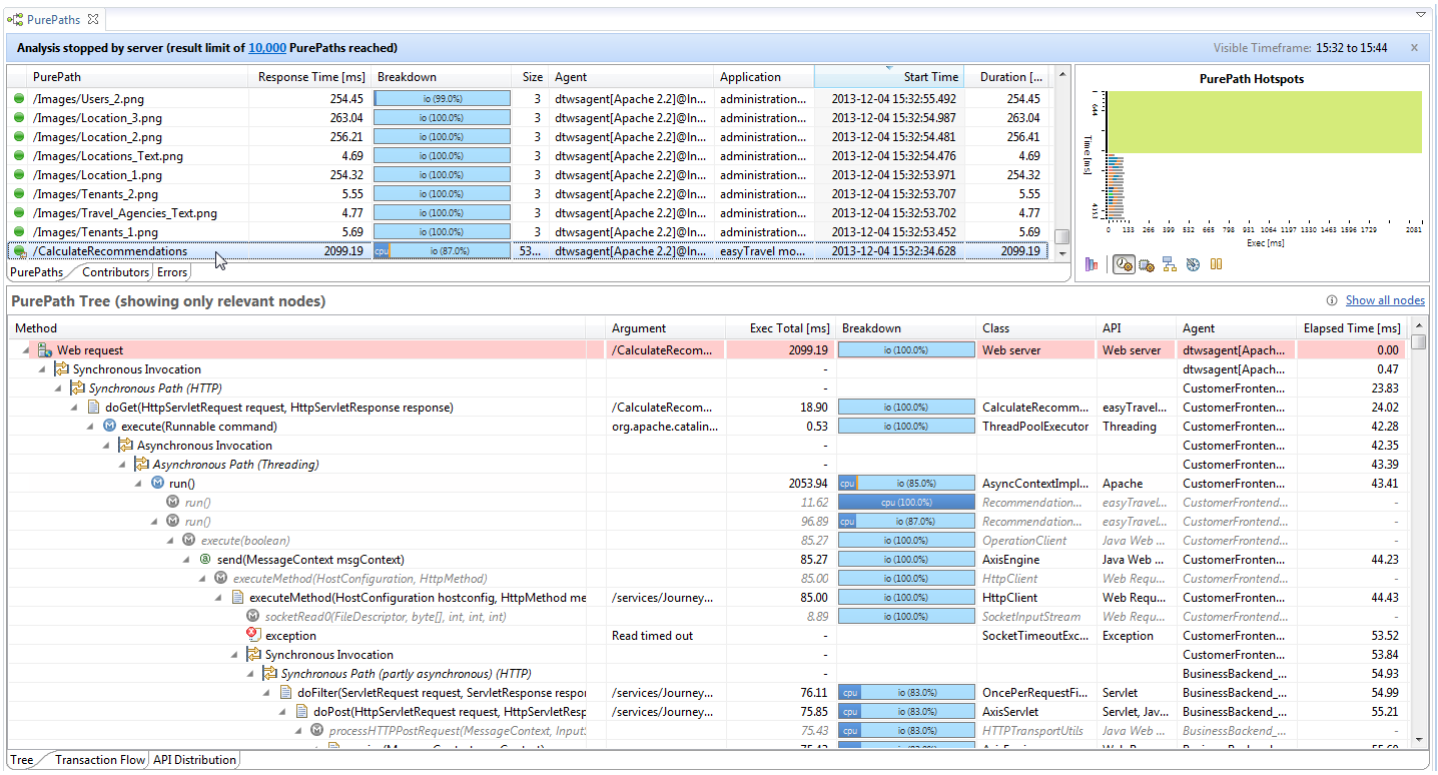
- **Execution**
- **CPU Execution**
- **Synchronization**
- **Wait**
- **Suspension** The bar color is defined by the most significant API of each method call; a gray bar indicates a set of aggregated APIs.

The **PurePath Hotspots** pane can also display the **API Call Hierarchy**. This option renders the hierarchy of the PurePath contributors, colored by the individual call's API, to give further insight into the PurePath's structure.

The pane's context menu gives access to the Details summary. This simplified version of the regular contributor details displays call statistics only. When you select a hotspot for multiple contributors, aggregated values are shown.

Tree tab

The **Tree** tab displays the contributors to the currently selected PurePath arranged as a tree that maps the calling hierarchy of the contributing methods. It contains the same columns as the **Contributors** tab. Methods highlighted in the **Contributors** tab are also highlighted in the **PurePath** Tree. Dependencies between instrumented methods are illustrated by the tree structure. When you select a method in the **Contributors** tab, the PurePath tree scrolls automatically to the selected contributor method, and a yellow arrow appears to the left of that method. With this functional interaction between the **Contributors** tab and the PurePath tree, you can efficiently navigate to desired methods within PurePaths to see context-referred information for this method.



Contributors Tree of the Selected PurePath

The whole tree or parts of it can be expanded or collapsed to focus on the important methods calls and to hide parts of the PurePath not relevant for the current performance diagnosis task. The state of the **Tree** tab is linked with the **Contributors** tab. If a new method is selected within the **Contributors** tab, the **Tree** tab is updated to also show the selected method. If required, the expanded and collapsed parts of the **Tree** tab are rearranged to focus on the new selection.

The call hierarchy of the selected PurePath can span multiple threads or processes, depending on the applied sensors. To indicate thread or process changes within a PurePath, the following special nodes are inserted within the tree view:

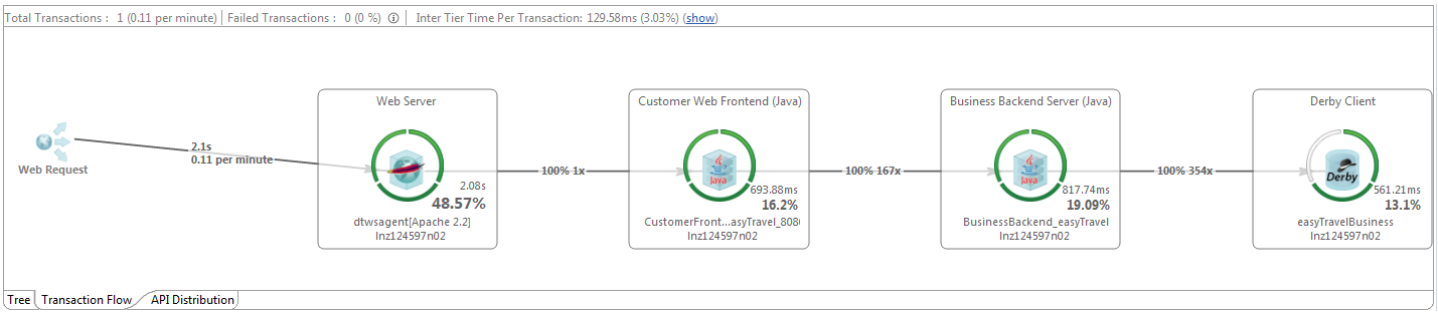
- **Synchronous Path**: A node that groups the nodes of the child path that is executed synchronously on another Agent/thread. The parent node of this Synchronous Path node blocks until the child path has finished its execution.
- **Asynchronous Path**: A node that groups the nodes of the child path that is executed asynchronously on another Agent/thread.
- **Asynchronous Invocation**: A grouping node of one or more Asynchronous Paths as a single call (e.g.. sending one JMS message) that can cause several threads/Agents to start processing (e.g. several subscribers on a JMS topic). A Synchronous Invocation grouping node is **not** shown for a Synchronous Path, because all synchronous invocations lead to exactly one Synchronous Path.

Tree context menu

- **Drill Down**: Right-click a PurePath and select a dashlet from the cascading menu to display information for just that PurePath in the selected dashlet.
- **Show in Dashboard**: Right-click a PurePath or select multiple PurePaths, then right-click the dashlet and select from the cascading menu to open a dashboard or create a new dashboard filtered for the selected PurePath.
- **Add Snapshot to Dashboard**: Display the dashlet in the selected dashboard with the time frame automatically set to **last 30 minutes**.
- **New Business Transaction**: Open the **Business Transaction Editor** to create a new Business Transaction based on the selection-related measures. This option is only available during realtime sessions.
- **Create Measures**: Open the **Measure Configuration** dialog box to add measures to a subscription. This option is only available during realtime sessions.
- **Source Lookup**: See Source Lookup Feature for a detailed description.
- **Add Sensor Rules** (only available on live sessions.
  - **Refine with Callee Methods**: All Methods that are called by the Selected Methods are retrieved and can be used as input for new Inclusion Sensor Rules.
  - **Exclude Selected Methods**: All Selected Methods can be used as input for new Exclusion Sensor Rules.
  - **Include Selected Methods**: All Selected Methods can be used as input for new Inclusion Sensor Rules.
- **Define API**: Open the **APIs** pane of the **System Profile Preferences** to assign an API to the System Profile.
- **Expand All**: Expand all subpaths starting from the selected node. This option is only available on the tree representation table.
- **Collapse All**: Collapse all subpaths starting from the selected node. This option is only available on the tree representation table.

Transaction Flow tab

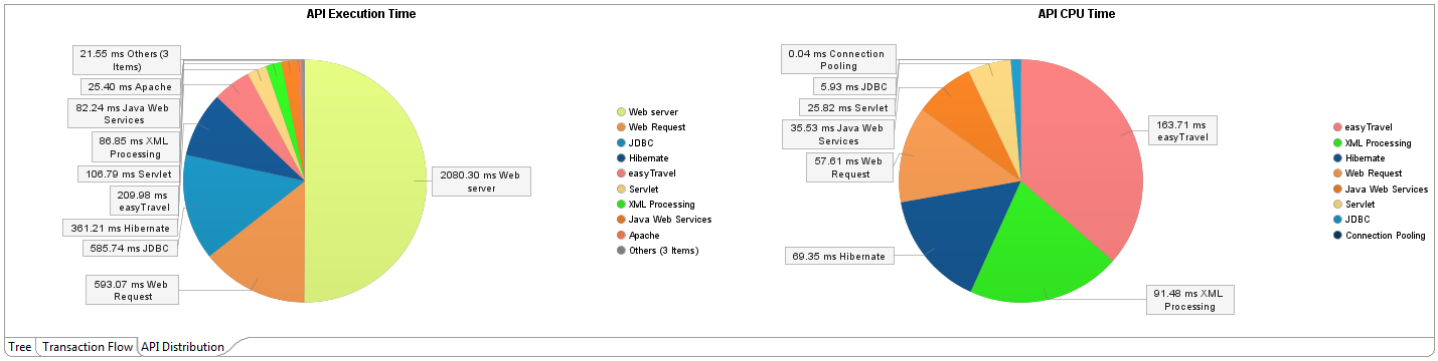
The **Transaction Flow** tab displays a diagram of the distribution of selected PurePaths over components or tiers of the system monitored by distinct AppMon Agent instances. The Transaction Flow Dashlet components involved in the execution of a selected PurePath are highlighted. PurePaths that span multiple components are indicated by dashed lines between the involved components. If several tiers are involved in a PurePath and if they communicate with one another directly, the communication channels are visualized by lines between communicating tiers.



Transaction Flow tab

API Distribution tab

The **API Distribution** tab provides an aggregated view of the distribution of execution time, CPU utilization, and synchronization over the APIs used by the selected PurePath(s).



API layer breakdown charts

Remoting dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Remoting metrics

Tuning guidelines

Related pages

Performance specific dashlets

Overview

For diagnosing and tuning remote method invocations (RMI), AppMon uses dynamic bytecode instrumentation to capture performance metrics, including remoting metrics, across horizontal and vertical application layers. A central AppMon Server correlates the resulting measurements to PurePaths Dashlet.

The **Remoting** dashlet displays the collected measurements.

Remoting metrics

This is a list of every remoting metric that can be collected by AppMon. Specific sensor implementations for individual remoting protocols may not gather the whole set of possible metrics. If a metric is not gathered by a specific Sensor implementation, a dash (-) displays in the column.

The matrix of supported metrics by protocol is available in the Dynatrace Community.

- **Error State**: The first, unlabeled column displays an icon that indicates the error state of the underlying PurePath: OK, Failed Transaction, Error.
- **Platform**: The second, unlabeled column displays an icon that indicates the software Platforms involved in the remote call: Java for a pure Java remote call, .NET for a pure .NET remote call, and Java/.NET for a heterogeneous call.
- **Protocol**: The type of the remoting protocol used for the call. Supported protocols are JRMP, T3, IIOP, and .NET remoting.
- **Side**: Indicates whether a client call (**C**) or a server-side execution (**S**) was detected. If both sides of the remote call were detected, the column shows **C/S**.
- **Type**: Indicates whether client and server involved in the remote method call run within distinct virtual machines.
  - **local**: Client and server run on the same virtual machine
  - **remote**: Client and server run on distinct machines.
  - **unknown**: Cannot be detected because of missing instrumentation on either the client or server side.
- **Method**: The name of the remotely called method.
- **Class**: The name of the class that provided the remotely called method.
- **Count**: The number of times the remote method was called.
- **Failed %**: The percentage of failures that occurred when calling the remote method.
- **Inter Tier Time [ms] (Avg)**: The average network latency, in milliseconds, of all remote calls of the method. See the note below the list for details.
- **Inter Tier Time [ms] (Min)**: The minimum network latency, in milliseconds, of all remote calls of the method.
- **Inter Tier Time [ms] (Max)**: The maximum network latency, in milliseconds, of all remote calls of the method.
- **Inter Tier Time [ms] (Sum)**: The accumulated network latency, in milliseconds, of all remote calls of the method.
- **Total [ms] (Avg)**: The average time elapsed for calls of the remote method as perceived by the client. The client-side total time includes the proxy overhead, network latency, delegation overhead, actual response time of the method implementation, and on the server side the delegation overhead and the actual response time of the method implementation.
- **Total [ms] (Min)**: The minimum time elapsed for a call of the remote method as perceived by the client.
- **Total [ms] (Max)**: Maximum time elapsed for a call of the remote method as perceived by the client.
- **Total [ms] (Sum)**: The accumulated time elapsed for all calls of the remote method as perceived by the client.
- **Exec [ms] (Avg)**: The average time to execute calls of the remote method as perceived by the client.

- **Exec [ms] (Min)**: The minimum time to execute a call of the remote method as perceived by the client.
- **Exec [ms] (Max)**: The maximum time to execute a call of the remote method as perceived by the client.
- **Exec [ms] (Sum)**: The accumulated time to execute calls of the remote method as perceived by the client.
- **CPU Total [ms] (Avg)**: The average CPU time used for calls of the remote method as perceived by the client.
- **CPU Total [ms] (Min)**: The minimum CPU time used for a call of the remote method as perceived by the client.
- **CPU Total [ms] (Max)**: The maximum CPU time used for a call of the remote method as perceived by the client.
- **CPU Total [ms] (Sum)**: The accumulated CPU time used for all calls of the remote method as perceived by the client.
- **CPU Exec [ms] (Avg)**: The average CPU time used for calls of the remote method as perceived by the client, excluding time of nested method calls.
- **CPU Exec [ms] (Min)**: The minimum CPU time used for a call of the remote method as perceived by the client, excluding time of nested method calls.
- **CPU Exec [ms] (Max)**: The maximum CPU time used for a call of the remote method as perceived by the client, excluding time of nested method calls.
- **CPU Exec [ms] (Sum)**: The accumulated CPU time used for all calls of the remote method as perceived by the client, excluding time of nested method calls.

Note

The latency is calculated as the difference between the client-side response time (total time) and the server-side response time (total time). The client-side response time is typically measured at the proxy/stub invocation, before marshaling/serialization takes place. The server-side response time is typically measured at the implementation or shortly before (e.g. the skeleton), after marshalling/serialization takes place.

Therefore:

- The whole communication over the network is included (both directions).
- Serialization times are included.
- XML serialization times are most probably included in the latency. Minor variations may occur between different platforms, technologies, stacks, and protocols.

Tuning guidelines

Follow these guidelines to tune performance based on the metrics described above:

- **Fields**: To reduce the number of objects serialized or to reduce the network traffic volume, use the transient keyword to omit serialization of fields that are not required on the remote location.
- **Transport classes**: Replace classes by special transfer classes that carry less information.
- **Local modifications**: If the number of remote method invocations is high, especially because of calls to getter and setter methods, rethink whether methods should be invoked on a remote object or whether this object should copied to the remote location, modified without network and serialization overhead, and sent back to the remote location.
- **Call merging**: Another technique to reduce the number of remote method invocations is to merge multiple remote method invocations into a single invocation. For example, multiple setter invocations could be merged into a single method call with multiple arguments. This merging reduces the network round trips to round trip.
- **Custom serialization for CPU optimization**: Use custom serialization to optimize for CPU usage (externalizable instead of serializable) to eliminate the overhead cost of Java reflection. Replacing UTF-8 encoded strings with `char[]` can reduce CPU cost for converting strings to the UTF-8 format, but it also doubles network traffic volume.
- **Custom serialization for size optimization**: Use custom serialization to optimize network usage. Multiple arguments could be merged together into a single `bytearray`. For example, merge two 16-bit short integers into a single 32-bit integer.
- **Network optimization**: You can achieve further optimization on the TCP/IP network layer, for example by modifying the `TcpWindow` size. However, network-level optimization is beyond the scope of this topic.

Response Time Hotspots dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Server transaction statistics

HotSpots by tier

HotSpots by API

Related pages

Performance specific dashlets

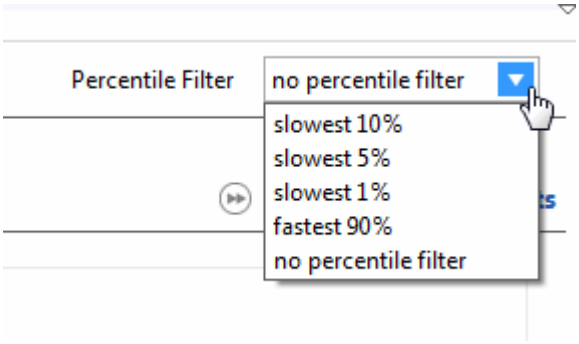
Watch and learn

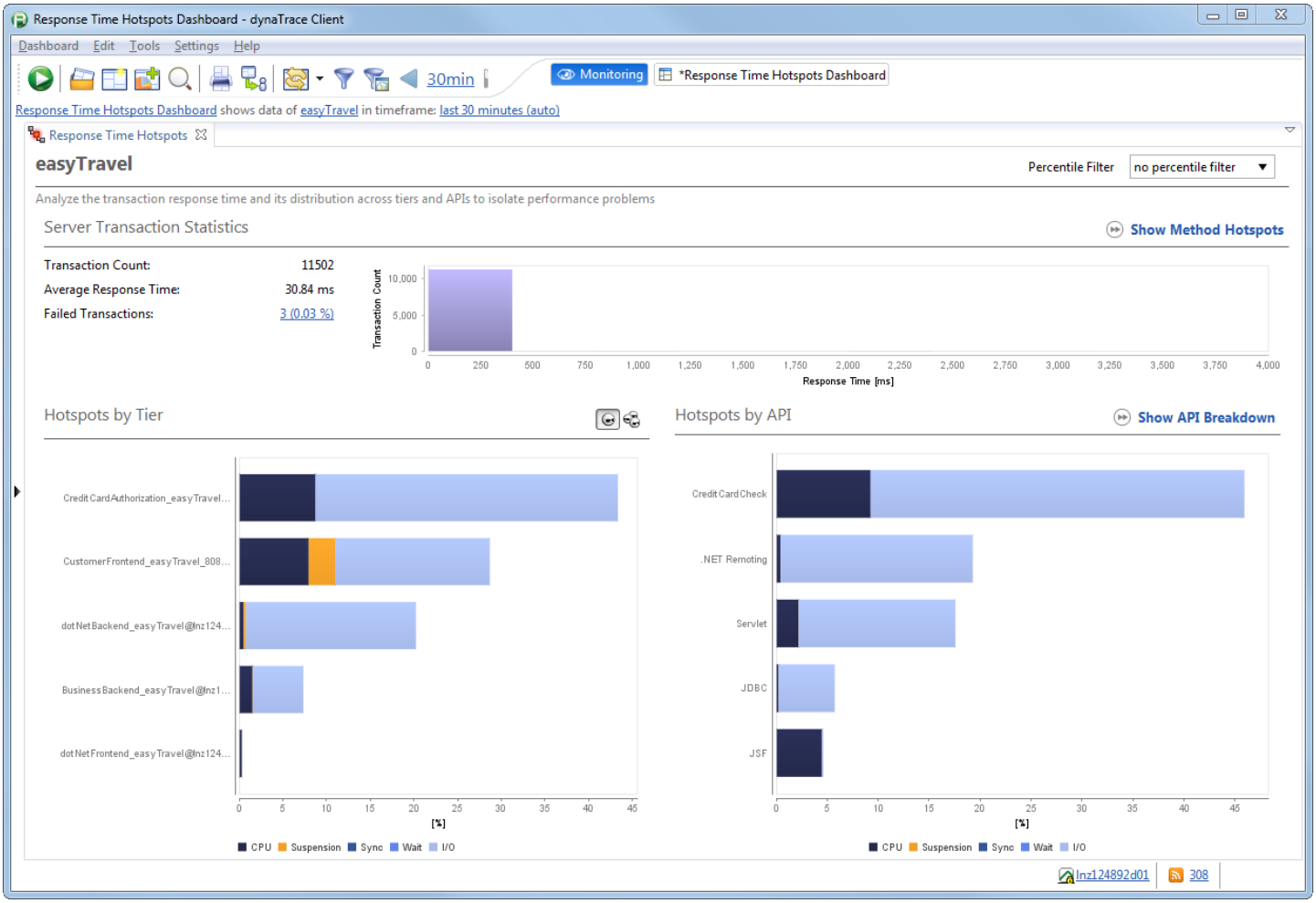
Identifying response time problems

Overview

The **Response Time Hotspots** dashlet provides a high-level view of a session to give you a quick impression of where the time is spent. A typical use case is to filter for Business Transactions to see how similar transactions are distributed regarding response time and where most of the time is contributed, by tier and by API.

You can use the filter list in the top right corner of the dashlet to filter the content to the slowest 1%, slowest 5%, slowest 10% or fastest 90% of transactions in the session.





Server transaction statistics

The top area shows the distribution of the PurePath response times. This chart gives a quick impression of the typical response times of the PurePaths. It is recommended to apply a timeframe filter so similar PurePaths are used for this calculation.

Response time formulas

When a PurePath is executed, AppMon captures CPU time used by the threads that are executing this transaction. It also captures the information on how much time was spent in sync (sync blocks), wait (waiting on an object) and runtime suspension (when garbage collection pauses the thread). The rest of the time is contributed by I/O.

The formula for response time therefore is:

```
Response Time = CPU + Sync + Wait + Runtime Suspension + I/O
```

I/O is calculated and everything else is measured:

```
I/O = Response Time - CPU - Sync - Wait - Runtime Suspension
```

See PurePath Metrics and Contributors Metrics for more information.

HotSpots by tier

This chart shows either the top five Agents or the top five Agent Groups sorted by the response times of the PurePaths executed there. Clicking a bar opens the Method Hotspots Dashlet dashlet showing the methods that contribute most to the PurePaths executed on this Agent or Agent Group.

HotSpots by API

The **Hotspots by API** chart shows the times spent per API. Clicking an API opens the Method Hotspots Dashlet dashlet showing the methods by their contribution to that API. Clicking a database-related API opens the Database Dashlet dashlet.

Sequence Diagram dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Detail levels

Related pages

Performance specific dashlets

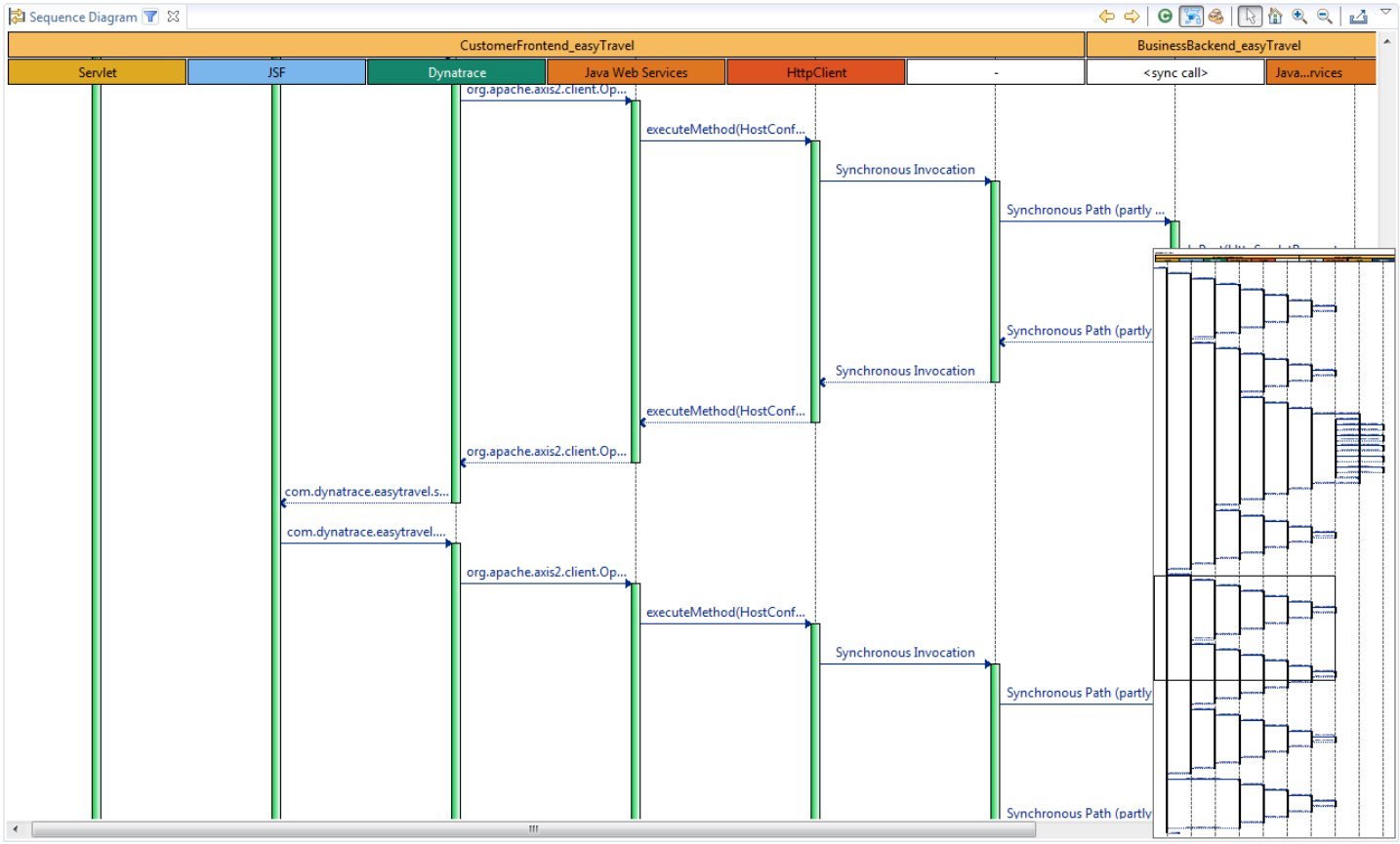
Overview

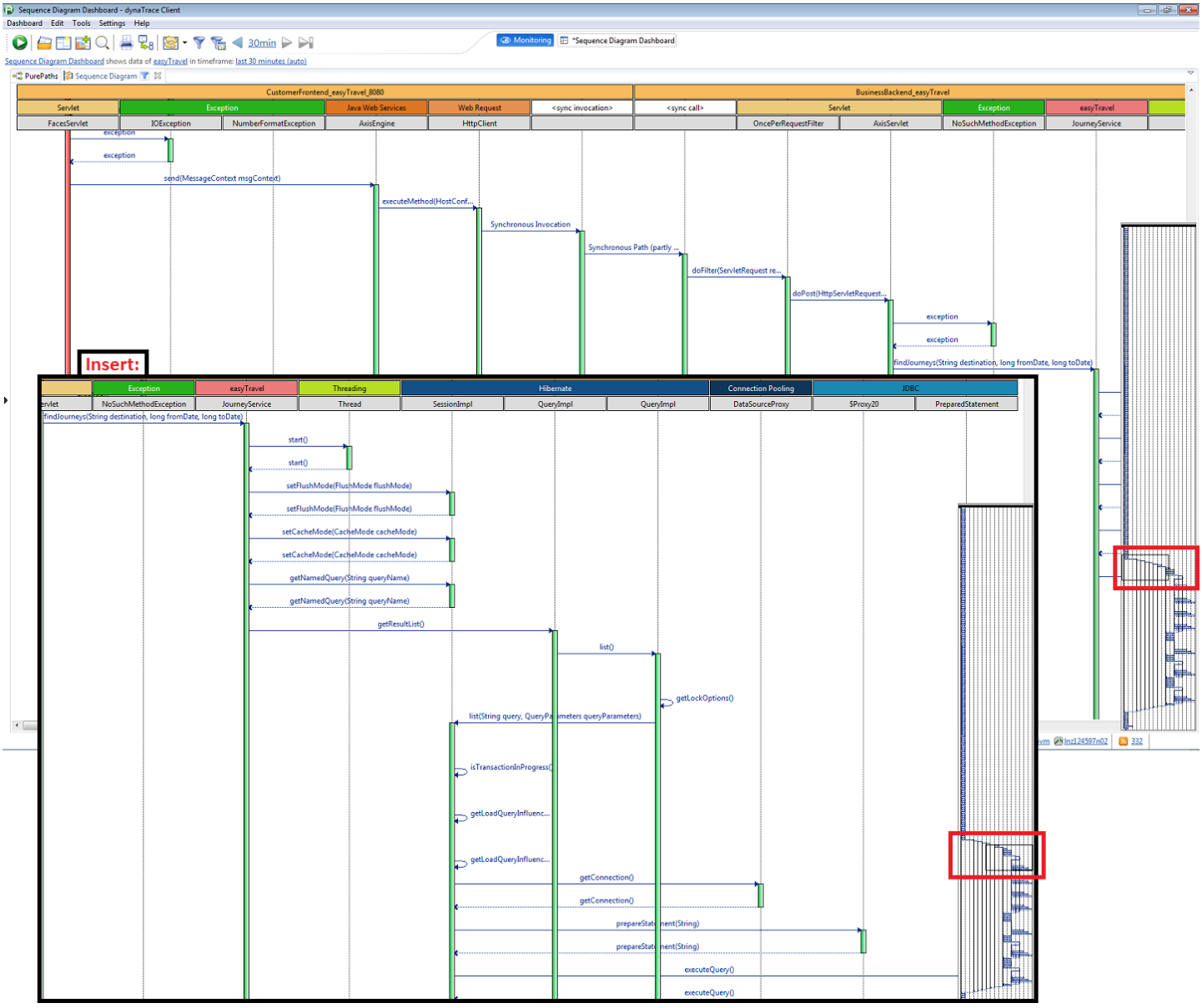
The **Sequence Diagram** dashlet shows a sequence diagram representation of a PurePath — a visualization of the Agents and APIs used.

If you open the dashlet from the **Cockpits**, the resulting dashlet shows the last PurePath that came in. You then can navigate through the PurePaths by clicking the arrow icons in the toolbar in the top right corner. To display the sequence diagram for a specific PurePath, right-click the row in the **PurePaths** dashlet and select **Drill Down > Sequence Diagram**.

If the content of the sequence diagram is larger than the visible area of the screen, the **Overview** button  appears in the lower right corner, at the intersection of the scrollbars. Click the button to open the overview popup window, then drag the cursor in the overview to update the visible area of the diagram to the match the cursor location in the overview. The shift and control keys can be used to slow down dragging for finer navigation in very large diagrams.

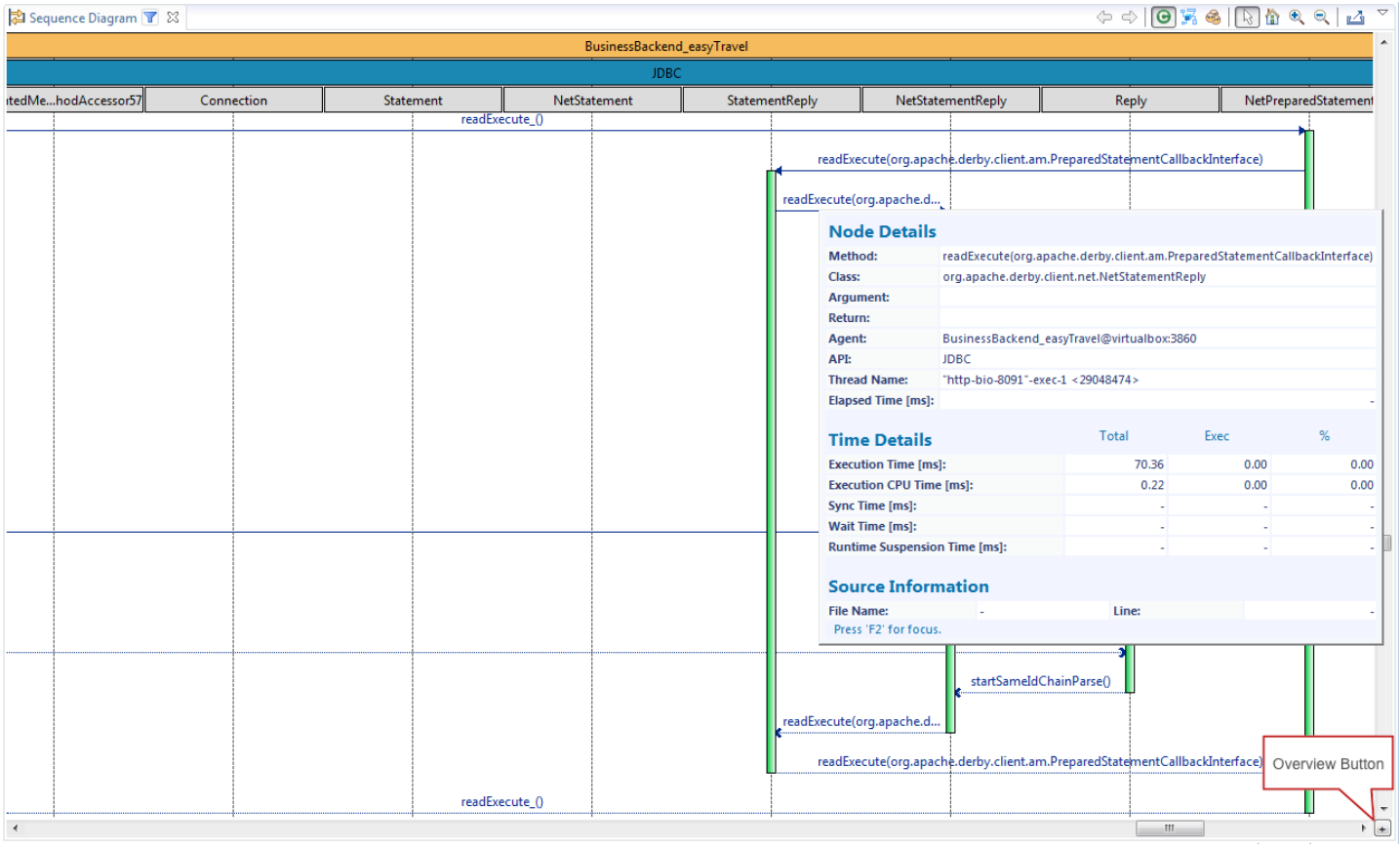
You can also resize the diagram by using the zoom out and zoom in buttons in the toolbar.


Sequence Diagram
CustomerFrontend_easyTravel
BusinessBackend_easyTravel
Servlet
JSF
Dynatrace
Java Web Services
HttpClient
-
<sync call>
Java...rvices
org.apache.axis2.client.Op...
executeMethod(HostConf...
Synchronous Invocation
Synchronous Path (partly ...
Synchronous Path (partly
Synchronous Invocation
executeMethod(HostConf...
org.apache.axis2.client.Op...
com.dynatrace.easytravel.s...
com.dynatrace.easytravel....
org.apache.axis2.client.Op...
executeMethod(HostConf...
Synchronous Invocation
Synchronous Path (partly
Synchronous Path (partly



When you select a call or return visualization object, you can use the right and left arrow keys to navigate through the call order.

To show the details of a call, hover over the object or right-click and select **Details**.



Details and Overview button

Toolbar

The toolbar is visible when the mouse pointer is inside the diagram.


Previous PurePath: Show the sequence diagram for the previous PurePath.
The arrows are only enabled when you opened the **Sequence Diagram** dashlet from the **Cockpits**. If you drilled down from the PurePaths dashlet, you are restricted to the one path node in the dashlet.

Next PurePath: Show the sequence diagram for the next PurePath.

Detail Level Classes: Show the sequence diagram containing all calls.

Detail Level APIs: Show the sequence diagram containing only calls that cross an API or Agent boundary.

Detail Level Agents: Show the sequence diagram containing only calls that cross an Agent boundary,

Reset Zoom: Reset a zoomed sequence diagram to its default size.

Zoom In: Zoom in the sequence diagram.

Zoom Out: Zoom out the sequence diagram.

Export Sequence Diagram: Export the sequence diagram in PNG or JPEG format.
For the optimum overview for large PurePaths with details still recognizable, use a combination of zoom and **Export Complete Sequence Diagram** to control the sequence diagram size and exported image size.

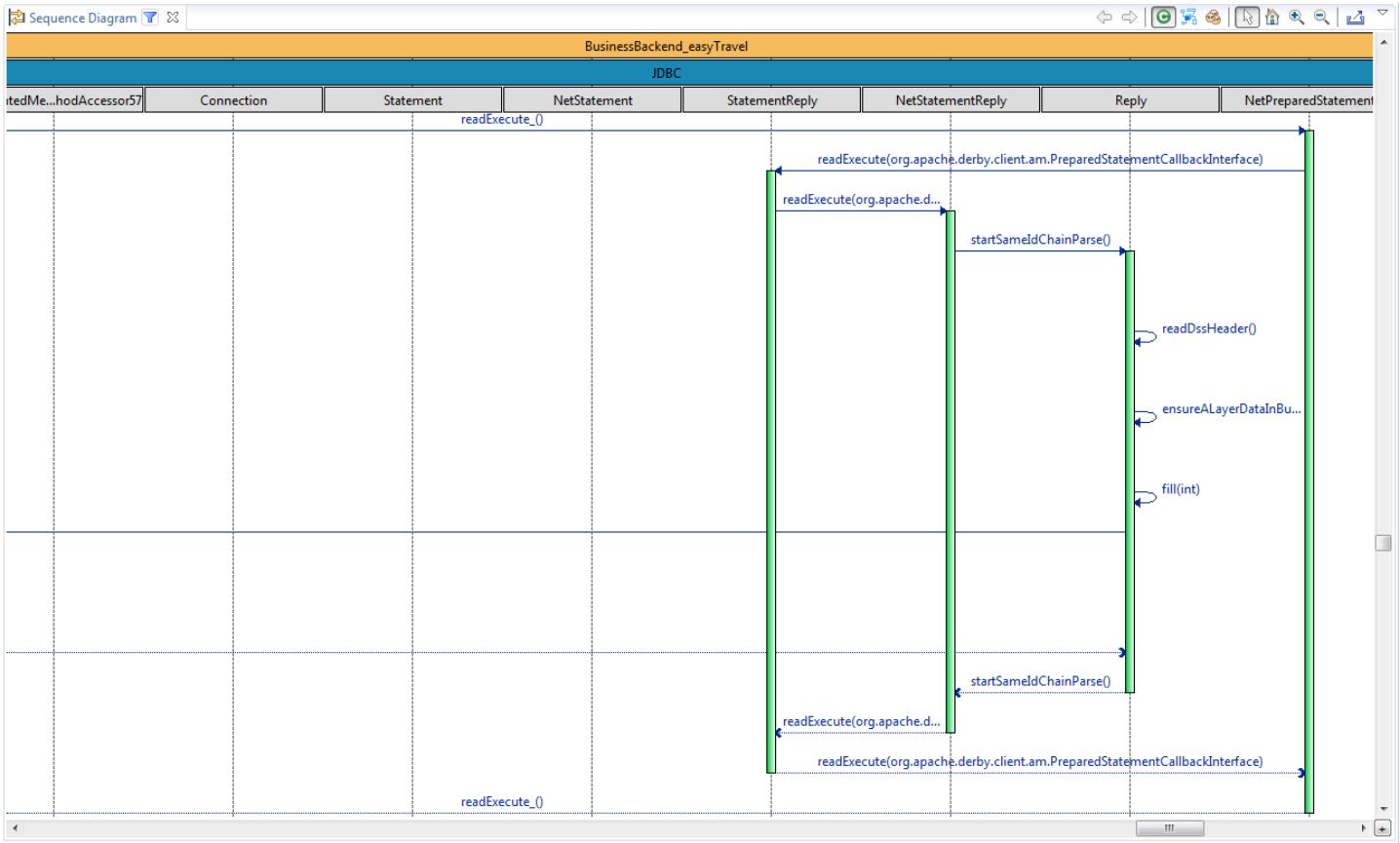
Detail levels

Use the toolbar buttons to select the detail level:

- Classes
- APIs
- Agents

Classes detail level

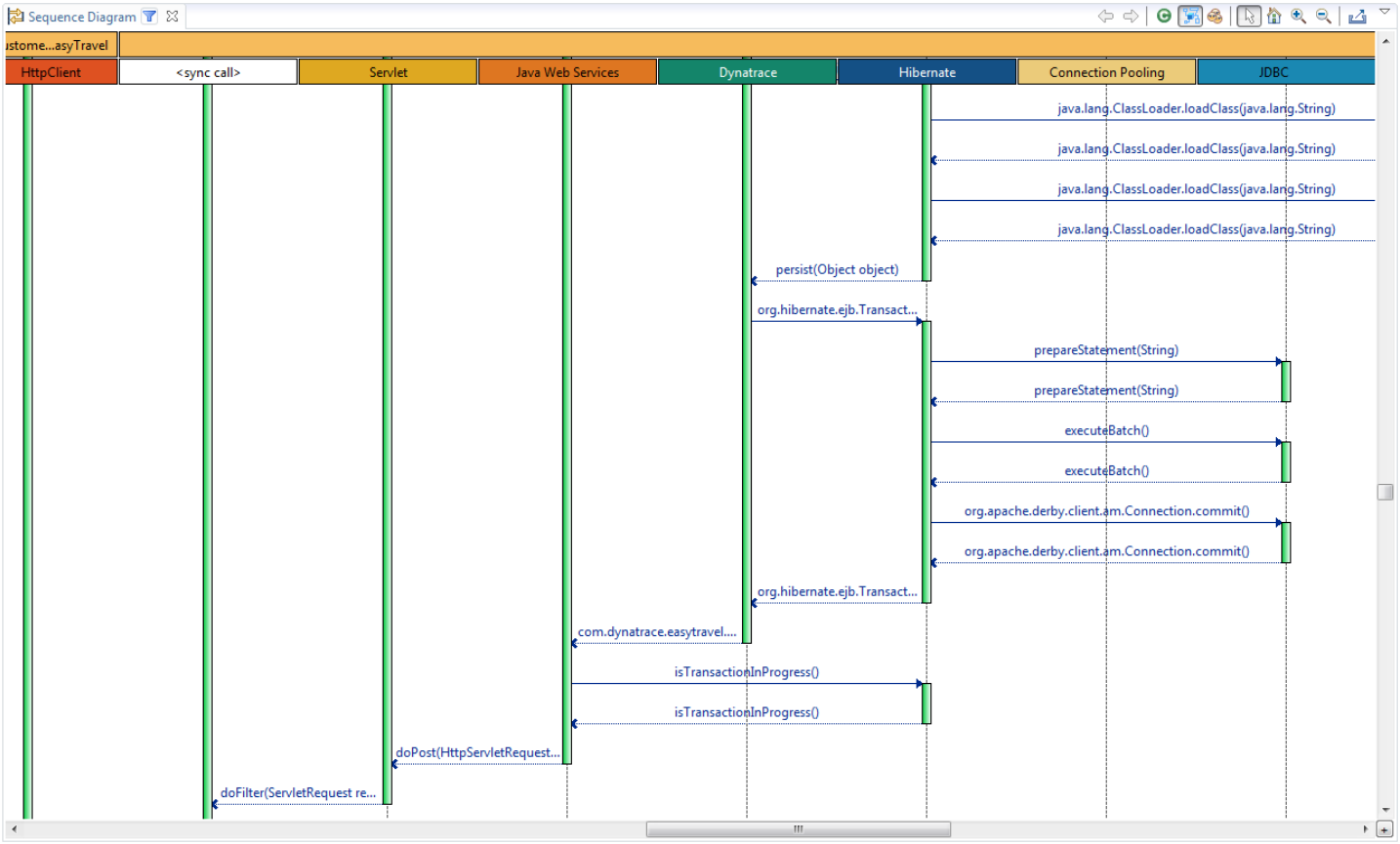
In the Classes Detail Level, the **Sequence Diagram** dashlet shows all method calls contained in the PurePath.



Classes detail level

APIs detail level

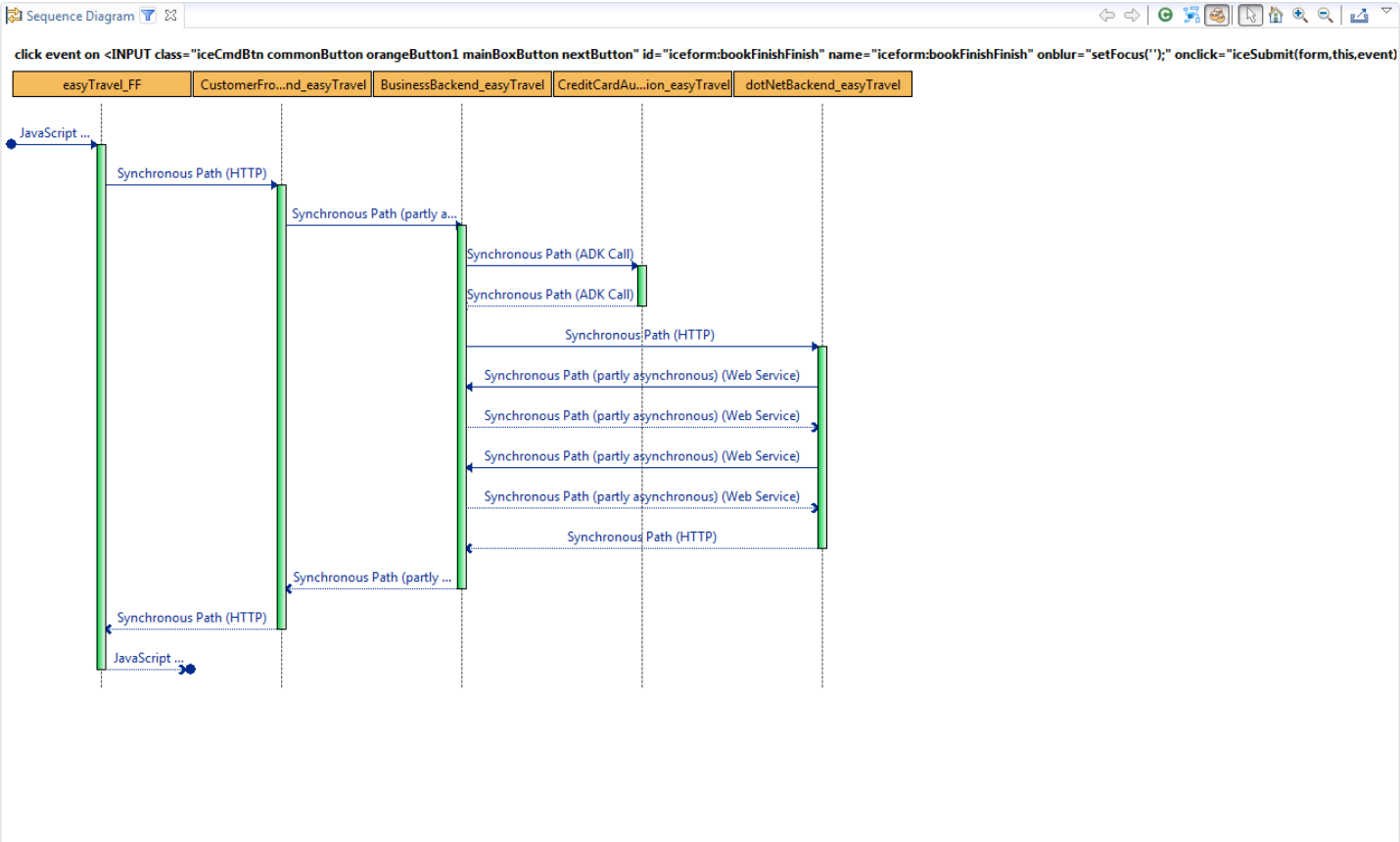
In the APIs Detail Level, the dashlet shows only method calls that cross the border to another API or Agent.



APIs detail level

Agents detail level

In the Agents Detail Level, the dashlet shows only method calls that cross the border to another AppMon Agent.



Agents detail level

Tagged Web Requests dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Grouping

Metrics

Aggregation options

Related pages

Performance specific dashlets

Load Test Overview dashboard

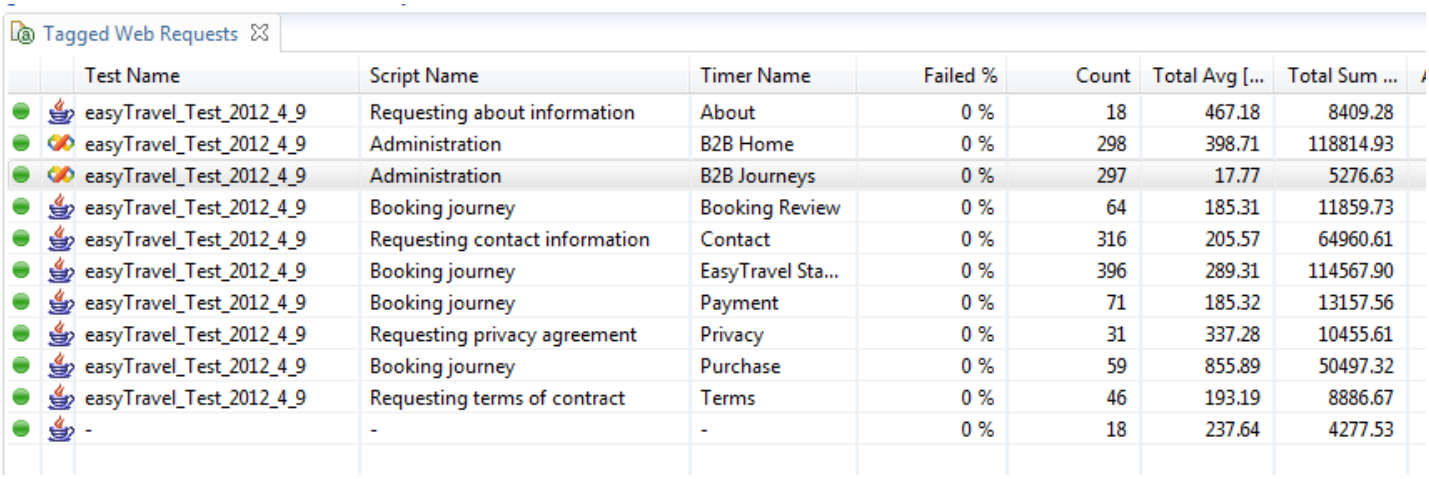
Load testing and monitoring tool integration

Overview

Use the **Tagged Web Requests** dashlet to analyze tagged web requests.

The load testing integration allows you to specify a **Timer Name**. You can also display additional context information such as **Script Name**, **Test Name**, or **Virtual User**. Columns are added to the default view to display this information. The following figure shows the **Tagged Web Request** dashlet with **Test Name** and **Script Name** added to the default **Timer Name** grouping.

Tagged Web Requests Dashlet



Test Name	Script Name	Timer Name	Failed %	Count	Total Avg [...	Total Sum ...
easyTravel_Test_2012_4_9	Requesting about information	About	0 %	18	467.18	8409.28
easyTravel_Test_2012_4_9	Administration	B2B Home	0 %	298	398.71	118814.93
easyTravel_Test_2012_4_9	Administration	B2B Journeys	0 %	297	17.77	5276.63
easyTravel_Test_2012_4_9	Booking journey	Booking Review	0 %	64	185.31	11859.73
easyTravel_Test_2012_4_9	Requesting contact information	Contact	0 %	316	205.57	64960.61
easyTravel_Test_2012_4_9	Booking journey	EasyTravel Sta...	0 %	396	289.31	114567.90
easyTravel_Test_2012_4_9	Booking journey	Payment	0 %	71	185.32	13157.56
easyTravel_Test_2012_4_9	Requesting privacy agreement	Privacy	0 %	31	337.28	10455.61
easyTravel_Test_2012_4_9	Booking journey	Purchase	0 %	59	855.89	50497.32
easyTravel_Test_2012_4_9	Requesting terms of contract	Terms	0 %	46	193.19	8886.67
-	-	-	0 %	18	237.64	4277.53

Tagged Web Requests dashlet

Grouping

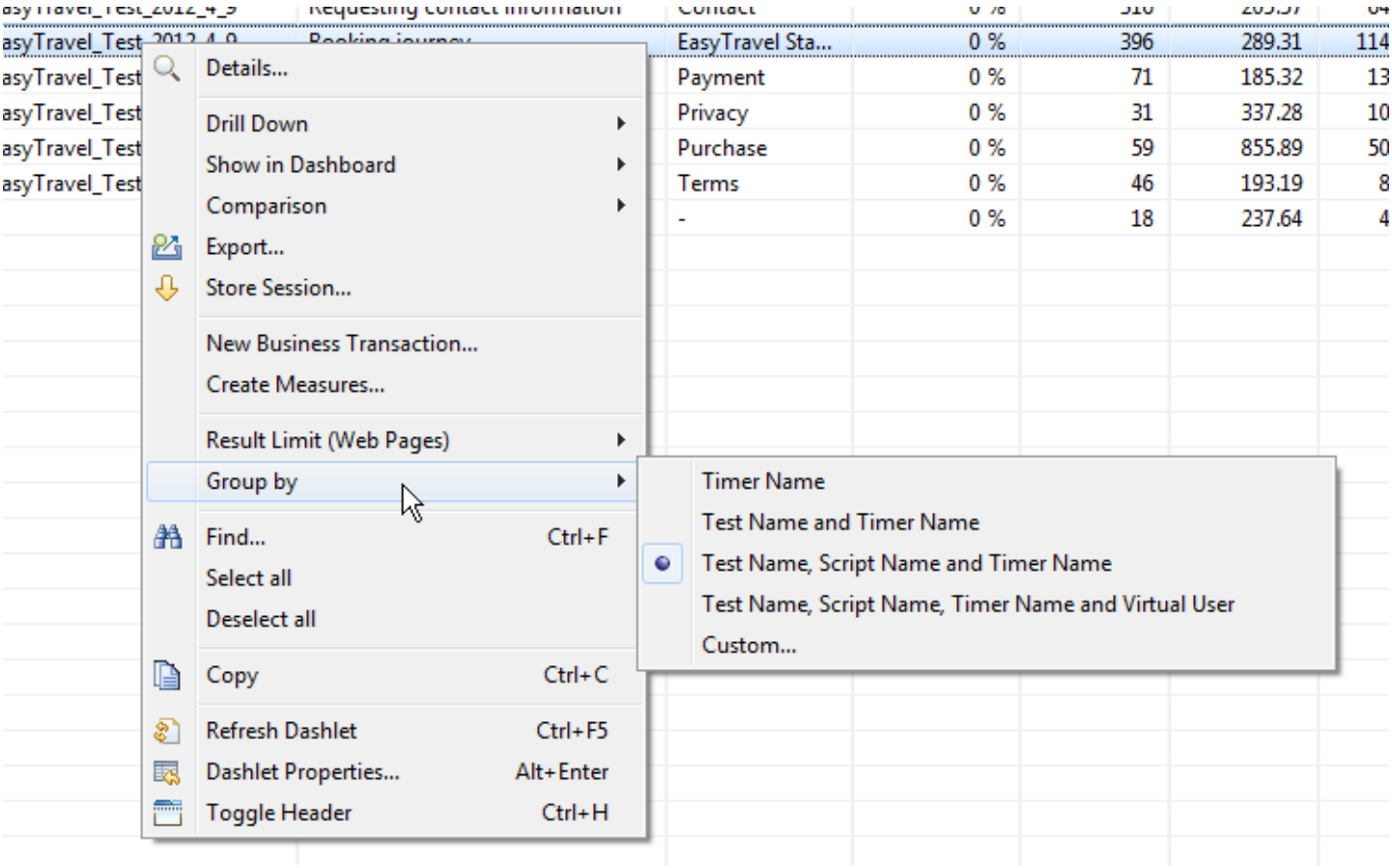
You can set different levels of grouping to analyze the data in a number of different granularities. You can select the groupings:

- Via the **grouping icons** at the top right corner of the dashlet.
- Through the **Group by** option on the context menu.
- In the **Analysis** tab of the **Dashlet Properties** dialog box.

Several predefined groupings are provided as well as the option to define a custom grouping.

The figure shows the options accessed through the context menu.

Group By Options



Group by options

Metrics

The following columns display by default to provide metrics for tagged web requests:

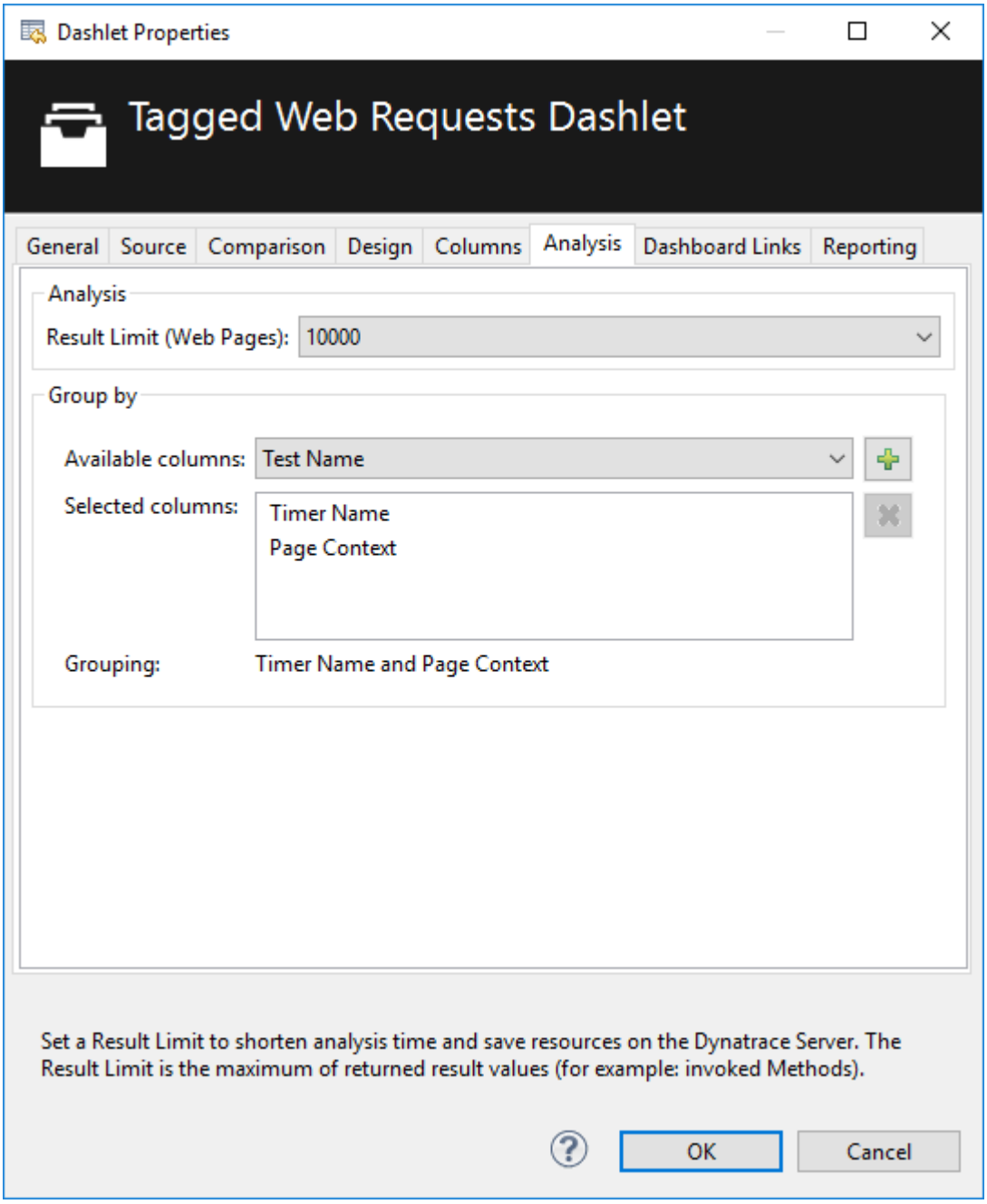
- **Timer Name**: The identifier for the web page, provided by a load test tool.
- **Page Context**: The identifier for a part of the web page, provided by a load test tool.
- **Count**: The number of times the web page has been rendered.
- **Total Avg [ms]**: The average overall time, in milliseconds, spent to render the web page.
- **Total Min [ms]**: The minimum overall time, in milliseconds, spent to render the web page.
- **Total Max [ms]**: The maximum overall time, in milliseconds, spent to render the web page.
- **Total Sum [ms]**: The accumulated overall time, in milliseconds, spent to render the web page.
- **Avg [bytes] Sent**: The average number of bytes sent as a response from the servlet.
- **Min [bytes] Sent**: The minimum number of bytes sent as a response from the servlet.
- **Max [bytes] Sent**: The maximum number of bytes sent as a response from the servlet.
- **Sum [bytes] Sent**: The accumulated number of bytes sent as a response from the servlet.
- **Avg [bytes] Rcvd**: The average number of bytes received as a request to the servlet.

- **Avg [bytes] Rcvd**: The minimum number of bytes received as a request to the servlet.
- **Avg [bytes] Rcvd**: The maximum number of bytes received as a request to the servlet.
- **Avg [bytes] Rcvd**: The accumulated number of bytes received as a request to the servlet.
- **CPU Time [ms]**: The average CPU time, in milliseconds, required to render the web page.
- **CPU Time [ms]**: The minimum overall CPU time, in milliseconds, required to render the web page.
- **CPU Time [ms]**: The maximum overall CPU time, in milliseconds, required to render the web page.
- **CPU Time [ms]**: The accumulated overall CPU time, in milliseconds, required to render the web page.

Aggregation options

You can define the aggregation type in the **Analysis** tab of the **Dashlet Properties** dialog box. Besides the predefined groupings, you can also define a custom grouping by adding all columns that should be part of the grouping.

Tagged Web Requests Dashlet Properties



Tagged Web Requests dashlet properties

Web Requests dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Performance metrics

Analysis options

Context menu options

Sensor Configuration

Performance tuning guidelines

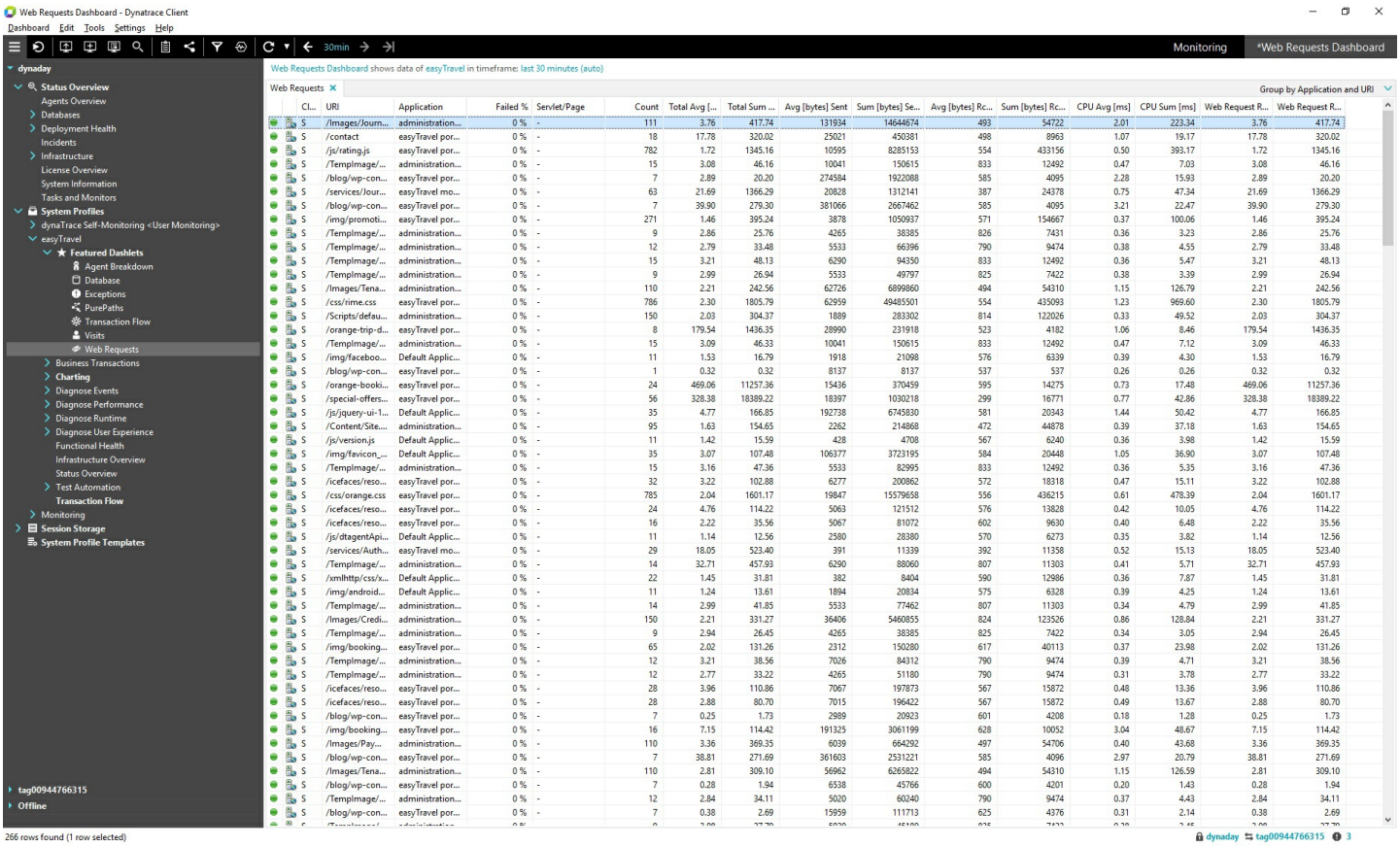
Related pages

Performance specific dashlets

Overview

For diagnosing and tuning software components that handle web requests like servlets and JSPs, AppMon uses dynamic bytecode instrumentation to capture performance metrics, including Java remoting metrics, across horizontal and vertical application layers. A central AppMon Server correlates resulting measurements to PurePaths Dashlet.

The **Web Requests** dashlet displays the collected information about web requests.



Web Request dashlet

Performance metrics

These columns are available in the dashlet. Not all columns display by default. To select which columns to display, right-click a column head and select or clear the columns as needed.

Column	Description
Error State	The first, unlabeled column displays icons that indicate the overall error state of the aggregation. For example, a green icon indicates that no errors occurred in the aggregated line.
Platform	The second, unlabeled column displays icons that identify the software platform that executed the web request: Java or .NET.
Client or Server	Indicates whether the client side (**C**), server side (**S**) or both sides (**C/S**) of the web request call were detected.
URI	The part of the URL that identifies the web request.

Application	The application associated with the web request.
Query	The query part that is contained in the web request URL.
Tier	The tier (Agent Group) that processed the web request.
Web Application ID	For servlet requests, the name of the web application as described in the `web.xml` descriptor. For ASP.NET pages, the name of the virtual application.
Servlet/Page	For servlet requests, the name of the servlet as specified in the `web.xml` descriptor. For ASP.NET pages, the name of the requested page.
Failed %	The percentage of failed web requests.
Count	The number of times the web request was called.
Total Avg [ms]	The average time, in milliseconds, spent on the web request.
Total Sum [ms]	The accumulated time, in milliseconds, spent on the web request.
Avg [bytes] Sent	The average number of bytes sent as a response from the web request.
Sum [bytes] Sent	The accumulated number of bytes sent as a response from all web requests.
Avg [bytes] Rcvd	The average number of bytes received as a request from the web request.
Sum [bytes] Rcvd	The accumulated number of bytes received as a request from all web requests.
CPU Avg [ms]	The average CPU time, in milliseconds, used for all invocations of the web request.
CPU Sum [ms]	The accumulated CPU time, in milliseconds, used for all invocations of the web request.
Total Min [ms]	The minimum time, in milliseconds, spent in a call to this web request.

Total Max [ms]	The maximum time, in milliseconds, spent in a call to this web request.
Exec Avg [ms]	The average time, in milliseconds, spent to call the web request, excluding time spent in nested calls to other methods.
Exec Min [ms]	The minimum time, in milliseconds, spent in a call to this web request, excluding time spent in nested calls to other methods.
Exec Max [ms]	The maximum time, in milliseconds, spent in a call to this web request, excluding time spent in nested calls to other methods.
Exec Sum [ms]	The accumulated time, in milliseconds, spent for all previous calls of this web request, excluding time spent in nested calls to other methods.
Min [bytes] Sent	The minimum number of bytes sent as a response from the web request.
Max [bytes] Sent	The maximum number of bytes sent as a response from the web request.
Min [bytes] Rcvd	The minimum number of bytes received as a request to the web request.
Max [bytes] Rcvd	The maximum number of bytes received as a request to the web request.
CPU Min [ms]	The minimum CPU time, in milliseconds, used for an invocation of this web request.
CPU Max [ms]	The maximum CPU time, in milliseconds, used for an invocation of this web request.
Web Request Response Time Avg [ms]	The average time, in milliseconds, spent on this web request, including (if applicable) Servlet 3 asynchronous processing time. Asynchronous Servlet 3 web requests are indicated by the Error State icon  .
Web Request Response Time Sum [ms]	The accumulated time, in milliseconds, spent on this web request, including (if applicable) Servlet 3 asynchronous processing time. Asynchronous Servlet 3 web requests are indicated by the Error State icon  .
Web Request Response Time Min [ms]	The minimum time, in milliseconds, spent in a call to this web request, including (if applicable) Servlet 3 asynchronous processing time. Asynchronous Servlet 3 web requests are indicated by the Error State icon  .

Web Request Response Time Max [ms]	The maximum time, in milliseconds, spent in a call to this web request, including (if applicable) Servlet 3 asynchronous processing time. Asynchronous Servlet 3 web requests are indicated by the Error State icon  .

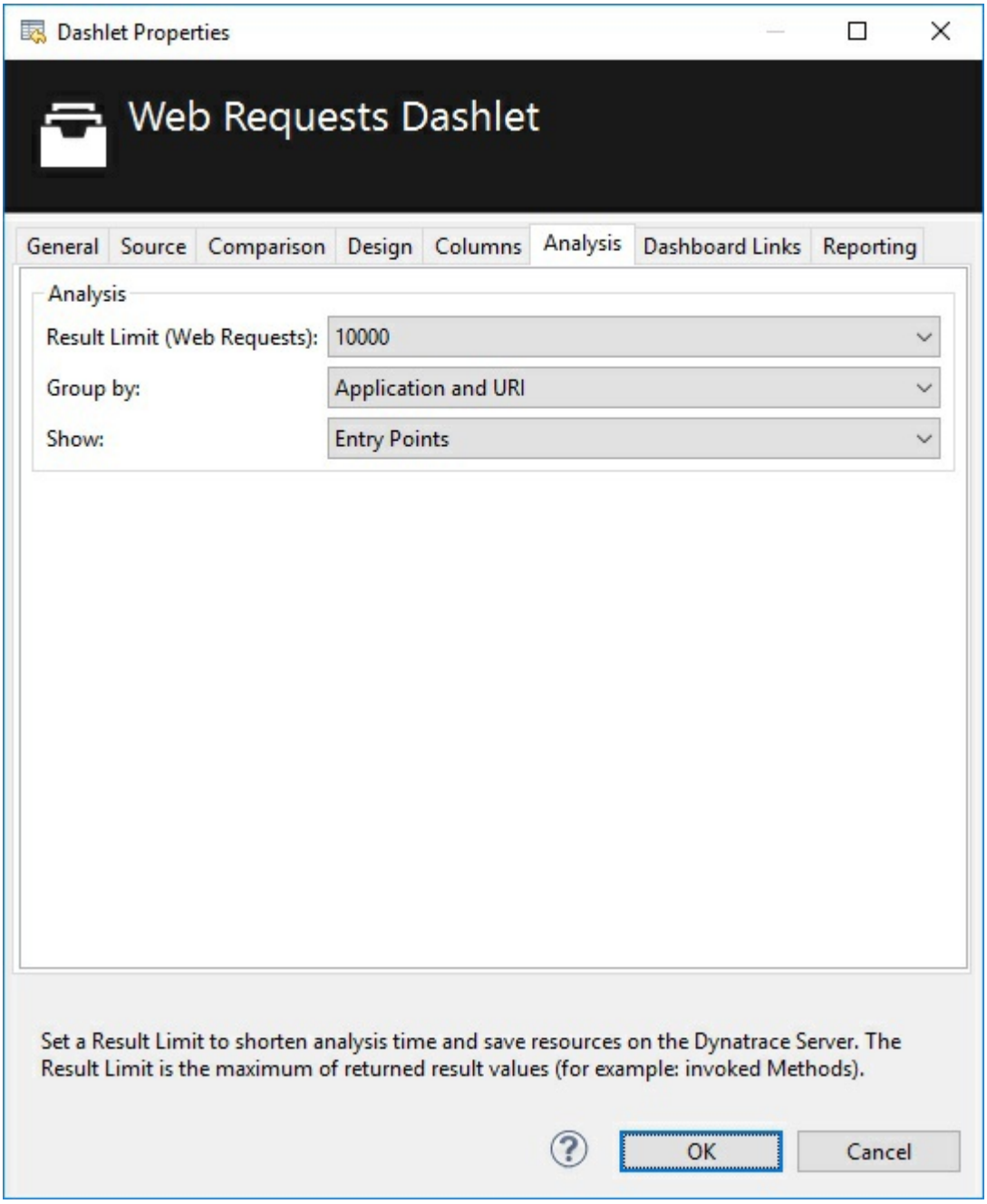
Note

Due to limitations in the .NET Framework, the metrics related to Bytes Sent cannot be calculated for .NET and are set to zero.

Analysis options

In the **Analysis** tab of the **Dashlet Properties** dialog box, you can customize the **Show**, **Result Limit**, and **Group by** options.

Web Request Dashlet Properties



Web Request dashlet properties

Show

- **All**: Show all web requests.
- **Entry Points**: Show only web requests that are entry points.

Result Limit

The **Result Limit** is the maximum number of returned result values (e.g. invoked methods). Set a limit to shorten analysis time and save resources on the AppMon Server.

Group by

You can specify different levels of grouping to analyze the data in different granularities:

- **Application, URI and Query**: Group by the application name defined in the System Profile, URI, and query.
- **Application and URI**: Group by application name and URI. This is the default setting.
- **Tier, URI and Query**: Group by tier (the name of the Agent Group), URI, and query.
- **Tier and URI**: Group by tier and URI.
- **Web Application ID**: Group by web application ID. For Java, this is the display name defined in `web.xml`. For ASP.NET, it is the virtual application name.
- **Web Application ID and Servlet-Name**: Group by the web application ID and the name of the servlet or page.

Context menu options

The context menu includes the following options:

- **Drill Down**: Right-click a PurePath and select a dashlet from the cascading menu to display information for just that PurePath in the selected dashlet.
- **Show in Dashboard**: Right-click a PurePath or select multiple PurePaths, then right-click the dashlet and select from the cascading menu to open a dashboard or create a new dashboard filtered for the selected PurePath.
- **Add Snapshot to Dashboard**: Display the dashlet in the selected dashboard with the time frame automatically set to **last 30 minutes**.
- **Comparison**: Use the options on the cascading menu to configure a Compare Source, select whether to show data from the primary source or compare source, and compare data from the two sources.
- **Export**: Open the **Export Session** dialog box to export the selected PurePaths to a user-defined session file.
- **Store Session**: Create a stored session on the AppMon Server.
- **New Business Transaction**: Open the **Business Transaction Editor** to create a new Business Transaction based on the selection-related measures. This option is only available during realtime sessions.

- **Create Measures**: Open the **Measure Configuration** dialog box to add measures to a subscription. This option is only available during realtime sessions.
- **Result Limit**: Set a result limit to shorten analysis time and save resources.
- **Show**: Show **All** web requests or only **Entry Points**.
- **Group by**: Select a level of grouping, as described above.

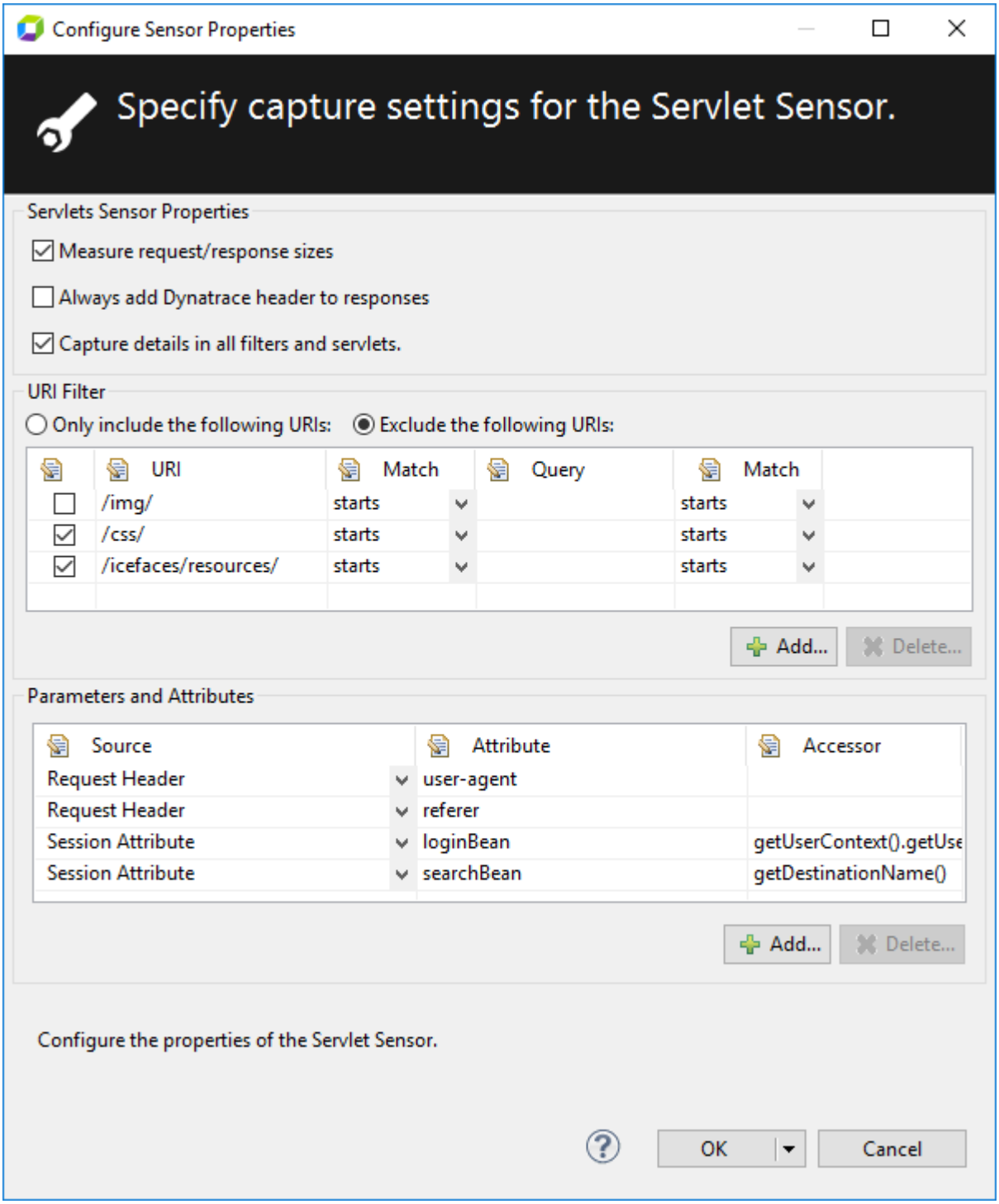
Sensor Configuration

Use the **Configure Sensor Properties** dialog box for the Servlet Sensor or ASP.NET Sensor to configure the URI filter to reduce the number of servlets that should be analyzed.

The filter can:

- **Include** or **exclude** servlet URIs.
- Specify whether AppMon should count the number of bytes in the requests and responses.
- Select whether to add the AppMon HTTP Header to every Servlet/ASP.NET request.
- Define which attributes of servlet invocations should be captured. By default, no attributes are captured.

Servlet Sensor Configuration



Always add AppMon header to responses

This option instructs the Servlet Sensor or ASP.NET Sensor to always send the AppMon HTTP header as part of the response back to the browser. By default, the AppMon HTTP header is only sent when the AppMon header is retrieved as part of the web request. This is usually the case when using an integration with testing or monitoring tools that use the AppMon header to identify the PurePath for the requests executed by these tools. Turning this feature on enables tools that are not able to send the HTTP header with each request to get the unique PurePath ID for every request that is processed by AppMon.

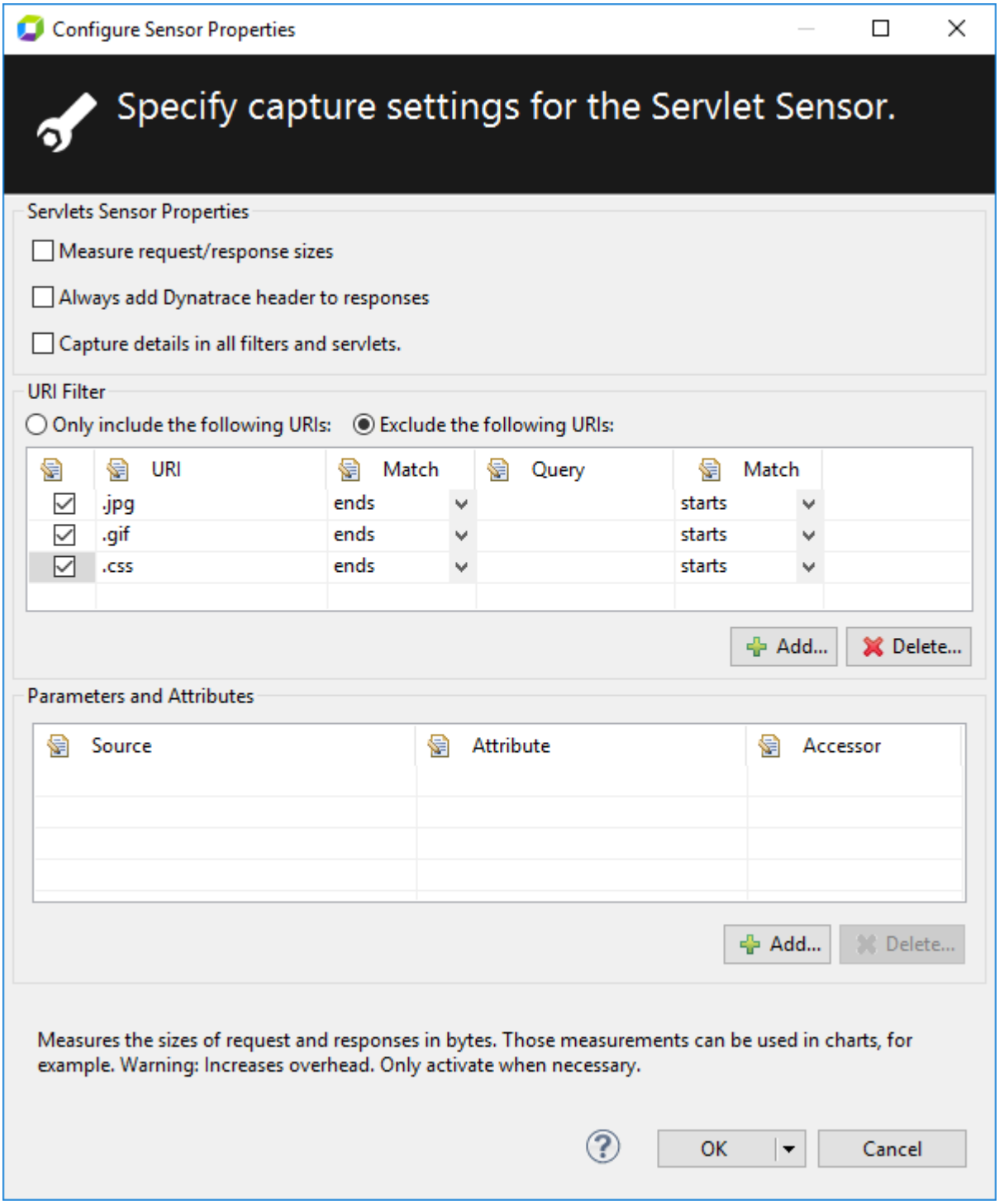
Parameters and attributes

You can use the **Parameters and Attributes** table to configure the Sensor to capture the following **Source** types:

- **Parameter**: Captures the form parameters of the request if the given tring is a substring of the parameter's name (ASP.NET and Java servlet).
- **Request Header**: Captures the HTTP headers sent with the request if the given tring is a substring of the header's name (ASP.NET and Java servlet).
- **Session Attribute**: Captures the session item for this request if the given tring is a substring of the session item's name (ASP.NET and Java servlet).
- **ServletContext Attribute**: Captures the servlet context attribute for this request if the given string is a substring of the servlet context attribute name (Java servlet only).
- **Request Attribute**: Captures the servlet request attribute for this request if the given string is a substring of the servlet request attribute name (Java servlet only).
- **Cookie**: Captures the cookie value of the request if the given string is a substring of the cookies name (ASP.NET only).
- **Server Variable**: Captures the server variable for the request if the given string is a substring of the server variable name (ASP.NET only).

For Tomcat and WebLogic platforms, servlets providing static content like images are excluded from diagnosis by default. However, the method `doFilter` is instrumented, which may result in creation of PurePaths for servlets that create static content on other platforms. To exclude such servlets, which usually listen on a URI ending with an image name, add exclusion rules for the expected image types to the Servlet Sensor Properties, as shown in the following figure.

Exclude Static Content



Performance tuning guidelines

Follow these guidelines to tune servlet performance:

- Minimize Java synchronization in servlets.
- Don’t use the single thread model for servlets.
- Use the servlet’s `init()` method to perform expensive one-time initialization.
- Avoid using `System.out.println()` calls.

- Create sessions sparingly. Session creation is not free. If a session is not required, do not create one.
- Use `javax.servlet.http.HttpSession.invalidate()` to release sessions when they are no longer needed.
- Keep session size small, to reduce response times. If possible, keep the size below 7 KB.
- Use the directive `<%page session="false"%>` in JSPs to prevent the application server from automatically creating sessions when they are not necessary.
- Avoid large object graphs in an `HttpSession`. They force serialization and add computational overhead. Generally, do not store large objects as `HttpSession` variables.
- Don't cache transaction data in an `HttpSession`. Access to data in an `HttpSession` is not transactional. Do not use it as a cache of transactional data, which is better kept in the database and accessed using entity beans. Transactions will rollback upon failures to their original state. However, stale and inaccurate data may remain in `HttpSession` objects. The application server provides read-only bean-managed persistence entity beans for cached access to read-only data.
- To improve class loading time, avoid having excessive directories in the server `CLASSPATH` setting. Put application-related classes into JAR files.
- Consider that HTTP response times depend on how the keep-alive subsystem and the HTTP server are tuned in general.
- If you are using Solaris 8, optimize SSL by using the `mtmalloc` library that provides a collection of `malloc` routines for concurrent access to heap space. To use `mtmalloc`:
  1. Get patch 111308-03 from Oracle and install it.
  2. Edit the `startserv` script located in `bin/startserv` for your domain, and define the `LD_PRELOAD` environment variable to be `/usr/lib/libmtmalloc.so`.
- Consider that the exact syntax to define an environment variable depends on the shell you use.
- Cache servlet results when possible.
- Deploy applications that do not contain EJBs as WAR files rather than as EAR files.

Web Services dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Web services performance metrics

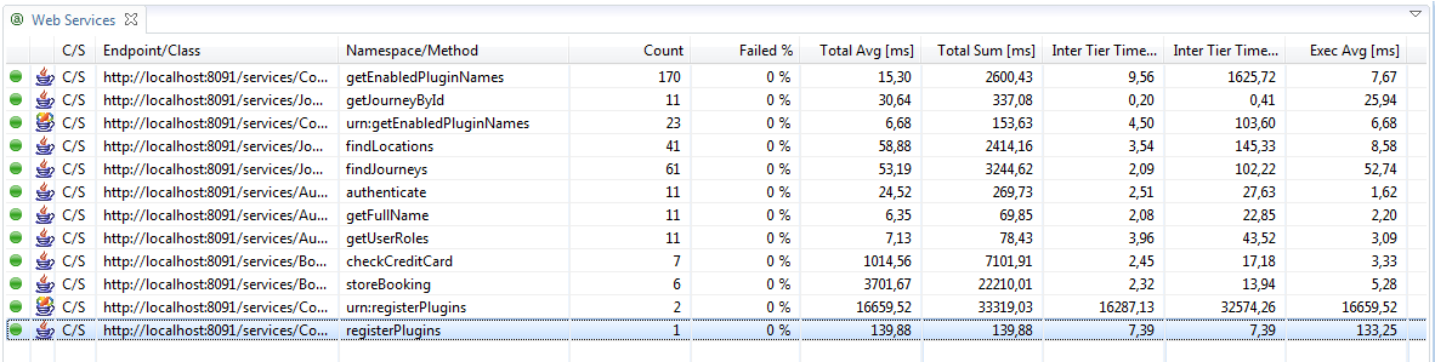
Related pages

Performance specific dashlets

Overview

The **Web Services** dashlet displays performance metrics for web services.

Web Services Dashlet



	C/S	Endpoint/Class	Namespace/Method	Count	Failed %	Total Avg [ms]	Total Sum [ms]	Inter Tier Time...	Inter Tier Time...	Exec Avg [ms]
	C/S	http://localhost:8091/services/Co...	getEnabledPluginNames	170	0 %	15,30	2600,43	9,56	1625,72	7,67
	C/S	http://localhost:8091/services/Jo...	getJourneyById	11	0 %	30,64	337,08	0,20	0,41	25,94
	C/S	http://localhost:8091/services/Co...	urn:getEnabledPluginNames	23	0 %	6,68	153,63	4,50	103,60	6,68
	C/S	http://localhost:8091/services/Jo...	findLocations	41	0 %	58,88	2414,16	3,54	145,33	8,58
	C/S	http://localhost:8091/services/Jo...	findJourneys	61	0 %	53,19	3244,62	2,09	102,22	52,74
	C/S	http://localhost:8091/services/Au...	authenticate	11	0 %	24,52	269,73	2,51	27,63	1,62
	C/S	http://localhost:8091/services/Au...	getFullName	11	0 %	6,35	69,85	2,08	22,85	2,20
	C/S	http://localhost:8091/services/Au...	getUserRoles	11	0 %	7,13	78,43	3,96	43,52	3,09
	C/S	http://localhost:8091/services/Bo...	checkCreditCard	7	0 %	1014,56	7101,91	2,45	17,18	3,33
	C/S	http://localhost:8091/services/Bo...	storeBooking	6	0 %	3701,67	22210,01	2,32	13,94	5,28
	C/S	http://localhost:8091/services/Co...	urn:registerPlugins	2	0 %	16659,52	33319,03	16287,13	32574,26	16659,52
	C/S	http://localhost:8091/services/Co...	registerPlugins	1	0 %	139,88	139,88	7,39	7,39	133,25

Web Services dashlet

Web services performance metrics

The dashlet contains the following columns by default.

Column	Description
Error State	The first, unlabeled column displays icons that indicate the overall error state of the aggregation. For example, a green icon indicates that no errors occurred in the aggregated line.
Platform	The second, unlabeled column indicates the software platform that executed the web request: Java or .NET.
Client or Service	Indicates whether the client side (**C**), server side (**S**) or both sides (**C/S**) of the web service call were detected.
Endpoint and Class	The endpoint of the web service and the class handling requests to the web service.
Namespace and Method	The name of the method or namespace that handled the service request.
Count	The number of times the web service was invoked by the client.
Failed %	The percentage of failed requests.
Total Avg [ms]	The average time, in milliseconds, spent for invocations of the web service.
Total Sum [ms]	The accumulated time, in milliseconds, spent for invocations of the web service.
Inter Tier Time[ms] Avg	The average network latency, in milliseconds, spent for invocations of the web service.
Inter Tier Time [ms] Sum	The accumulated network latency, in milliseconds, spent for invocations of the web service.
Exec Avg [ms]	The average time, in milliseconds, spent for invocations of the web service, excluding time spent for nested method calls.
Exec Sum [ms]	The accumulated time, in milliseconds, spent for invocations of the web service, excluding time spent for nested method calls.
Request Bytes (Avg)	The average number of bytes sent to or from the web service method, for all invocations.
Request Bytes (Sum)	The accumulated number of bytes sent to or from the web service method, for all invocations.

Response Bytes (Avg)	The average number of bytes received by or from the web service method, for all invocations.
Response Bytes (Sum)	The accumulated number of bytes received by or from the web service method, for all invocations.
CPU Avg [ms]	The average CPU time spent for executions of the web services on monitored participants (client, server, or both).
CPU Sum [ms]	The accumulated CPU time spent for executions of the web services on monitored participants (client, server, or both).
Webapplication Id	The application that is associated with this request. See System Profile - Applications.

Event specific dashlets

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Errors dashlet

Exceptions dashlet

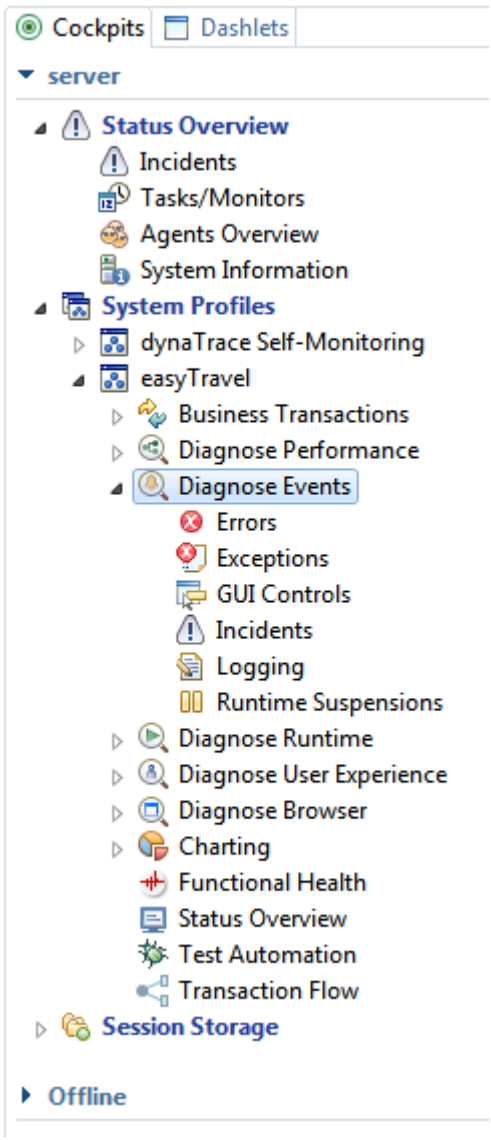
GUI Controls dashlet

Incidents dashlet

Logging dashlet

Runtime Suspensions dashlet

AppMon has many ways to handle events that are raised by the application or by monitoring. Select **Diagnose Events** in the Cockpit to get an overview via event specific dashlets.



Event Category	Description
Errors Dashlet	Overview of all errors that are detected in a specified time frame.
Exceptions Dashlet	Shows all of the exceptions captured by the Exception Sensor.
GUI Controls Dashlet	Shows the PurePaths that were started using WinForms, WPF and Java AppMon Clients.
Incidents Dashlet	Shows the incidents defined in a System Profile and instances where the set rules are violated.
Logging Dashlet	Shows an aggregated view of the logging events, grouped by the class and method that issued the logging event.
Runtime Suspensions Dashlet	Shows the instrumented methods in your PurePaths that are impacted by Garbage Collection.

Errors dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Error overview pane

Error details pane

Related pages

Event specific dashlets

Exceptions dashlet

GUI Controls dashlet

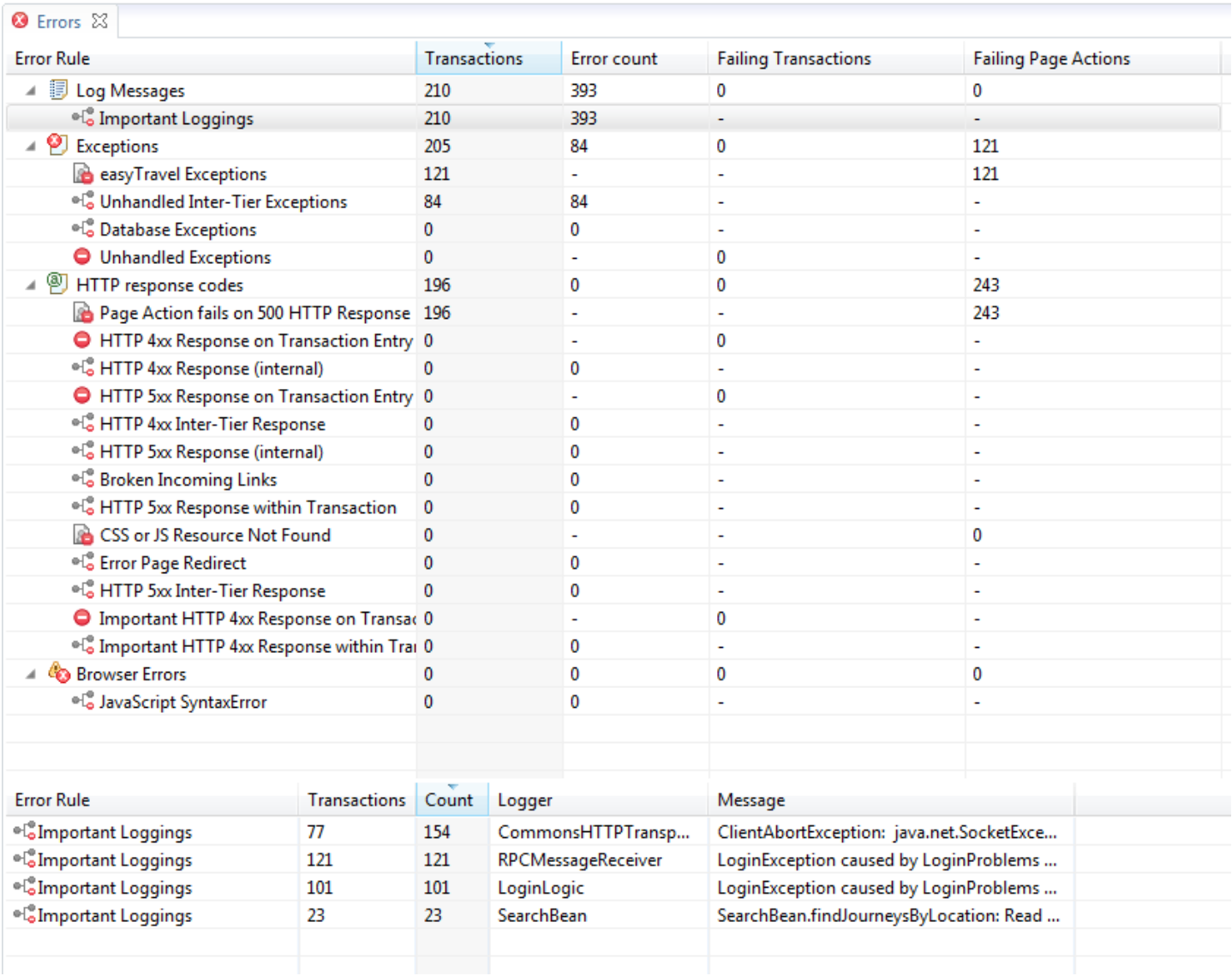
Incidents dashlet

Logging dashlet

Runtime Suspensions dashlet

The **Errors** dashlet provides an overview of all errors that are detected in a specified time frame. For more information, see Error Analysis. Like many other dashlets, the **Errors** dashlet is split into two parts:

- Error overview pane (top) displays configured error rules. For more information, see System Profile - Error Detection.
- Error details pane (bottom) displays details of a selected error category or error rule.



Error overview pane

This pane shows all the detected errors sorted by error categories. By default, errors can be caused by HTTP response codes, exceptions, logging messages or browser errors.

The pane displays the following columns:

- **Error Rule**: Name of the rule that specifies the error
- **Transactions**: Number of the transactions (PurePaths) where errors are detected
- **Error count**: Number of errors that are detected for one particular error category or error rule
- **Failing Transactions**: Number of failed transactions this error category or error rule has caused
- **Failing User Actions**: Number of failed user actions this error category or error rule has caused

Typically, only one of the last three columns contains a value. These represent the impact the user selects in the error rule. In the first HTTP error rule, the impact changes during runtime, which results in values in two of the columns.

Error details pane

This pane shows the details of a row that is selected above. All columns depend on the selected error category or the error rule, except for the following first three columns:

- Error Rule
- Transactions
- Count

The following represents one entry in the error details pane:

- HTTP errors with equal HTTP response code.
- URI and Referrer.
- Exceptions with equal exception class and message.
- Logging errors with equal logger and message.

The following columns display on this pane:

- **Error Rule**: Name of the associated error rule
- **Transactions**: Number of transactions in one error entry. The total transaction count of all rows is the total transaction count in the **Overview** pane.
- **Count**: Number of errors that happened for one type of error entry
- **HTTP response code (HTTP errors only)**: Exact HTTP response code that causes the error
- **URI (HTTP errors only)**: Requested URI that causes the error
- **Referrer (HTTP errors only)**: Referrer to the requested URI
- **Exception Class (exception errors only)**: Class name of the exception that causes the error
- **Logger (logging errors only)**: Name of the logger
- **Message (exception and logging errors only)**: Message that explains the problem

Exceptions dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Aggregation options

Configuration

Context menu options

Related pages

Event specific dashlets

Errors dashlet

GUI Controls dashlet

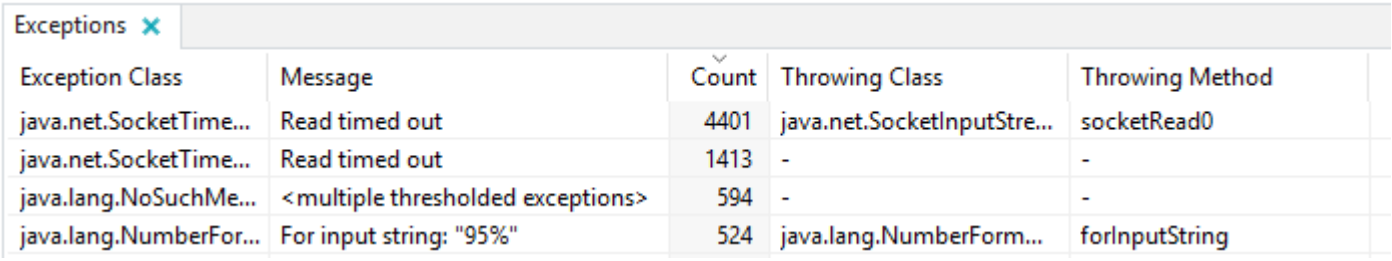
Incidents dashlet

Logging dashlet

Runtime Suspensions dashlet

Overview

The **Exceptions** dashlet uses a novel approach to diagnose exceptions. In Java, exceptions provide error-handling capabilities for applications. AppMon makes use of dynamic bytecode instrumentation to capture throwable classes, messages, and stack traces of java classes that extend `java.lang.Exception`.


Exceptions

Exception Class	Message	Count	Throwing Class	Throwing Method
java.net.SocketTime...	Read timed out	4401	java.net.SocketInputStre...	socketRead0
java.net.SocketTime...	Read timed out	1413	-	-
java.lang.NoSuchMe...	<multiple thresholded exceptions>	594	-	-
java.lang.NumberFor...	For input string: "95%"	524	java.lang.NumberForm...	forInputString

Exceptions

Metrics

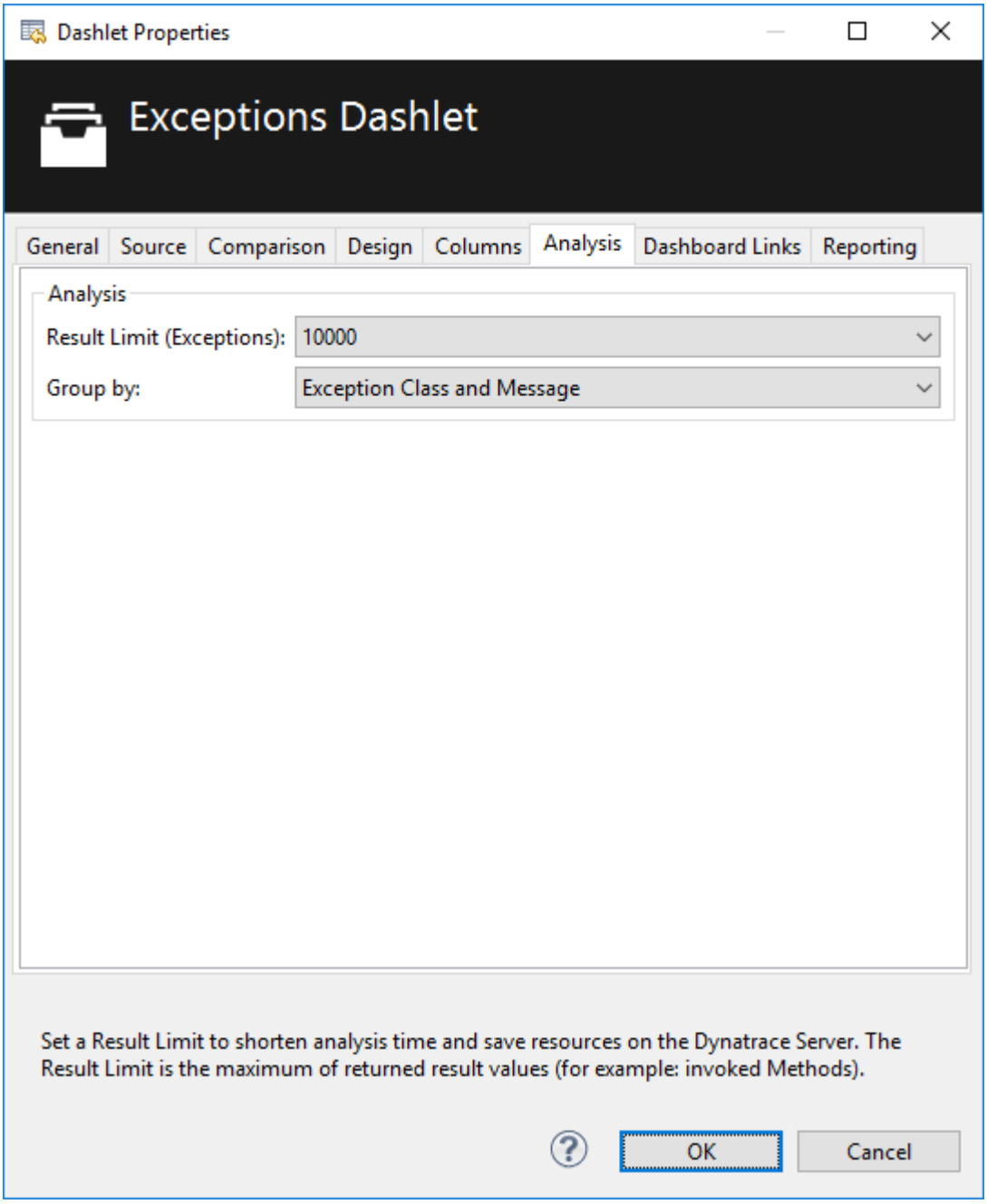
Exception metrics

- **Exception Class**: Full qualified class name of the instantiated exception.
- **Message**: Message of the instantiated exception. This is only available if the grouping is set to *Throwable Class/Message*.
- **AppMon Agent**: Name of the AppMon Agent that recorded the instantiation of the exception.
- **Count**: Number of times an instance of this throwable, with this message, is instantiated, if grouping is set to *Throwable Class/Message.* If the grouping is set to *Throwable Class*, it is the number of times this throwable class is instantiated, regardless of the message text.
- **Throwing Class**: Full qualified name of the throwable class that is used initialize the new instantiated throwable class.
- **Throwing Method**: Signature of the method that instantiates the throwable.

Aggregation options

Click the **Analysis** tab on the **Properties** dialog box to select one of the following aggregations:

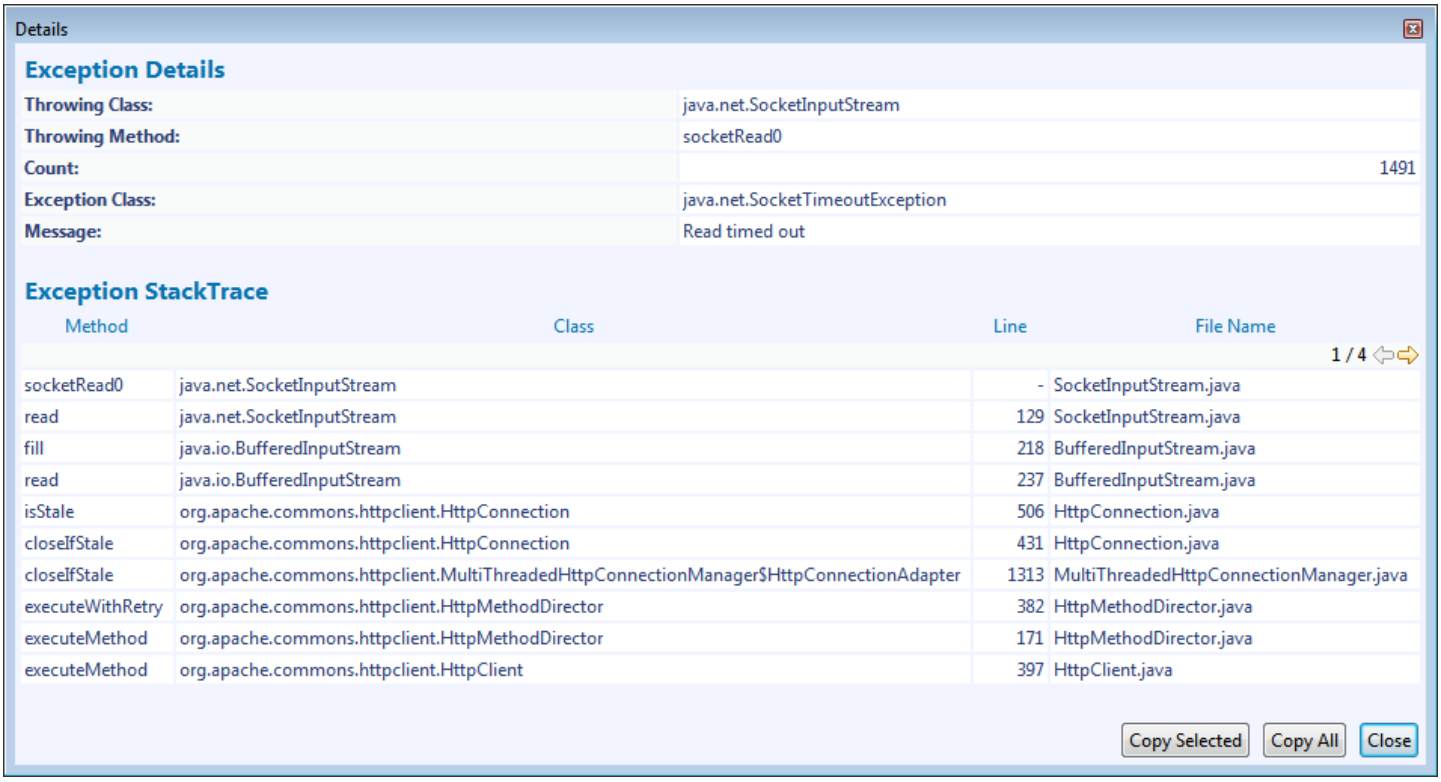
- **Exception Class**: Exception Records are listed in the Diagnostics view and grouped by their throwable classes. They ignore message contents.
- **Exception Class/Message**: Default setting for the Exception view. Exception records are listed in the Diagnostics view and grouped by their throwable classes and messages. Exceptions with the same throwable class, but with varying message contents, are shown as individual exception records.



Exceptions Dashlet

Details

- **Stack Trace**: A list of the methods executed in sequence that leads to the exception.



Exception Details

Configuration

Use the **Exception Sensor Configuration** dialog box to define filter criteria. This targets the set of exceptions to analyze. See Sensor Properties, Java Exception Sensor for more information.

Context menu options

- Subscribe Measures: Opens the **Measure Picker** dialog box to add measures to a subscription. This option is only available during real-time sessions.
- New Business Transaction: Opens the **Business Transaction** dialog box to create a new business transaction based on the Measures. This option is only available during real-time sessions.

GUI Controls dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Aggregation options

Related pages

Event specific dashlets

Errors dashlet

Exceptions dashlet

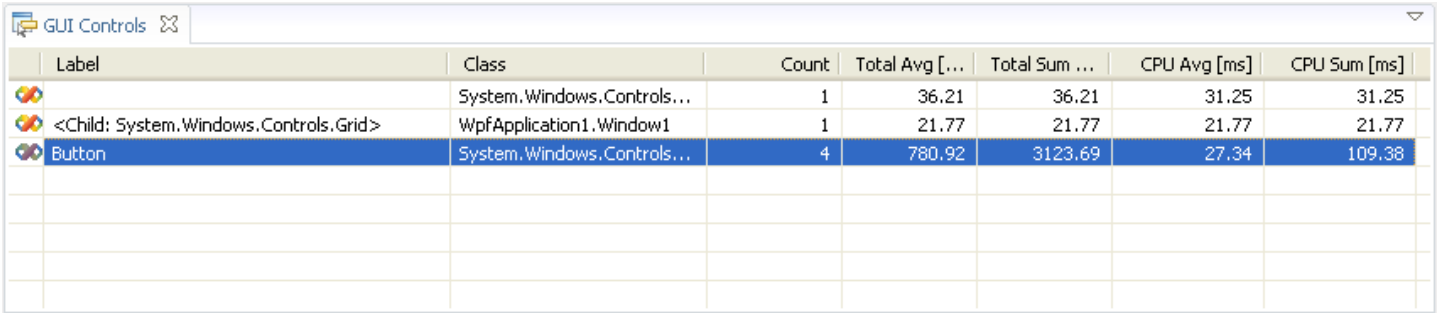
Incidents dashlet

Logging dashlet

Runtime Suspensions dashlet

Overview

The following figure shows the **GUI Controls** dashlet.



Metrics

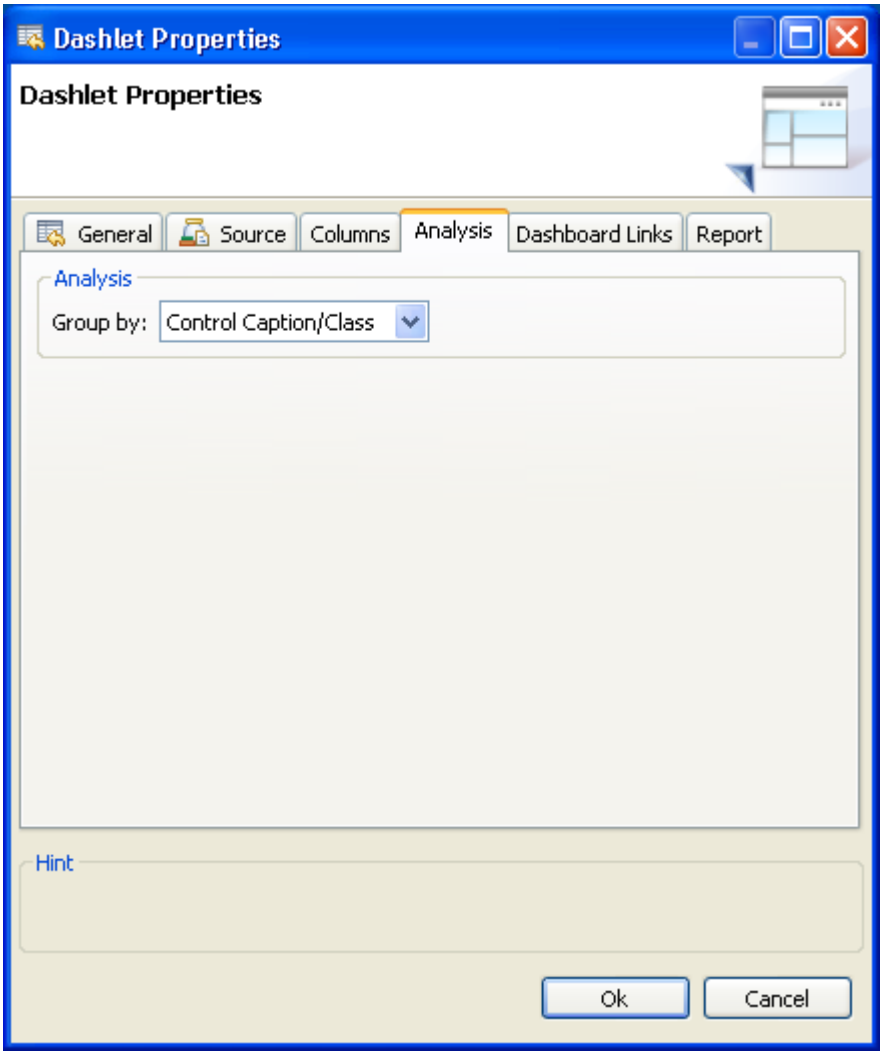
Metrics on the GUI Controls dashlet

- **Platform**: Software platforms (Java/.NET).
- **Label**: Control label.
- **Class**: Control class.
- **Count**: Number of control activations.
- **Total Avg [ms]**: Average time elapsed during the control activation.
- **Total Min [ms]**: Minimum time elapsed during the control activation.
- **Total Max [ms]**: Maximum time elapsed during the control activation.
- **Total Sum [ms]**: Total time elapsed during the control activation.
- **CPU Avg [ms]**: Average CPU time elapsed during the control activation.
- **CPU Min [ms]**: Minimum CPU time elapsed during the control activation.
- **CPU Max [ms]**: Maximum CPU time elapsed during the control activation.
- **CPU Sum [ms]**: Total CPU time elapsed during the control activation.

Aggregation options

Click the **Analysis** tab on the dashlet's **Properties** dialog box to choose the following aggregation options:

- **Control Caption**: Aggregate all controls with the same caption.
- **Control Caption/Class**: Aggregate controls with the same caption if they are declared in the same class.


Dashlet Properties
Dashlet Properties
General
Source
Columns
Analysis
Dashboard Links
Report
Analysis
Group by:
Control Caption/Class
Hint
Ok
Cancel

Incidents dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Dashlet filters and Incident Rules

Incident metrics

Context menu options

Related pages

Event specific dashlets

Errors dashlet

Exceptions dashlet

GUI Controls dashlet

Logging dashlet

Runtime Suspensions dashlet

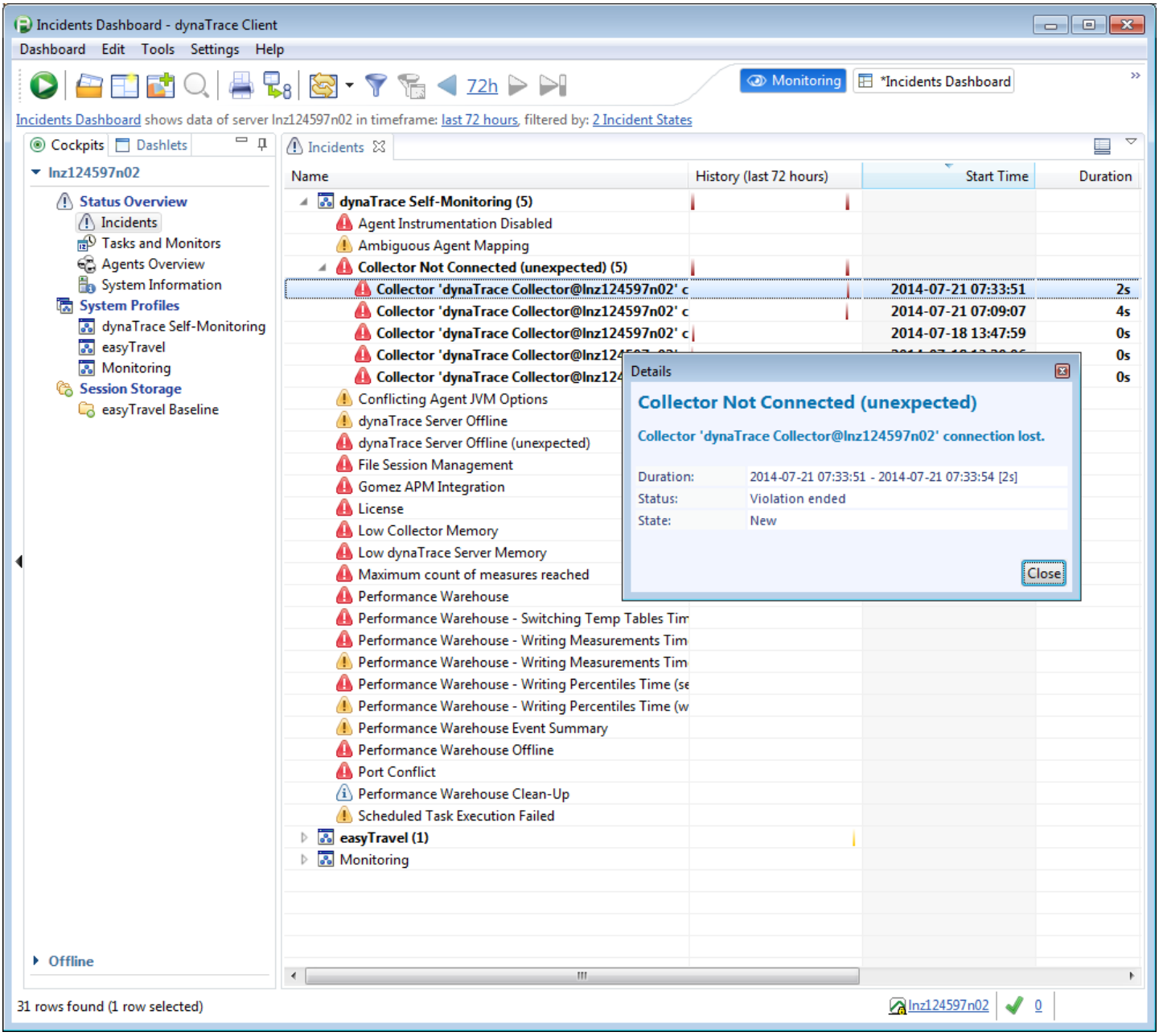
Overview

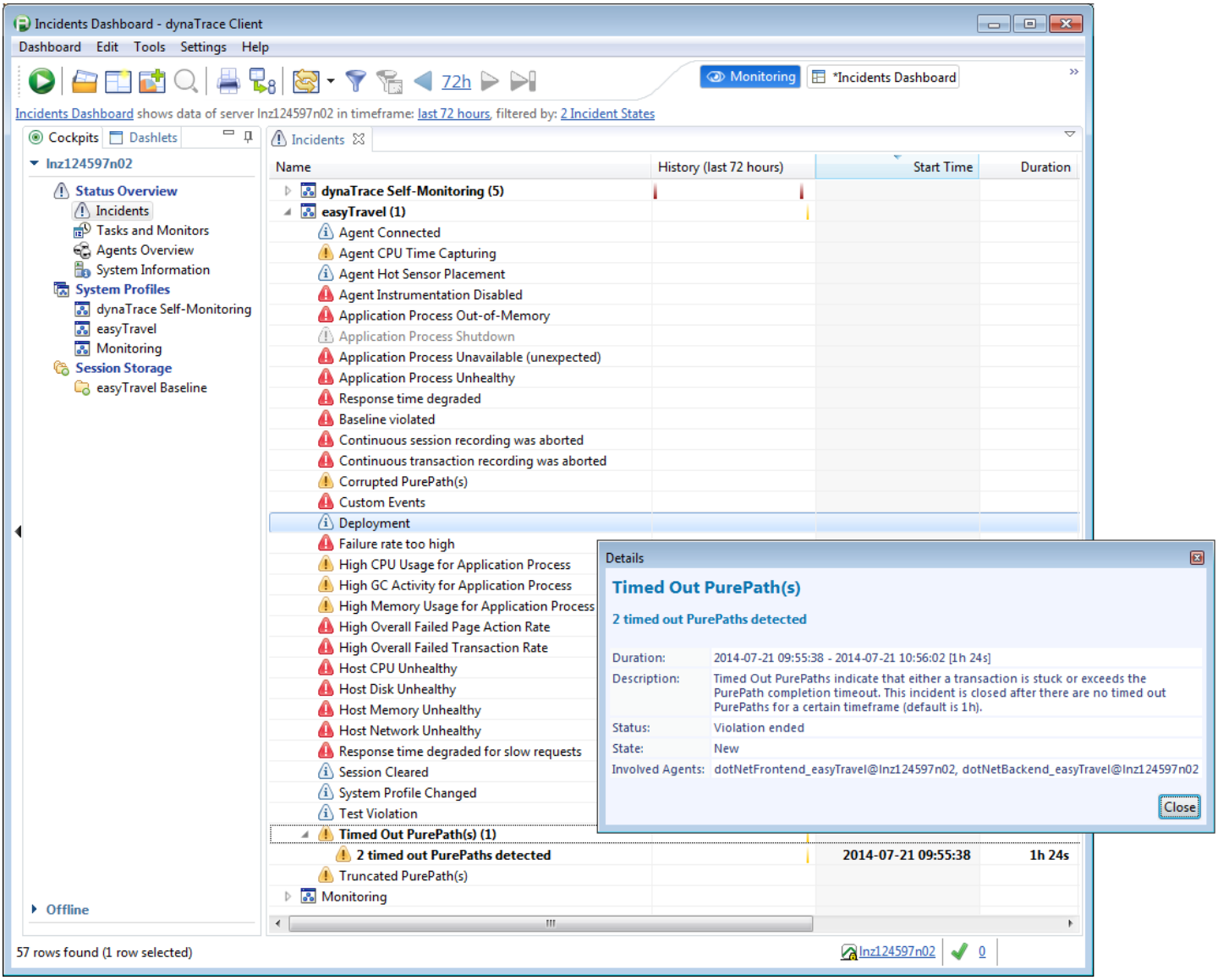
The **Incidents** dashlet displays the incidents defined in a System Profile and violations thereof. See Incidents and Alerting > Configuring Incident Rules for more information on incident rules.

Notification based on incident rules is more flexible than plain threshold notification, because it defines complex conditions that must be met to generate a notification. AppMon generates notifications if two or more measure values exceed their thresholds simultaneously.

Incident notifications are useful starting points for analysis.

Incidents Dashlet / Dashboard

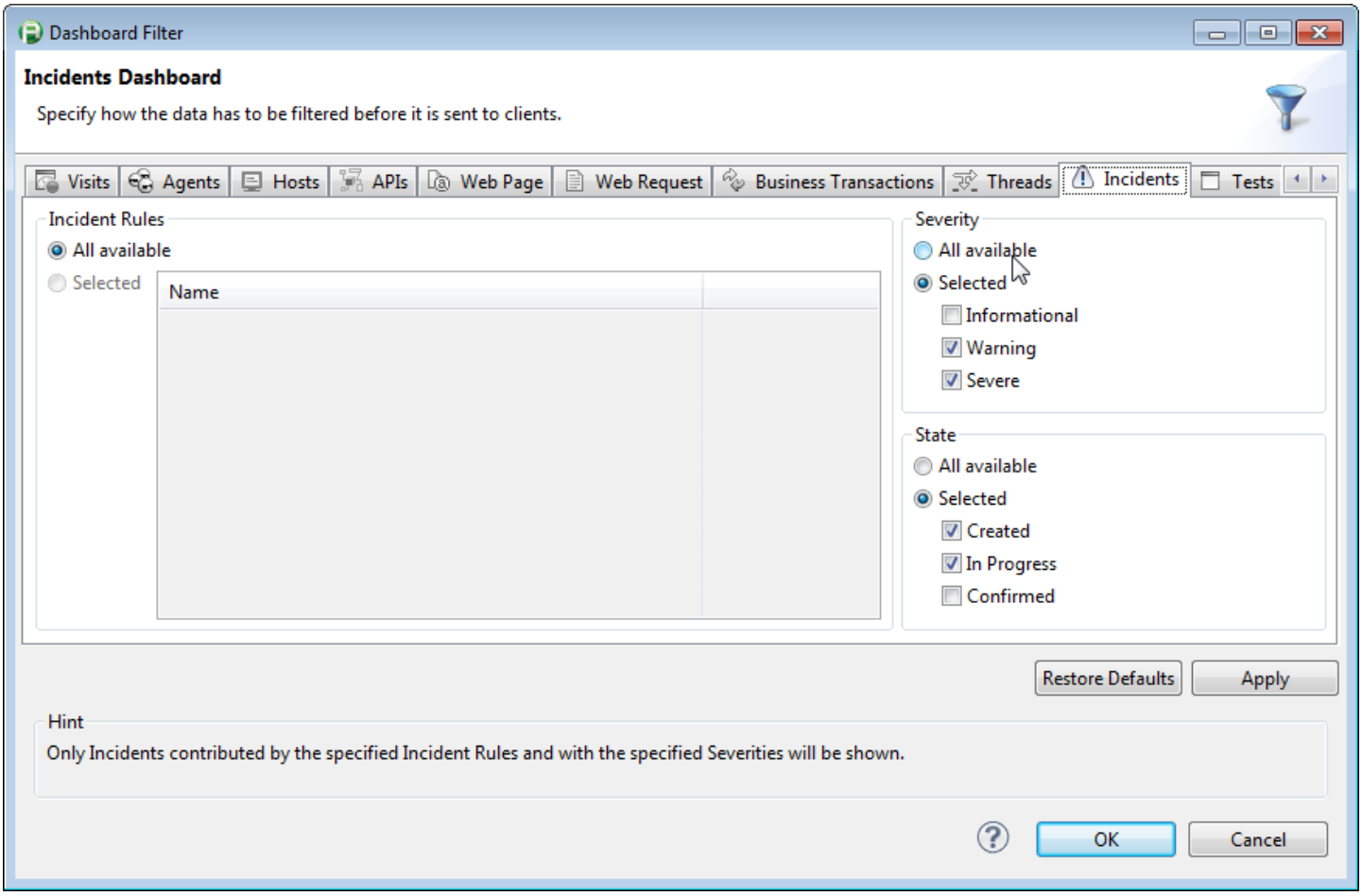


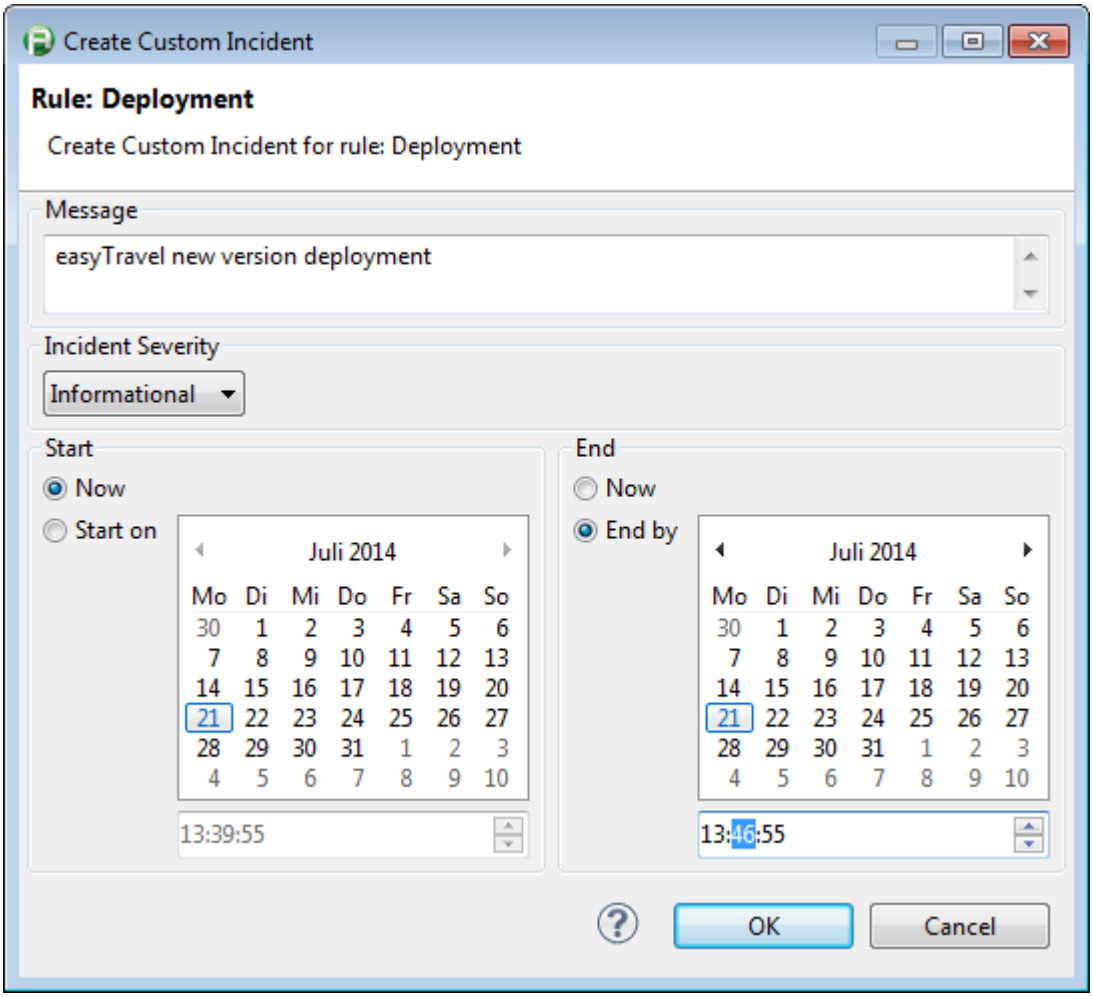


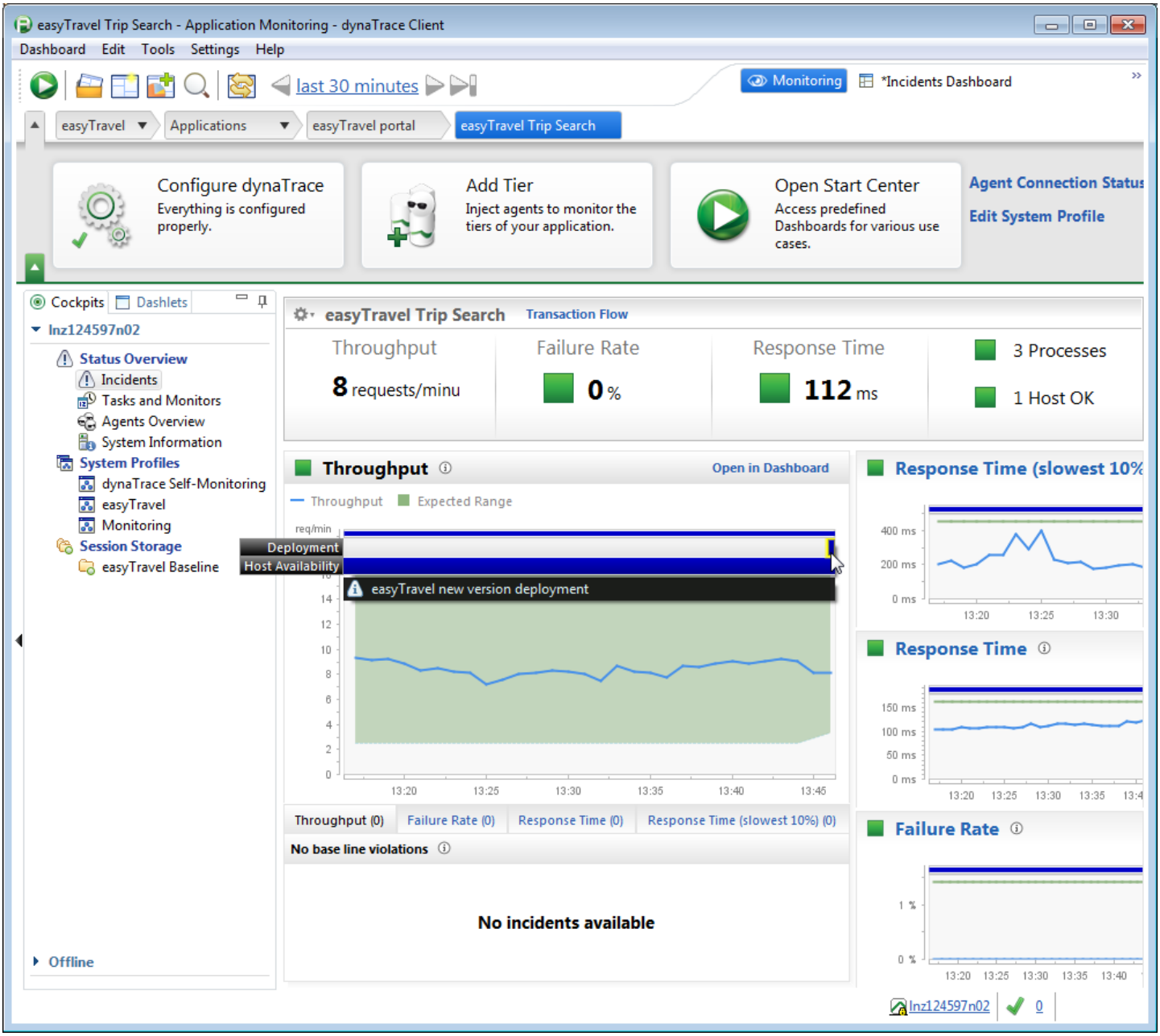
Dashlet filters and Incident Rules

To see information such as the new informational Deployment Incident, switch the dashboard's Incidents filter from default Warning and Severe to All by right-clicking the Incidents dashlet and selecting **Dashlet Properties > Incidents**.

For a new Deployment incident rule, right-click > Create Custom Incident… In our case we got an easyTravel deployment, so you will e.g. see in Baseline Dashboard > Heat Maps that an easyTravel deployment was the reason you got no data for several minutes.


Dashboard Filter
Incidents Dashboard
Specify how the data has to be filtered before it is sent to clients.
Visits
Agents
Hosts
APIs
Web Page
Web Request
Business Transactions
Threads
Incidents
Tests
Incident Rules
All available
Selected
Name
Severity
All available
Selected
Informational
Warning
Severe
State
All available
Selected
Created
In Progress
Confirmed
Restore Defaults
Apply
Hint
Only Incidents contributed by the specified Incident Rules and with the specified Severities will be shown.
OK
Cancel


Create Custom Incident
Rule: Deployment
Create Custom Incident for rule: Deployment
Message
easyTravel new version deployment
Incident Severity
Informational
Start
Now
Start on
Juli 2014
Mo Di Mi Do Fr Sa So
30 1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31 1 2 3
4 5 6 7 8 9 10
13:39:55
End
Now
End by
Juli 2014
Mo Di Mi Do Fr Sa So
30 1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31 1 2 3
4 5 6 7 8 9 10
13:46:55
OK
Cancel



Incident metrics

- **Severity**:
  - Informational
  - Warning
  - Severe
- **Start Time**
- **Duration [s]**: Duration in seconds from the first significant threshold violation until no longer violations happen.
- **End Time**
- **Incident Rule**: Rule that triggered the incident.

- **Message**: Describes the type of the incident.
- **Source**: Represents the category of the raised incident, foe example *License* for AppMon license based events and *Threshold* for defined threshold violations.

Context menu options

For more incident details and to see the conditions that causes the incident to be thrown, go to the **Context** menu and click **Details**.

Logging dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Configuration

Context menu options

Related pages

Event specific dashlets

Errors dashlet

Exceptions dashlet

GUI Controls dashlet

Incidents dashlet

Runtime Suspensions dashlet

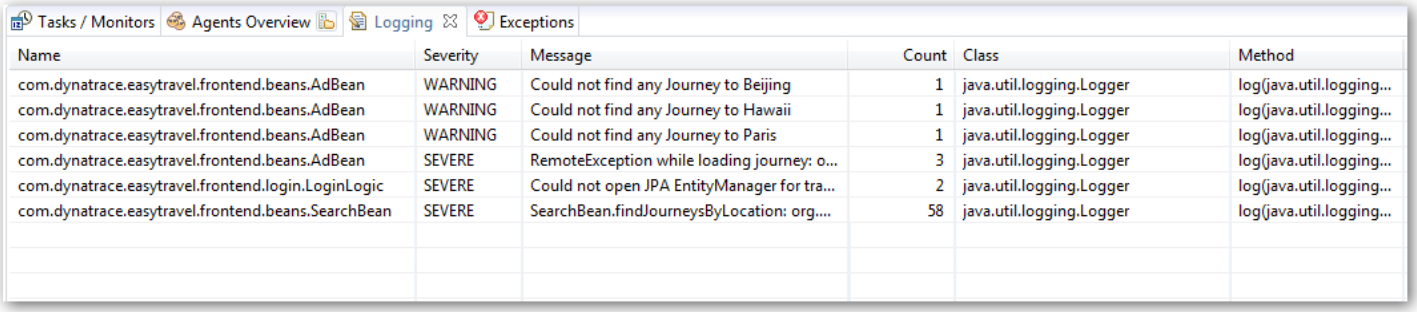
Overview

AppMon diagnoses logging calls executed by the Java Logging API, the Log4J Logging Framework, and the .NET Log4NET Framework. It provides information about the severity of logging messages. A developer can control the amount of detail that is included in the logging information via APIs and frameworks. The **Logging** dashlet shows an aggregated view of the logging events grouped by the following:

- Issuing class and method
- Recording AppMon Agent

- Message string
- Severity
- Logger name

Logging View



Name	Severity	Message	Count	Class	Method
com.dynatrace.easytravel.frontend.beans.AdBean	WARNING	Could not find any Journey to Beijing	1	java.util.logging.Logger	log(java.util.logging...
com.dynatrace.easytravel.frontend.beans.AdBean	WARNING	Could not find any Journey to Hawaii	1	java.util.logging.Logger	log(java.util.logging...
com.dynatrace.easytravel.frontend.beans.AdBean	WARNING	Could not find any Journey to Paris	1	java.util.logging.Logger	log(java.util.logging...
com.dynatrace.easytravel.frontend.beans.AdBean	SEVERE	RemoteException while loading journey: o...	3	java.util.logging.Logger	log(java.util.logging...
com.dynatrace.easytravel.frontend.login.LoginLogic	SEVERE	Could not open JPA EntityManager for tra...	2	java.util.logging.Logger	log(java.util.logging...
com.dynatrace.easytravel.frontend.beans.SearchBean	SEVERE	SearchBean.findJourneysByLocation: org....	58	java.util.logging.Logger	log(java.util.logging...

Metrics

Logging metrics

- **Name**: User defined logger name.
- **Severity**: Severity level assigned to this logging message.
- **Message**: Message string received by the logging framework.
- **Agent**: Name of the AppMon Agent that recorded the logging event.
- **Count**: Number of times a logging event is issued by the class method in this record and recorded by an AppMon Agent:
  - From this particular logger
  - With this particular message string
  - With this severity, issued by the class method listed in this record
- **Class**: Fully qualified name of the class that issued the logging event.
- **Method**: Method name that issued the logging event.

Configuration

Use the **Logging Sensor Configuration** dialog box to define filter criteria to target the set of extended logging events for analysis. See Sensor Properties for more information.

The filter criteria are based on severity levels. Include or exclude this in the analysis. The .NET Logging sensor, Java JDK Logging, and Log4J sensor provide this configuration option.

Note

This sensor configuration applies only to Java or .NET based logging frameworks.

Context menu options

- **Subscribe Measures**: Opens the **Measure Picker** dialog box to add measures to a subscription. This option is only available during real-time sessions.
- **New Business Transaction**: Opens the **Business Transaction** dialog box to create a new business transaction based on the Measures. This option is only available during real-time sessions.

Runtime Suspensions dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics on the Runtime Suspension dashlet

Related pages

Event specific dashlets

Errors dashlet

Exceptions dashlet

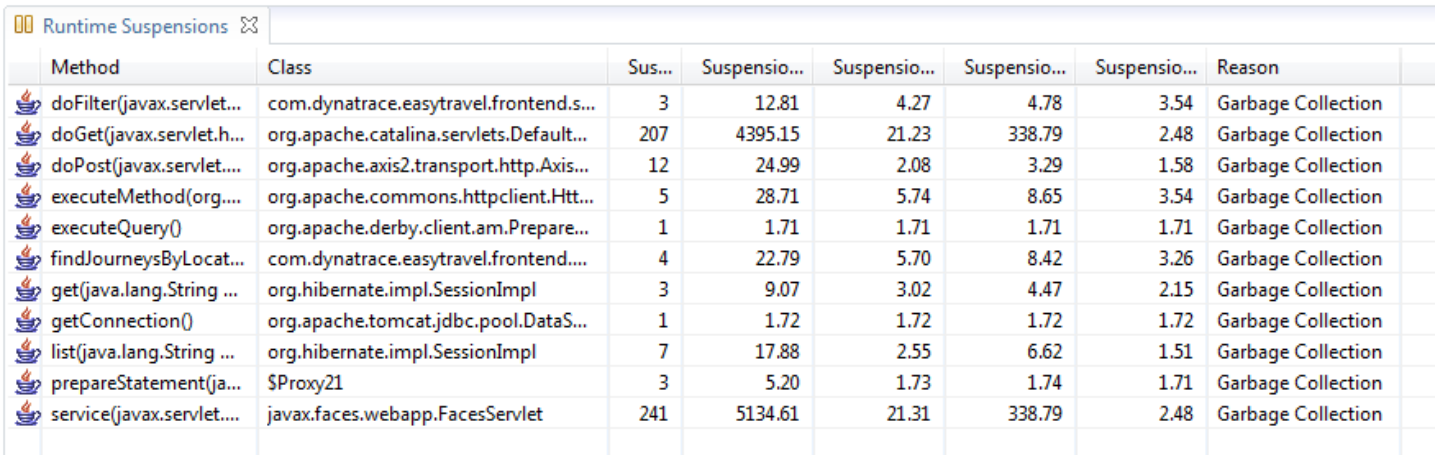
GUI Controls dashlet

Incidents dashlet

Logging dashlet

Overview

Runtime suspensions are intervals in which the JVM or CLR don't execute application threads. Typically, this is because a runtime suspension is a garbage collector run. Other reasons include a debugger suspending execution or a CLR shutdown of an app-domain.


Runtime Suspensions

Method	Class	Sus...	Suspensio...	Suspensio...	Suspensio...	Suspensio...	Reason
doFilter(javax.servlet...	com.dynatrace.easytravel.frontend.s...	3	12.81	4.27	4.78	3.54	Garbage Collection
doGet(javax.servlet.h...	org.apache.catalina.servlets.Default...	207	4395.15	21.23	338.79	2.48	Garbage Collection
doPost(javax.servlet....	org.apache.axis2.transport.http.Axis...	12	24.99	2.08	3.29	1.58	Garbage Collection
executeMethod(org....	org.apache.commons.httpclient.Htt...	5	28.71	5.74	8.65	3.54	Garbage Collection
executeQuery()	org.apache.derby.client.am.Prepare...	1	1.71	1.71	1.71	1.71	Garbage Collection
findJourneysByLocat...	com.dynatrace.easytravel.frontend....	4	22.79	5.70	8.42	3.26	Garbage Collection
get(java.lang.String ...	org.hibernate.impl.SessionImpl	3	9.07	3.02	4.47	2.15	Garbage Collection
getConnection()	org.apache.tomcat.jdbc.pool.DataS...	1	1.72	1.72	1.72	1.72	Garbage Collection
list(java.lang.String ...	org.hibernate.impl.SessionImpl	7	17.88	2.55	6.62	1.51	Garbage Collection
prepareStatement(ja...	$Proxy21	3	5.20	1.73	1.74	1.71	Garbage Collection
service(javax.servlet....	javax.faces.webapp.FacesServlet	241	5134.61	21.31	338.79	2.48	Garbage Collection

Metrics on the Runtime Suspension dashlet

- **Platform**: Runtime Suspension (Java/.NET) software platform.
- **Method**: Method that is suspended during Runtime Suspension.
- **Class**: Class of suspended method.
- **Suspension Count**: Number of Runtime Suspension.
- **Suspension Sum [ms]**: Total time elapsed during the Runtime Suspension.
- **Suspension Avg [ms]**: Average time elapsed during the Runtime Suspension.
- **Suspension Max [ms]**: Maximum time elapsed during the Runtime Suspension.
- **Suspension Min [ms]**: Minimum time elapsed during the Runtime Suspension.
- **Reason**: Reason for the Runtime Suspension (typically garbage collector run).

Runtime Specific dashlets

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

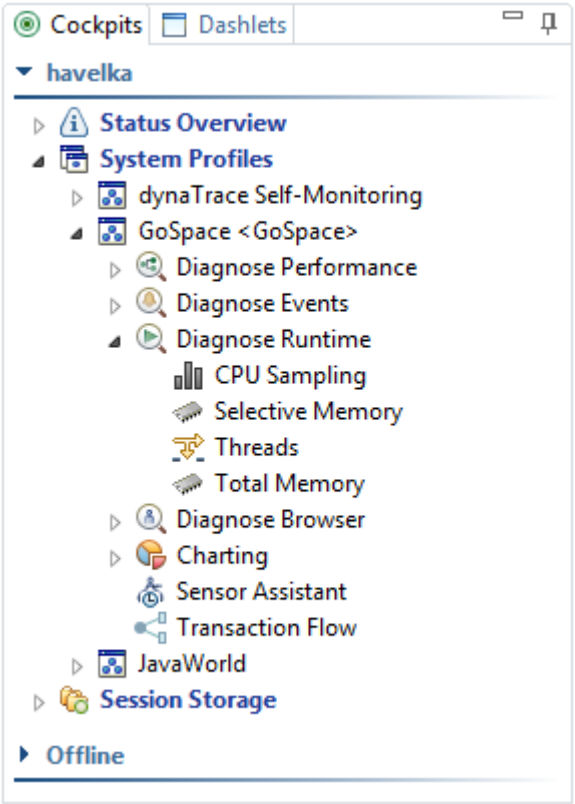
CPU Sampling dashlet

Selective Memory dashlet

Threads dashlet

Total Memory dashlet

AppMon offers four possibilities to analyze snapshots of a system at runtime.



Runtime Analysis	Description
CPU Sampling Dashlet	Use the **CPU Sampling** dashlet to identify an entry point and to analyze background threads.

Selective Memory Dashlet	Use the **Selective Memory** dashlet to create or open snapshots and analyze the instances in the snapshot.
Threads Dashlet	Use the **Threads** dashlet to create or open thread dumps and analyze the thread stack.
Total Memory Dashlet	Use the **Total Memory** dashlet for snapshot information about currently allocated objects, regardless of their connection to recorded PurePaths or configured memory rules.

CPU Sampling dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Runtime Specific dashlets

Selective Memory dashlet

Threads dashlet

Total Memory dashlet

When to use sampling

AppMon automatically captures performance relevant method invocations (shown in the **CPU Sampling** dashlet) and does not define explicit sensors. Therefore, you would use CPU sampling to:

- Identify an entry point: Find your custom entry points to define Custom Entry Point Sensors.
- Analyze background threads: If AppMon does not capture threads as PurePaths, analyze them with CPU sampling.

Sampling FYIs

- Sampling investigates the current stack of all running Java or .NET application threads to determine if a method is long running or called very often.
- Sampling doesn't see calls across thread or application boundaries as do PurePaths Dashlet.
- CPU times are measured on a per thread basis, not per method. To visualize it per method, CPU time is assigned to methods based on the number of times a method is seen as the executing one.

Limitations

CPU sampling for Java is available for Java versions 1.5+ and .NET versions 1.1+.

Wording

leaf method	A method that has been seen at least once at the top of a stack trace. When the method sees the stack trace, it is executing.
active thread	A method is counted active if the thread is currently in a runnable state.
inactive thread	A method is counted inactive if the thread it is executed in is blocked or waiting.

Sampling sessions

Sampling sessions receive an *incomplete* state when sampling stops unexpectedly. This occurs when the collector at the agent connection stops, or when the agent itself stops. You can open incomplete sampling sessions like you open correctly finished sessions.

Sampling session interpretation

Stacktraces grouping

Hotspot view

In the Hotspot tree, all methods that are seen executing are root nodes. The callers of the methods are the tree children.

Group by root methods

In this view, all grouped stack traces ignore the thread in which they were executed. This is useful if you don’t care about the thread in which your method is executed. This is true for code that is executed by thread pools (like worker threads of web-applications).

Group by thread

All stack traces are grouped by the thread in which they were executed.

Filtering capabilities

Click `Ctrl-F` to filter the tree. The following predefined filtering capabilities are at the top-right:

- The Package filter hides methods that are called from the same package. This shows only methods where the parent node in the tree is a method of another package.
- The Hide Delegates filter hides methods that are always seen together with the calling method.
- The Only Active filter ignores threads that are only seen inactive. This is enabled by default. If you are looking for synchronization or similar problems, disable this filter.

The nodes that match your filter text always display, even if the Hide Same Package filter or Hide Delegates filter is enabled. When you switch to search mode, the behavior is different. Nodes that match your search do not display if they are hidden by these filters.

Accuracy

Methods may be shown as hotspots, however these methods are cheap. The JVM doesn't hold the threads exactly when the stack traces are requested, but only on so-called *safe points*. Safe points have their origins in garbage collection. The Hotspot JVM sets a safe point when it exits a method and will see a stack trace when a method just returns.

Errors

No methods consuming CPU time have been found

Reasons:

- No methods have been seen active while the thread in which they were running consumed CPU time.
- The sampling session was created with an AppMon version lower than 3.5, where this data wasn't written.

No waiting methods have been found

Reasons:

- No methods have been seen in inactive state. This applies to .NET because sampling only regards managed code.
- The sampling session was created with an AppMon version lower than 3.5, where this data wasn't written.

CPU usage in overview differs from an operating system process monitor output

This can happen because AppMon sampling does not apply to native threads.

Selective Memory dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Show content

Compare selective memory snapshots

Related pages

Runtime Specific dashlets

CPU Sampling dashlet

Threads dashlet

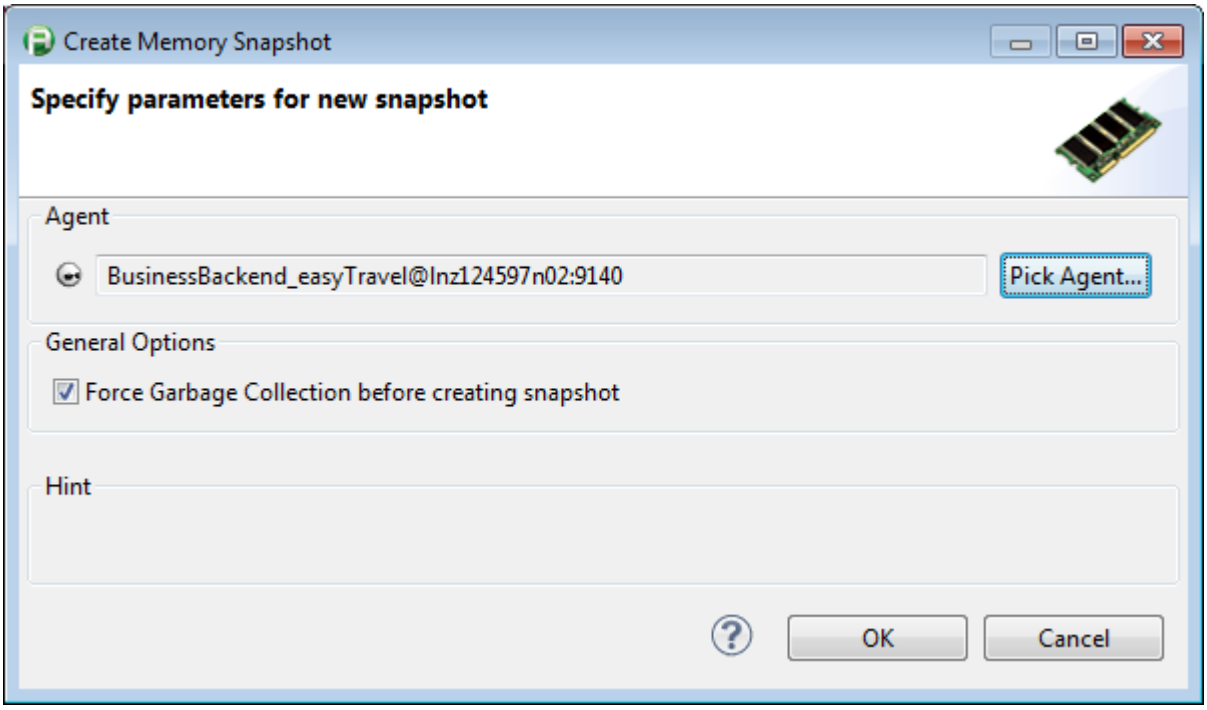
Total Memory dashlet

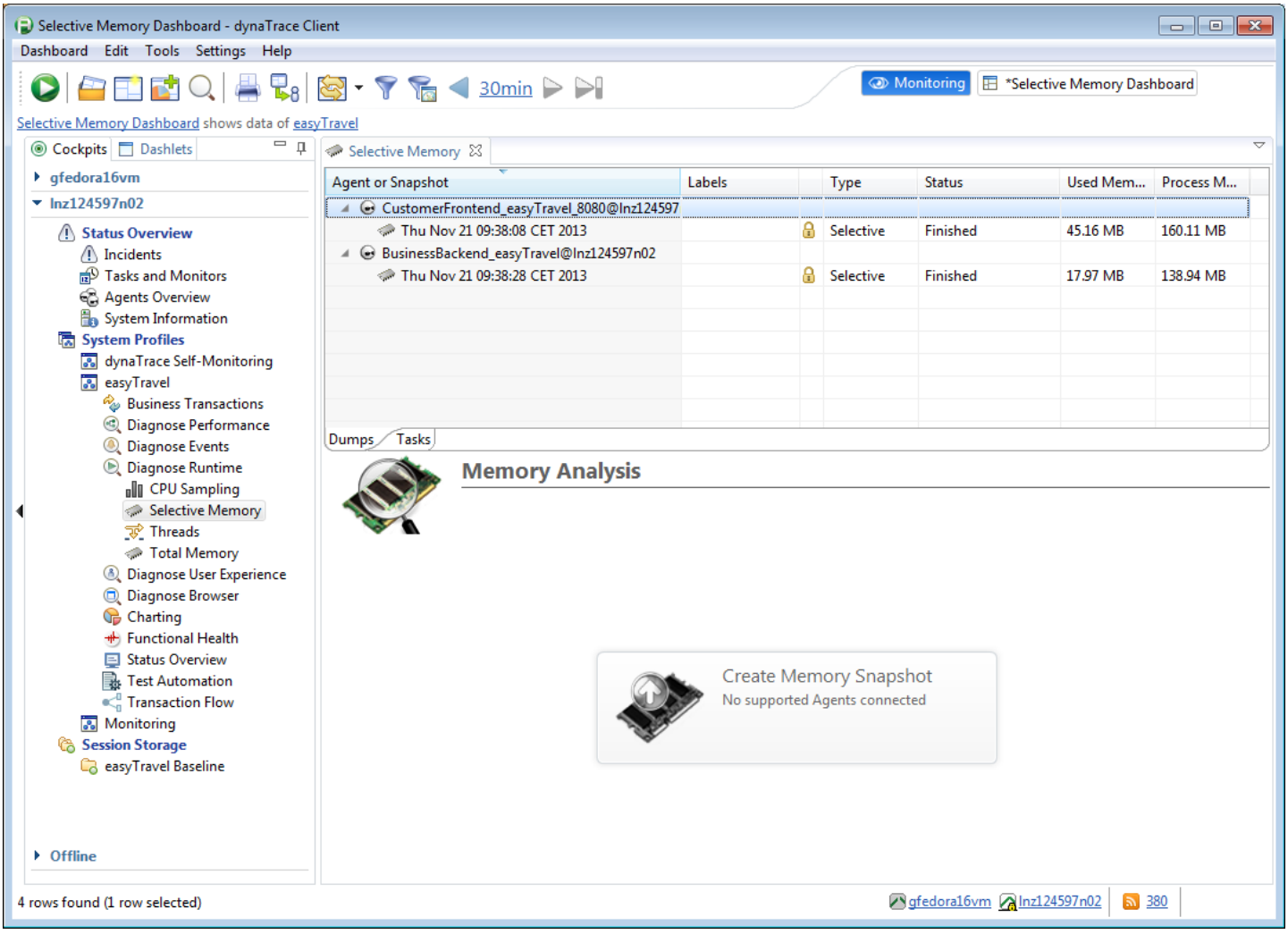
Overview

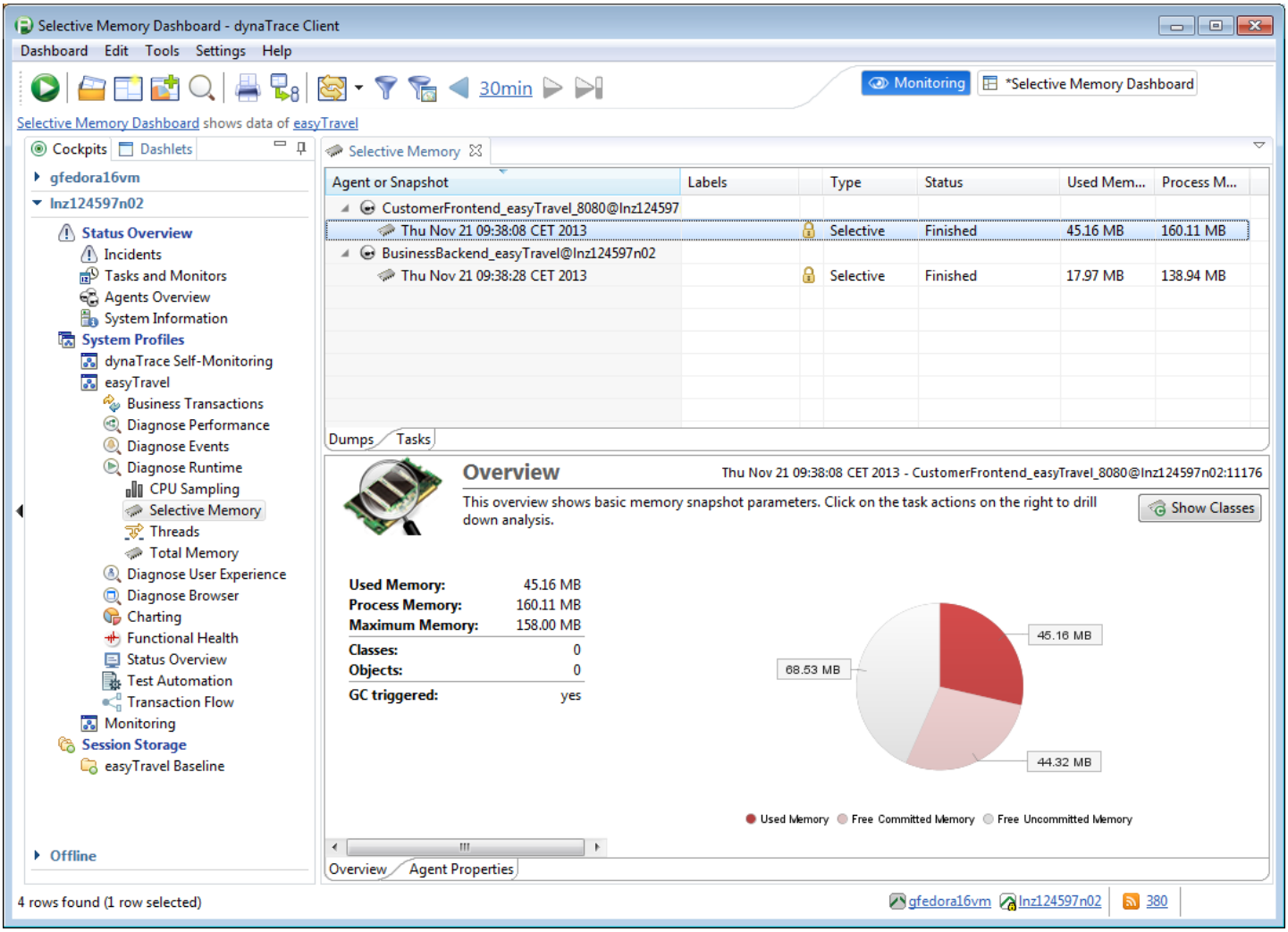
Use the **Selective Memory** dashlet to create or open **Selective Memory** snapshots and analyze the instances in the snapshot. A selective memory dump uses memory sensor rules for instance-allocation book keeping. You get a list with all allocated instances for which you have created memory sensor rules. For more information, see Memory Diagnostics.

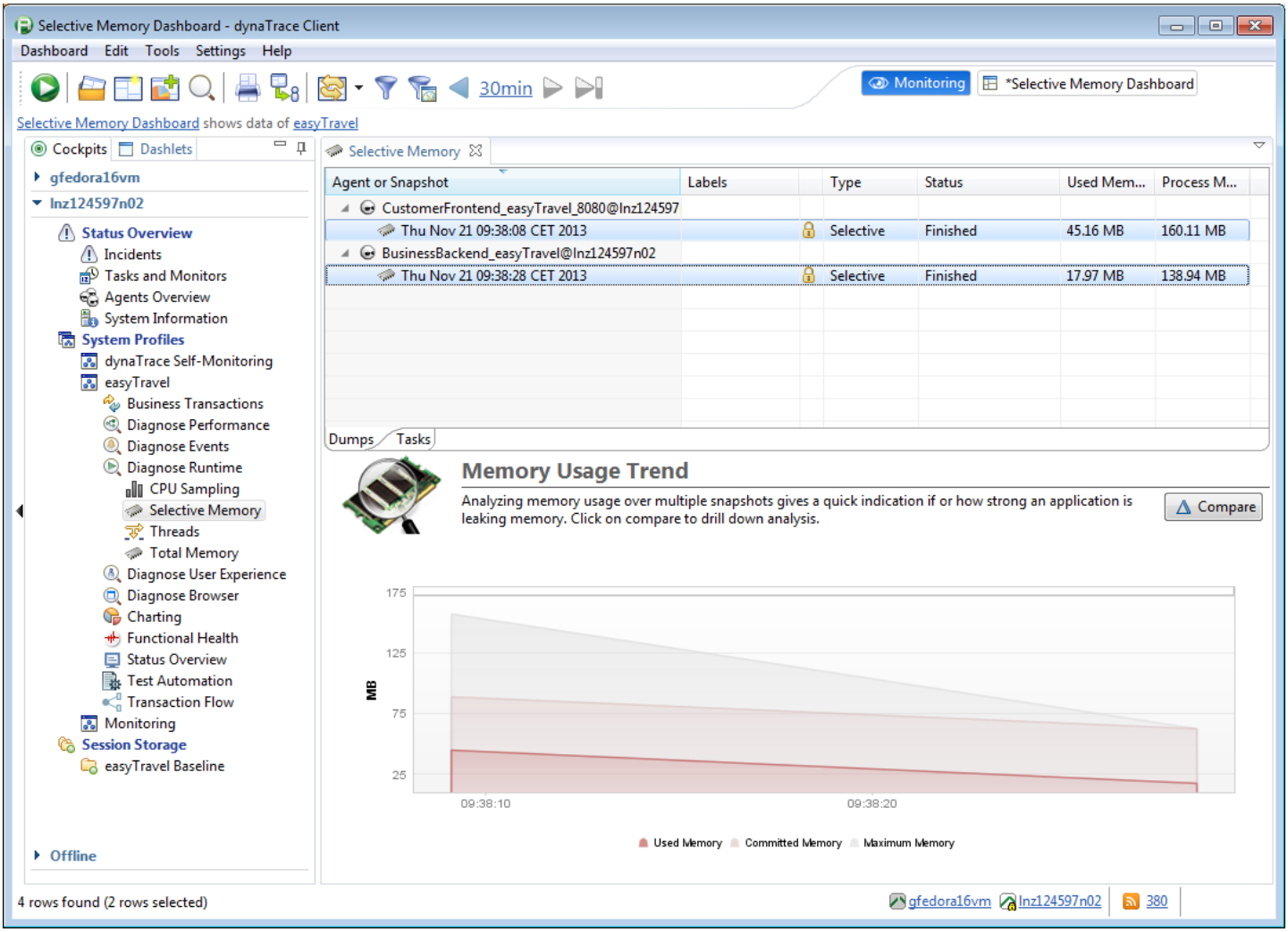
How to create a new snapshot?

Click on the “create Memory Snapshot”-Button in the lower half of the dashlet. This option only appears when an agent or nothing is selected in the upper table of the dashlet. If you select an existing snapshot, an overview of the snapshot appears. If you select multiple snapshots, a chart appears that compares the overview data.


Create Memory Snapshot
Specify parameters for new snapshot
Agent
BusinessBackend_easyTravel@lnz124597n02:9140
Pick Agent...
General Options
Force Garbage Collection before creating snapshot
Hint
OK
Cancel


Selective Memory Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*Selective Memory Dashboard
Selective Memory Dashboard shows data of easyTravel
Cockpits
Dashlets
gfedora16vm
lnz124597n02
Status Overview
Incidents
Tasks and Monitors
Agents Overview
System Information
System Profiles
dynaTrace Self-Monitoring
easyTravel
Business Transactions
Diagnose Performance
Diagnose Events
Diagnose Runtime
CPU Sampling
Selective Memory
Threads
Total Memory
Diagnose User Experience
Diagnose Browser
Charting
Functional Health
Status Overview
Test Automation
Transaction Flow
Monitoring
Session Storage
easyTravel Baseline
Offline
Selective Memory
Agent or Snapshot
Labels
Type
Status
Used Mem...
Process M...
CustomerFrontend_easyTravel_8080@lnz124597
Thu Nov 21 09:38:08 CET 2013
Selective
Finished
45.16 MB
160.11 MB
BusinessBackend_easyTravel@lnz124597n02
Thu Nov 21 09:38:28 CET 2013
Selective
Finished
17.97 MB
138.94 MB
Dumps
Tasks
Memory Analysis
Create Memory Snapshot
No supported Agents connected
4 rows found (1 row selected)
gfedora16vm
lnz124597n02
380


Selective Memory Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*Selective Memory Dashboard
Selective Memory Dashboard shows data of easyTravel
Cockpits
Dashlets
gfedora16vm
lnz124597n02
Status Overview
Incidents
Tasks and Monitors
Agents Overview
System Information
System Profiles
dynaTrace Self-Monitoring
easyTravel
Business Transactions
Diagnose Performance
Diagnose Events
Diagnose Runtime
CPU Sampling
Selective Memory
Threads
Total Memory
Diagnose User Experience
Diagnose Browser
Charting
Functional Health
Status Overview
Test Automation
Transaction Flow
Monitoring
Session Storage
easyTravel Baseline
Offline
Selective Memory
Agent or Snapshot
Labels
Type
Status
Used Mem...
Process M...
CustomerFrontend_easyTravel_8080@lnz124597
Thu Nov 21 09:38:08 CET 2013
Selective
Finished
45.16 MB
160.11 MB
BusinessBackend_easyTravel@lnz124597n02
Thu Nov 21 09:38:28 CET 2013
Selective
Finished
17.97 MB
138.94 MB
Dumps
Tasks
Overview
Thu Nov 21 09:38:08 CET 2013 - CustomerFrontend_easyTravel_8080@lnz124597n02:11176
This overview shows basic memory snapshot parameters. Click on the task actions on the right to drill down analysis.
Show Classes
Used Memory: 45.16 MB
Process Memory: 160.11 MB
Maximum Memory: 158.00 MB
Classes: 0
Objects: 0
GC triggered: yes
45.16 MB
68.53 MB
44.32 MB
Used Memory
Free Committed Memory
Free Uncommitted Memory
Overview
Agent Properties
4 rows found (1 row selected)
gfedora16vm
lnz124597n02
380



Show content

Double click a snapshot or click **Show Classes** to view the instances that the snapshot collects in a new dashlet.

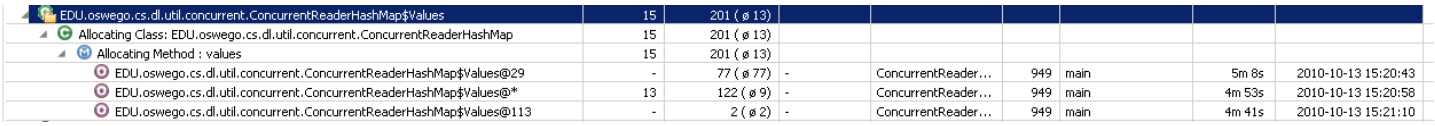
Selective memory metrics

- **Class/Instances**: Name of class that is monitored for instantiations. The map/collection sensor includes all classes that implement the corresponding interfaces. When you expand the class, the individual instances are grouped by their allocating class, allocating method, allocating line, and their time of allocation.
- **Instances**: Number of monitored class instances allocated by one method of one allocating class.
- **Sum Collection Size** For collections or maps, this represents the sum of the collection.size() / map.size() values of all Instances. It also contains an average of the size of all Instances of this class.
- **Allocating Class**: Class that allocates the instance. Hidden by default.
- **Allocating Method**: Method that allocates the instance. Hidden by default.

- **Domain**: The Appdomain (only for .NET)
- **File**: Name of the source file of the allocating class, if available.
- **Line**: Line number within the source file were the allocation is performed. If allocation takes place in a native method the line number is set to “-“
- **Allocating Thread**: Name of the thread that allocated the instance.
- **Object Age**: Age of the instance at the time the snapshot is created. Use this to find very long living instances.
- **Allocation Time**: When the Instance is allocated.

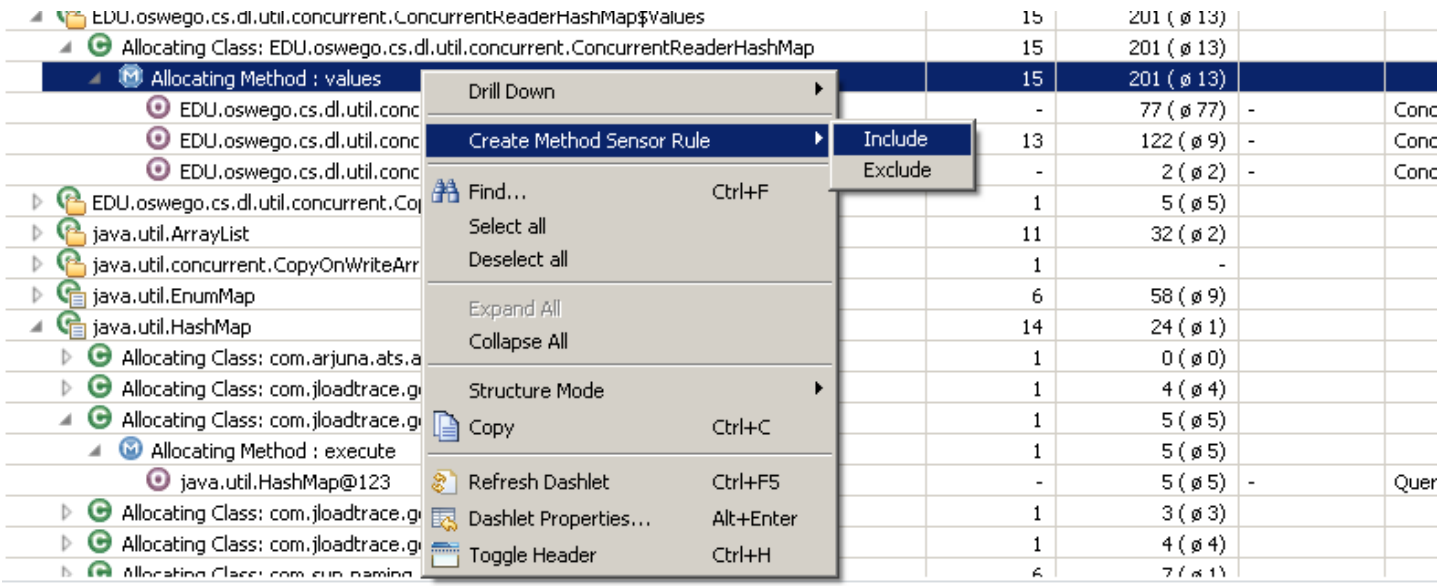
Group by allocation time

AppMon groups instances by allocating class, allocation method, allocating line and their time of allocation. If the same class/method/line, within the same second, allocates multiple instances of a class, AppMon groups them into a single node to improve visibility. The following figure shows three *ConcurrentReaderHashMap$Values* nodes. The first shows the allocation of a single entry. The second shows the allocation of 13 entries within a single second. The third shows the allocation of a single instance.



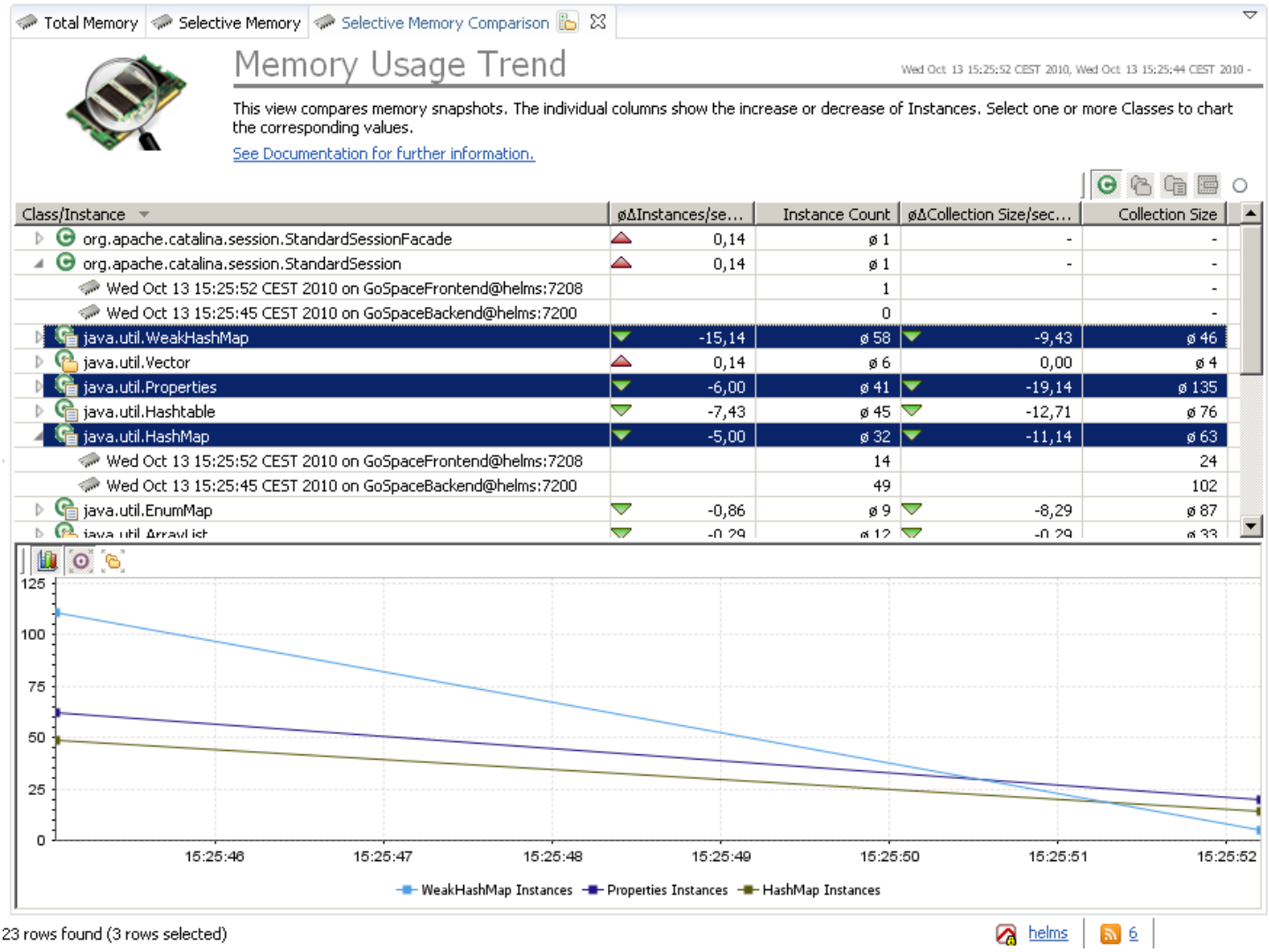
Create Method Rules from selective memory content

Use the **Selective Memory Content** dashlet to create include or exclude method rules to allocate methods. When you create an include rule, the corresponding method that allocates an instance is included into PurePaths. To create method rules, use the context-menu on an allocating method node as shown in the figure below.



Compare selective memory snapshots

To open the **Selective Snapshot Compare** dashlet, select multiple snapshots and click the compare button in the overview (or use the context-menu). This dashlet compares the instance counts and collections/maps classes size of the selected snapshots. You cannot compare individual collection/map instances by size. All instances of a specific class must be aggregated.



This View Compares Memory Snapshots

The view includes the following columns:

- **Class/Instance**: Class that is compared. Expand to see the individual compared snapshots.
- **Average Delta Instances per second**: Average change of number instances per second for this class.
- **Instance Count**: Average number of instances over all snapshots for first-level entries (classes). Number of instances in this snapshot for second-level entries (snapshots).
- **Average Delta Collection size**: Average change of the sum of the sizes of all instances of a class for collections/maps only.
- **Collection size**: Sum of the size of all instances of that class for collections/maps only.

Threads dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Metrics

Stack trace

Owned monitors

Tasks

Related pages

Thread diagnostics

Runtime Specific dashlets

CPU Sampling dashlet

Selective Memory dashlet

Total Memory dashlet

Overview

Use the **Threads** dashlet to create or open thread dumps and analyze the thread stacks.
This view is divided into two areas. The upper area shows existing thread dumps for certain Agents. The bottom area shows information about the selected dump or the possibility to invoke a thread dump, if no dump is selected in the upper area. AppMon supports thread dumps for .NET.

Metrics

Thread performance metrics

- **Name**: Name of the thread.
- **Threads**: Number of threads.
- **Group**: Name of the thread group the thread belongs to. This column may be empty because not all threads must belong to a group.
- **CPU Time [ms]**: Time, in milliseconds, that the thread used the CPU. (This information is only available for J2SE 5.0 and higher)
- **State**: Current state of the thread. There are six defined states: * New: A that is created, but has not been started. There is no stack trace available for this thread.
  - Runnable: Ready to execute or currently executing code.
  - Blocked: A thread waiting to gain a monitor.
  - Waiting: A thread that has yet to be notified, is sleeping, or is parked.
  - Timed waiting: A thread that is waiting a fixed period of time.
  - Terminated: A thread that has finished.
- **Waiting On**: Monitor (if any) that the thread is waiting on.
- **Owned By**: Owner of the monitor that this thread is waiting for (if any). A monitor may not have an owner.
- **Owned Monitors**: Number of monitors owned by this thread.
- **Start Time**: Time when the thread was started.
- **Thread Id**: JVM: Identity hashcode of the thread, for correlation. .NET: managed thread id

Stack trace

- **#**: Number of the stack trace element, i.e. the stack depth of the corresponding element.
- **Class**: Fully qualified name of the class, which contains the method that produced the stack trace element.
- **Method**: Name of the method.
- **Line**: Line number of the source file that is currently executing.
- **Source File**: Name of the source file for the executed method.

Owned monitors

- **Object**: Object ID, i.e. its identity hash code
- **Class**: JVM: Monitor class, .NET: Lock class
- **Entry Count**: Number of times that the monitor was entered in this thread.
- **Waiter Count**: Number of threads that are waiting to gain this monitor.

- **Notify Waiter Count**: Number of threads that are waiting to be notified by this monitor.

Tasks

Use the **Tasks** tab to modify or add tasks that can automatically create periodic thread dumps.

Total Memory dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Create a total memory snapshot
Snapshot overview tabs
Content of a total memory snapshot
Comparing memory snapshots
Analysis drill-downs

Related pages

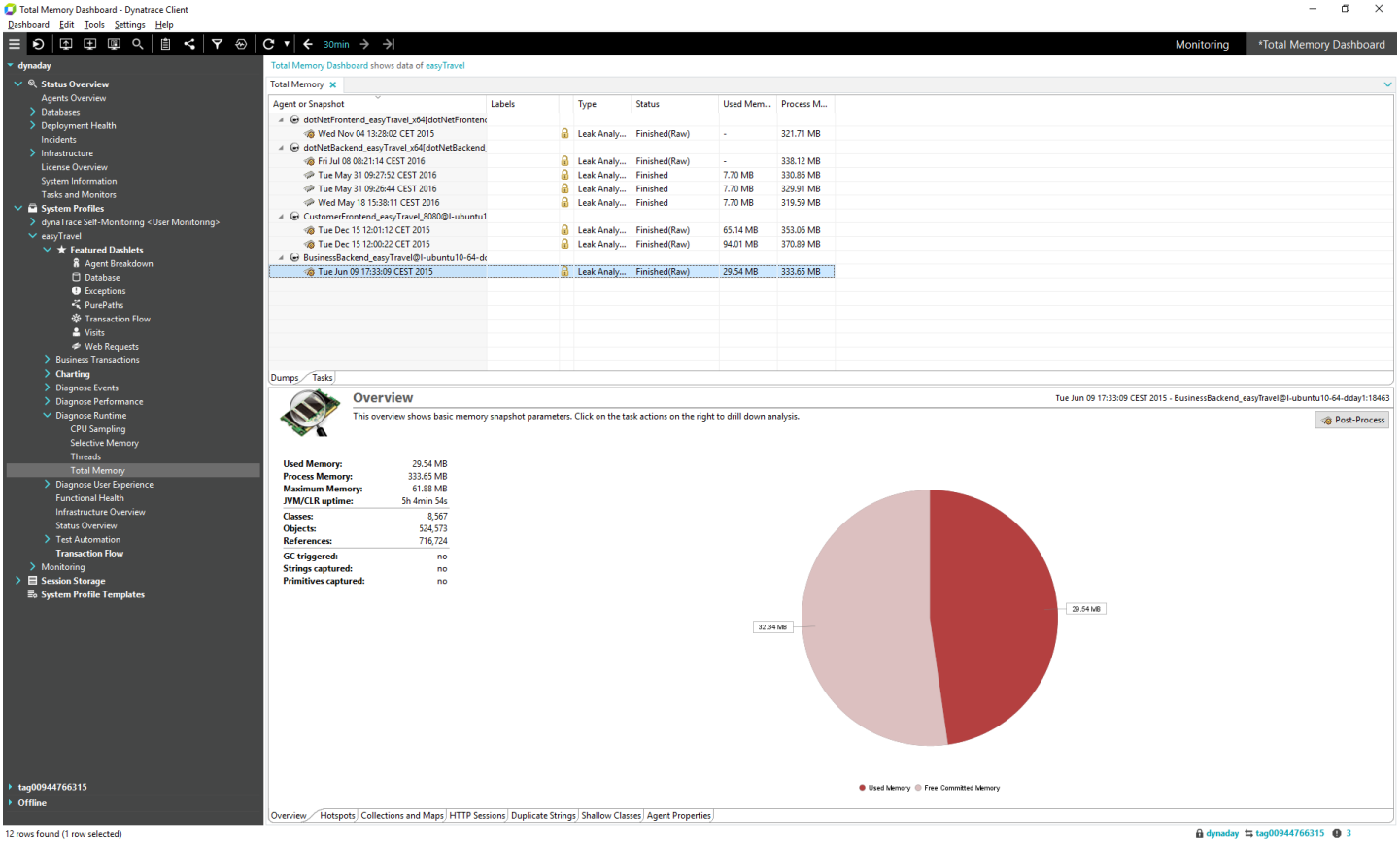
Duplicate Strings Content dashlet
Fields and values
Follow References dashlet
Keep Alive Set dashlet
Shortest Root Paths dashlet
Total Memory Comparison dashlet
Total Memory Content dashlet
Runtime Specific dashlets
CPU Sampling dashlet
Selective Memory dashlet
Threads dashlet

The **Total Memory** dashlet provides snapshot information about currently allocated objects, regardless of their connection to recorded PurePaths or to configured memory rules. Total memory snapshots

belong to one of the following two types:

- **Trending Snapshot**: These contain information about classes, instance counts and shallow sizes.
- **Leak-Analysis Snapshot**: These contain information about classes, the individual instances, references between instances and the individual instances GC size.

The **Total Memory** dashlet shows information about all objects that exist in the virtual machine heap memory at the time AppMon creates the memory snapshot. See Memory Diagnostics for more information.



[dynaTrace] Severe Incident occurred: Application Process Out-of-Memory

If there are problems with how this message is displayed, click here to view it in a web browser.

Sent: Do 20.06.2013 09:46

To:

Severe Incident occurred: Application Process Out-of-Memory

Application Process reported Out-of-memory. A Memory Dump will we automatically triggered if not disabled in the Agent group settings.

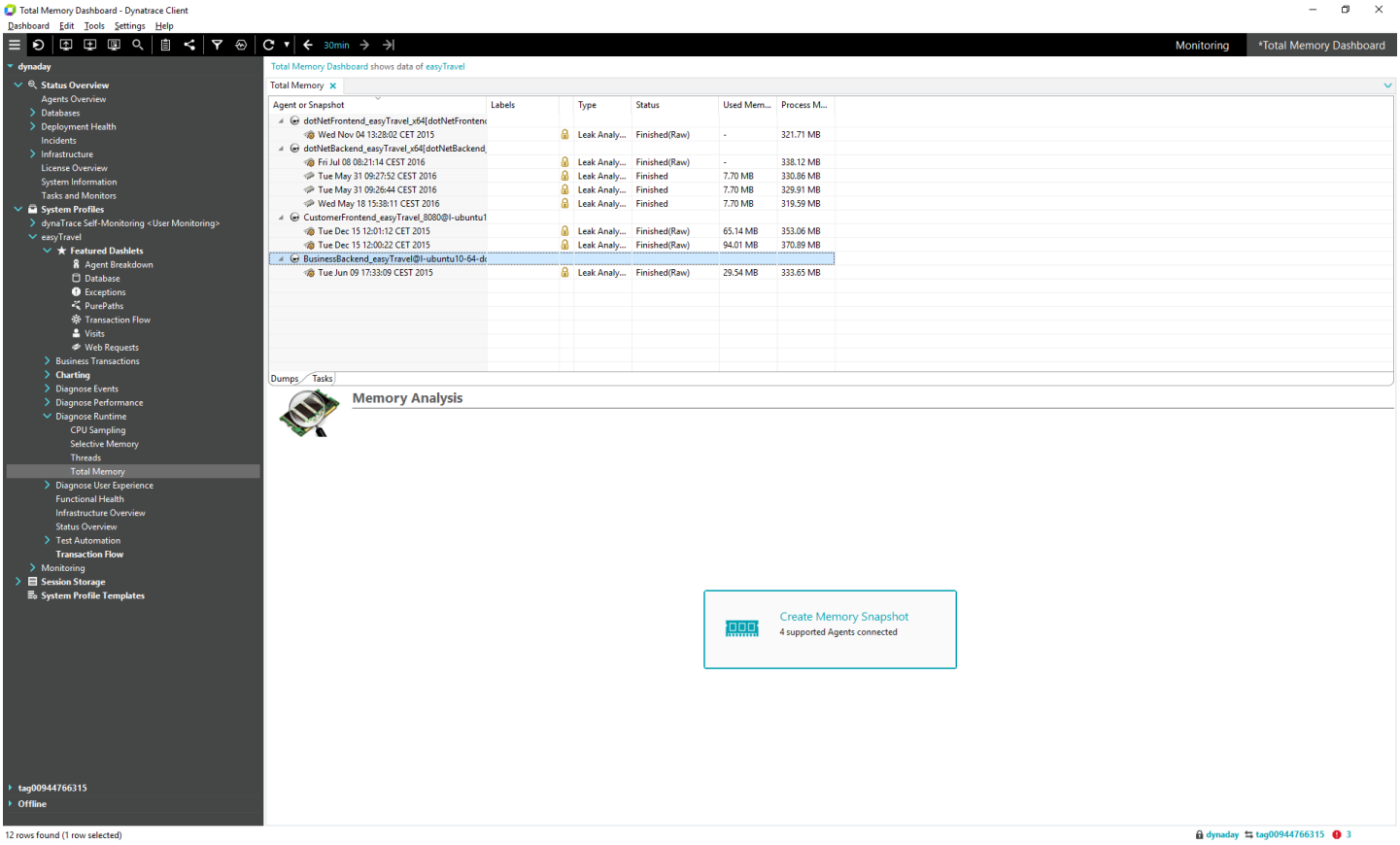
Details

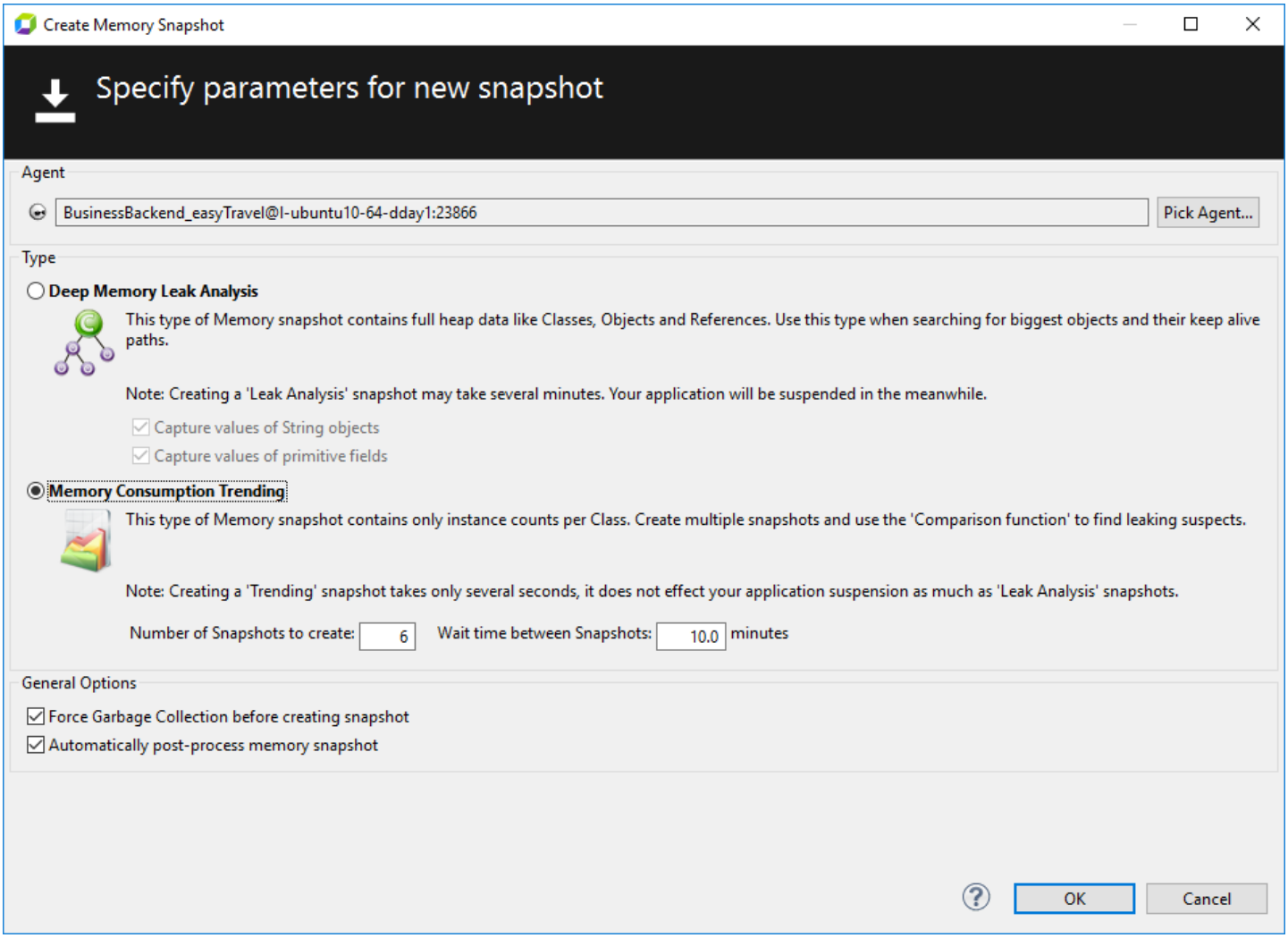
Time:	2013-06-20 09:45:17
Agents:	BusinessBackend_easyTravel@gfedora16vm
Hosts:	gfedora16vm
System Profile:	easyTravel
dynaTrace Server:	gfedora16vm

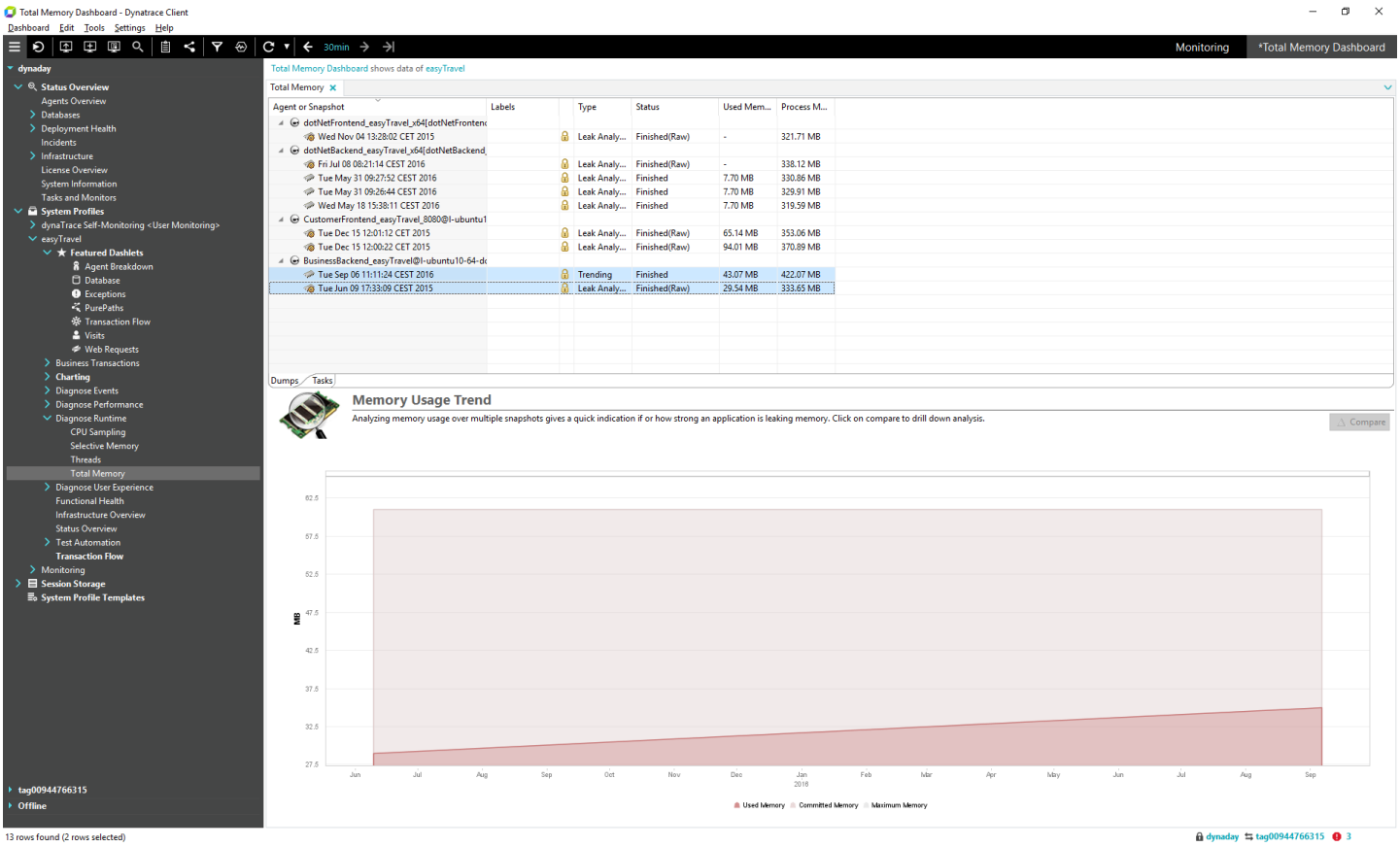
Violations

Out-of-Memory on Agent 'BusinessBackend_easyTravel@gfedora16vm:3237': Java heap space.	A memory snapshot has already been triggered upon a previous incident. There have been 2 out of memory events until now. The latest out of memory event occurred 6min 31s ago.

COMPUWARE APM Open in dynaTrace Open in browser


Total Memory Dashboard - Dynatrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*Total Memory Dashboard
dynaday
Status Overview
Agents Overview
Databases
Deployment Health
Incidents
Infrastructure
License Overview
System Information
Tasks and Monitors
System Profiles
dynaTrace Self-Monitoring <User Monitoring>
easyTravel
Featured Dashlets
Agent Breakdown
Database
Exceptions
PurePaths
Transaction Flow
Visits
Web Requests
Business Transactions
Charting
Diagnose Events
Diagnose Performance
Diagnose Runtime
CPU Sampling
Selective Memory
Threads
Total Memory
Diagnose User Experience
Functional Health
Infrastructure Overview
Status Overview
Test Automation
Transaction Flow
Monitoring
Session Storage
System Profile Templates
tag00944766315
Offline
Total Memory Dashboard shows data of easyTravel
Total Memory
Agent or Snapshot
Labels
Type
Status
Used Mem...
Process M...
dotNetFrontend_easyTravel_x64[dotNetFronten
Wed Nov 04 13:28:02 CET 2015 Leak Analy... Finished(Raw) - 321.71 MB
dotNetBackend_easyTravel_x64[dotNetBackend
Fri Jul 08 08:21:14 CEST 2016 Leak Analy... Finished(Raw) - 338.12 MB
Tue May 31 09:27:52 CEST 2016 Leak Analy... Finished 7.70 MB 330.86 MB
Tue May 31 09:26:44 CEST 2016 Leak Analy... Finished 7.70 MB 329.91 MB
Wed May 18 15:38:11 CEST 2016 Leak Analy... Finished 7.70 MB 319.59 MB
CustomerFrontend_easyTravel_8080@l-ubuntu1
Tue Dec 15 12:01:12 CET 2015 Leak Analy... Finished(Raw) 65.14 MB 353.06 MB
Tue Dec 15 12:00:22 CET 2015 Leak Analy... Finished(Raw) 94.01 MB 370.89 MB
BusinessBackend_easyTravel@l-ubuntu10-64-d
Tue Jun 09 17:33:09 CEST 2015 Leak Analy... Finished(Raw) 29.54 MB 333.65 MB
Dumps
Tasks
Memory Analysis
Create Memory Snapshot
4 supported Agents connected
12 rows found (1 row selected)
dynaday tag00944766315 3


Create Memory Snapshot
Specify parameters for new snapshot
Agent
BusinessBackend_easyTravel@l-ubuntu10-64-dday1:23866
Pick Agent...
Type
Deep Memory Leak Analysis
This type of Memory snapshot contains full heap data like Classes, Objects and References. Use this type when searching for biggest objects and their keep alive paths.
Note: Creating a 'Leak Analysis' snapshot may take several minutes. Your application will be suspended in the meanwhile.
Capture values of String objects
Capture values of primitive fields
Memory Consumption Trending
This type of Memory snapshot contains only instance counts per Class. Create multiple snapshots and use the 'Comparison function' to find leaking suspects.
Note: Creating a 'Trending' snapshot takes only several seconds, it does not effect your application suspension as much as 'Leak Analysis' snapshots.
Number of Snapshots to create: 6
Wait time between Snapshots: 10.0 minutes
General Options
Force Garbage Collection before creating snapshot
Automatically post-process memory snapshot
OK
Cancel



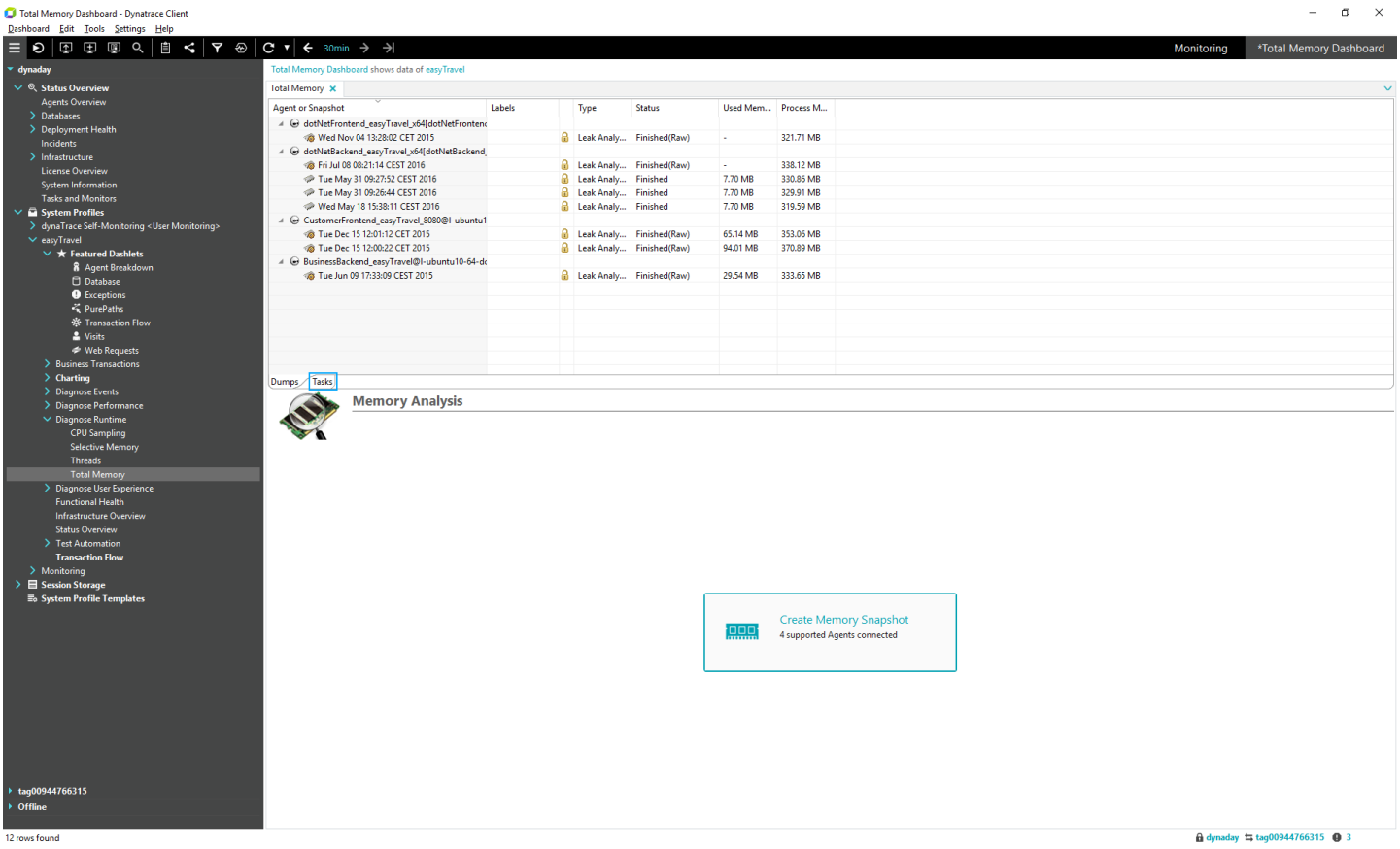
To create a snapshot, click **Create Memory Snapshot** in the lower pane. In the trending screenshot, memory usage falls a bit after the second auto-shot. This is because the user selected **Create Memory Snapshot > General Options > Force Garbage Collection**.

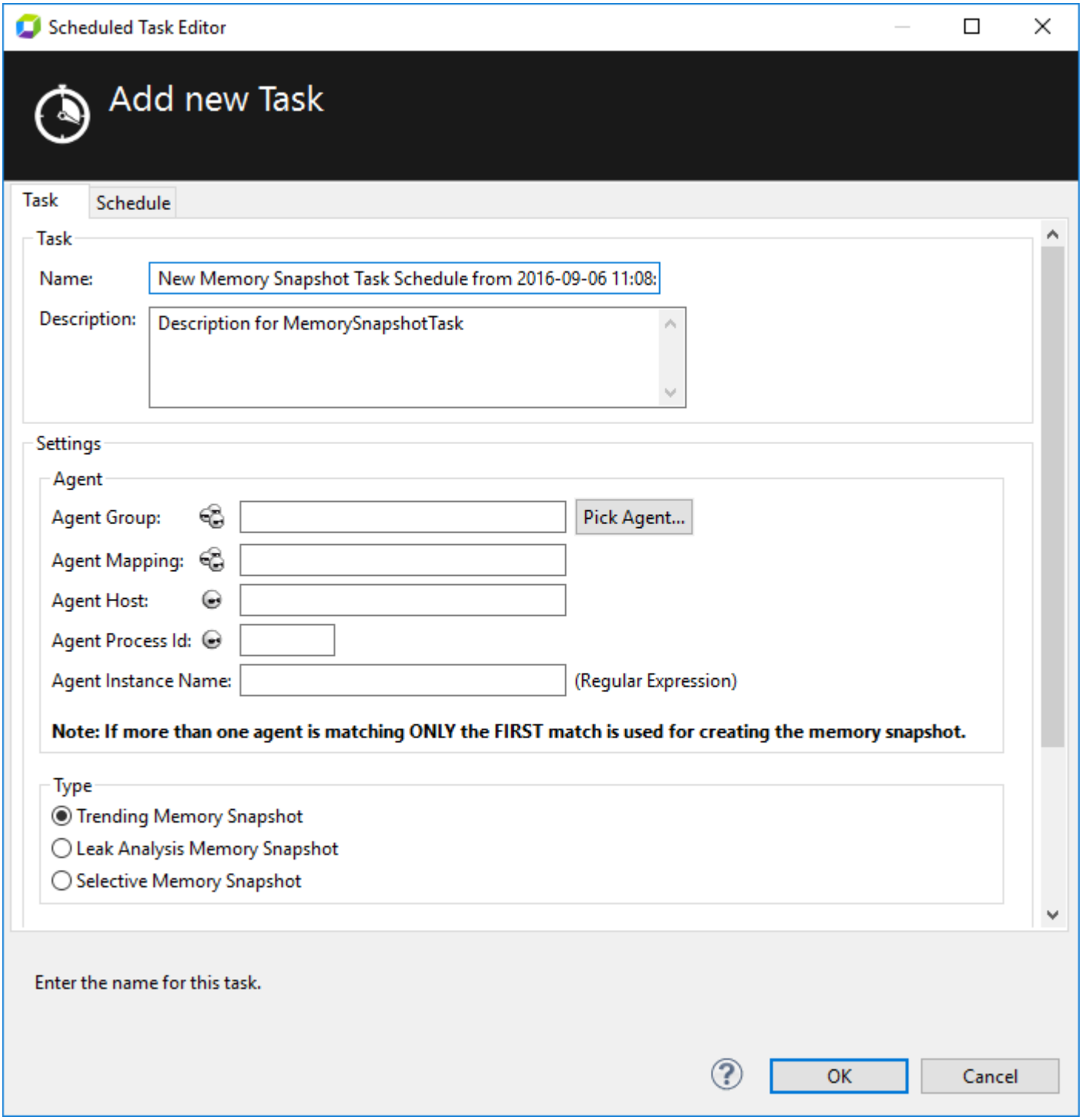
The upper pane of the dashlet displays all snapshots of the corresponding system-profile, grouped by Agents. By default, the table contains the following information for each snapshot:

- **Agent/Dump**: Contains the name of the snapshot (it's creation time by default). Right-click and select **Edit** to change the snapshot name.
- **Labels**: Snapshots can have custom labels assigned. To attach information to a snapshot, right-click and select **Edit** to assign or delete labels.
- **Locking State**: Determines if you can delete the snapshot automatically if the server is running low on disk-space. Right-click and select **Edit** to change the locking state.
- **Type**: The type of the memory snapshot, either Leak Analysis or Trending.
- **Status**: The state of the snapshot, Finished, Finished (raw) or Not completed. A finished snapshot is ready for analysis, a raw-snapshot needs post-processing before you can analyze. A Not completed status indicates problems you discover while you create or post-process the memory snapshot.
- **Used Memory**: The amount of allocated memory at the time the snapshot is created.

> **Note**
>
> Right-click on the table header for additional columns.

Additionally, the upper pane of the dashlet contains a **Task** tab to schedule memory snapshots. (See below)



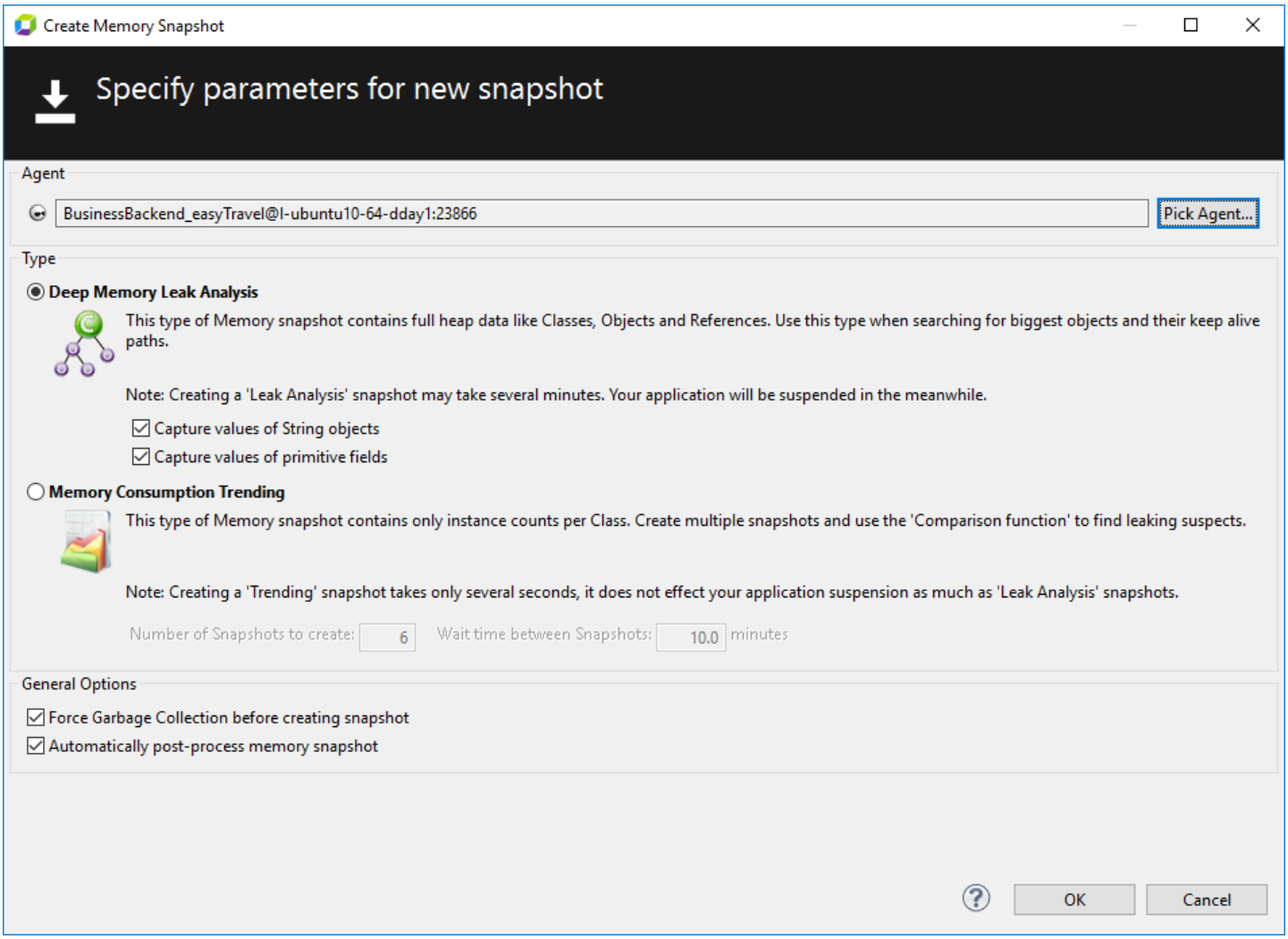


Create a total memory snapshot

The dialog box for creating a memory snapshot allows to determine if a single Leak-Analysis snapshot or one or multiple trending snapshots should be created. Click **Pick-Agent** to select the Agent for which a snapshot should be created. Only currently connected agents are listed. You should select a single agent-instance (no agent-group or agent-mapping). At the bottom of the dialog box, you can deactivate a default triggered Garbage Collector. By default, the snapshot is automatically post-processed. You can disable this in the dialog box.

If you enable **Capture values of String objects**, AppMon performs an automatic string duplicate analysis while it post-processes. If you capture the values of string, the time required to create the snapshot increases because string values have to be transferred. This option works only with Java VMs higher or equal to 6.0, and CLRs higher or equal 2.0.

If the option **Capture values of primitive fields** is enabled, the values of primitives can be inspected in the Fields and Values dialog box for each instance. This options is only available for Java VMs higher or equal 6.0.



Hint

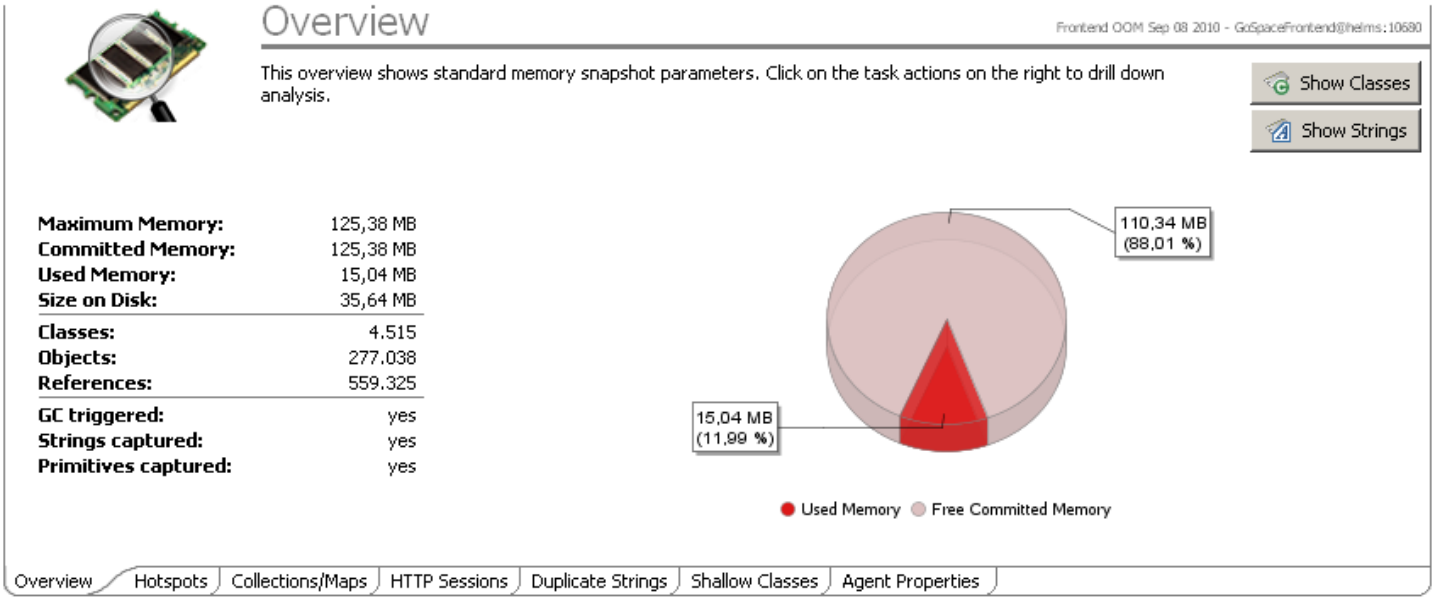
Memory Snapshots initiated by this dialog box are performed via tasks that are visible in the Tasks tab. Trending Snapshots can be paused or stopped there using the context-menu.

Snapshot overview tabs

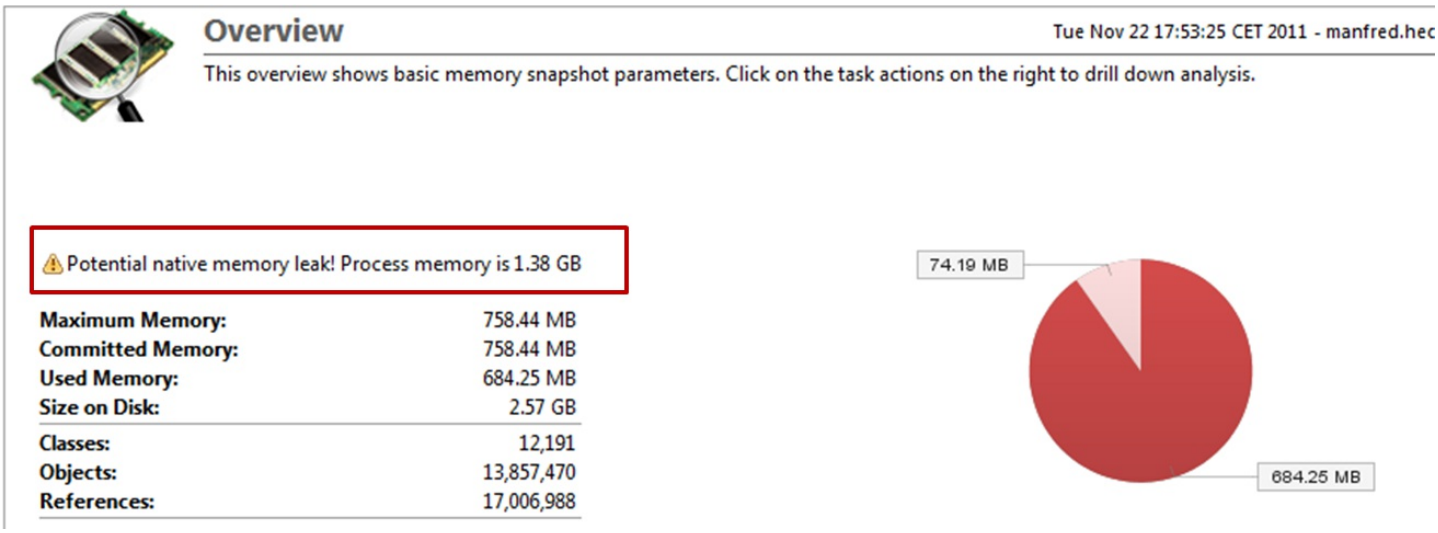
If an existing snapshot (which is already post-processed) is selected in the upper pane, overview information of the corresponding snapshot is presented in the individual tabs in the lower pane as depicted in the following figures. When selecting a non-post-processed snapshot, a button in the lower-pane allows to post-process it.

Overview Tab

This tab shows general information regarding the Heap usage at the time the snapshot was created, including maximum Heap size, currently used and committed memory, the number of Classes and (in case of an Leak-Analysis snapshot) the number of Objects and References.
CLick the **Show Classes** button to open the Memory Snapshot content, including all classes and instances. Click the **Show Strings** button to show all Strings contained in the snapshot. This is only available if enabled during snapshot creation.



In case the overview contains a message that a potential native memory leak has been detected as depicted in the image below, the process-memory of the application is higher than 120 % of the maximum heap size of your VM. Please note that this does not necessarily mean that there is a problem with native memory, it just highlights the possibility of such a leak. Native memory leaks often occur when UI resources are not closed correctly or when a problem with native buffers exists.

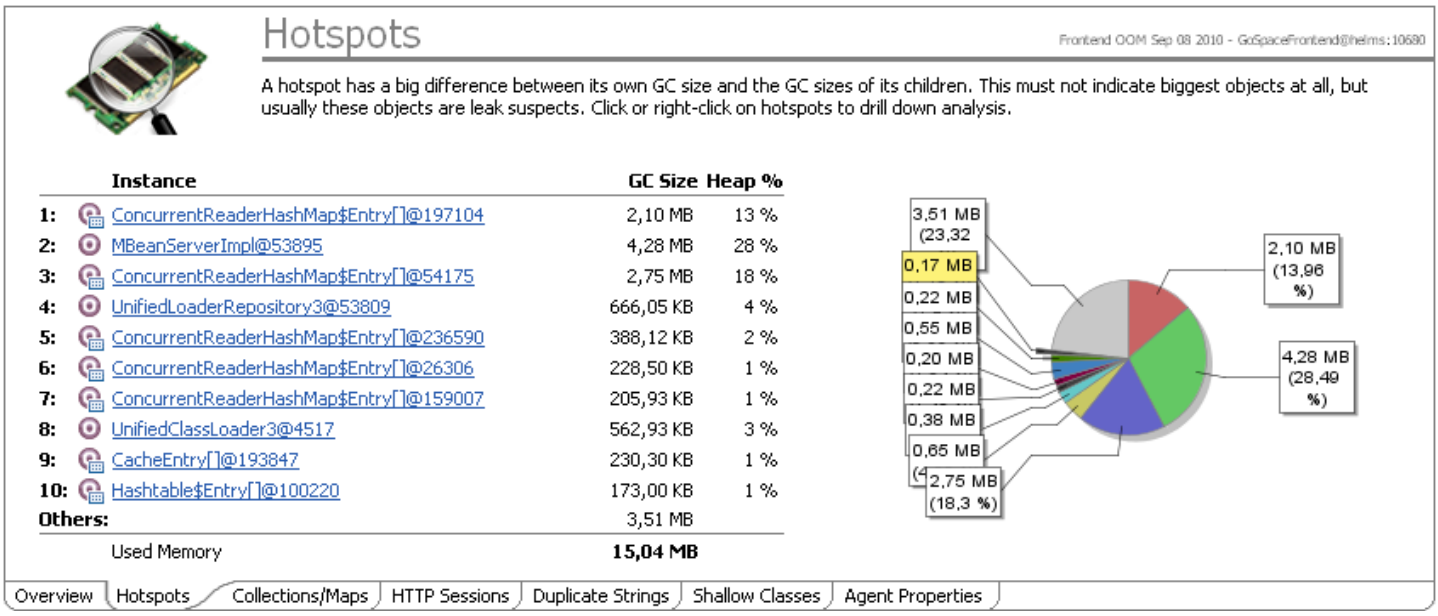


Hot spots

The Hot Spots tab shows the potentially top interesting objects in the Memory Snapshot. A Hot Spot is an object that's difference between its own GC size and the maximum GC size of its child is high. A Hot Spot thus 'accumulates' a lot of other objects.

> **Note**
>
> Please note that Hot Spots are not ordered by their GC size but by the difference of their size to maximum size of an objects children.

Left-click a Hot Spot to open the Shortest Path to Root By Keep Alive in a new dashlet, which displays how it is kept alive. For further analysis of a HotSpot (e.g. analyzing the objects it keeps alive), right-click on it and select the corresponding drilldown.



Top collections/maps

The Top Collections/Maps tab shows the largest (by GC size) objects on the Heap that implement the Collection or Map interfaces.
Collections and Maps are shown in a separate tab since they are very frequently the root-cause of a memory leak. Similar to the other tabs, use the context-menu for further analysis of the individual objects.

Top HTTP-sessions

This tab contains the largest (by GC size) HttpSession objects on the Heap. As for the other tabs, the context-menu allows to drill-down to further analysis Dashlets.

Top duplicate strings

If capturing of Strings is enabled while snapshot creation, this tab shows the top duplicate Strings, ordered by their saving potential by String internalization. The saving potential is calculated by the size of a string multiplied by its duplicate-count. Use the context-menu to drill-down to further analysis Dashlets.

Agent properties

Shows various extended information regarding the Application, JVM/CLR and the AppMon agent of the snapshot.

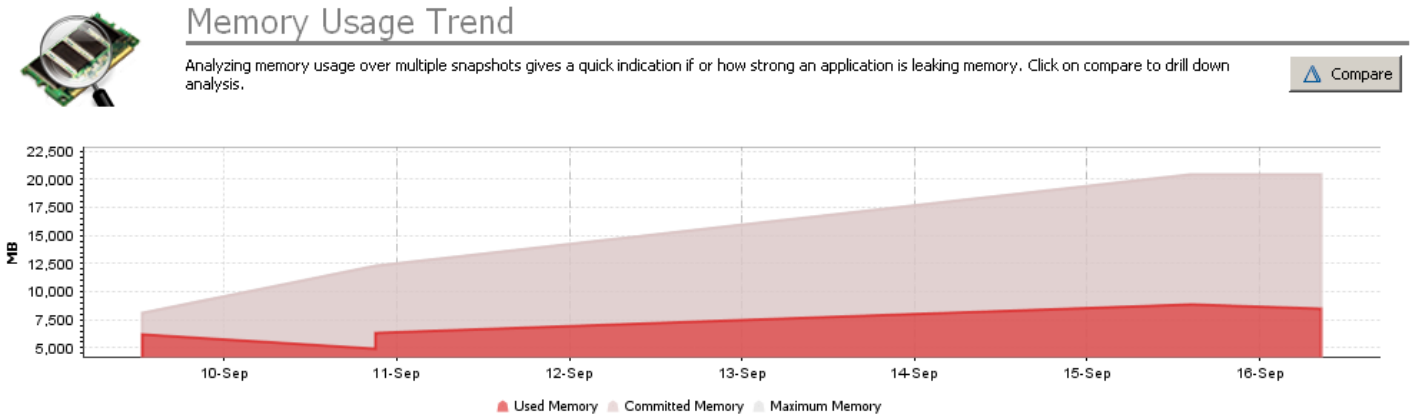
Content of a total memory snapshot

Click **Show Classes** in the overview or double-click on a Memory snapshot to open the corresponding snapshot in a new Dashlet. See Total Memory Content Dashlet for more information.

If String capturing was enabled while snapshot creation, the **Show Strings** button in the overview opens all String duplicates within in the snapshot. See Duplicate Strings Content Dashlet for further details.

Comparing memory snapshots

When selecting multiple memory snapshots, the overview presents a Chart that shows Heap statistics over time as depicted below. Click on the **compare** button to compare the selected snapshots. See Total Memory Comparison Dashlet for more information.



Analysis drill-downs

Within a Leak-Analysis snapshot the following drill-down analysis Dashlets are available for individual objects and usually also aggregated for classes. An aggregated analysis means that the corresponding analysis is done for multiple (or all) instances of a class. Use the context-menu on a class or instance for the following analysis Dashlets:

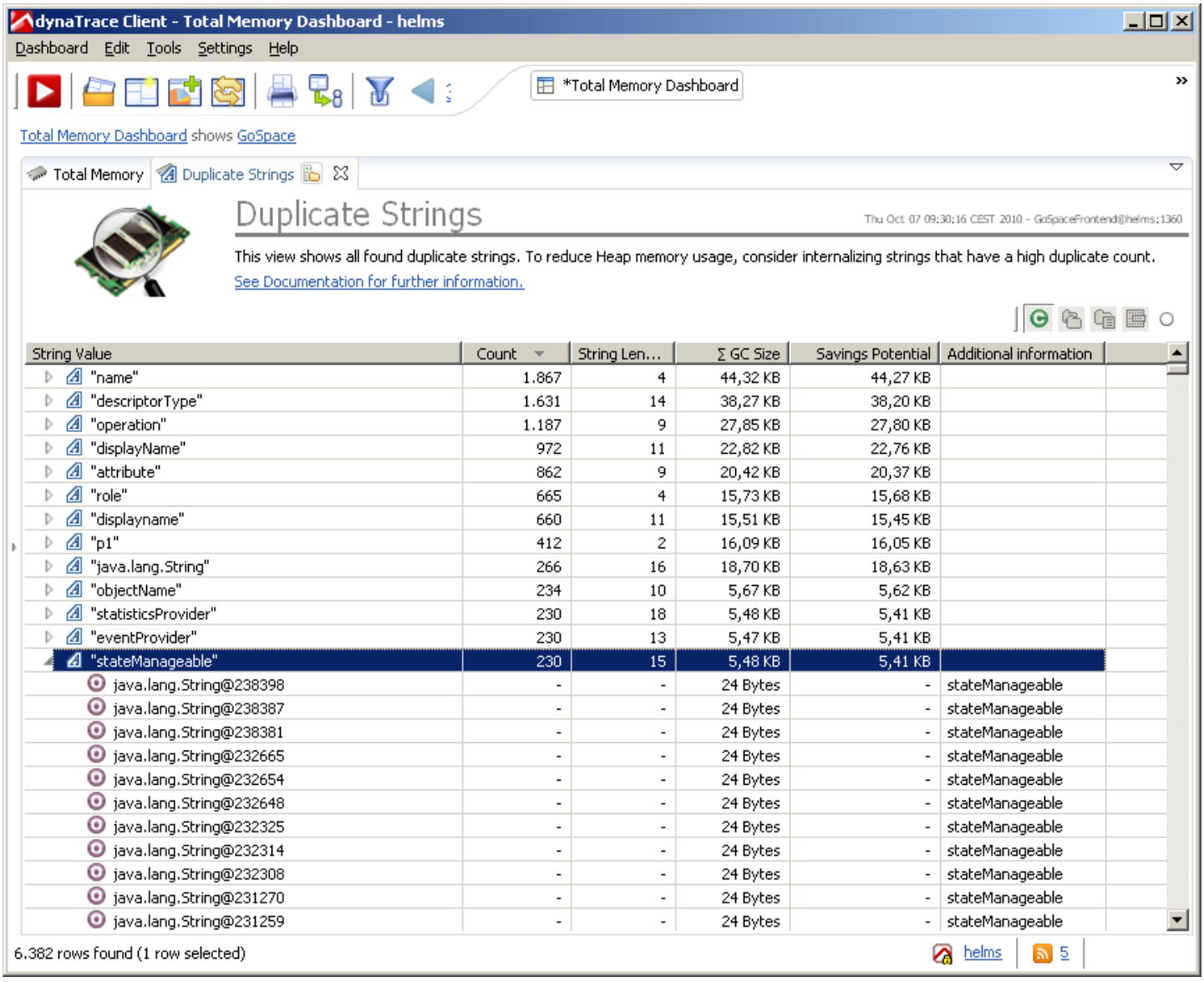
- Follow References - Direct
- Follow References - By Keep Alive
- Shortest Root Paths - Direct
- Shortest Root Paths - By Keep Alive
- Keep Alive Set Dashlet
- Fields and Values

Duplicate Strings Content dashlet

To prevent duplicates, internalize strings or create a string cache. Strings are stored in the Perm Gen, which may fill up. You can build a custom string cache to *long-term reference* only a single string Instance. Usually, you have to look up a string.

If strings are not massively duplicated you should not have to remove duplicates. If a snapshot contains many strings and the Garbage Collection Size of these strings is quite high, however, check string duplicates. To investigate where the string duplicates are referenced, use the Follow References Dashlet drilldown on the string record.

Eliminate string duplicates

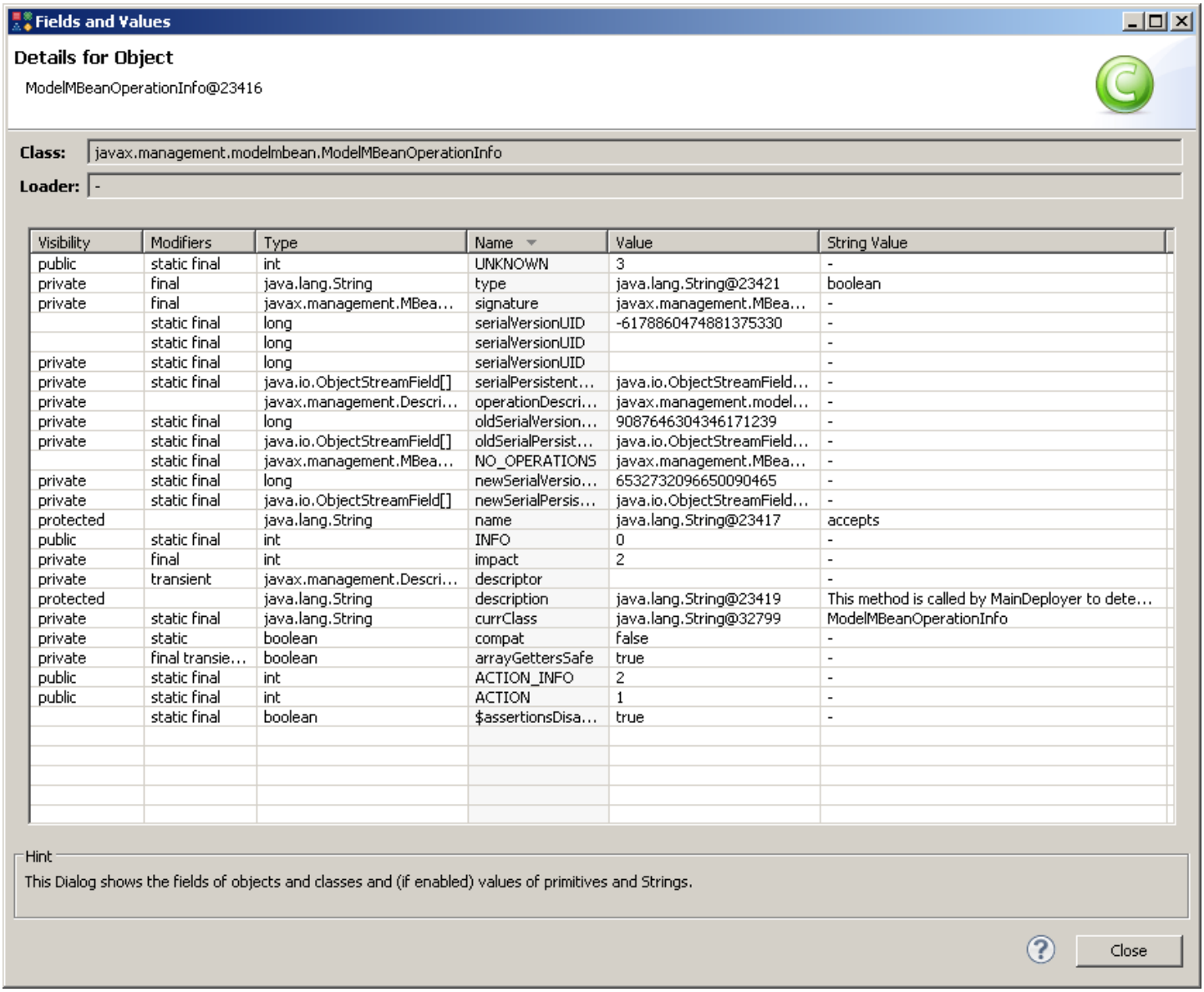


Total Memory Dashboard

- **String value**: String value of the corresponding string Instances. For an expanded string, shows the individual string instances that allow further analysis drilldown. Please note that for the string value records (first-level) the Follow References Dashlet drilldown is available, showing where the corresponding string duplicates are referenced.
- **Count**: Number of string duplicates.
- **String length**: String length in characters.
- **Sum GC Size**: Sum of the Garbage Collection Size of the corresponding string duplicates.
- **Saving Potential**: Amount of bytes that you can save potentially, if all duplicates are eliminated.

To open the **Duplicate Strings Content** dashlet, click **Show Strings** in the **Overview** tab, or use the context-menu of a memory snapshot. It shows the post-processing result of the string duplicate analysis. The corresponding information is only available if you enable string value capturing while you create the snapshot. The table shows all strings that are included at least twice in the snapshot. The individual columns show the following information:

Fields and values



Fields and Values dialog box

- **Visibility**: Private or protected.
- **Modifiers**: Final, static, static final, or final transient.
- **Type**: Double or any object type such as java.lang.String.
- **Name**: The name of the field as defined by the source code.
- **Value**: For primitive fields, the corresponding primitive value. For object fields, the corresponding referenced object. This column only shows values if primitive values are captured
- **String value**: The option for string value must be enabled during snapshot creation.

The following columns as depicted in the figure below are shown:

The top of the dialog box shows the instance or class name that is used in other analysis dashlets. The Classloader specifies which corresponding class is loaded. A value of “-“ means that the class was not created by a Classloader, or was loaded by the bootstrap class loader.

If you do not enable primitive value capturing, while you create the snapshot, the corresponding field values are not shown. However, field-names and modifiers are shown. To capture primitive values, you must have Java with VMs that are equal to or greater than 6.0.

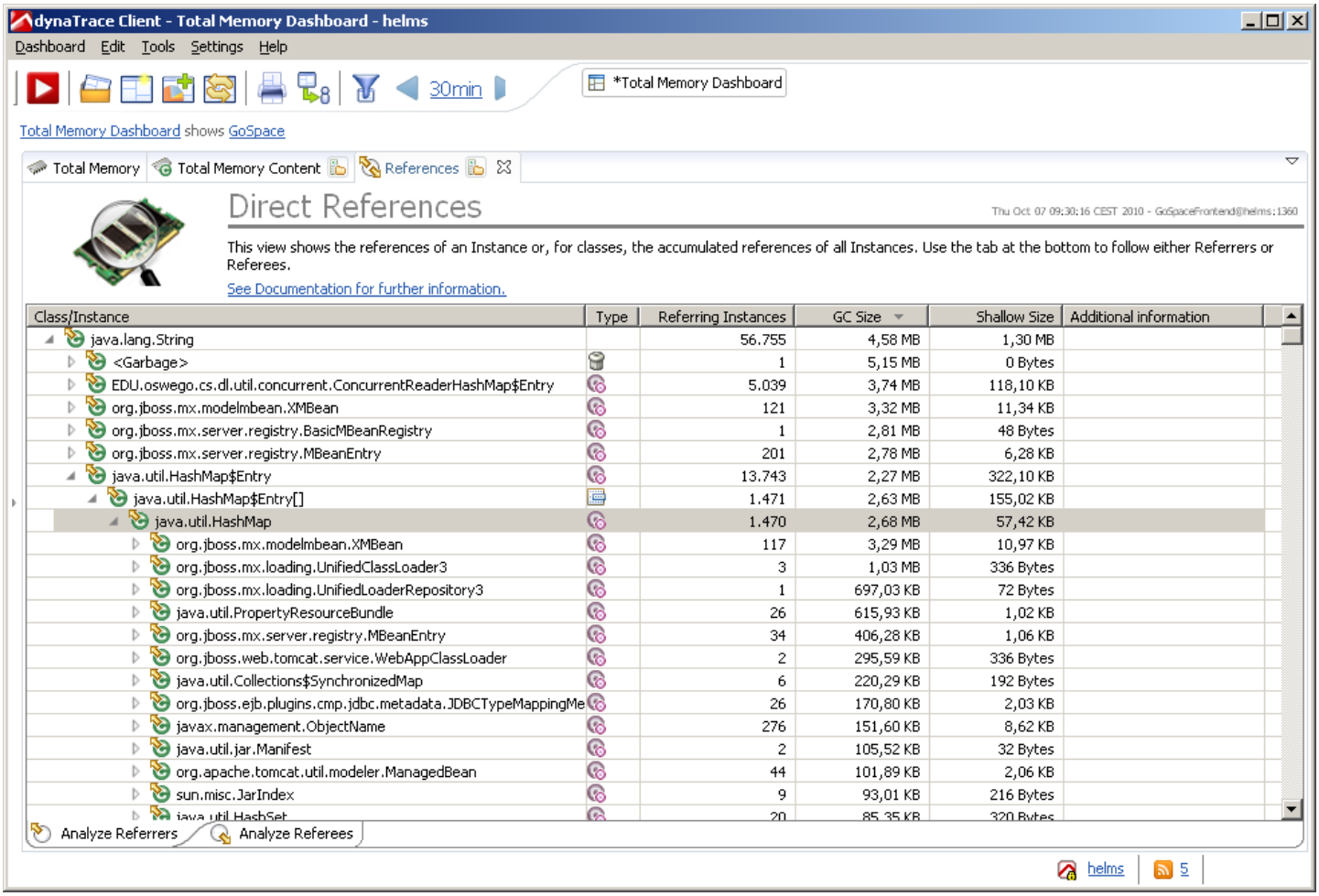
Use the **Fields and Values** dialog box to inspect the fields and their values of single instances and/or classes. It shows all fields declared in the corresponding class, all fields of parent classes, and all fields defined statically.

Overview

Follow References dashlet

For aggregated reference analysis, the GC Size column displays the Garbage Collection Size of the referring / referenced instances only, whereas shared children are considered. In the example above the GC size of 2.68MB for HashMap's is the size of the referring 1.470 HashMap instances, whereas shared children of these Instances are considered.

The columns Referring / Referenced instances count the number of instances that are referring / referenced by a higher level node. In the example above, 13.743 HashMap$Entry instances refer a string instance. To further analyze those 13.743 referring instances, select **Show Instances** in the context menu to open them in a new **Total Memory Content Dashlet** dashlet. Expand the HashMap$Entry node to show the referrers of those 13.743 HashMap$Entry's (only those which refer strings). They are referred by Entry[] and further by 1.470 HashMap's. Those 1.470 HashMaps are thus all HashMaps on the heap that contain at least one string.

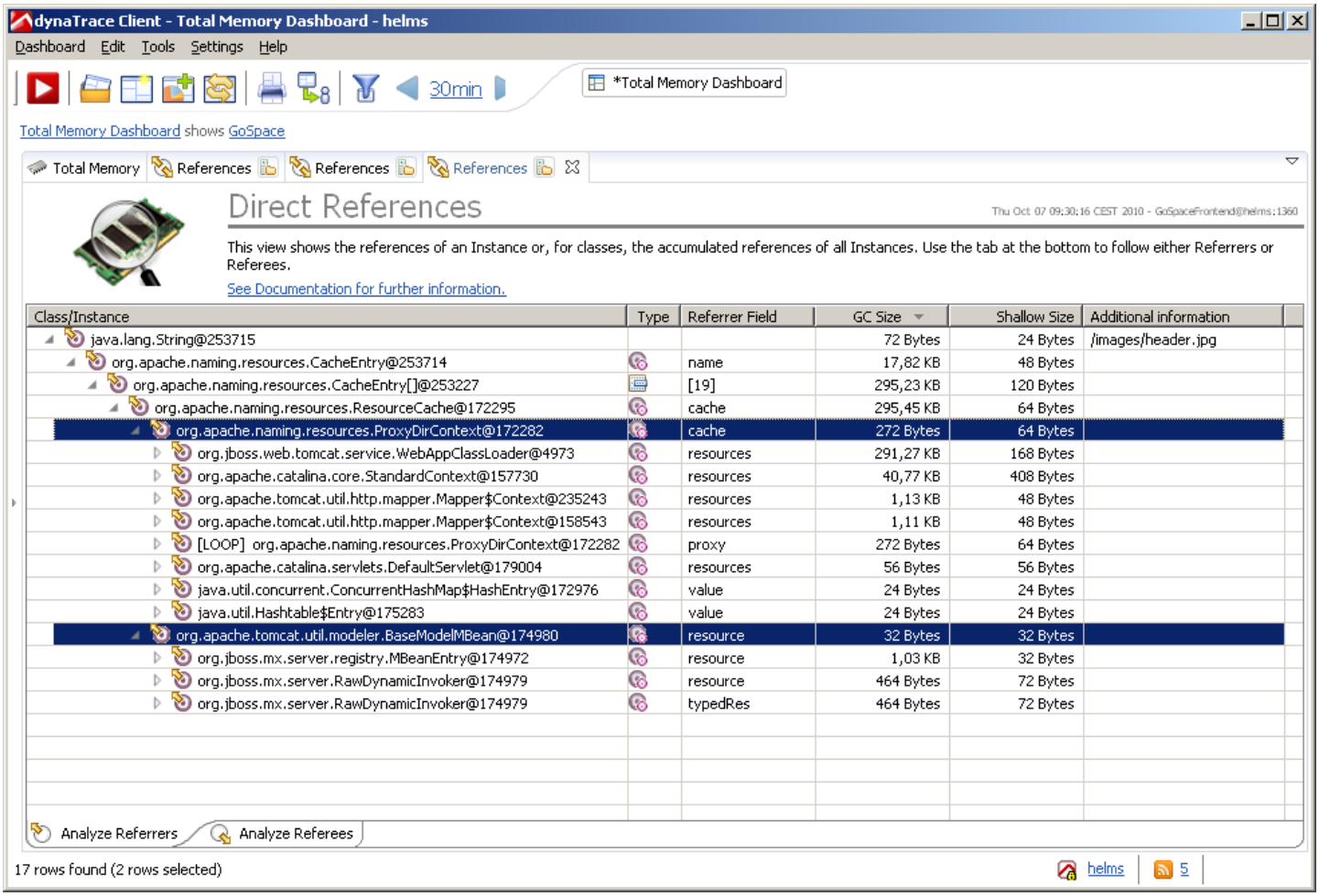


Direct References

Click a class record to follow the references of all instances of a class (e.g. in the Total Memory Content Dashlet dashlet). This generates an aggregated view of Referrers or Referees. Every instance of the class is investigated, and the corresponding references are grouped by their classes.

Aggregated references

The example above shows the direct Referrers of the same instance as shown in the Follow References By Keep Alive. Please note that the referrer-tree splits into to sub trees on the “Resource-Cache” node. This is the reason why in the keep-alive references Dashlet <VMRoot> is shown as the node keeping this instance alive; obviously, this instance is referenced indirectly from two Heap root instances.



Direct References

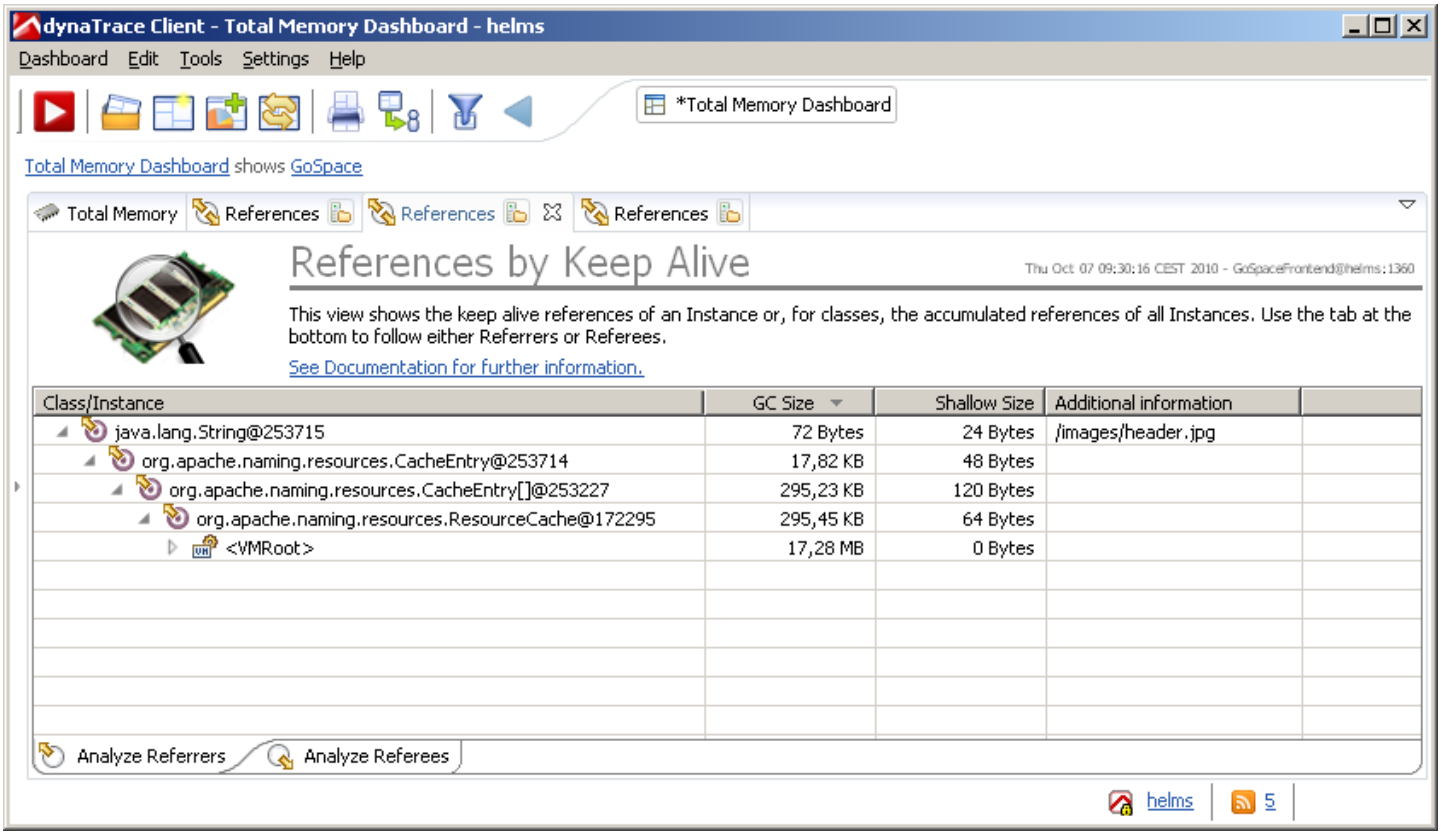
- **Class/Instance**: Class or instance that is referring or referenced. If the record is already contained in the tree up to the original record, a **LOOP** tag is added to indicate a loop. It usually does not make

sense to further follow loops.

- **Type**: Reference type as reported by the VM. Tooltips show a textual representation of the shown icon.
- **Referrer Field / Referee Field**: Referring/referenced field name as defined in the source code of the corresponding class.
- **GC Size**: Garbage collection size of the instance.
- **Shallow Size**: Shallow size of the instance.
- **Additional Information**: Data related to the record. For strings, it shows the string value if string-value capturing was enabled. For most collections, it shows the collection size if capturing primitive values was enabled.

When you follow References Direct, the instance Referrer/Referee is shown as found in the VM/CLR Heap. The table shows the following columns:

Direct



References by Keep Alive

Follow references by Keep Alive to show the (usually indirect) Referrer that keeps an instance alive, or

Referee that is kept alive by an instance. The references are not the direct references of the object graph on the heap but the Referrers or Referees on the Keep Alive Graph. References by Keep Alive are more lightweight than direct references. Use them to determine why an instance is not garbage collected. Please note that only one single instance can be the Keep-Alive-Referrer of another instance. If an instance is referred by two instances, the Keep Alive Referrer is the first Referrer up the Referrer-Hierarchy that exclusively and indirectly references the instance. If there is no such record (as the instance is indirectly referenced by multiple Heap roots) the virtual `<VMRoot>` record is shown. For more information, see Keep Alive Graph.

By keep alive

Note

By default, the **Follow References** dashlet does not show references from or to classes. Click **Ctrl+F** to use the filter bar and change this.



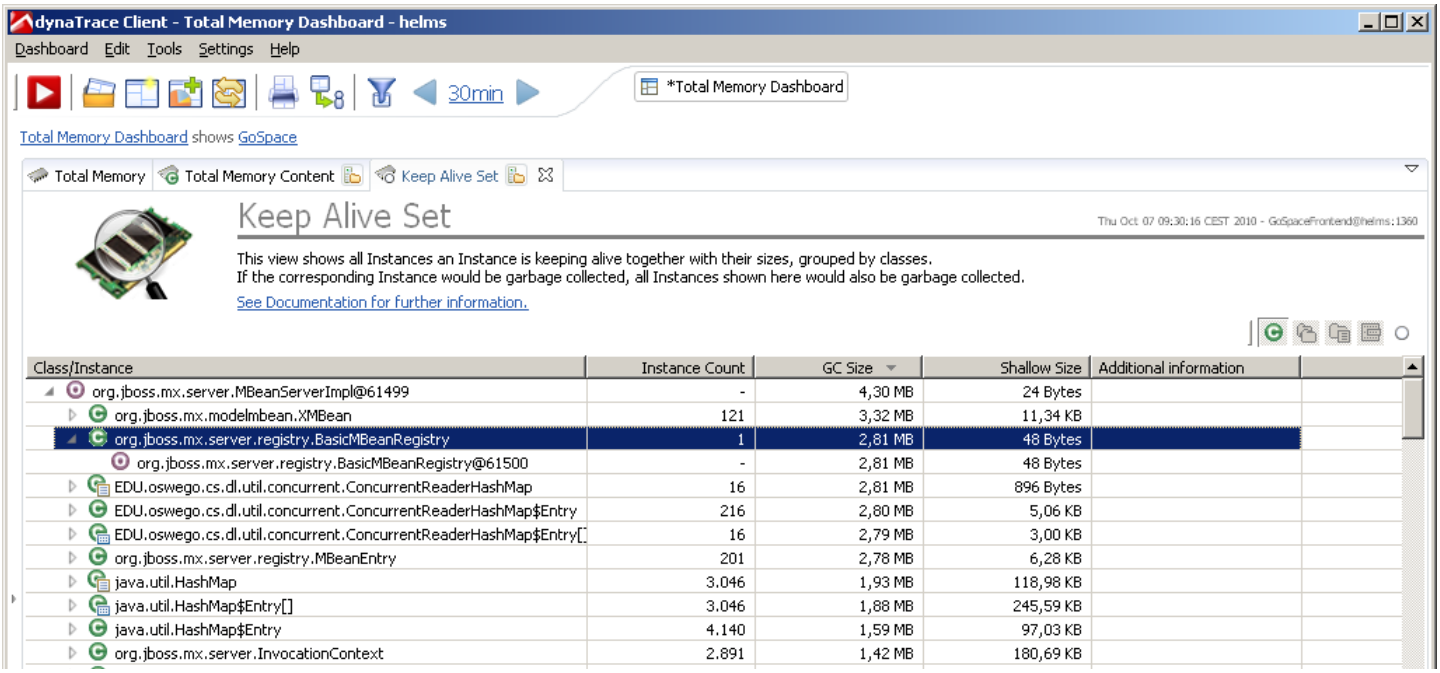
Use the **Follow References** dashlet to inspect the references for an instance, or a set of all Instances of a class. Since it requires instance information, it is only available for Leak Analysis snapshots. Use it to follow referring instances (Referrers), as well as referenced Instances (Referees). Use the tabs at the bottom of the dashlet to switch between Referrer and Referee analysis.

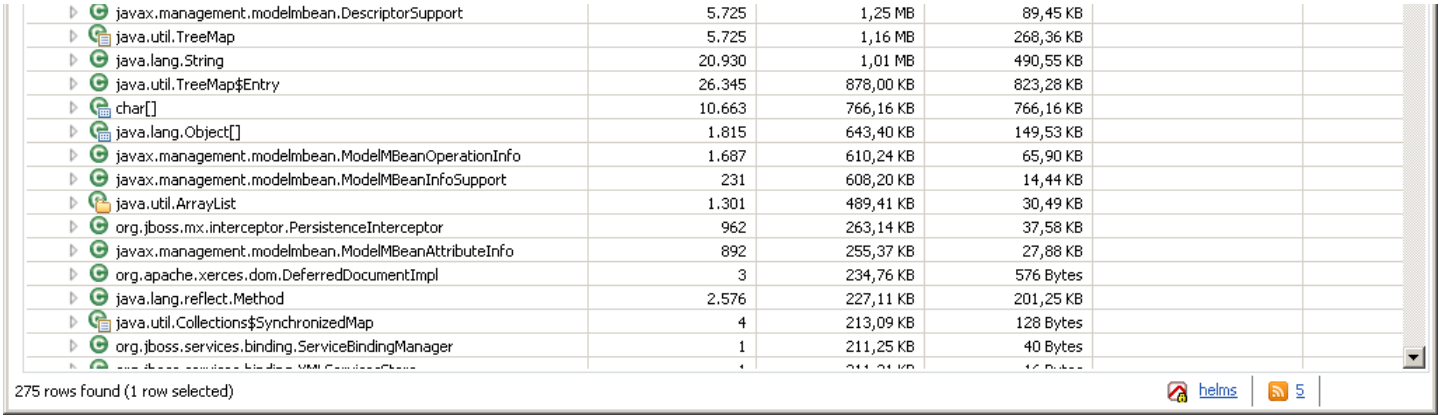
Keep Alive Set dashlet

> **Note**
>
> Click **Show Instances** on the context menu to further analyze a set of instances in a new **Total Memory Content Dashlet** dashlet.

- **Class/Instance**: Instances kept alive are grouped by their class. Expand the corresponding class to view the individual Instances kept alive. For both classes and Instances, all Memory Snapshot analysis drill-downs are available.
- **Instance Count**: Number of Instances of the corresponding class.
- **GC Size**: Garbage Collection Size of the instances kept alive. For the class, this is the GC size of all instances kept alive, whereas shared children are considered. Because instances may have shared children, the GC size shown for classes may not be simply the sum of the GC size of all instances.
- **Shallow Size**: Shallow size of the instances kept alive. For classes, this is the sum of the shallow size of the instances.
- **Additional Information**: Information related to the corresponding records. For example, string values for string instances and size values for collections.

The individual columns show the following information:





Keep Alive Set dashlet

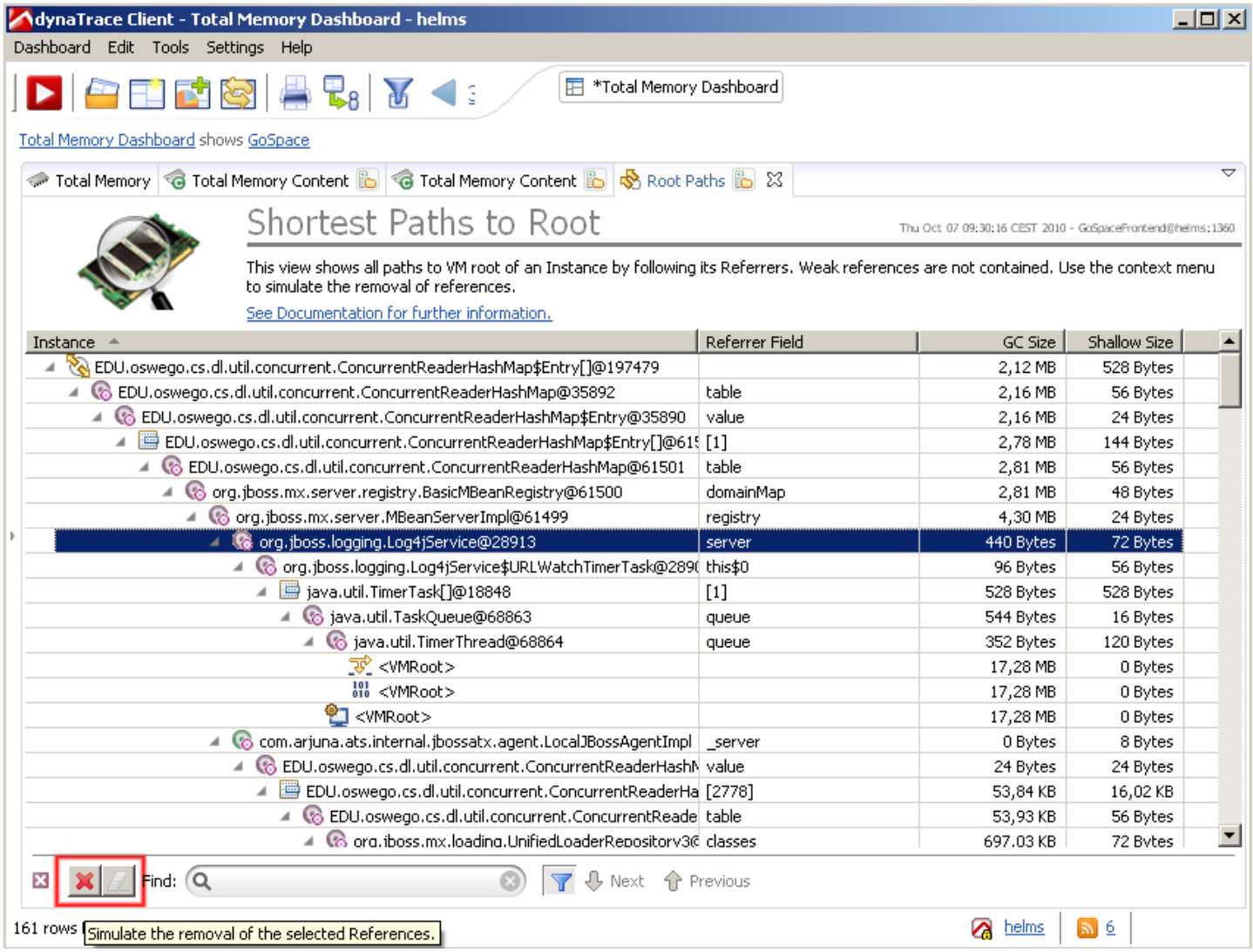
Keep Alive Set Dashlet

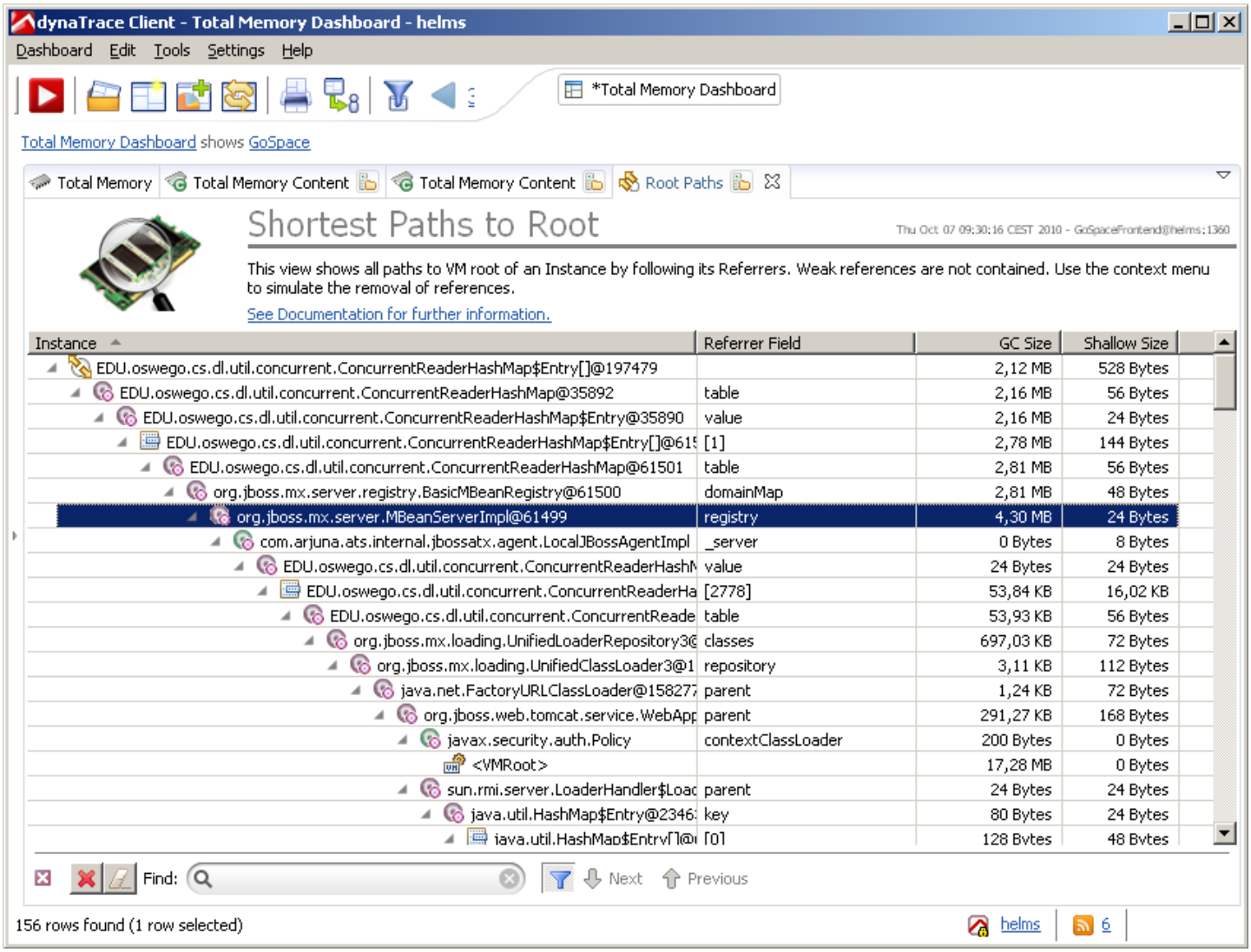
The following figure illustrates a **Keep Alive Set** dashlet.

The Keep Alive Set calculation is based on instances. Therefore, it is only available for Leak Analysis snapshots.
The **Keep Alive Set** dashlet is the set of instances that are “held” by an object instance. If the corresponding instance is garbage-collected, all instances held by this object can also be garbage-collected.

Shortest Root Paths dashlet

After you remove `Log4jService`, the dashlet shows only one path to the heap root, and runs over the class `LocalJBossAgentImpl`. You can see that this is a static reference by the item to the left of the class name.





In the example below, we have a Path to Root for the `MBeanServerImpl` instance. Assume that the path that runs over `Log4jService` is correct. Use the context menu or the controls in the filter bar (Ctrl+F5) to simulate removal. To re-include the reference, use the context menu or the filter bar. To restore a single removed reference, use the context menu. Use the filter-bar to restore all removed references.

As described above, the **Direct Path to Root** dashlet does not show every path to a root instance because in a real world heap this results in numerous paths. Use the context menu or the controls in the filter bar (Ctrl+F5) to simulate a reference removal and recalculate the corresponding Direct Paths to Root again. Use this to verify that an instance is no longer reachable from a heap root if you remove a reference. Also, you can use this to remove references within a Root Path that are correct, until you find a suspicious reference.

Simulate reference removal

Reference from an instance to another instance



Reference from a class to an instance (static)



Reference from an instance to a class object



System class root reference or a .NET finalizer reference



JNI Global reference, a JNI local reference or a .NET Handle reference



Reference that is not reachable from a heap root



Weak reference



Thread root reference



Root reference to an instance on the stack



Monitor root reference

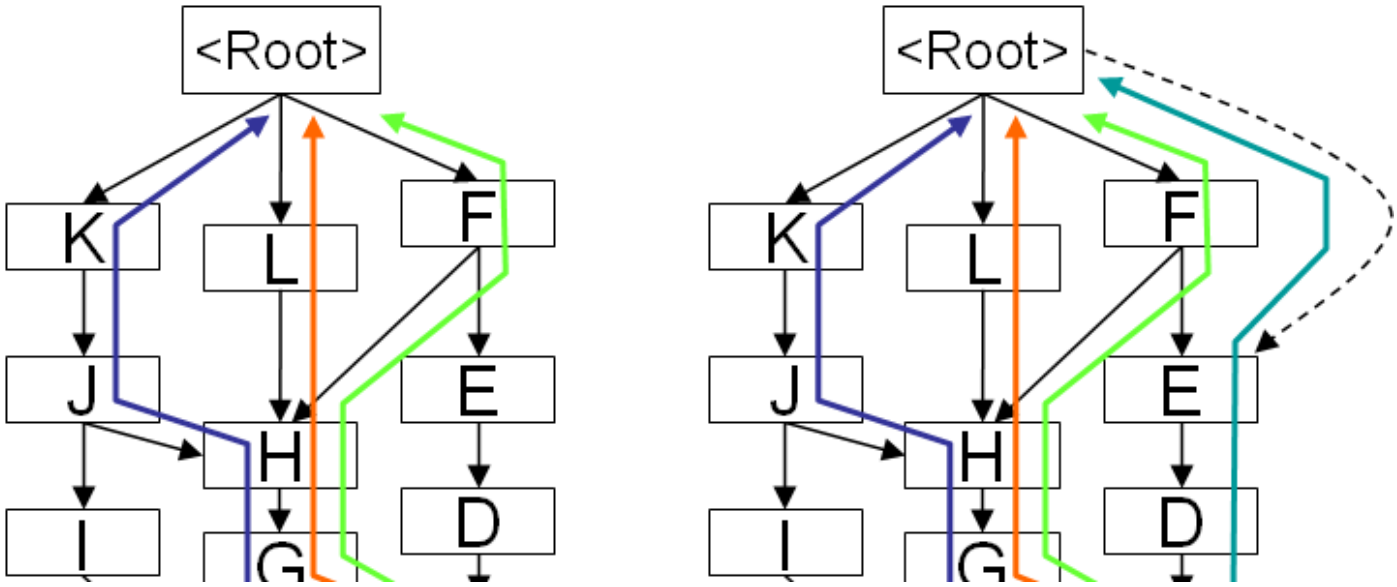


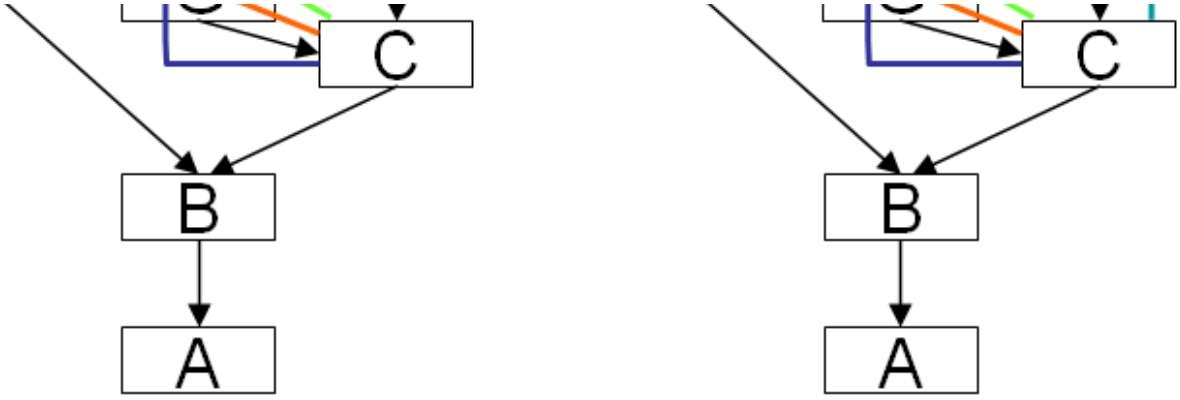
Reference from an array to one of its elements

The **Direct Path to Root** dashlet uses the icons below to visualize references between two instances or classes. The tooltip displays text for the references.

Reference types

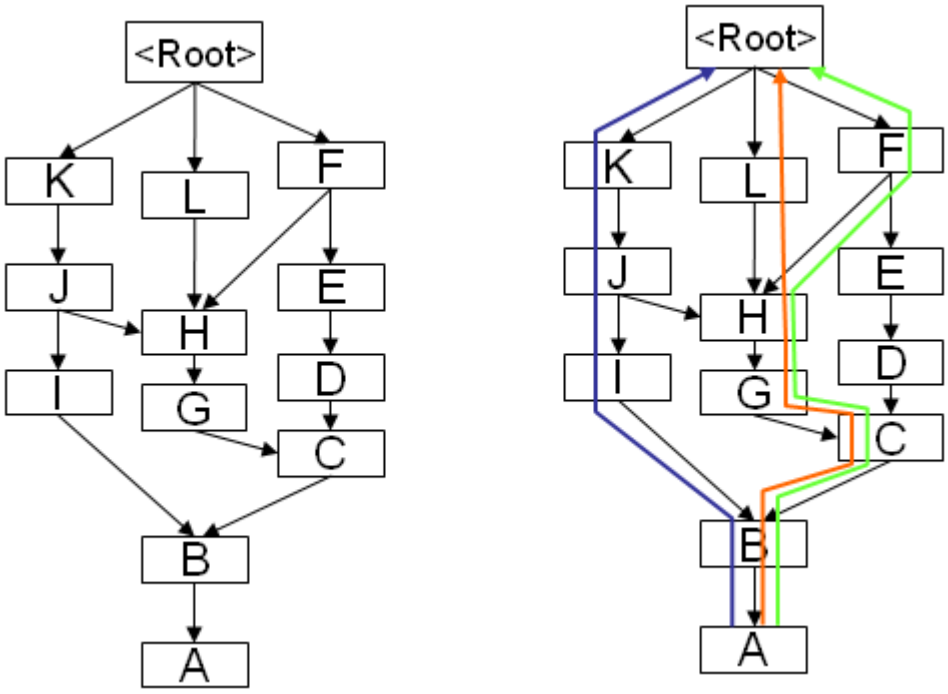
Before the modification, we have three paths to root, as in instance A. After the modification, we have an additional path, C->D->E, because E has no other path to root.





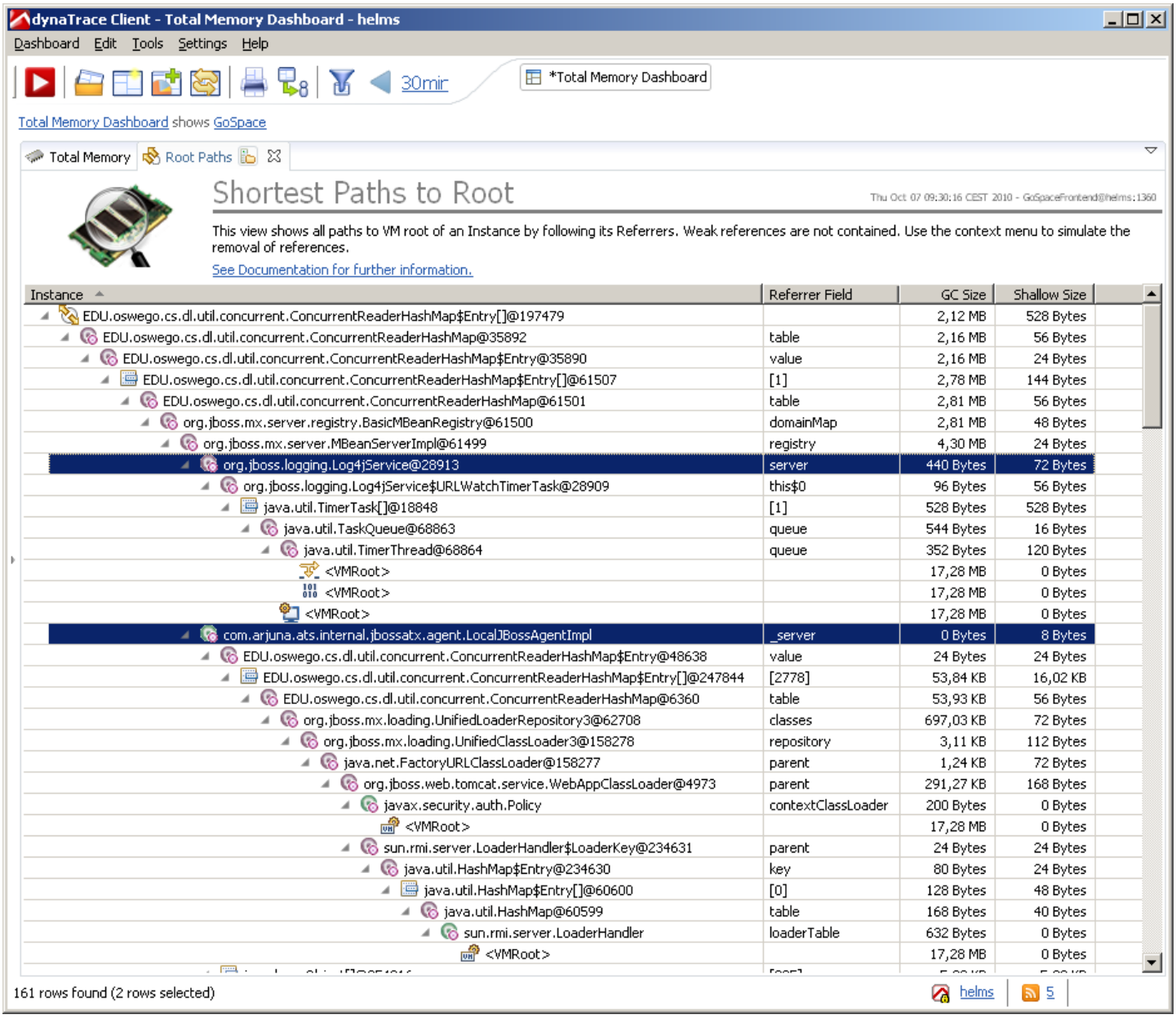
Modify the above reference graph: The heap root directly references instance E. AppMon reported the following Direct Paths to Root for instance C:

The reference graph of this example leads to three Paths to Root for instance A. The first (blue) runs via A->B->I->J->K. The second (orange) runs via A->B->G->H->L. The third (green) runs via A->B->C->G->H->F. Additional paths must root via A->B->C->G->H->J->K and A->B->C->D->E->F. So why are those paths not shown? The path via H->J is not shown because there is already a path running via J and this path is shorter (5 vs. 7 records). The path via C->D is not shown because it runs over F, which is already contained in the green path of an equal length.



The Direct Path to Root dashlet does not show each path to root. If an instance has two referrers that lead to the heap root, but run over the same parent instance, only one path is shown to prevent numerous paths. Consider the following example:

Shortest Paths to Root

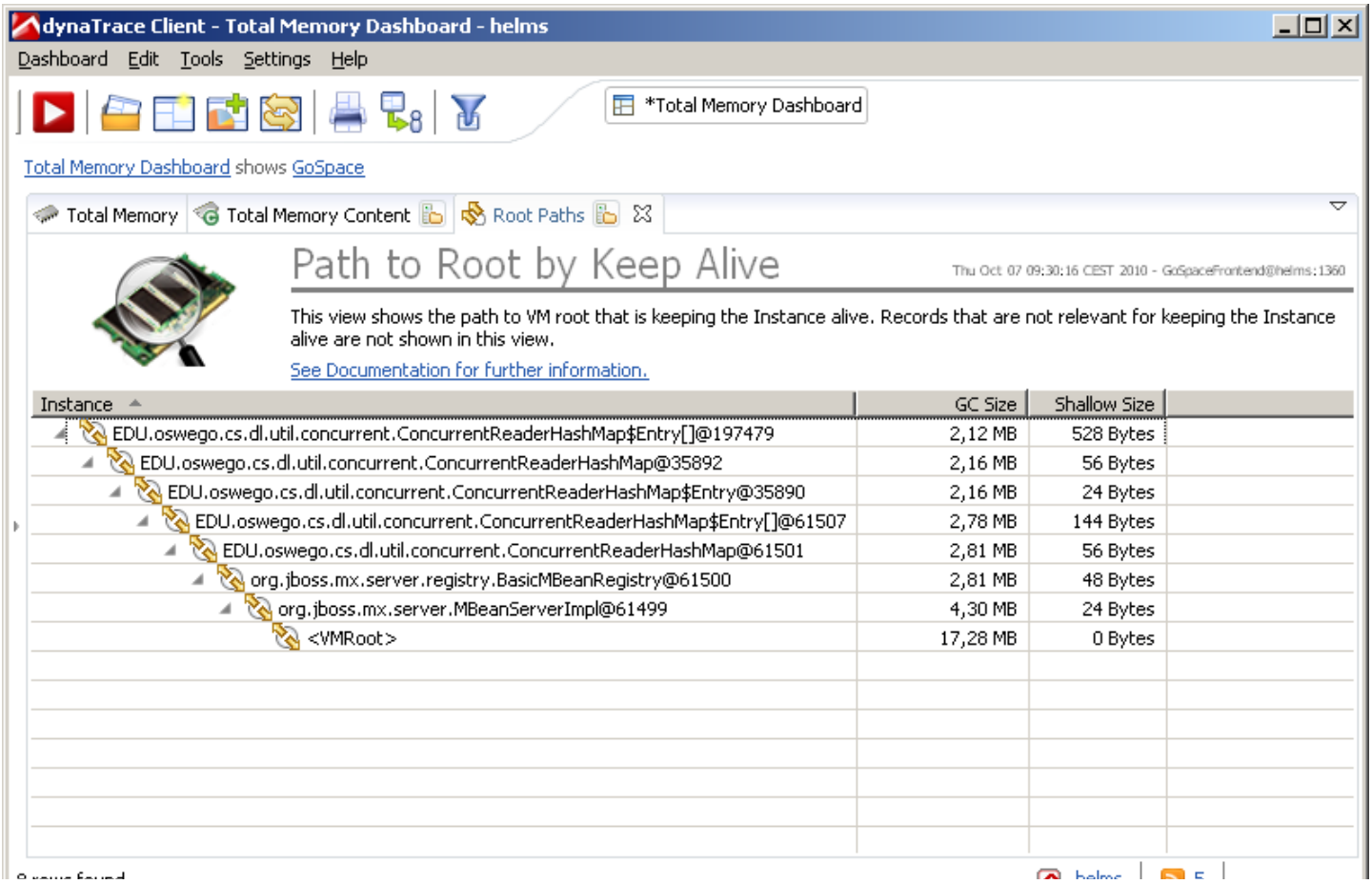


The Direct Path to Root dashlet tracks the referrers on the reference graph, until they reach the heap root. Most instances have more than one path to the heap root. The example below describes these paths. The following figure illustrates the Path to Root Direct for the same instance as the Path to Root

by Keep Alive shown above. The instance MBeanServerImpl@61499 has two referrers that use two different paths to the heap root. In the Path to Root by Keep Alive above, the record above the MBeanServerImpl is the <VMRoot> instance, because there is no instance above MBeanServerImpl that exclusively references it. For more information, see Keep Alive graph. In contrast, the Path to Root Direct dashlet displays these direct referrers and the path to root that they spawn.

Direct

Path to Root by Keep Alive





To calculate the Path to Root by Keep Alive, follow the Keep Alive referrer instance in the Keep Alive graph, until you reach the heap root instance. In the Keep Alive graph, each instance has exactly one Keep Alive referrer. The Root Path by Keep Alive is quickly calculated, but does not always show the reference graph's direct references. However, it does provide a good overview of why an instance is kept alive. If you require direct references, use Follow References Direct.

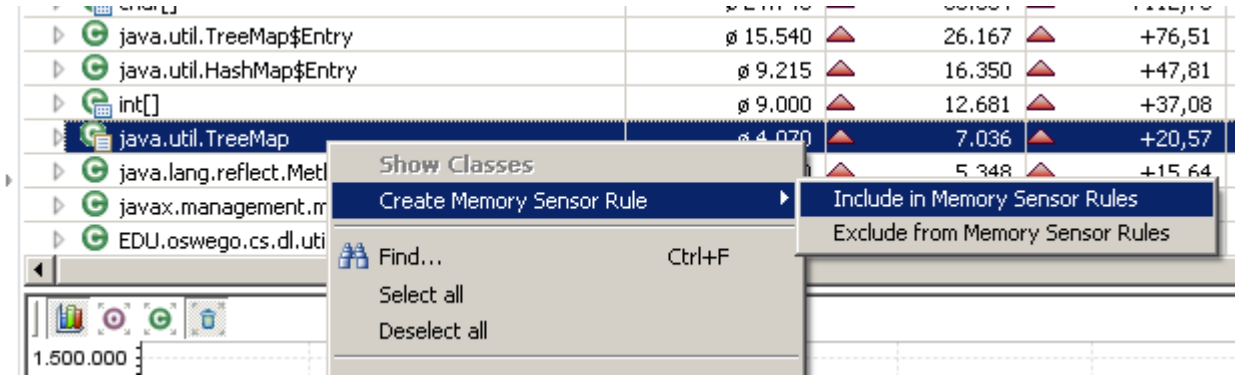
By keep alive

The **Shortest Root Paths** dashlet calculates the path of referrers to heap roots on the reference graph or on the Keep Alive graph. Use this to analyze why an instance is not garbage collected.

Total Memory Comparison dashlet

If you select a class in the upper pane of the dashlet, the lower pane shows a chart of the currently selected measure of the selected class. You can adapt the corresponding measure (instance count, GC size, or shallow size) with the filter toolbar in the lower pane. You can chart a maximum of 25 classes simultaneously.

Class comparison chart



Create Memory Sensor Rule Option

Use the class record context menu to define Memory Sensor rules in the** Comparison** dashlet. Use this to create Sensor rules to grow classes for further Selective Memory Dashlet analysis. This analysis shows where, in code, the corresponding instances are allocated, as well as where this allocations occurred along a PurePath.

Create a Memory sensor rule

The Average Delta per second values for instances, GC size, and shallow size show the change rate of the corresponding value over time. A value of e.g. +4.23MB for GC size means that in average, the GC size of all instances of a class increased by 4.23MB per second. This value is heavily influenced by the time between the individual snapshots. For example, if you have two hypothetical snapshots where the GC size for class A is 100MB and 200MB, and these snapshots have a time difference of 100 seconds, the average GC size delta per second is +1MB. If you have the same GC sizes with two snapshots that have a time difference of only 25 seconds, the average GC size delta per second is +4MB.

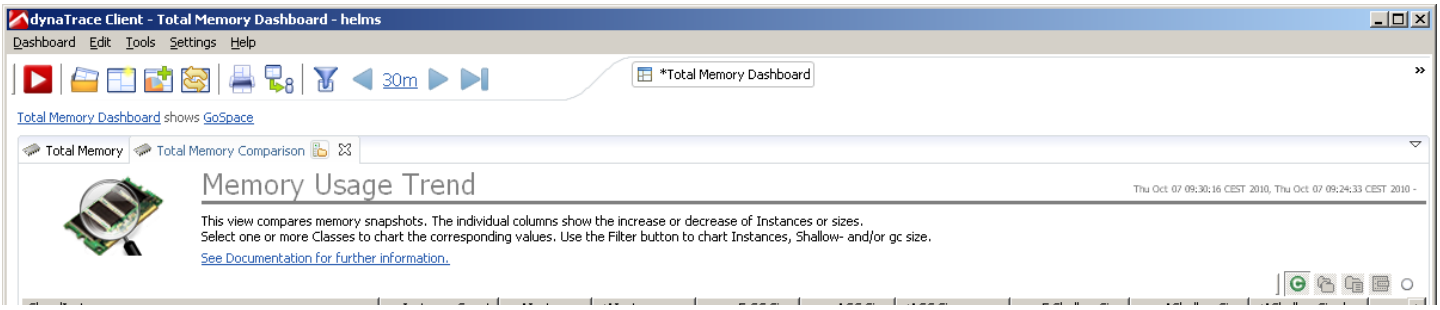
Average delta per second

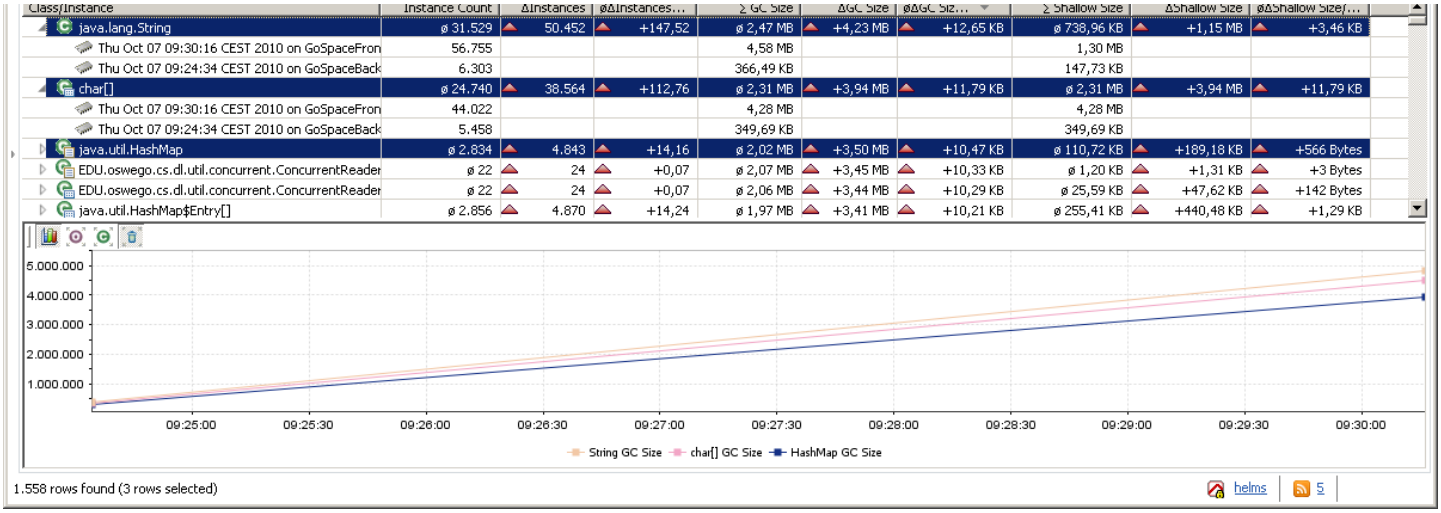
Average delta per second

- **Class/Instance**: Contains the corresponding compared class or the corresponding the value for this class in the corresponding snapshot on second level.
- **Instance Count**: Average number of instances for this class. For the second level, it's the number of instances in the corresponding snapshot.
- **Instance Delta**: The absolute difference of the number of instances for the oldest to the newest Snapshot.
- **Average Instance Delta per second**: The average change of the number of instances per second. See Average Delta per second for more information.
- **GC Size**: For classes, the average garbage collection size. For individual snapshots on level two, the corresponding GC size of the class in this snapshot. This value is not available for Trending snapshots.
- **Class GC Size Delta**: The absolute difference in garbage collection size between the oldest and the newest snapshot of the corresponding class. This value is not available for Trending snapshots.
- **Average GC Size Delta per second**: The average change of the garbage collection size per second (see Average Delta per second, below) for the corresponding class. This value is not available for Trending snapshots.
- **Shallow Size**: The average Shallow Size of the class, i.e. the average sum of the shallow sizes of all instances of the class. For second level records, the shallow size of this class in the corresponding snapshot.
- **Shallow Size Delta**: The absolute difference of the Shallow Size of the class between the oldest to the newest snapshot.
- **Average Shallow Size Delta per second**: The average change of the Shallow Size per second of this class (see Average Delta per second, below).

The upper table contains all the classes that are in the snapshots with the comparison data. AppMon only shows classes with instances and comparisons that have differences. Use the toolbar in the upper left corner to change to filter for maps, collections, and arrays. Expand a class to show the corresponding values for each compared snapshot. The table contains the following columns:

Class comparison



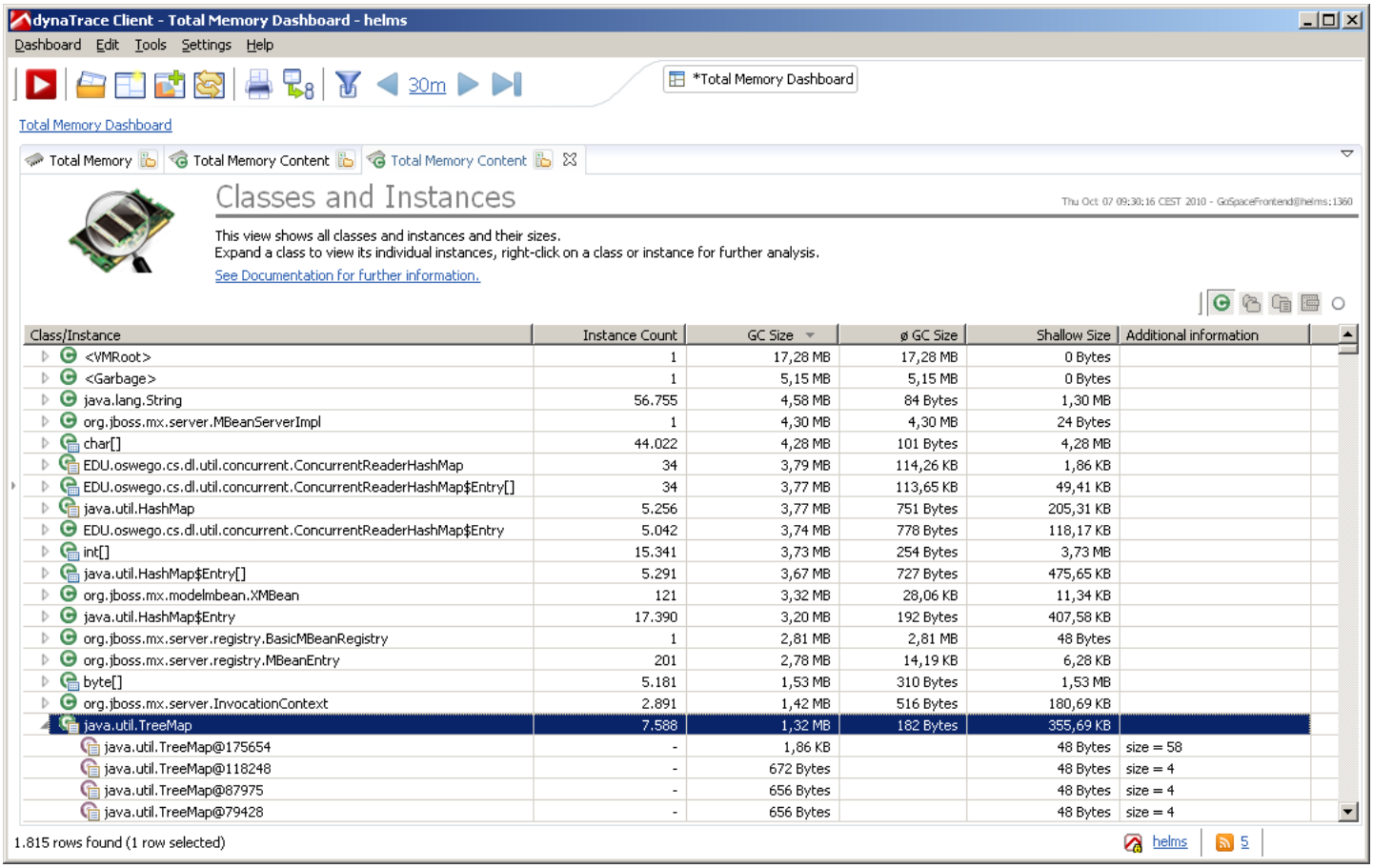


Total Memory Comparison Dashlet Comparing Two Snapshots

The **Total Memory Comparison** dashlet compares the instance counts, shallow sizes, and GC sizes of multiple Memory snapshots. If you compare Trending snapshots and Leak Analysis snapshots, there are no values for GC sizes for Trending snapshots.

Total Memory Content dashlet

Total Memory Dashboard Classes and Instances



Class/Instance	Instance Count	GC Size	ø GC Size	Shallow Size	Additional information
<VMRoot>	1	17,28 MB	17,28 MB	0 Bytes	
<Garbage>	1	5,15 MB	5,15 MB	0 Bytes	
java.lang.String	56.755	4,58 MB	84 Bytes	1,30 MB	
org.jboss.mx.server.MBeanServerImpl	1	4,30 MB	4,30 MB	24 Bytes	
char[]	44.022	4,28 MB	101 Bytes	4,28 MB	
EDU.oswego.cs.dl.util.concurrent.ConcurrentReaderHashMap	34	3,79 MB	114,26 KB	1,86 KB	
EDU.oswego.cs.dl.util.concurrent.ConcurrentReaderHashMap$Entry[]	34	3,77 MB	113,65 KB	49,41 KB	
java.util.HashMap	5.256	3,77 MB	751 Bytes	205,31 KB	
EDU.oswego.cs.dl.util.concurrent.ConcurrentReaderHashMap$Entry	5.042	3,74 MB	778 Bytes	118,17 KB	
int[]	15.341	3,73 MB	254 Bytes	3,73 MB	
java.util.HashMap$Entry[]	5.291	3,67 MB	727 Bytes	475,65 KB	
org.jboss.mx.modelmbean.XMBean	121	3,32 MB	28,06 KB	11,34 KB	
java.util.HashMap$Entry	17.390	3,20 MB	192 Bytes	407,58 KB	
org.jboss.mx.server.registry.BasicMBeanRegistry	1	2,81 MB	2,81 MB	48 Bytes	
org.jboss.mx.server.registry.MBeanEntry	201	2,78 MB	14,19 KB	6,28 KB	
byte[]	5.181	1,53 MB	310 Bytes	1,53 MB	
org.jboss.mx.server.InvocationContext	2.891	1,42 MB	516 Bytes	180,69 KB	
java.util.TreeMap	7.588	1,32 MB	182 Bytes	355,69 KB	
java.util.TreeMap@175654	-	1,86 KB		48 Bytes	size = 58
java.util.TreeMap@118248	-	672 Bytes		48 Bytes	size = 4
java.util.TreeMap@87975	-	656 Bytes		48 Bytes	size = 4
java.util.TreeMap@79428	-	656 Bytes		48 Bytes	size = 4

To further investigate the content of a Memory snapshot (show Shortest Root Path - By Keep Alive or Follow References Dashlet), use the context menu to open the corresponding dashlets. Each analysis is not available for the classes (Keep Alive Set Dashlet).

Analysis drilldown

> **Hint**
>
> When you sum up the GC size of all instances of a class, the result may not be the same as that in the GC Size column. Instances may have shared children. These are considered when you calculate the GC class size.

The most important column for Leak Analysis snapshots is the GC Size column. For classes, it shows

The most important column for Leak Analysis snapshots is the GC Size column. For classes, it shows the sum of the GC size of all instances of that class. It considers the shared children between the individual instances. For instances, it shows the garbage collection size of the individual instance.

The GC size column

The figure below shows two special classes: <VMRoot> class and <Garbage> class. These are virtual classes and do not exist in the SUD heap. They show the corresponding amount of total memory used and memory that is no longer referenced from any root object. Use the analysis drilldown below on those records. For example, the Keep Alive Set Dashlet of the <Garbage> instance, shows the instances that might be garbage-collected.

To show the instances of a class, click on the arrow symbol left to the class. This shows the first 30 records of the corresponding class, based on the current sorting order. When the sort order changes, a new request that reflects the new sort order refreshes these records. Each current instance of each class has an entry that fetches the next 30 records. To show all instances of a class, use the **Show all** option in the context menu.

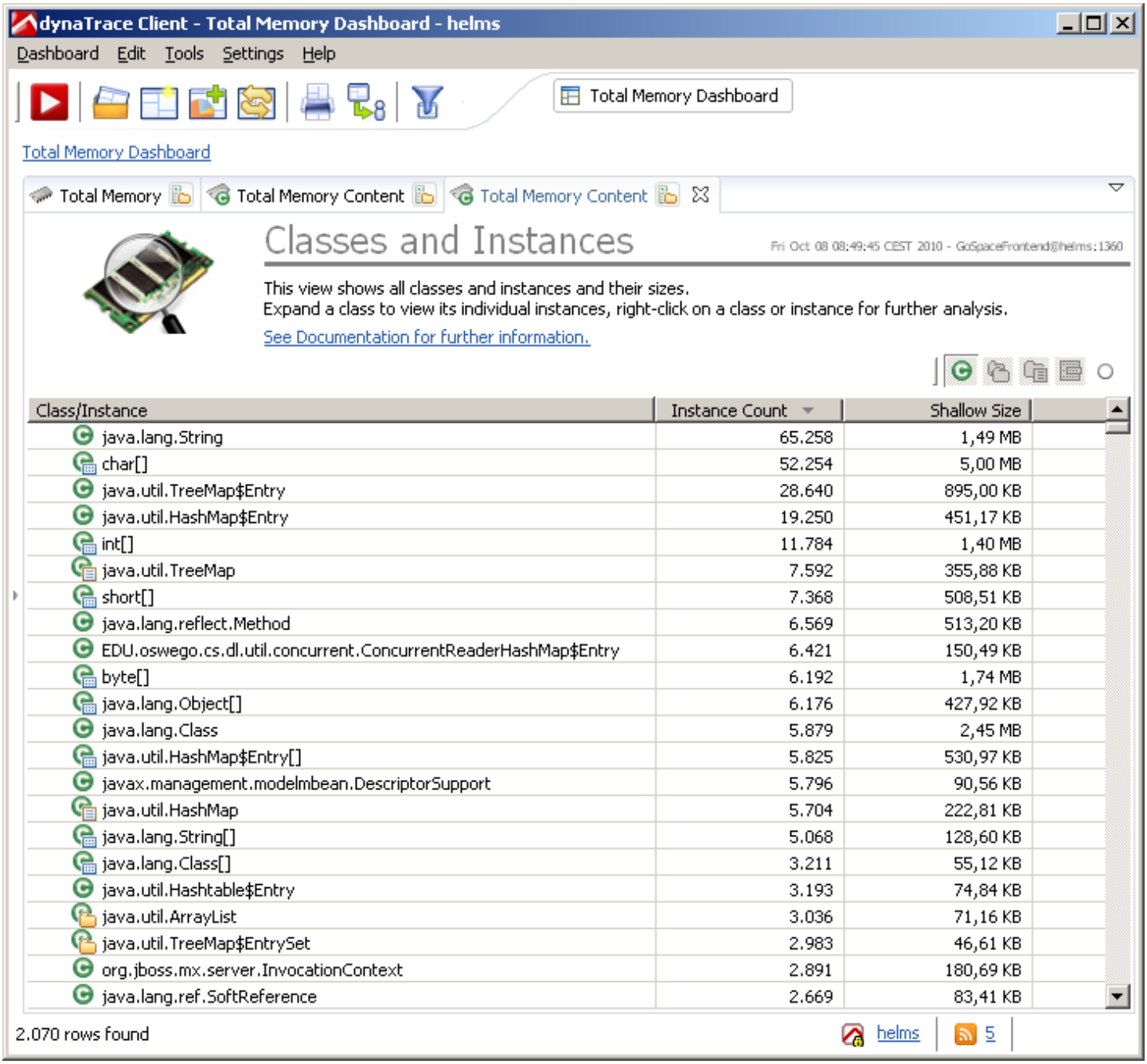
Show instances

- **Class/Instance**: Corresponding class or if a class is expanded, its corresponding instances.
- **Instance Count**: Number of instances of a class.
- **GC Size**: Garbage collection size of all instances of a class, or the garbage collection size of a single instance.
- **Average GC Size**: For classes, the average garbage collection size of its instances. Use the average GC size to identify exceptional outlier instances.
- **Shallow Size**: For classes, the sum of the Shallow Size of all instances. For instances, the Shallow Size of the corresponding Instance.
- **Additional Information**: Information related to the individual record. This column shows string values for string objects (if string capturing is enabled) and the size for most classes that implement the Collection or Map interfaces (if primitive field value capturing is enabled).

For Leak Analysis snapshots, this dashlet contains the following columns:

Total memory content for leak analysis snapshots

Total Memory Dashboard



Use the toolbar at the top right to filter and show classes that have no instances.

For trending dumps, AppMon displays all classes with their instance counts and Shallow Size. Trending snapshots do not contain Instances, so there is no garbage collection size available. Trending snapshots show which classes are instantiated and how often. Trending snapshots have no further drilldowns.

Total memory content for trending snapshots

Hint

Use **Ctrl+F** to filter the shown classes/instances by name.

-  - Normal Instances
-  - Collection Instances
-  - Map Instances
-  - Array Instances
-  - Unreferenced Instances (garbage)
-  - Special virtual Instances (VM-root and garbage-root)

The following icons are used for Instances

-  - Normal classes
-  - Collection classes
-  - Map classes
-  - Arrays

The **Total Memory Content** dashlet and the drilldown dashlets use the following icons to illustrate class types:
The **Total Memory Content** dashlet shows classes and instances (Leak Analysis snapshot) in a Memory snapshot. It shows which classes are instantiated most often and which classes require the most memory.

User Experience dashlets

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Client Errors dashlet

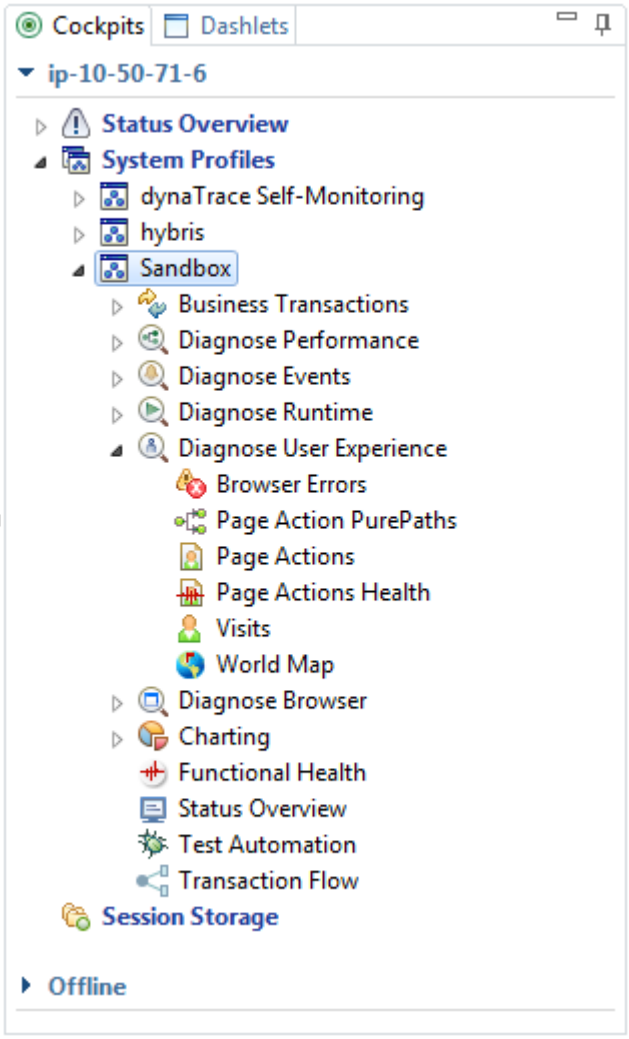
User Action PurePaths dashlet

User Actions dashlet

User Actions Health dashlet

Visits dashlet

Access AppMon's **User Experience** dashlets through the **Start Center** or in the AppMon Client cockpit, click **<system profile> > User Experience**.



User Experience Analysis	Description
Client Errors Dashlet	Overview of JavaScript errors captured by the JavaScript agent, mobile errors, exceptions, and crashes.
User Action PurePaths Dashlet	Represents all captured PurePaths starting in the browser.
User Actions Dashlet	Overview of user action performance, errors, and affected visits.
User Actions Health Dashlet	Overview of system load and failure rate compared to a previous time period.
Visits Dashlet	Overview of visit information including visit duration, bounce rate, and stream count.
World Map Dashlet	Provides a way to visualize visits or user actions in a geographical representation.

Client Errors dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Client errors overview table

Client Errors Detail table

Custom filtering and splitting

Related pages

User Experience dashlets

User Action PurePaths dashlet

User Actions dashlet

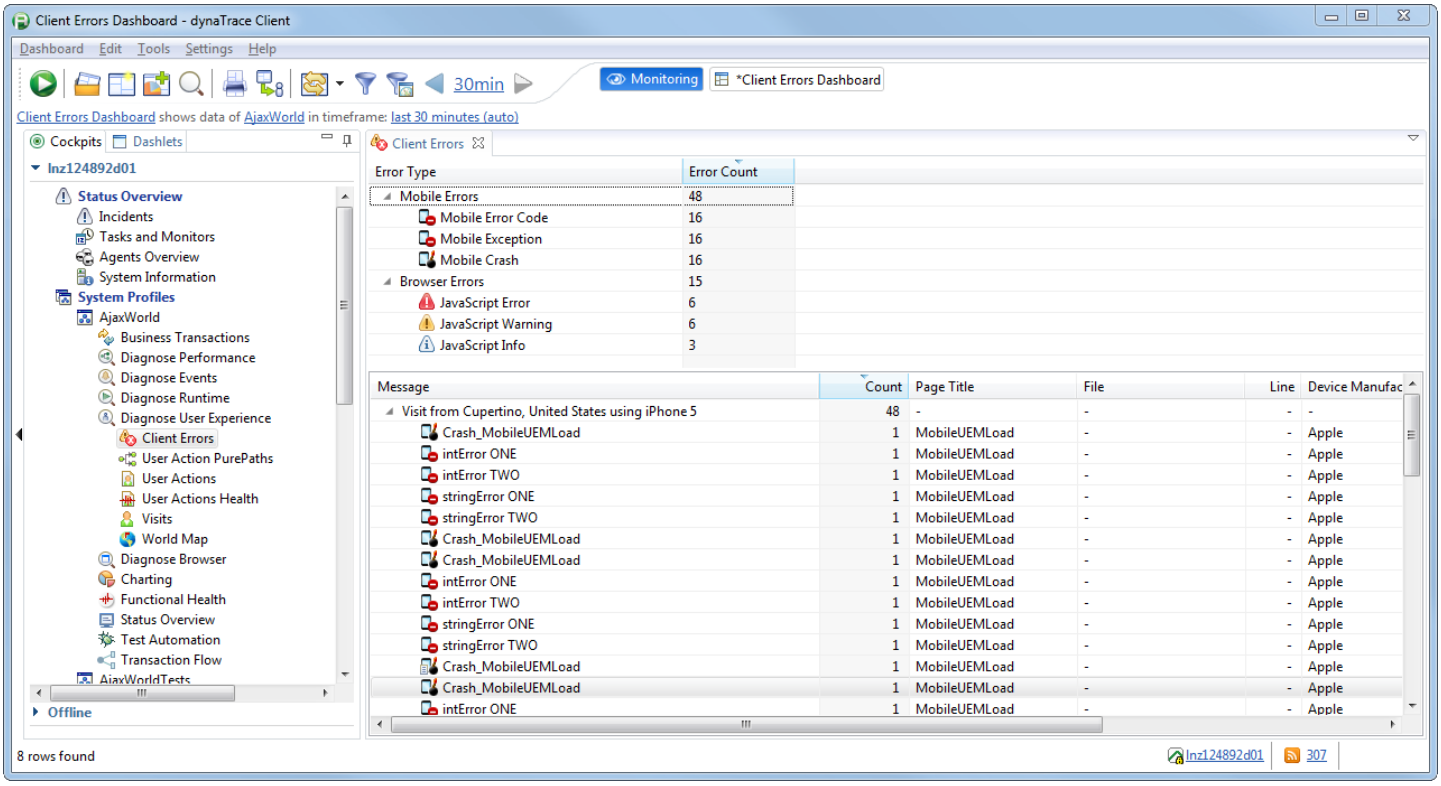
User Actions Health dashlet

Visits dashlet

World Map dashlet

Overview

The **Client Errors** dashlet displays JavaScript errors captured by the JavaScript agent, mobile errors, exceptions, and crashes. Detailed information for errors such as URL, browser, page title, device, and manufacturer is provided and can be used for splitting and aggregating errors.



Client Errors Dashlet

Client errors overview table

At the top of the **Client Errors** dashlet is an overview of the different error types. Additionally, a count indicates the number of errors that occurred within the specified time frame. The error types are:

- **Browser Errors**
  - **JavaScript Error**: Errors, which occurred within the JavaScript code.
  - **JavaScript Warning**: Warnings captured by the JavaScript agent.
  - **JavaScript Info**: Such as log messages.
- **Mobile Errors**
  - **Mobile Exceptions**: Exceptions from mobile applications.
  - **Mobile Error Codes**: Error codes from mobile apps.
  - **Mobile Crashes**: Mobile crashes can contain a crash report, indicating the cause of the crash.

Selecting one or more error types from the overview table changes the details table's content.

Client Errors Detail table

Analyze client errors by different aspects. The dashlet provides different structure modes and filters in the tool bar of the dashlet

. The following aggregation options are available:

- **Error**: Aggregates errors according to the error message given by the browser or mobile application.
- **Client Type, Client Family**: Groups all errors by client type (e.g. desktop browser, mobile browser, and mobile app) and client family with version or device information. For example, IE Mobile - IE Mobile 9.0, or Apple - iPod touch 4G. Another example is an error that occurred using Mozilla Firefox 6 is grouped under Desktop Browser - Firefox - Firefox 6.0.
- **Application**: Groups errors by application.
- **Visit**: All errors are grouped according to their visit within which it occurred.

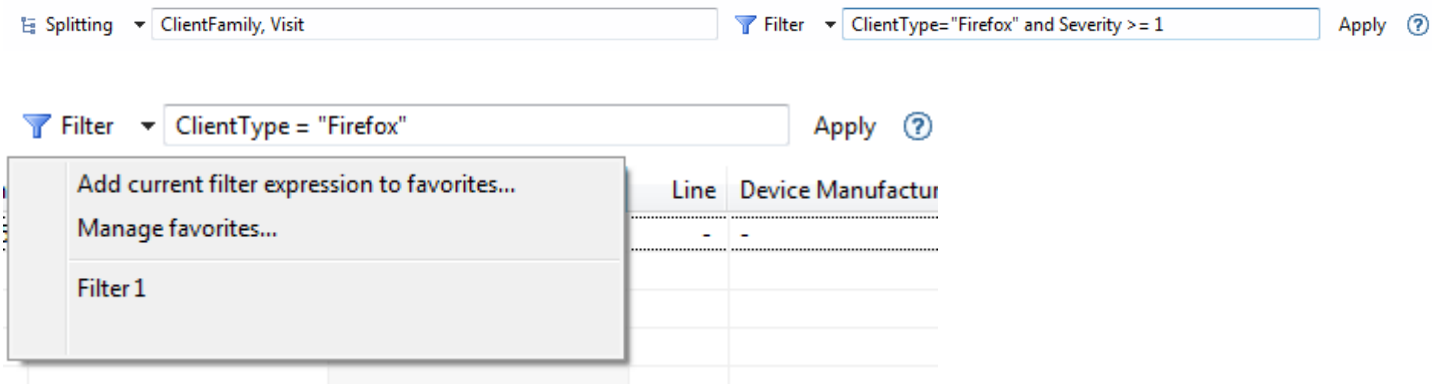
Drill-Downs to other dashlets (e.g. User Actions Dashlet, User Action PurePaths Dashlet, and Visits Dashlet) are provided for deeper analysis.

Custom filtering and splitting

Click **Custom Query** on the toolbar for custom filtering and splitting

. Filtered expressions support a Java-like syntax. Splitting expressions consist of a comma separated list of properties. Click **<CTRL>-<SPACE>** for a list of available filter and splitting values. Store favorite filters and splittings for future use.

Custom Filter



Supported syntax

Expressions support a Java-like syntax. The following tables describe constants and operators recognized within expressions:

Constants	Sample
Bool	true, false
Number	123, 3.14
String	“text/html”, ‘Firefox’
Regular expressions	/[0-9A-F]*/

Operators	Description
, , or, OR	logic or
&&, &, and, AND	logic and
!=, <>	not equals
==, =	equals
<, >, <=, >=	relational operators for *smaller* , *greater* , *smaller than or equals to* and *greater than or equals to*
=~, ~=	matches a regular expression (right hand side is a regular expression)
!=~, !~=, <>~	does not match (right hand side is a regular expression
like, LIKE	SQL style string match function using ‘%’ as joker character

Filter Examples

`FamilyFamily="Firefox" and Severity >= 1`: Filters client errors for Firefox browsers that have a severity greater than 1.
`Path like "%community%"`: Only show client errors whose path contains the phrase community.

Splitting Example

`ClientType, Visit`: Groups client errors by their client type and visit.

User Action PurePaths dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

PurePaths

Contributors

Errors

Tree

Transaction Flow

API Distribution

Related pages

User Experience dashlets

Client Errors dashlet

User Actions dashlet

User Actions Health dashlet

Visits dashlet

World Map dashlet

System Profile - User Experience

System Profile - Error Detection

System Profile - User Experience

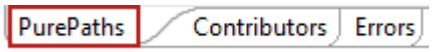
System Profile - Applications

PurePaths Dashlet

Overview

The **User Action PurePaths** dashlet represents all captured PurePaths that start in the browser. Use this to analyze user actions like page loading or button clicks. A user action PurePath begins at the start of an action like loading the https://www.dynatrace.com/en page. It contains all web requests to instrumented system components (web server, application server). You can analyze all actions that begin in the browser to track functional problems and user performance issues.

The dashlet is divided into upper and lower areas. The upper part consists of the following tabbed views:

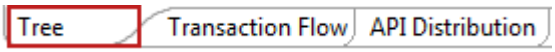
- 

PurePaths: A list of PurePaths, each with corresponding dashboard or dashlet filter (timeframe, application, etc.), and basic information (name, response time, start time) for each PurePath.
- 

Contributors: A list of all contributing actions and methods of the selected PurePath. Colors represent contributors with the highest execution times. The selected contributor is represented in the tree in the lower part of the dashlet.
- 

Errors: All errors in the selected PurePath. Errors are defined via several rules. For more information, see System Profile - Error Detection.

The lower part consists of the following tabbed views:

- 

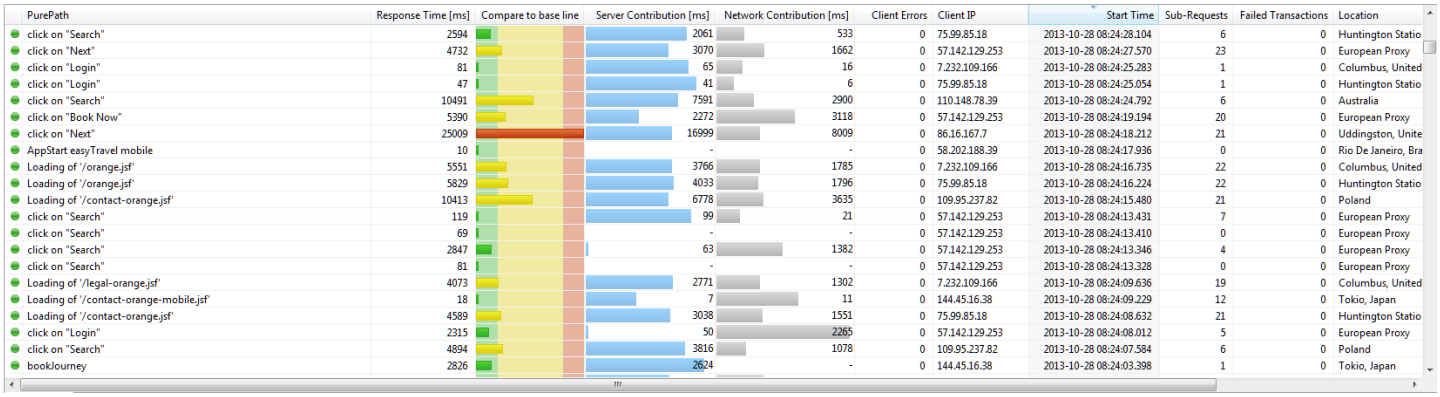
Tree: The sequential representation of all contributors in a tree.
- 

Transaction Flow: The PurePath contributing system components visualized in a transaction flow.
- 

API Distribution: Execution and CPU time distribution of the PurePath's contributing APIs.

PurePaths

The PurePaths View shows a list of all user action PurePaths that match the applied dashboard or dashlet filters such as timeframe or application. Each PurePath receives basic information such as name, start time and client IP.



PurePath	Response Time [ms]	Compare to base line	Server Contribution [ms]	Network Contribution [ms]	Client Errors	Client IP	Start Time	Sub-Requests	Failed Transactions	Location
click on "Search"	2594		2061	533	0	75.99.85.18	2013-10-28 08:24:28.104	6	0	Huntington Statio
click on "Next"	4732		3070	1662	0	57.142.129.253	2013-10-28 08:24:27.570	23	0	European Proxy
click on "Login"	81		65	16	0	7.232.109.166	2013-10-28 08:24:25.283	1	0	Columbus, United
click on "Login"	47		41	6	0	75.99.85.18	2013-10-28 08:24:25.054	1	0	Huntington Statio
click on "Search"	10491		7591	2900	0	110.148.78.39	2013-10-28 08:24:24.792	6	0	Australia
click on "Book Now"	5390		2272	3118	0	57.142.129.253	2013-10-28 08:24:19.194	20	0	European Proxy
click on "Next"	25009		16999	8009	0	86.16.167.7	2013-10-28 08:24:18.212	21	0	Uddingston, Unite
AppStart easyTravel mobile	10		-	-	0	58.202.188.39	2013-10-28 08:24:17.936	0	0	Rio De Janeiro, Bra
Loading of '/orange.jsf'	5551		3766	1785	0	7.232.109.166	2013-10-28 08:24:16.735	22	0	Columbus, United
Loading of '/orange.jsf'	5829		4033	1796	0	75.99.85.18	2013-10-28 08:24:16.224	22	0	Huntington Statio
Loading of '/contact-orange.jsf'	10413		6778	3635	0	109.95.237.82	2013-10-28 08:24:15.480	21	0	Poland
click on "Search"	119		99	21	0	57.142.129.253	2013-10-28 08:24:13.431	7	0	European Proxy
click on "Search"	69		-	-	0	57.142.129.253	2013-10-28 08:24:13.410	0	0	European Proxy
click on "Search"	2847		63	1382	0	57.142.129.253	2013-10-28 08:24:13.346	4	0	European Proxy
click on "Search"	81		-	-	0	57.142.129.253	2013-10-28 08:24:13.328	0	0	European Proxy
Loading of '/legal-orange.jsf'	4073		2771	1302	0	7.232.109.166	2013-10-28 08:24:09.636	19	0	Columbus, United
Loading of '/contact-orange-mobile.jsf'	18		7	11	0	144.45.16.38	2013-10-28 08:24:09.229	12	0	Tokio, Japan
Loading of '/contact-orange.jsf'	4589		3038	1551	0	75.99.85.18	2013-10-28 08:24:08.632	21	0	Huntington Statio
click on "Login"	2315		50	2265	0	57.142.129.253	2013-10-28 08:24:08.012	5	0	European Proxy
click on "Search"	4894		3816	1078	0	109.95.237.82	2013-10-28 08:24:07.584	6	0	Poland
bookJourney	2826		2624	-	0	144.45.16.38	2013-10-28 08:24:03.398	1	0	Tokio, Japan

PurePaths View

- **(Error) State**: Error State of the PurePath (Failed Transaction, Error).
- **(Internal) State**: Current state of the PurePath. Please find detailed information about PurePath states here.
- **PurePath**: String representing the PurePath identifier.
- **Domain**: The requested domain.
- **Duration**: The total response time of the action, in milliseconds.
- **Response Time**: Total duration of PurePaths (in ms).
- **Compare to performance baseline**: Response time compared to performance baseline, configured for the application to which the PurePaths is assigned. See System Profile - User Experience for details.
- **Server Contribution**: Time that is spent on the server to process requests with an immediate impact on the user action response time.
- **Network Contribution**: Estimated time that was spent to transfer resources with an immediate impact on the user action response time.
- **User Expreience**: The user experience index (Satisfied, Tolerating, Frustrated).
- **Client Errors**: Number of JavaScript or mobile errors in the PurePath.
- **Client IP**: IP address of the client that triggers the user action.
- **Start Time**: The time when the PurePath started.
- **Sub-Requests**: Number of web requests caused by the user action.
- **Failed Transaction**: Number of failed transactions in the PurePath.
- **Location**: Geographical location of the client that triggers the user action.
- **Client Family**: Client family of the client that triggers the user action.
- **Tag ID**: Any tag associated with the user action.

- **Application**: Application, where the PurePath is assigned. See System Profile - Applications for application configuration details.
- **Bandwidth**: The connection bandwidth.
- **Action Group**: The name of the action group used to calculate the user experience.
- **Threshold**: The action groups defined threshold, in milliseconds.
- **Performance metric**: The metric defined for the user action that is compared to the threshold.
- **Performance value**: The value of the metric being compared to the threshold, in milliseconds.
- **Factor reason**: The reason why the factor was applied to the user experience thresholds.
- **Threshold factor**: The factor that was applied to the user-experience thresholds, due to external conditions.
- **Visually complete**: The time at which page appears visually complete.
- **Speed index**: The average time when visible parts of the page display.

End-to-end response

The **User Action PurePaths** dashlet calculates how much of the response time is split between the client, server and network time. The algorithm to calculate the three fractions works in the following way:

1. **Server time or server contribution**: The sum of the calls to blocking APIs in server-side PurePaths, considering overlaps. Simply summing up the PurePaths durations (even considering overlaps) creates a wrong impression, since the network bandwidth influences the PurePath: Blocking writes on the server-side prolong the PurePath. A client's (browser's) slow network connection can prolong the PurePath.
2. **Network time or network contribution**: The sum of all response sizes sent to the client / bandwidth of the client.
3. **Client time or client contribution**: The rest of the user action response time is client time.

> **Note**
>
> To calculate the network duration you must enable response size capturing and bandwidth detection. Response size capturing is disabled by default, but you can enable it in the Web server, Servlet or ASP.NET sensors. Bandwidth detection is disabled by default. To enable it click **System Profile > User Experience**. See System Profile for more information.

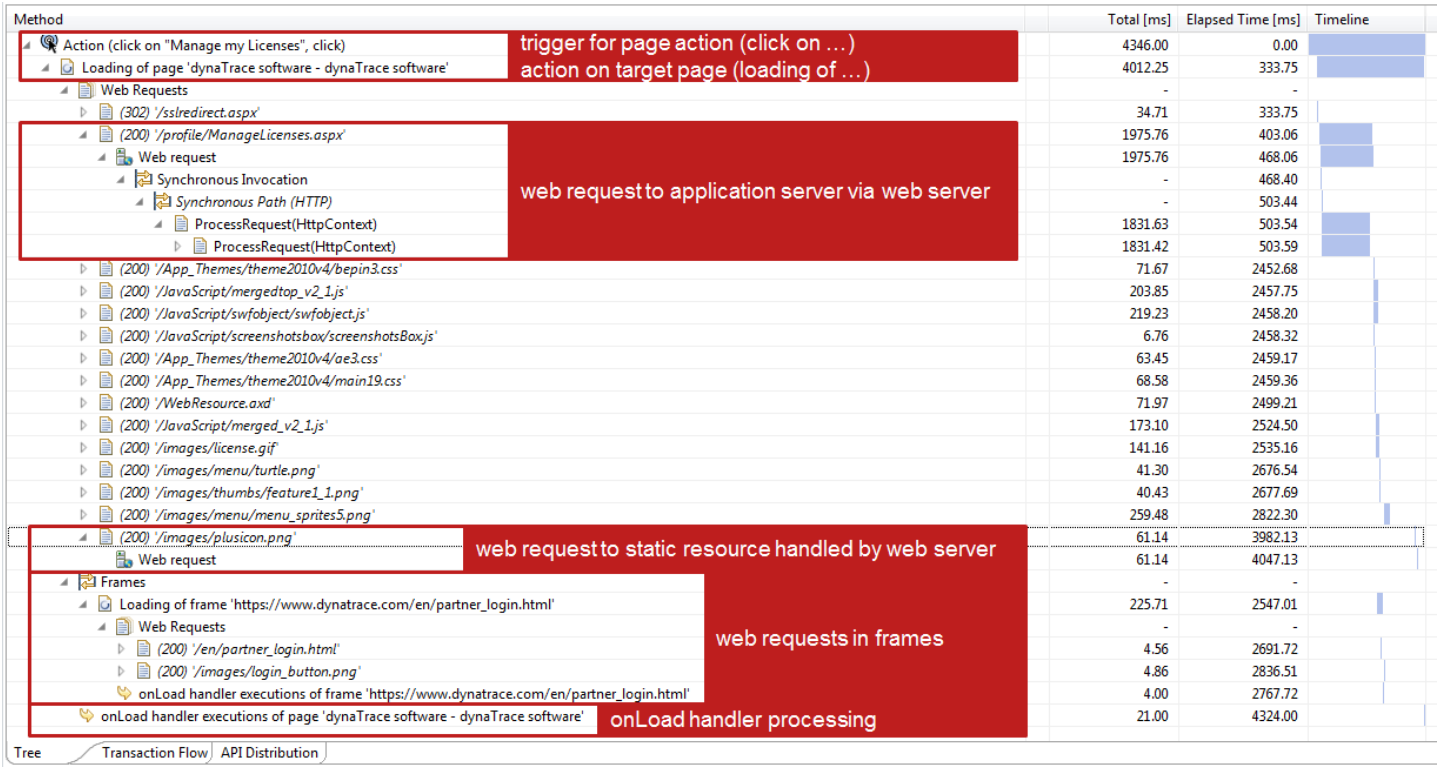
Contributors

See PurePaths dashlet for more information.

Errors

See PurePaths dashlet for more information.

Tree



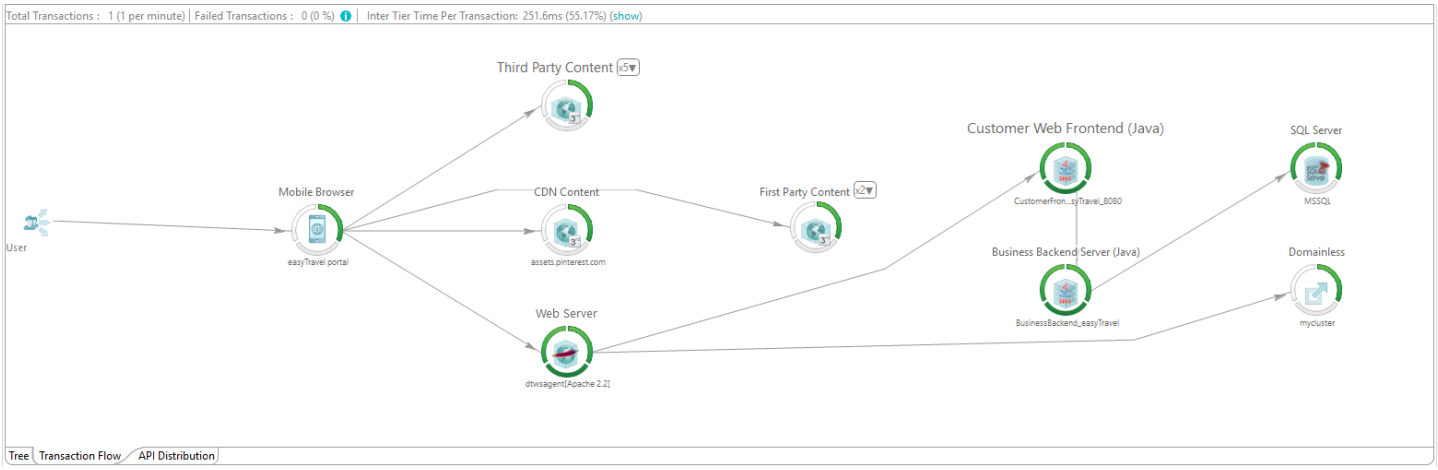
Tree View

The user action PurePath tree is rooted at the trigger of an action such as **click on** or **loading of**, and contains all web requests to instrumented web and application servers. Web requests within frames are grouped in a separate section at the bottom of the tree. The user action PurePath ends with the onLoad handler node, which shows the time spent in activities after the page is loaded.

To analyze single web requests, select a request in the tree and drill down to an applicable dashlet.

Transaction Flow

The Transaction Flow view shows the distribution of PurePaths over system components or tiers, monitored by distinct AppMon Agent instances. Components that are part of a selected PurePath execution are highlighted. PurePaths that span multiple components are represented by dashed lines between the components.



Transaction Flow View

API Distribution

See PurePaths Dashlet for more information.

User Actions dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

User Actions list

Action groupings

Response time charts

Web requests

Custom filtering and splitting

Related pages

User Experience dashlets

Client Errors dashlet

User Action PurePaths dashlet

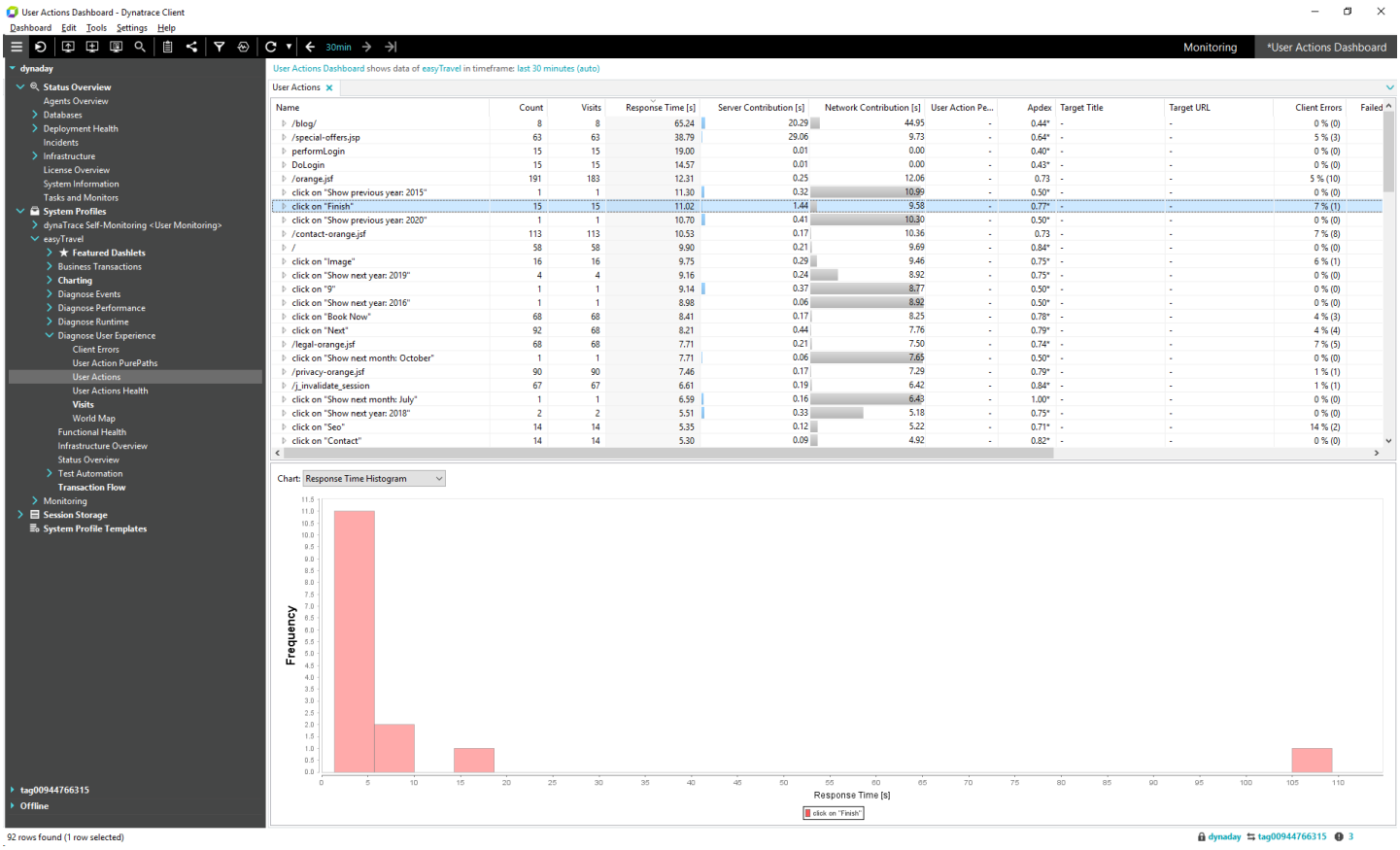
User Actions Health dashlet

Visits dashlet

World Map dashlet

Overview

The **User Actions** dashlet shows information about performance, errors and affected visits. The lower pane displays a chart with response time distribution, or a list of specific user action requests.



The User Actions Dashlet

User Actions list

UEM Metrics include Visits, Duration, End-To-End Response, User Action Performance, Apdex, Client Time, Server Time, and Network Time.

Default columns in the dashlet include:

- **Name**: Action name comprised of a user action and the page URL, or the caption of an element where the action was captured.
- **Count**: Number of occurrences for a user action.
- **Visits**: Number of visits that executed a user action.
- **Duration[s]**: Average duration of user actions. The action duration is calculated from the start time (for example, click, page load) until the triggered request is processed (for example, a new page is loaded) and includes asynchronous time.
- **User Action Performance**: User experience for a user action as calculated by the Apdex range. For more information, see Apdex.

- **Apdex**: Apdex of user actions. An asterisk (*) indicates that the Apdex was calculated from a small number of samples (< 100). See Apdex for more information.
- **Client Errors**: Rate of user actions affected by a client error (JavaScript error, or mobile error).
- **Failed Actions**: Rate of failed actions.
- **Total Duration**: Summarized duration of user actions.
- **Server Contribution [s] or Server Time**: Time spent on server to process requests that have an immediate impact on the user action response time.
- **Client Contribution [s] or Client Time**: Time spent on the client including user action, server time and network time. If you have not turned on bandwidth detection, the client time also contains the network time.
- **Network Contribution [s] or Network Time**: Estimated time spent on transferring resources that have an immediate impact on the user action response time.

Example: `doGet 100ms`

- Async Path 1
  - `doSomethingAsync: 500ms`
- Async Path 2
  - `doSomethingAsync: 550ms`

Response Time: 100ms, because `doGet` finishes after 100ms which means the transaction actually delivers the response back to the end user in 100ms.
Duration: 650ms. Assume that the two asynchronous paths start at the same time. The duration is 650ms, because the PurePath is completed (= last event received) after 650ms.
Execution Total: 1150ms. It takes1150ms to execute the transaction although some of it is executed in parallel.

Action groupings

You can analyze the user actions by different aspects. Use the different structure modes in the dashlet tool bar:

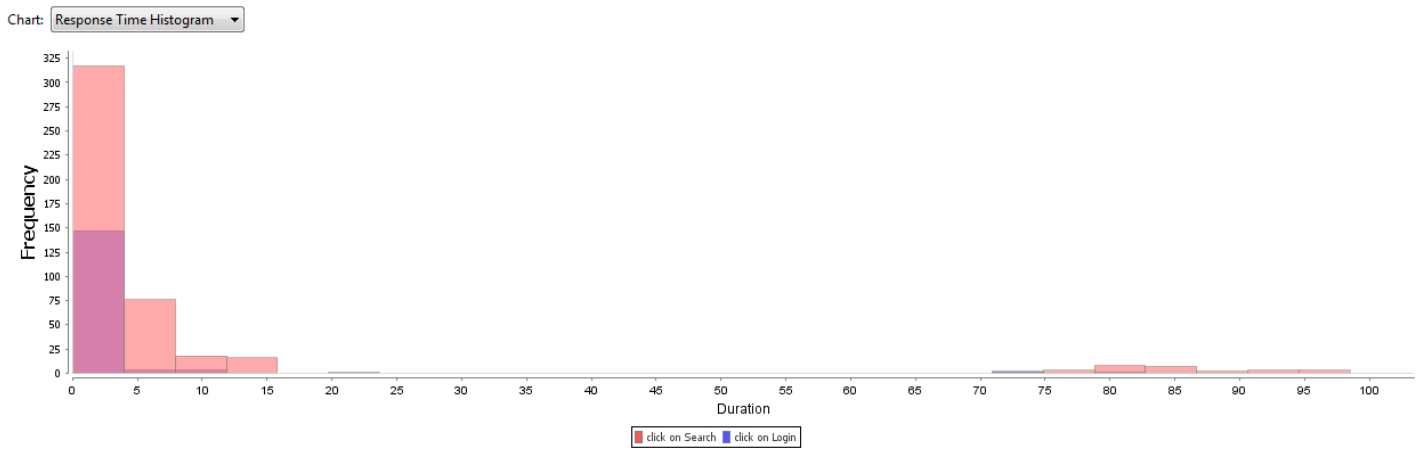
. Aggregate user actions by the following aspects:

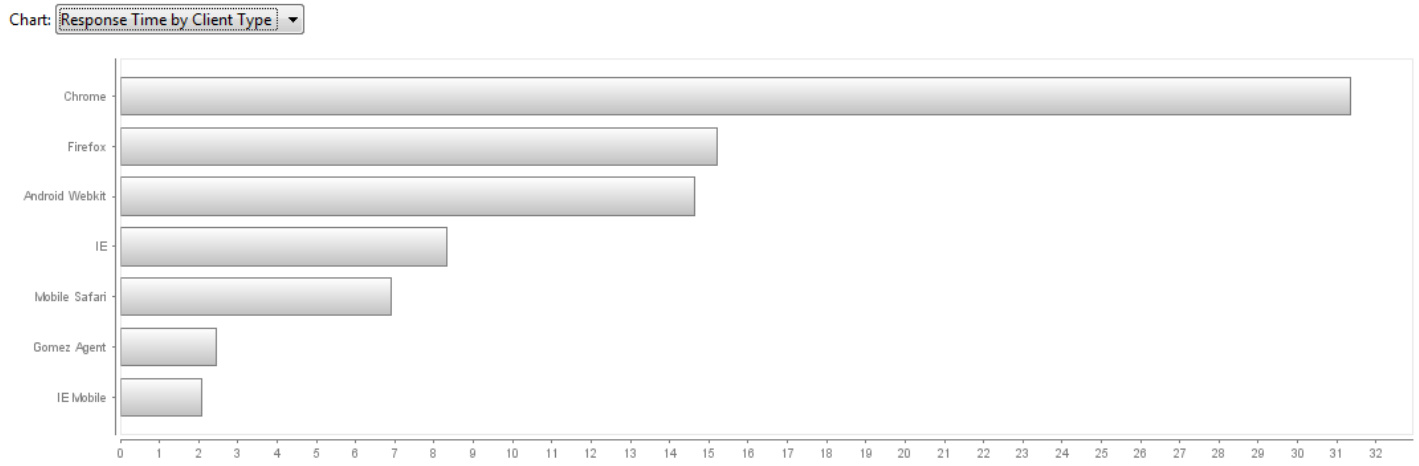
- **User Action**: Aggregates User Actions by their action type (click, key press or load action). Page load actions are grouped further by the requested URI.

- **ClientType, ClientFamily**: Groups all user actions by client type (for example, desktop browser, mobile browser, and mobile app) and client family, with version or device information (for example, IE Mobile-IE Mobile 9.0, or Apple-iPod touch 4G). For example, a Mozilla Firefox 6 user action belongs in the Desktop Browser-Firefox-Firefox 6.0 group.
- **Application**: This is defined in the System Profile.
- **URI (= uniform resource identifier) Title**: The page title is used as grouping criteria.
- **Visit**: Groups all visitor user actions.

Response time charts

The lower pane of the dashlet provides two different response time charts. Response times are displayed as a histogram or are grouped by client family. These charts focus on the selected group of user actions and support multi-selection.





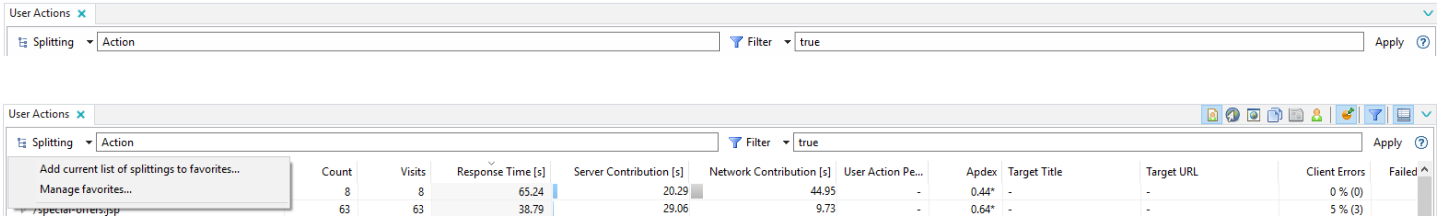
Web requests

If you select a specific user action, the lower pane displays the list of web requests captured for the selected user action, instead of one of the response time charts.

Custom filtering and splitting

To filter and split an expression, click **Custom Query** on the toolbar

. Filter expressions support a Java-like syntax. To split an expression, use a comma-separated properties list. Click **CTRL + SPACE** for a list of available filter and splitting values. You can store favorite filters and splittings for future use.



Supported syntax

Expressions support a Java-like syntax. The following tables describe constants and operators recognized within expressions:

Constants	Sample
Bool	true, false
Number	123, 3.14
String	“text/html”, ‘Firefox’
Regular expressions	/[0-9A-F]*/

Operators	Description
,	, or, OR	logic or
&&, &, and, AND	logic and

!=, <>	not equals
==, =	equals
<, >, <=, >=	relational operators for *smaller*, *greater*, *smaller than or equals to* and *greater than or equals to*
=~, ~=	matches a regular expression (right hand side is a regular expression)
!=~, !~=, <>~	does not match (right hand side is a regular expression
like, LIKE	SQL style string match function using '%' as joker character

Filter Examples

ClientFamily = Firefox and ClientFamilyVersion = %4%: Filters a Firefox browser's user actions with a version string that contains the character 4.
URI = %booking%: Shows only user actions with a URI that contains the phrase *booking* .

Splitting Examples

Country, Region, City: Groups user actions by a geographical location hierarchy that begins with the user's country.
OperatingSystem: Groups user actions by the detected operating system.

User Actions Health dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Selected time frame

Previous time frame

Error HotSpots

Available drilldowns

Related pages

User Experience dashlets

Client Errors dashlet

User Action PurePaths dashlet

User Actions dashlet

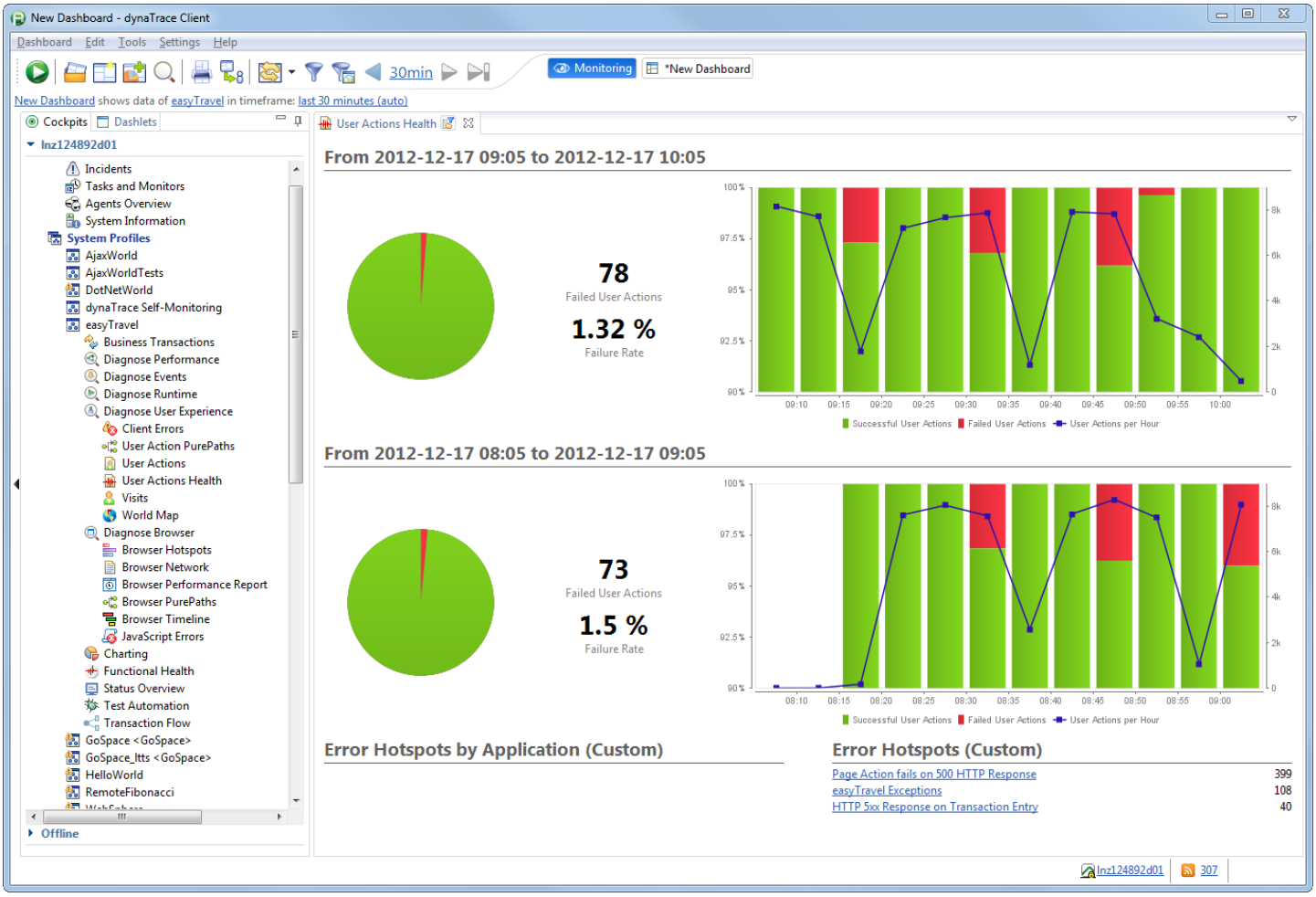
Visits dashlet

World Map dashlet

System Profile - Error Detection

Functional Health dashlet

The **User Actions Health** dashlet shows the current system load and failure rate compared to a previous time period.



Page Actions Health Dashlet

Selected time frame

The upper part of the dashlet shows the overall failure rate of user actions and the total number of failed user actions. The chart at the right displays the system load and failure rate. Click on the chart and drill down to the **Errors** dashlet.

The pie chart at the very left shows the total error rate within the specified time frame.

Previous time frame

The pane for the previous time frame displays equivalent information necessary to compare the current state with a previous one.

Error HotSpots

The bottom part of the dashlet shows the error hotspots by application or detected errors. The numbers refer to the selected dashlet time frame.

Available drilldowns

Hover over the details chart bar. When the cursor changes, click to open the **Errors** dashlet. It opens with a time frame filter applied to the selected period.

Clink a link in the Error Hotspots table to open the **Errors** dashlet. It opens on a selected application or a corresponding error entry.

- See System Profile - Error Detection to learn how to configure error detection rules.
- See description of Functional Health Dashlet dashlet to see the health of your application based on captured transactions captured.

Visits dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Visits list

Visits details

Custom filtering and splitting

Example usage

Related pages

User Experience dashlets

Client Errors dashlet

User Action PurePaths dashlet

User Actions dashlet

User Actions Health dashlet

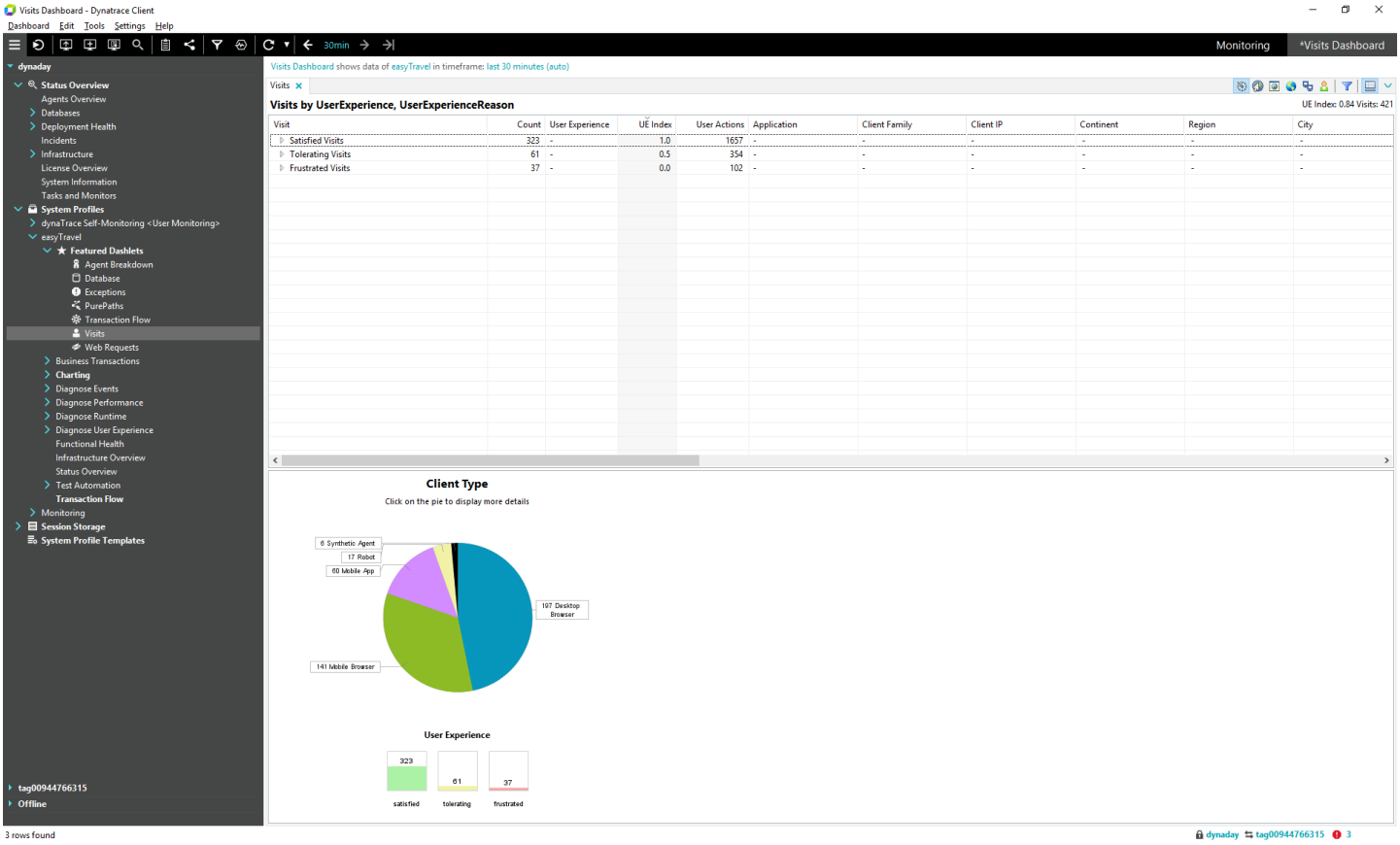
World Map dashlet

System Profile - User Experience

Mobile UEM

Overview

Use the **Visits** dashlet to analyze visits that experience slow response times or errors.



Visits Dashlet

The **Visits** dashlet is divided into two sections: A visits list and a details section that shows the statistics of a group of selected visits or the click path of a specific visit.

Visits list

The following table describes the visit metrics:

Column	Description

User Experience	Shows how users perceive the application performance. Visits fall into three categories: *satisfying*, *tolerating* and *frustrating*. Users have a *frustrating* experience if the following occurs: • The last action failed, or… • The last action was frustrating, or… • More than 50% of all actions are frustrating. Users have a *tolerating* experience if the following occurs: • The last action was not frustrating and did not fail, and… • Less than 50% of all actions are satisfying, or… • More than 50% of all actions are tolerating. Users have a *satisfying* experience if the following occurs: • No action failed, and… • More than 50% of all user actions are satisfying.
Apdex	The Apdex rate for the pages that a visitor viewed. Apdex (Application Performance Index) is an open standard that defines a method for reporting and comparing website performance. The Apdex method converts many measurements into one number on a uniform scale of 0 to 1 (0 = no users satisfied, 1 = all users satisfied). Everything below 0.5 is considered unacceptable and everything above 0.93 is considered excellent
Visit Duration	The time a user is active on the page. To configure the time, click **Edit <System Profile name> > User Experience vertical tab > General > Visits tree > Set Visit ends after a user has been inactive for … minutes**.
User Actions	The number of user actions that are executed, grouped by visitor or client type. Typically, visitors use several user actions, like those listed in Click Path below.
Failed Actions	Shows how many user actions are marked as failed. A user action can fail based on error rules
Client Errors	Indicates if client errors (JavaScript errors, or mobile errors) occur per visitor. The aggregated column per type (region or client) shows a total of client errors and a percentage.
Exit Action Failed	Indicates if the visitor's exit action failed.
Bounce Rate	Shows the percentage of visitors who enter the site and leave (bounce), rather than continue to view other pages.
Visitor Tag	Shows the user ID for this visit. Use the visit search dialog box to search for known users.
Bandwidth	Shows the users network bandwidth. You must enable bandwidth calculation in the System Profile.
Converted	Indicates that a visit meets a goal. To see these goals, click **System Profile > User Experience > Conversion.** This is important in an e-commerce scenario. A visitor becomes a customer when they complete payment services. However, conversion can mean any condition that is met, like account creation, login, or a search.

Converted by	Lists the names of the Measures that converts the visit. The single Measures are separated by a ";"
Streaming	Indicates the types of streaming media that AppMon consumes during a visit. This can be audio, video or audio+video.
Stream Count	Indicates the number of streaming media items that AppMon consumes during a visit. This can be audio+video.

Grouping modes

Access the grouping modes via the toolbar

. The **Visits** dashlet provides various structure modes:

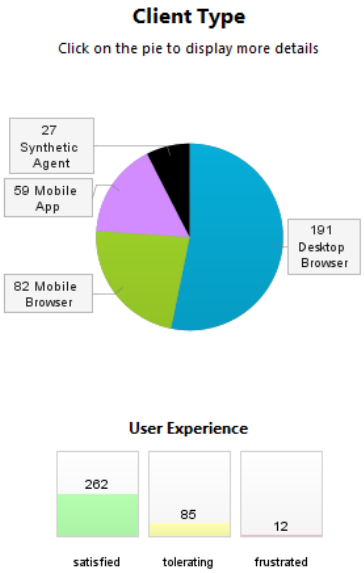
- **User Experience**: Groups visits by the categories satisfying, tolerating or frustrating.
- **ClientType, ClientFamily**: Groups all visits by client type (e.g. desktop browser, mobile browser, mobile app) and client family. It also includes version or device information (e.g.IE Mobile - IE Mobile 9.0, or Apple - iPod touch 4G). For example, a visitor who uses Mozilla Firefox 6 is grouped in Desktop Browser - Firefox - Firefox 6.0.
- **Application**: Groups visits by applications that the user configures in System Profile.
- **Location**: Groups visits by their identified geographic location. The groups are hierarchically divided into continent, country, region and city e.g. North America, United States, California, Los Angeles.
- **OS - Operating System**: Groups by operating system family and version, e.g. OS Family: Windows, Windows 7.
- **No Grouping**: A flat list of all visits.

Visits details

Visits Details shows the client statistics for a group of selected visits or the click path of a specific visit.

Client statistics

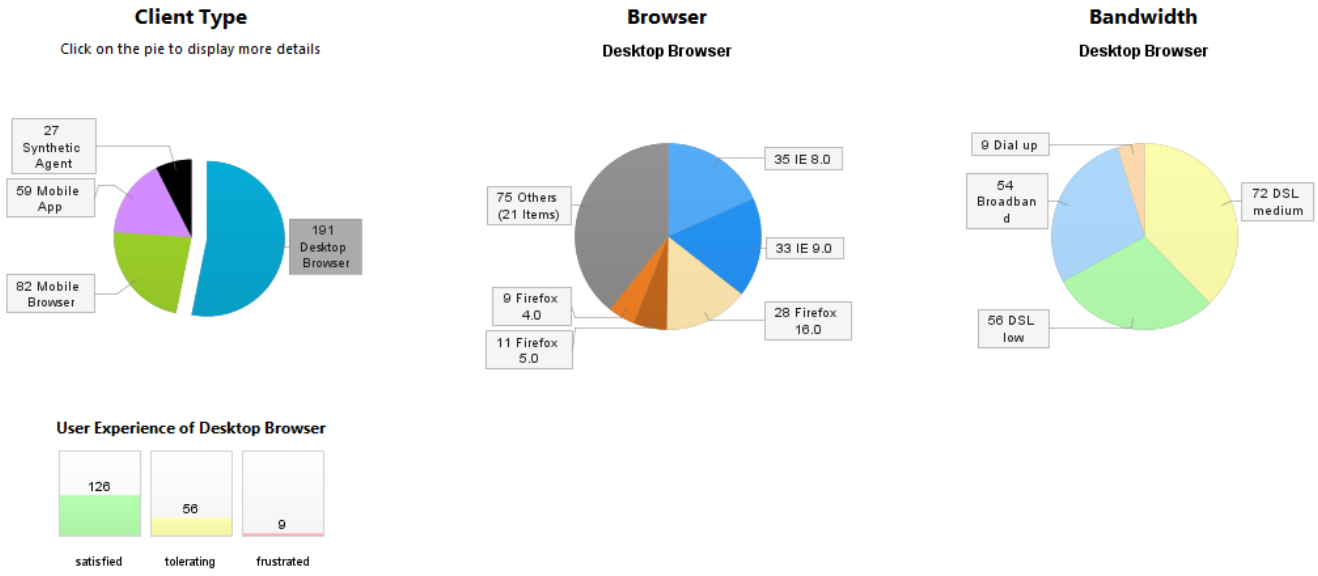
The **Visit** dashlet displays two charts at the left. The pie chart displays the available client type distribution. The meter chart, below, displays the user experience distribution. These charts dynamically update if the visit list selection changes.



Initial Client Statistics

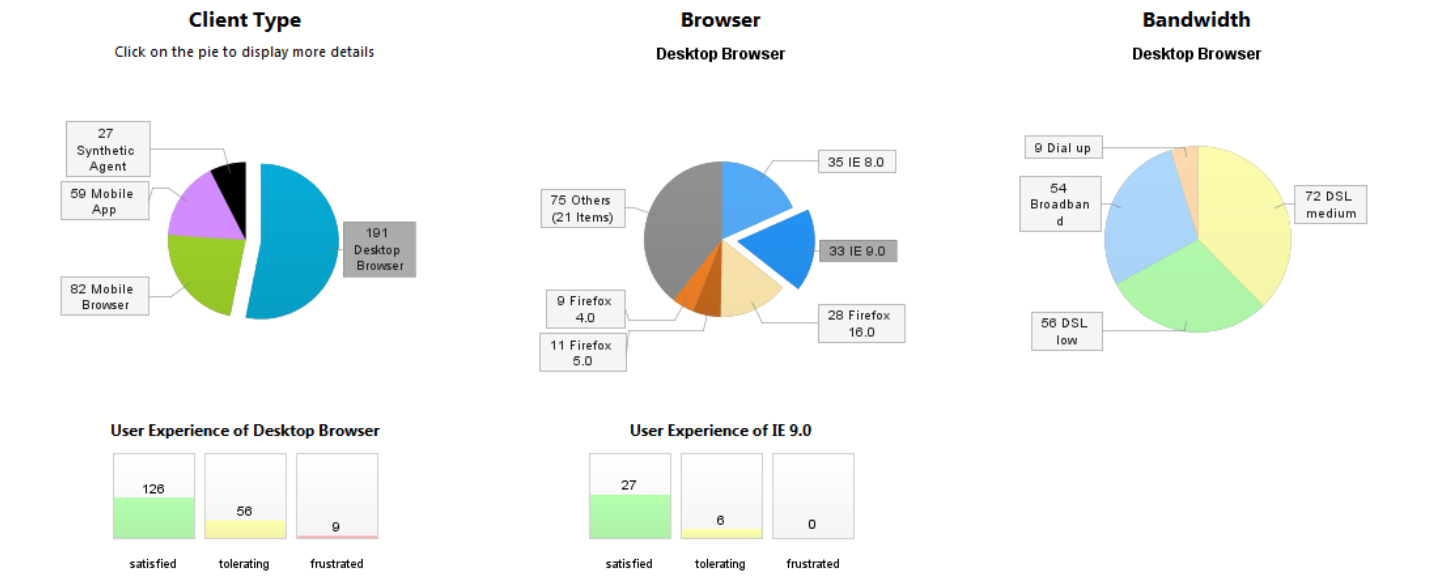
Click on a piece of the pie chart and two more pie charts appear that display the following:

- **Browser or Device**: Shows the distribution of recognized browser versions and devices.
- **Bandwidth**: Shows the measured bandwidth that the user experiences.



Client Statistics With Selection

If you select a piece of these charts, they display that selection's user experience distribution. For example, you can drill down the user experience of the Firefox 16 browser version.



Selection of a Client Type and a Browser

Configure the chart layout in the **Browser Statistics** tab of the dashlet properties.

Click path

The Click Path view shows all of a specific visit's user actions.

User Action	Response Ti...	Server Contribution [ms]	Network Contribution [ms]	User Experience	Start-Time	Size	User Actio...	Client Errors	Failed Actions
Loading of Page /orange.jsf	40.00	26.41	13.59	satisfied	2013-10-28 08:07:36	22	1	0	0
click on "Login" on /orange.jsf	82.60	77.41	5.19	satisfied	2013-10-28 08:07:39	5	2	0	0
click on "Search" on /orange.jsf	44.00	37.19	6.81	satisfied	2013-10-28 08:07:43	6	3	0	0
click on "Book Now" on /oran...	93.61	68.36	25.24	satisfied	2013-10-28 08:07:46	20	4	0	0
click on "Next" on /orange-bo...	371.19	334.93	36.25	satisfied	2013-10-28 08:07:49	23	5	0	0
Loading of Page /legal-orang...	21.49	12.04	9.45	satisfied	2013-10-28 08:07:52	19	6	0	0
Loading of Page /privacy-ora...	52.06	29.31	22.75	satisfied	2013-10-28 08:07:55	20	7	0	0
Loading of Page /j_invalidate_...	45.16	32.27	12.90	satisfied	2013-10-28 08:07:58	23	8	0	0

In addition to the visit's user actions it also shows web requests which belong to the visit, but not to user actions. These so-called "orphaned web requests" are just captured as server-side requests and linked to the visit.

User Action	Response Ti...	Server Contribution [ms]	Network Contribution [ms]	User Experience	Start-Time	Size	User Actio...	Client Errors	Failed Actions
Loading of Page /	1883.00	385.16	12.32	satisfied	2014-02-27 08:28:53	43	1	0	0
click on "About" on /	667.00	82.17	0.13	satisfied	2014-02-27 08:29:19	1	2	0	0
1 Web Request	*18.77*	-	-		*2014-02-27 08:29:19*	-	-	-	-
click on "Home" on /about-or...	516.00	84.63	0.33	satisfied	2014-02-27 08:29:21	6	3	0	0
7 Web Requests	*1236.89*	-	-		*2014-02-27 08:29:21*	-	-	-	-
click on "Contact" on /orange...	1122.00	35.31	0.13	satisfied	2014-02-27 08:29:54	1	4	0	0
1 Web Request	*2.11*	-	-		*2014-02-27 08:29:54*	-	-	-	-
Loading of Page /maps/maps...	470.00	1.29	0.01	satisfied	2014-02-27 08:29:54	1	5	6	0
Loading of Page /maps/maps...	481.00	10.19	0.08	satisfied	2014-02-27 08:29:54	7	6	6	0
click on "Home" on /contact-...	495.00	47.47	0.29	satisfied	2014-02-27 08:29:59	2	7	0	0
20 Web Requests	*1893.94*	-	-		*2014-02-27 08:29:59*	-	-	-	-
click on "Terms of Use" on /or...	851.00	37.19	0.20	satisfied	2014-02-27 08:30:32	1	8	0	0
click on "Home" on /legal-ora...	442.00	33.47	0.29	satisfied	2014-02-27 08:30:33	2	9	0	0
17 Web Requests	*1860.82*	-	-		*2014-02-27 08:30:33*	-	-	-	-

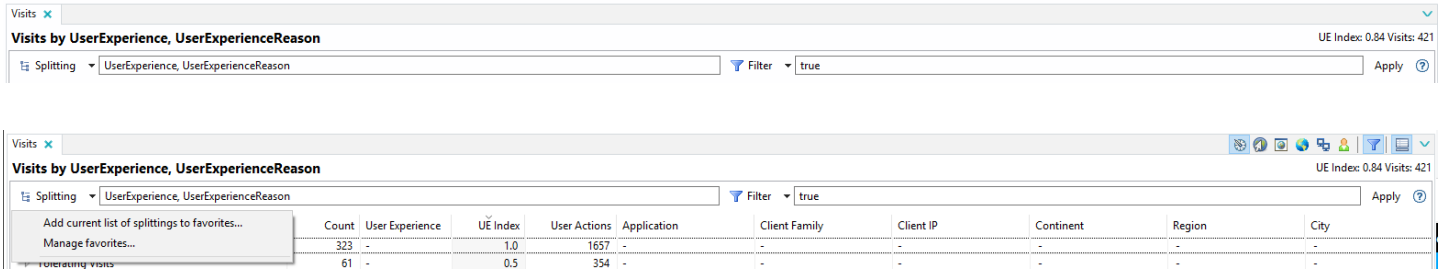
Column	Description
Server Contribution	The time spent on the server to process requests with an immediate impact on the user action response time.
Network Contribution	The estimated time spent to transfer resources with an immediate impact on the user action response time.
Failed Actions	Displays the number of server side transactions that were marked as failed.

Reports

The Visits Details section is also included in most report types sych as HTML and PDF, but it is not included in XML reports.

Custom filtering and splitting

To filter and split an expression, click **Custom Query** on the toolbar. Filter expressions support a Java-like syntax. To split an expression, use a comma-separated properties list. Click **CTRL + SPACE** for a list of available filter and splitting values. You can store favorite filters and splittings for future use.



Supported syntax

Expressions support a Java-like syntax. The following tables describe constants and operators recognized within expressions.

Constants	Sample
Bool	true, false
Number	123, 3.14

String	“text/html”, ‘Firefox’
Regular expressions	/[0-9A-F]*/

Operators	Description
, , or, OR	logic or
&&, &, and, AND	logic and
!=, <>	not equals
==, =	equals
<, >, <=, >=	relational operators for *smaller*, *greater*, *smaller than or equals to* and *greater than or equals to*.
=~, ~=	matches a regular expression (right hand side is a regular expression)
!=~, !~=, <>~	does not match (right hand side is a regular expression
like, LIKE	SQL style string match function using ‘%’ as joker character

Filter Examples**

ClientFamily = Firefox and ClientFamilyVersion = %4%: Filters a Firefox browser’s user actions with a version string that contains the character 4.
URI = %booking%: Shows only user actions with a URI that contains the phrase *booking* .

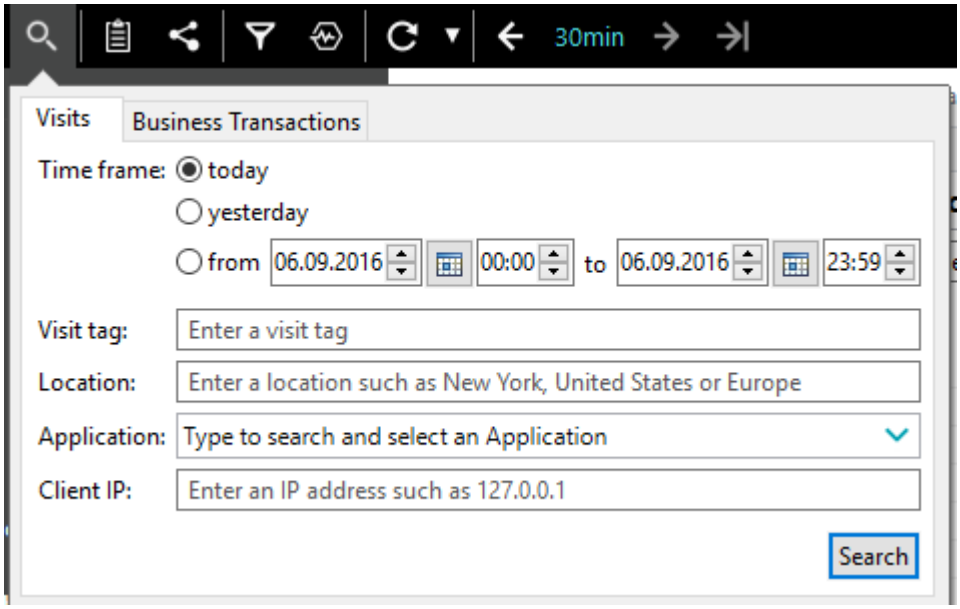
Splitting Examples**

Country, Region, City: Groups user actions by a geographical location hierarchy that begins with the user’s country.
OperatingSystem: Groups user actions by the detected operating system.

Example usage

Find a specific user

Use AppMon to identify visitors and get information about the related pages they visit. Choose **Edit > Search** in the menu or click **User Experience > Customer Complaint Resolution > Search** in the Start Center. Use the search dialog box options to search for a specific user. The search result generates the **Visits** dashlet in a flat structure. Use this to analyze the user's click path. Look at the actions that contain errors or slow response times.



The Search dialog box for Visits

See System Profile - User Experience to configure Measures to capture the user ID and detect a visit conversion. See Mobile Agent to add mobile applications to your visits.

Other dashlets

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Session Browser dashlet

Test Overview dashlet

Test Results dashlet

Text and image dashlet

Text and measurement dashlet

Web Page dashlet

There are several other dashlets available for your use:

- Session browser: Displays stored sessions in its data source.
- Test overview: Displays general status of tests in your environment.
- Test results: Visualizes results of various tests in your environment (Test Center edition only).
- Text and image: Displays custom text and images to add notes to a dashboard.
- Text and measurement: Displays the value of a measure, along with the custom text.
- Web page: Displays a web page as the part of your dashboard.

Session Browser dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Stored session attributes

Context menu options

Grouping options

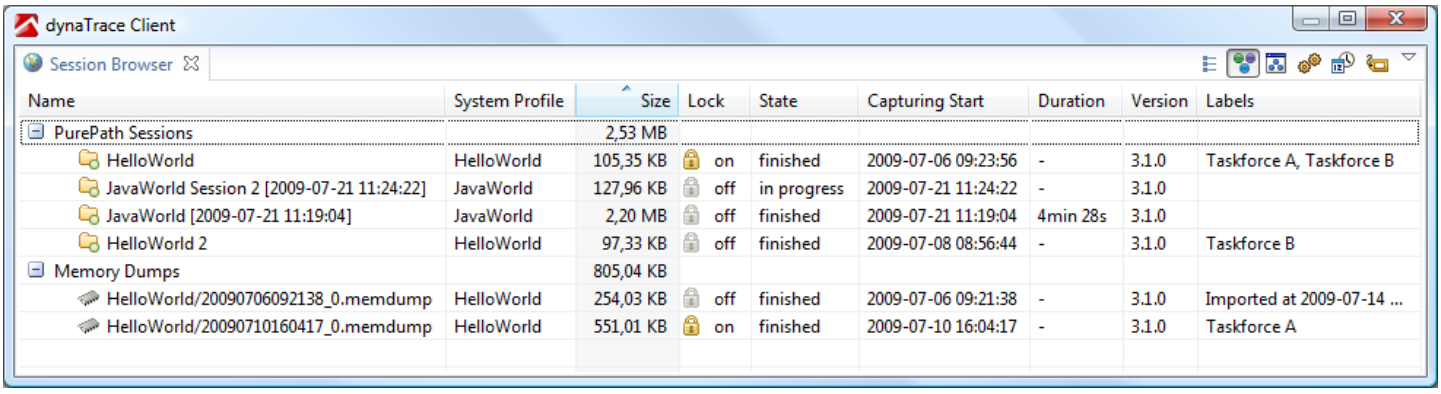
Related pages

Using sessions

Overview

The **Session Browser** dashlet displays information about stored sessions available in its data source.

A stored session can be a PurePath session, a memory dump, a thread dump, or a sampling that is stored on an AppMon Server, or offline on the client. See Usage of Sessions for more information.



Name	System Profile	Size	Lock	State	Capturing Start	Duration	Version	Labels
PurePath Sessions		2,53 MB						
HelloWorld	HelloWorld	105,35 KB	on	finished	2009-07-06 09:23:56	-	3.1.0	Taskforce A, Taskforce B
JavaWorld Session 2 [2009-07-21 11:24:22]	JavaWorld	127,96 KB	off	in progress	2009-07-21 11:24:22	-	3.1.0	
JavaWorld [2009-07-21 11:19:04]	JavaWorld	2,20 MB	off	finished	2009-07-21 11:19:04	4min 28s	3.1.0	
HelloWorld 2	HelloWorld	97,33 KB	off	finished	2009-07-08 08:56:44	-	3.1.0	Taskforce B
Memory Dumps		805,04 KB						
HelloWorld/20090706092138_0.memdump	HelloWorld	254,03 KB	off	finished	2009-07-06 09:21:38	-	3.1.0	Imported at 2009-07-14 ...
HelloWorld/20090710160417_0.memdump	HelloWorld	551,01 KB	on	finished	2009-07-10 16:04:17	-	3.1.0	Taskforce A

Stored session attributes

A stored session has the following attributes:

- **Name**: Name of the stored session.
- **Description**: Optional description of the stored session.
- **Type**: Possible values are `PurePath Session`, `Total Memory (Simple)`, `Total Memory (Extended)`, `Selective Memory Dump`, `Thread Dump`, and `Sampling`.
- **System Profile**: System Profile from which the stored session was created.
- **Size**: Size of the stored session in Bytes, KB, MB or GB.
- **Lock**: Switch to **on** to protect the stored session from the automatic deletion procedure.
- **Directory**: Directory in which the session is stored.
- **State**: Possible values are `finished`, `incomplete`, `in progress` and `corrupt`.
- **Capturing Start**: Start date and time of the stored session capture.
- **Duration**: Stored session capture duration.
- **Version**: File format version in which the session is stored.
- **Labels**: Labels that are assigned to the stored session. Use label assignments to group multiple stored sessions.
- **Recording Type**: Possible values are `manual` and `incident`.

Context menu options

- **Open**
  
  : Opens the selected stored sessions in separate dashboards.
- **Make Available Offline**: Downloads a stored session from the AppMon Server to an AppMon Client. See Usage of Sessions for more information.
- **Convert to latest version**: Converts the selected session files to match the server version. The menu item is only available for sessions with an old version.
- **Export**
  
  : Opens the **Export Session** dialog box and exports the stored session to a user-defined session file.
- **Import**
  
  : Opens the **Import Session** dialog box in order to import a stored session from a user-defined session file.
- **Lock**
  

- **Unlock**
  
  : Toggles the deletion lock to enable or disable protection against automatic deletion procedure.
- **Edit**: Opens a dialog box to modify stored session properties like name, description, deletion lock, or the assigned Labels.
- **Drill Down**: Opens a view associated with a selected stored session.
- **Copy**
  
  : Copies the attributes of the selected stored sessions to the clipboard.
- **Delete**
  
  : Deletes the selected stored sessions from the dashlet's data source.
- **Details**
  
  Shows the selected stored session attributes.

Grouping options

Use the options at the top right dashlet corner to change the structure of the displayed stored sessions.



- **Flat**
  
  : Displays a simple list of all stored sessions and doesn't group them.
- **Group by Type**
  
  : Groups the stored session by type such as `PurePath Sessions`, `Memory Dumps`, `Thread Dumps`, and `Samplings`.
- **Group by System Profile**
  
  : Groups the stored sessions by the System Profile from which they are created.
- **Group by State**
  
  : Groups the stored sessions by their state. Possible groups are `finished`, `incomplete`, `in progress`, and `corrupt`.
- **Group by Capturing Start**
  
  : Groups the stored sessions by the capture start date.

- **Group by Label**
  
  : Groups all stored sessions by label. If a stored session has multiple labels, it can be a member of multiple groups.

Test Overview dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

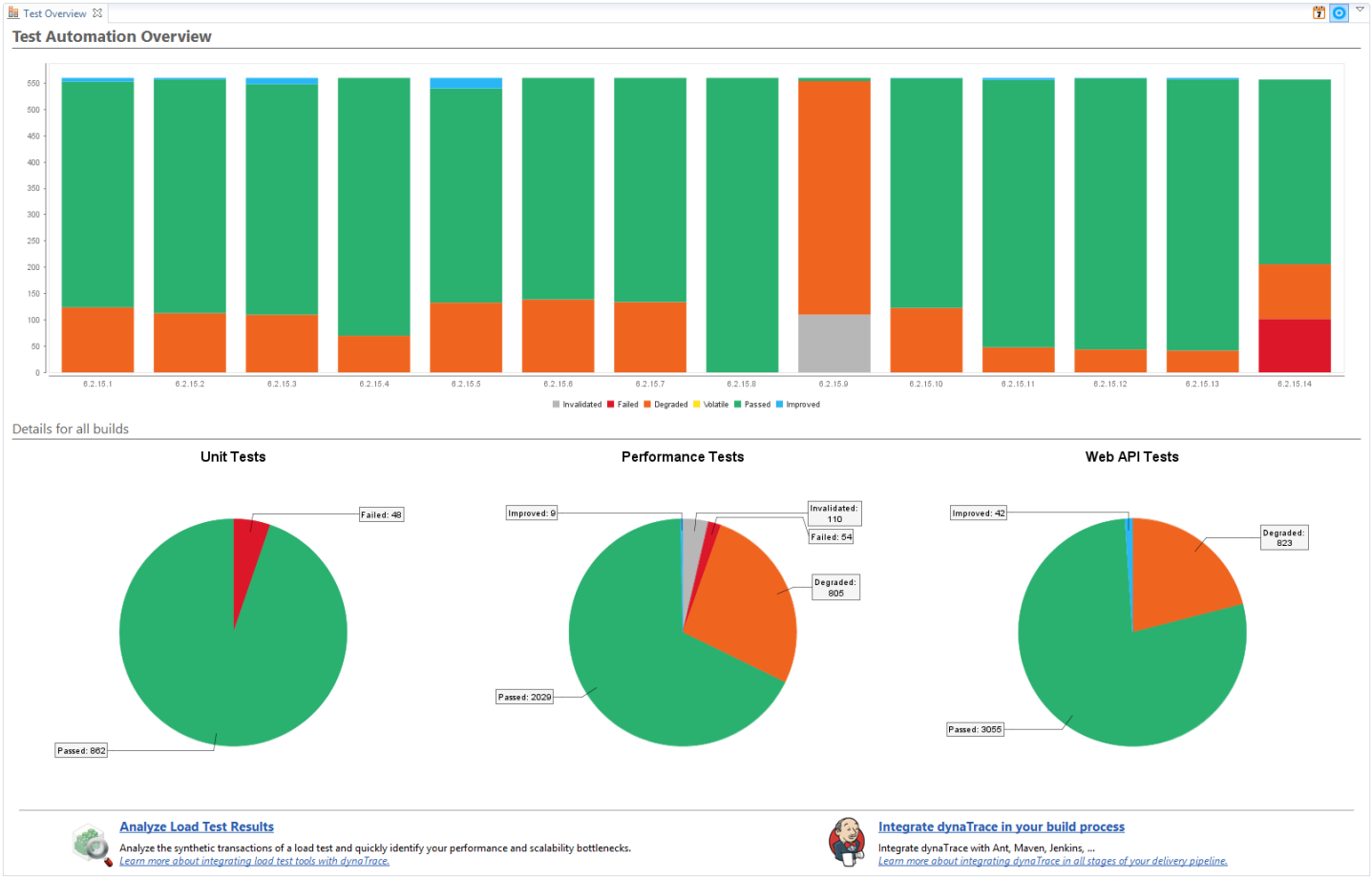
On this page

Related pages

Test automation explained

Test Results dashlet

The **Test Overview** dashlet displays the general status of tests in your environment.



Note

This dashlet is only available in the Pre-Production (formerly Test Center) Edition.

Bar chart section

In the upper section of the dashlet, a bar chart displays information about the overall number of test executions with each status. The X-axis contains the aggregation specific keys and the Y-axis shows the number of matching test executions. You can choose the aggregation type in the dashlet header.

Aggregation types

The following aggregation types are available:

- **By build**: Test executions with the same build version parameters (major.minor.revision.build) are aggregated.
- **By day**: Test executions done in the same day are aggregated.

Pie charts section

Below the **Bar Chart** section, pie charts show the number of test executions with each status, as well as the specified test category such as Unit Tests, Performance Tests, Web API Tests, Browser Tests. There is one pie chart per each test category. Only pie charts with a test category that have at least one test execution are shown. As a result, one to four pie charts can be visible in this section.

By default, the pie charts filter test executions by matching test category and show a consolidated view of the details for all your builds for the selected timeframe.

Selecting any bar in the **Bar Chart** section by clicking it once also filters the data shown in **Pie Charts** section by aggregation specific key, as shown in the bar's X-axis. Clear the selection by clicking the empty surface of the chart above the bar or between the bars. The current selection has a thin border around the bar and the header of the **Pie Charts** section.

Drilldowns to Test Results dashlet

You can drill down from the Test Results Dashlet dashlet to see more detailed, filtered data by doing one of the following:

- **Double click** on any bar in the **Bar Chart** section. This filters data by an aggregation specific key, as shown in the bar's X-axis.
- **Single click** on the chart or chart title in **Pie Charts** section. This filters the data by a related test category to the chart and the aggregation specific key, as shown in the bar's X-axis if any bar is selected. You can also check it on the header of **Pie Charts** section.

A **Test Results** dashlet opened by drilldown inherits all filters applied to the source **Test Overview** dashlet. For example, the selected timeframe.

Footer section

Load tests results

Load tests can be analyzed using a separate Load Test Overview dashboard. You can open it by clicking the **Analyze Load Test Results** link in **Test Overview** dashlet footer. See Load Tests for more information.

The **Footer** section also includes useful documentation links about tests.

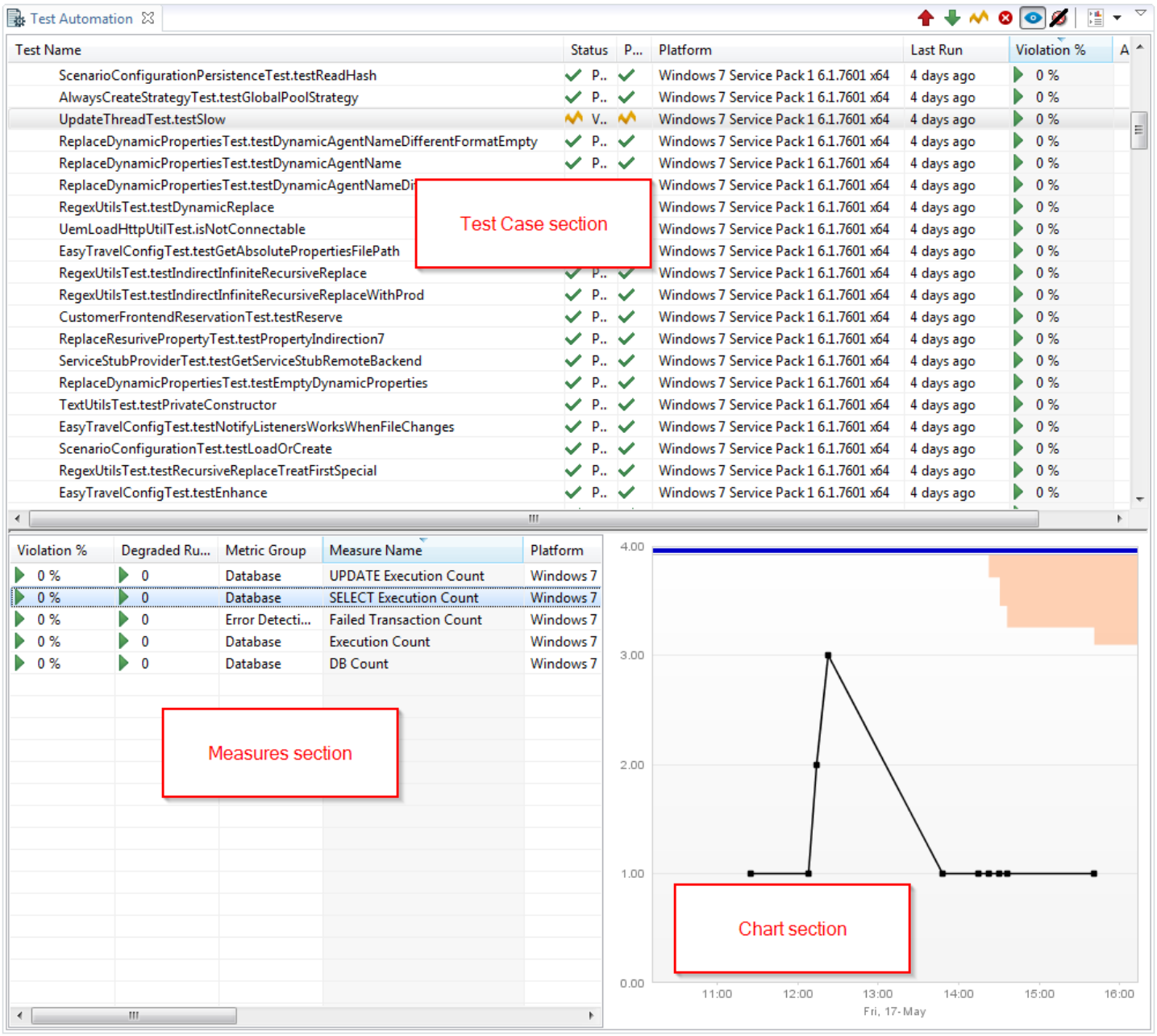
Test Results dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Test automation explained

The **Test Results** dashlet visualizes the results of various tests in your environment.



This dashlet is only available in the AppMon Test Center edition.

Test case section

The upper section of the dashlet lists the test cases, organized by defined test category and package. Each test case entry shows an aggregated view of the KPIs.

Test status

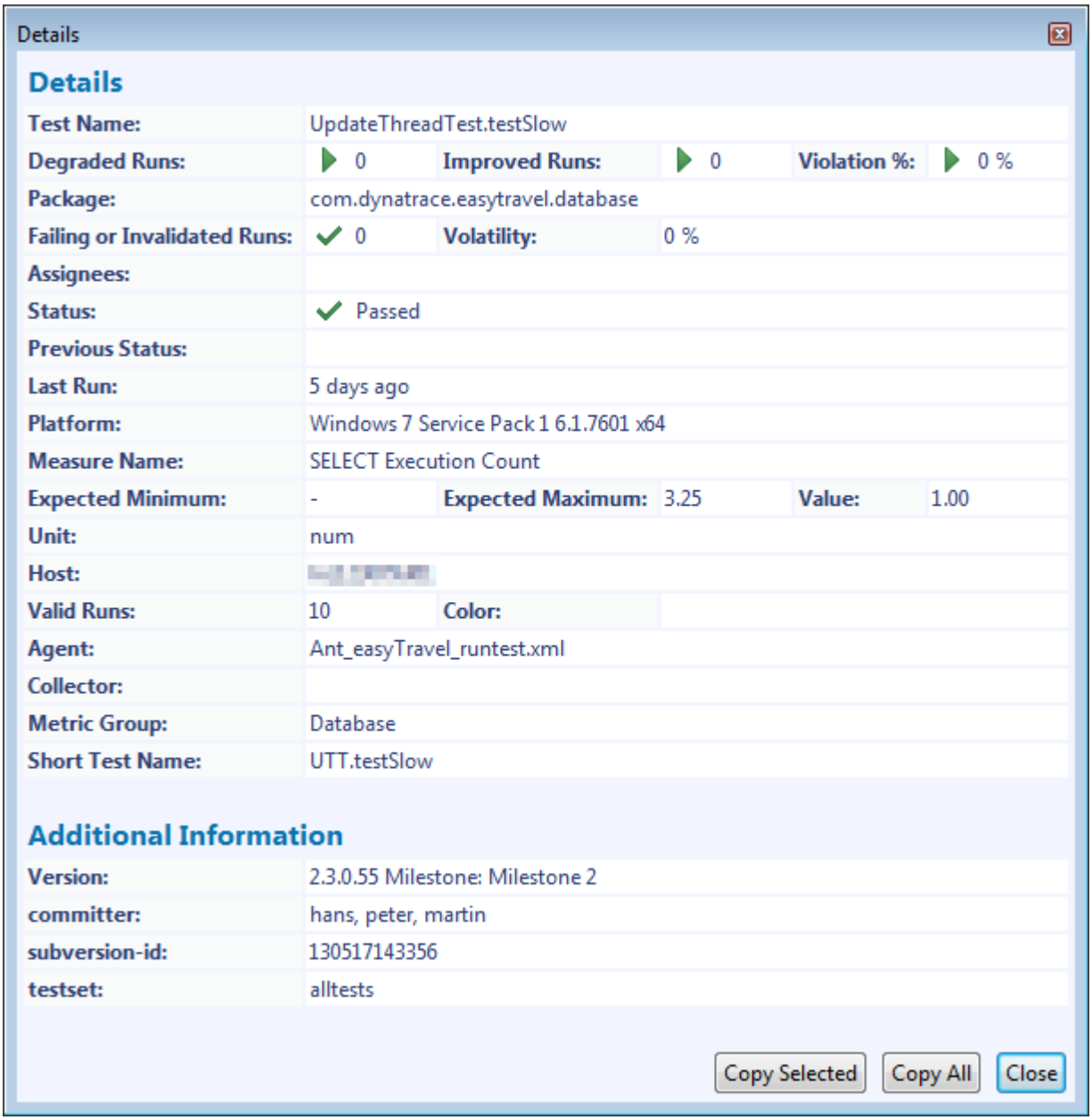
Every test case has a status:

- 
  OK: Test case executed correctly.
- 
  Degrading: Test case runs have become slower.
- 
  Improving: Test case runs have become faster.
- 
  Volatile: Test case has a very volatile outcome — sometimes faster, sometimes slower — so changes in performance might not be recognized correctly. This especially happens with very short-running tests. Try to increase the duration of the test (e.g. by executing the same operation multiple times).
- 
  Failing: Test case has a functional problem, so no performance data could be recorded.
- 
  Invalidated: The last test run of this test case was manually invalidated by the user.
  When you click a test status icon in the toolbar in the upper right corner of the **Test Results** dashlet, related test status columns display in the measure section. For example, click the **Degraded** icon to include the **Degraded Runs** column or the **Volatile** icon to include the **Volatility** column.

Grouping by package can be toggled by clicking the **Group by packages** icon.

Viewing details

Right-click the item and select **Details** to view details for a test case, measure, or test run. You can copy information from the **Details** dialog box.



Measures section

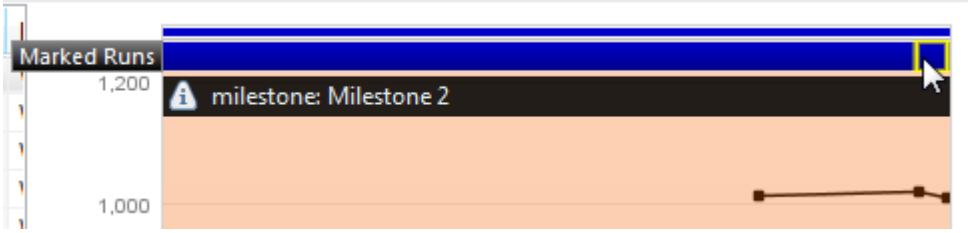
The measure section in the lower left of the dashlet shows the latest values of the KPIs associated with a test case, and indicates whether those values are within, above, or below the corridor.

Chart section

The chart section shows the historical values of a KPI for a test case, including the calculated performance corridor.

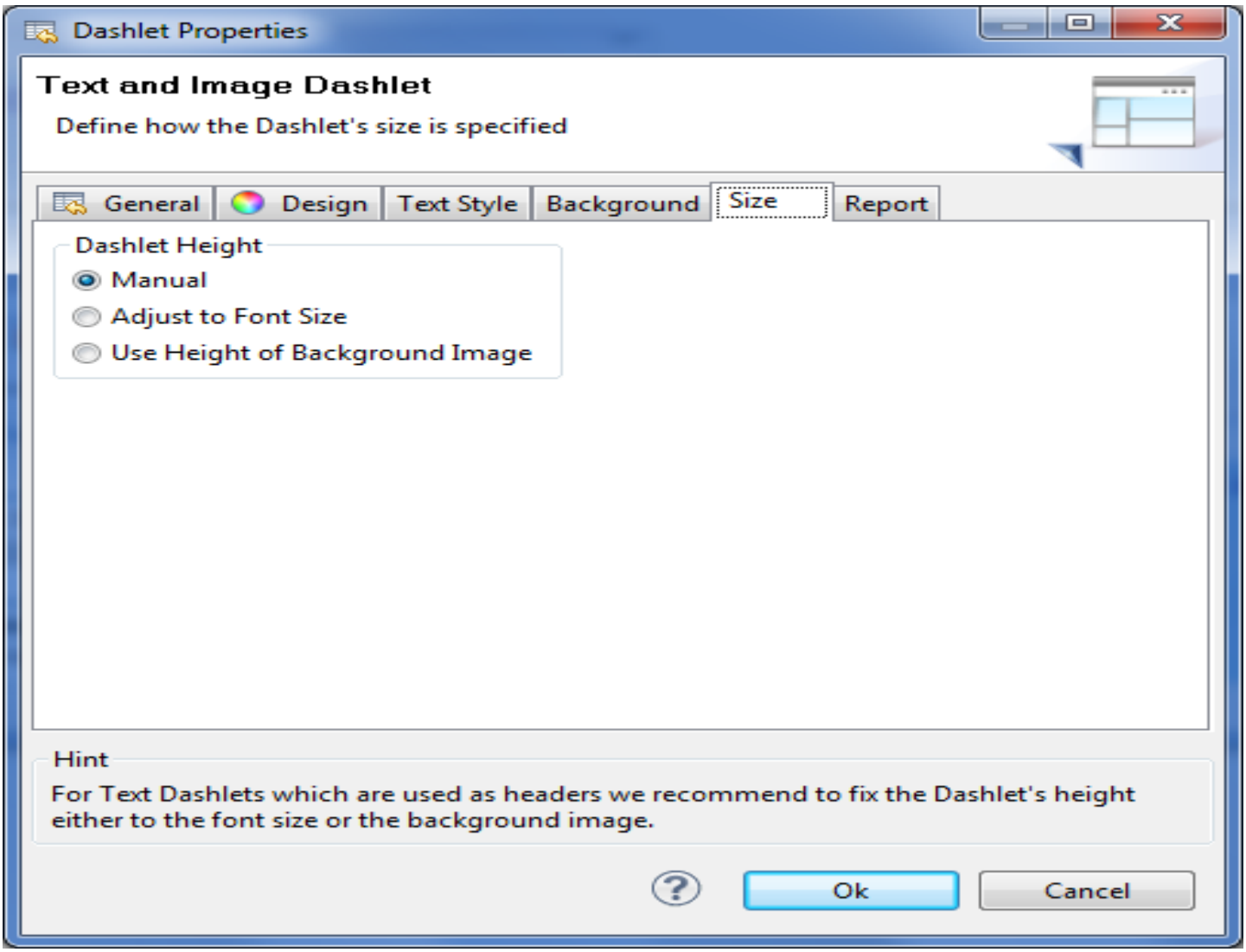
Viewing markers

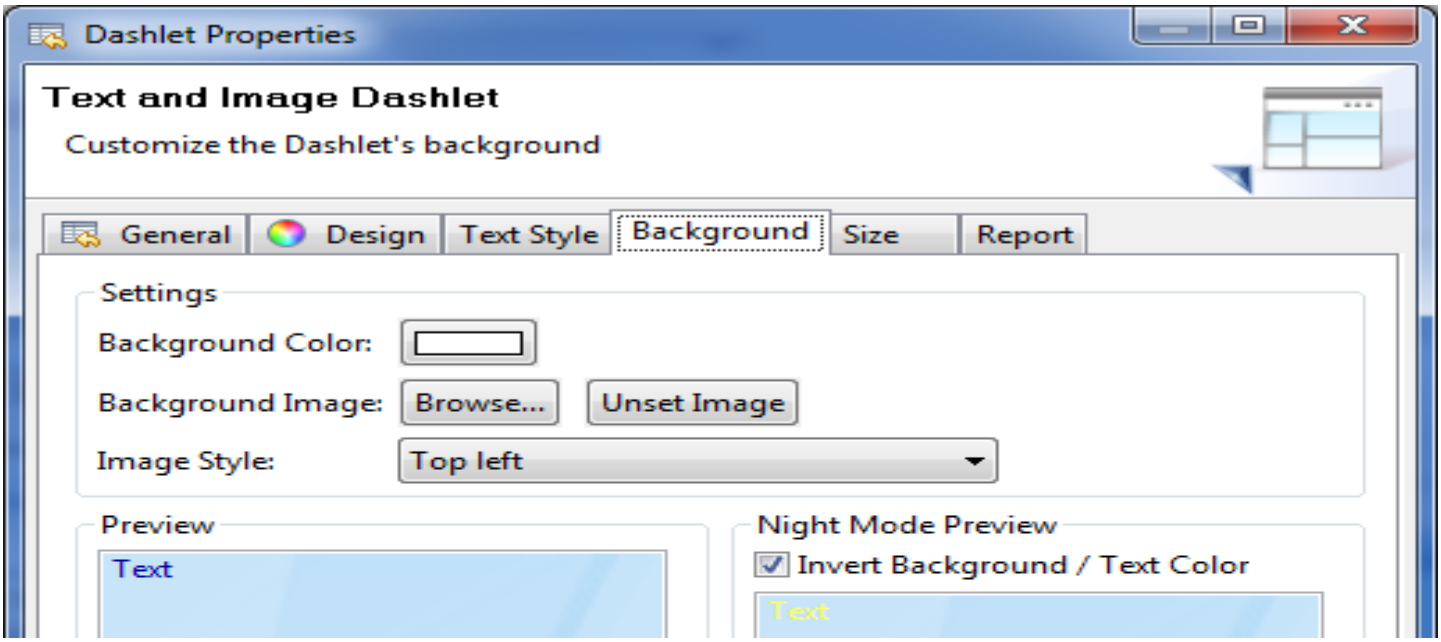
Any marker set in the test metadata displays in the chart's heat field.

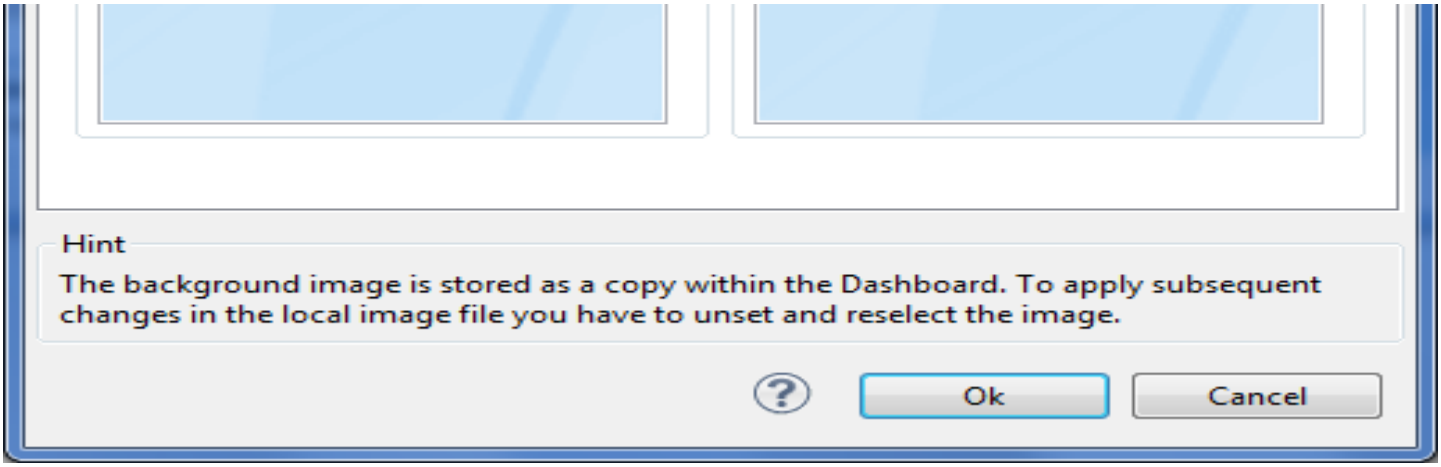


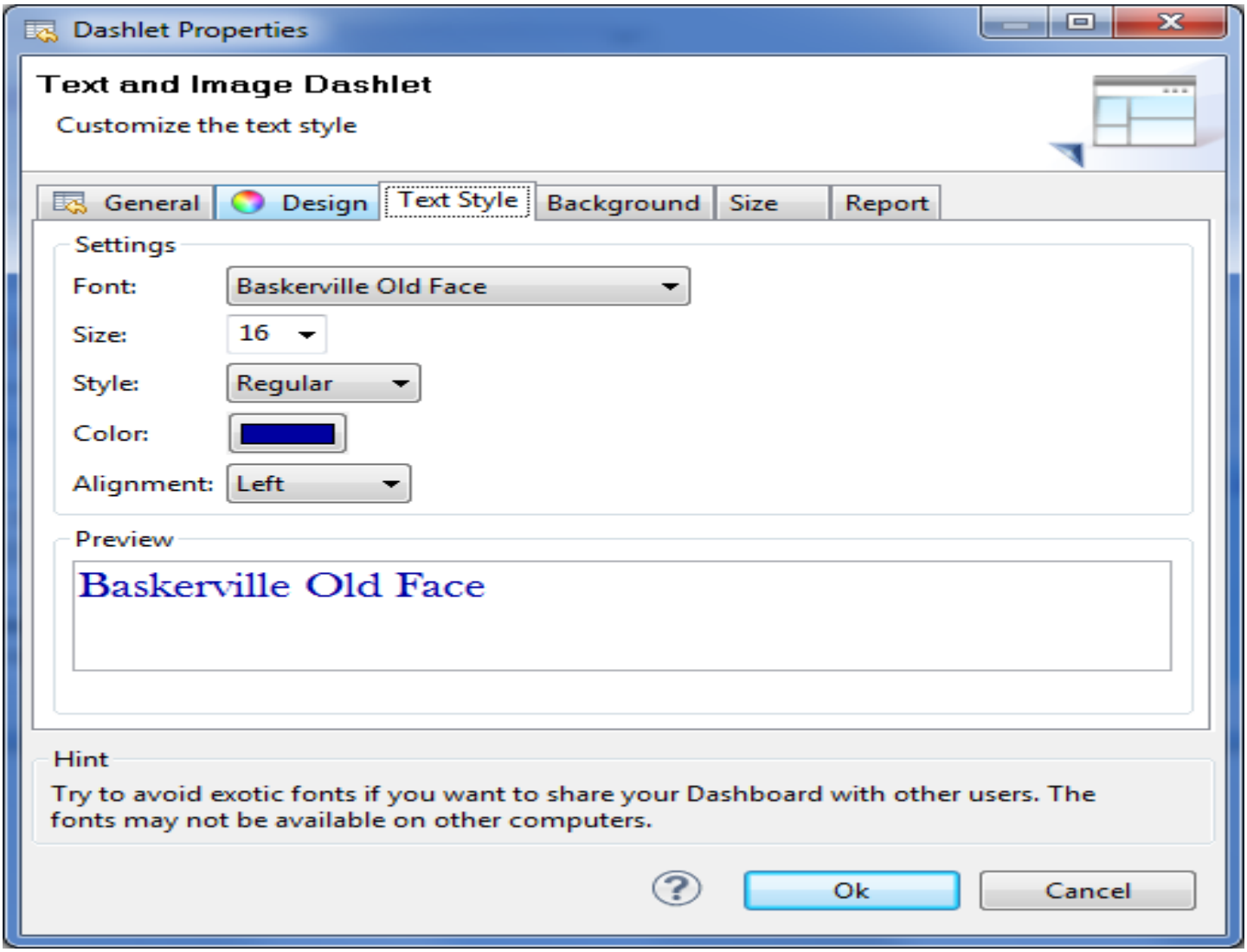
See the Test Automation FAQ for more information.

Text and image dashlet

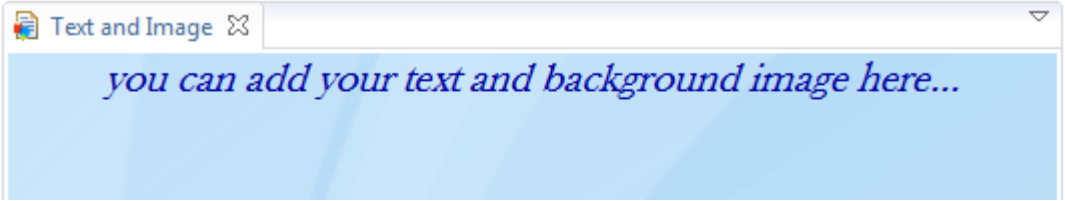








Use the customization options to change text and background images.



Use the **Text and Image** dashlet to add notes to a dashboard.

Text and measurement dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Web tile

Tile specific filters

AppMon Client dashlet

Use the **Text and Measurement** dashlet/tile to show the value of a measure, along with the custom text.

The web **Text and Measures** tile supports plain text, measure placeholders, Markdown, and custom tags. You can use up to 40 measures here. In case of splitting each part counts as one measure.
The AppMon Client **Text and Measurement** dashlet supports plain text and measure placeholders. You can use only one measure here.

Web tile

To add a measure:

1. In the **Tile configurator** click **Add measure**.
2. Select the required measure and click **Next**. You can type the name of you measure to filter the list.
3. Configure aggregation and splitting options as you need and click **Done**.
   The added measure and its placeholder are displayed in the list.

Web tile supports Markdown, and custom tags for color and style. Type the code right in the input area, the **Preview** panel will display it live. The size of the preview is limited, and exceeding content will not be shown here, you have to save the dashlet to see in the actual dashboard.

To enable Markdown and custom tags:

1. In the AppMon Client, select **Settings menu > Dynatrace Server > Services item > Dynatrace Web tab**.
2. Check the **Allow custom HTML** checkbox.

Placeholders

The following placeholders are supported in web **Text and Measurement** tile:

Measurement: Inserts the value of the measure. The syntax is `{1}`, `{2}`, etc. for un-splitted measures, and `{2.1}`, `{2.2}`, etc. for splitted ones. You can find placeholder for the particular measure in the list of measures.
Server base URL: Inserts the basic URL of the server. The syntax is `{server-url}`.
Server protocol: Inserts protocol, used by the server (http or https). The syntax is `{server-protocol}`.
Server host name: Inserts the host name of the server. The syntax is `{server-name}`.
Server port: Inserts the port of the server. The syntax is `{server-port}`.
Dashboard timeframe: Inserts the timeframe of the dashboard, in hours. The syntax is `{dashboard-timeframe}`.
Tile timeframe: Inserts the timeframe of the dashlet, in hours. The syntax is `{tile-timeframe}`.

Format text

You can use Markdown to format your text, or add additional elements, like tables, links, images. You can find the Markdown documentation in the Markdown Cheatsheet, or at the Commonweb.org website.

The following Markdown features are supported:
Headings: You can use heading levels from 1 to 6. Markdown and placeholders work in headings.
Emphasis: You can add italic and bold using asterisks, underscores, and combination of both.
Lists: You can create numbered, or bulleted lists, and also create sub-list items. The **List** button of the **Tile configurator** creates a bulleted list.
Links: You can use absolute URLs, or anchors to Dynatrace Web, for example `#dashboards`. You can find the anchors in the address string of your browser. **Images**: Images are inserted inline, so standalone picture requires its own line.
Code highlighting: You can highlight one or several words inline, or create a code block. Markdown is ignored inside the highlight, but placeholders and custom tags work.
Tables: Markdown, measure placeholders, and custom tags work inside cells.
Blockquotes: Markdown, measure placeholders, and custom tags work inside quotes.
Horizontal rules
Line breaks

Custom tags

You can use custom tags to change the color and style of your text. To add syntax, just select the text, and click the required button of the editor.

Color tag. The `{color}` custom tag adds color to the font. The syntax is `{color:<color_definition>}Your text{color}`. You can define color by name or HEX code, such as #6f2da8.
Style tag. The `{style}` custom tag allows you to apply CSS styles to your text. The syntax is `{style:<CSS property>:<value>}Your text{style}`. You can find list of CSS parameters and their values in the CSS tutorial.

Tile specific filters

In web tiles you can set a particular application as a data source, and the custom timeframe. These are independent from the dashboard data source and timeframe settings, so changing the dashboard data source or timeframe won't affect tile data source and timeframe.

Custom data source

To set a particular application as the tile data source:

1. In the **Tile configurator** click **Advanced settings**.
2. In the **Application** list, select the required application.
3. Set the required timeframe.
4. Click **Done**.

Custom timeframe

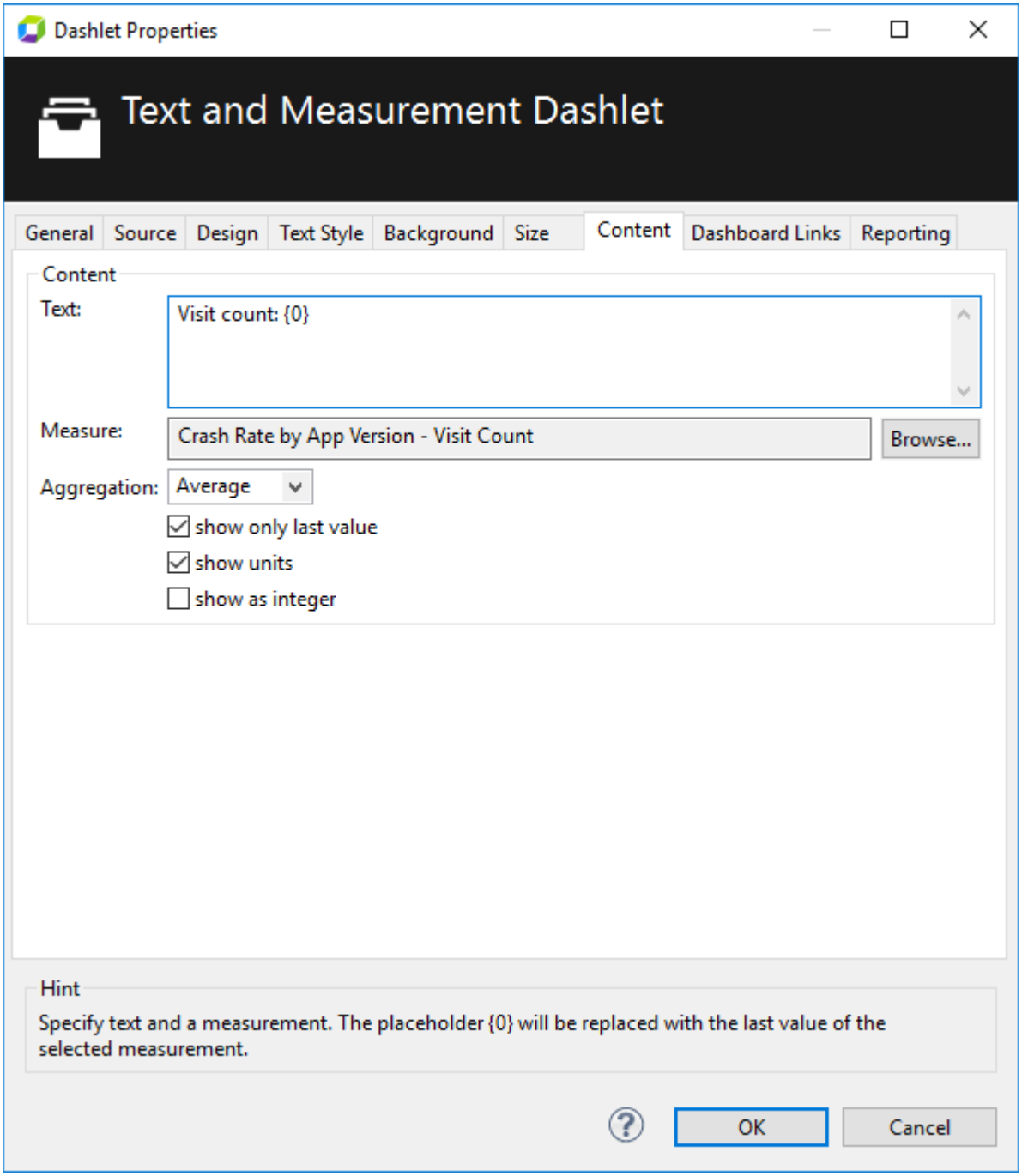
To set custom timeframe:

1. In the **Tile configurator** click **Advanced settings**.
2. In the **Timeframe** section, select the **Tile specific timeframe** radio button.
3. Set the required timeframe.
4. Click **Done**.

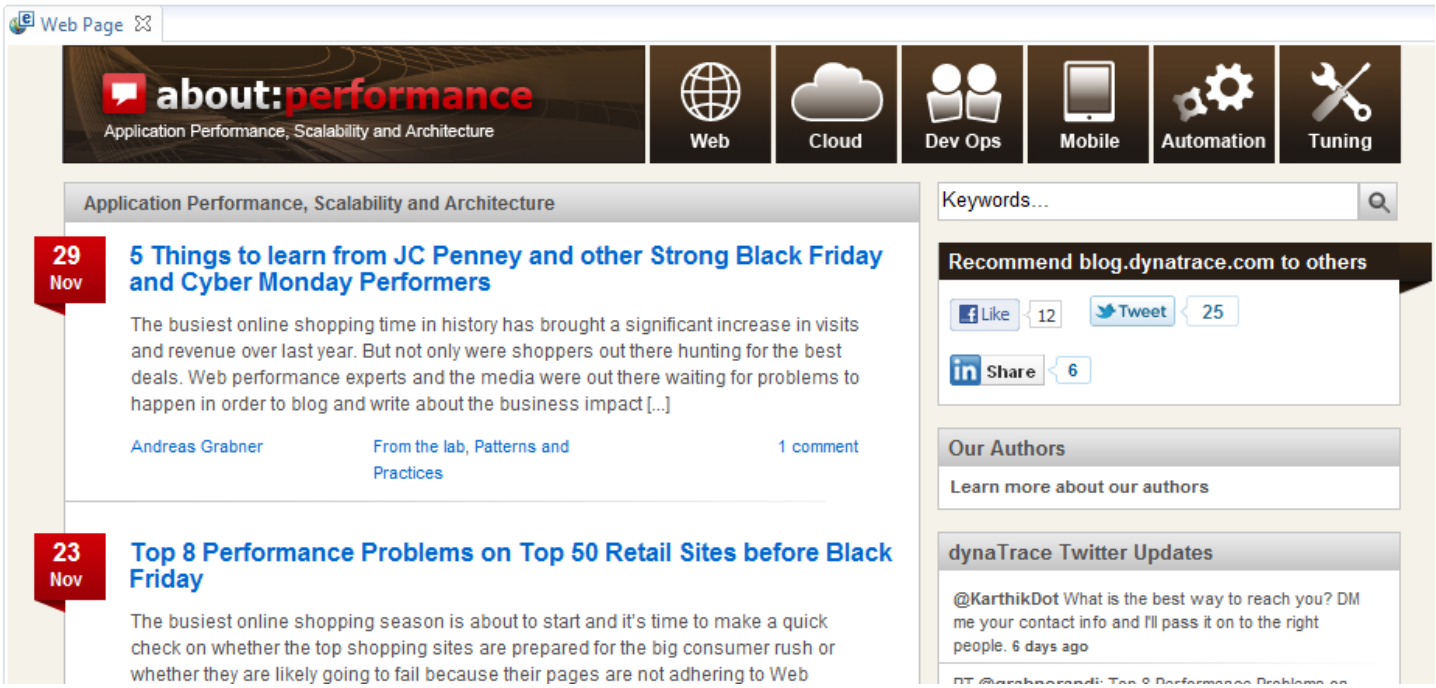
AppMon Client dashlet

Add the measure and type your text in the **Dashlet properties** dialog box.

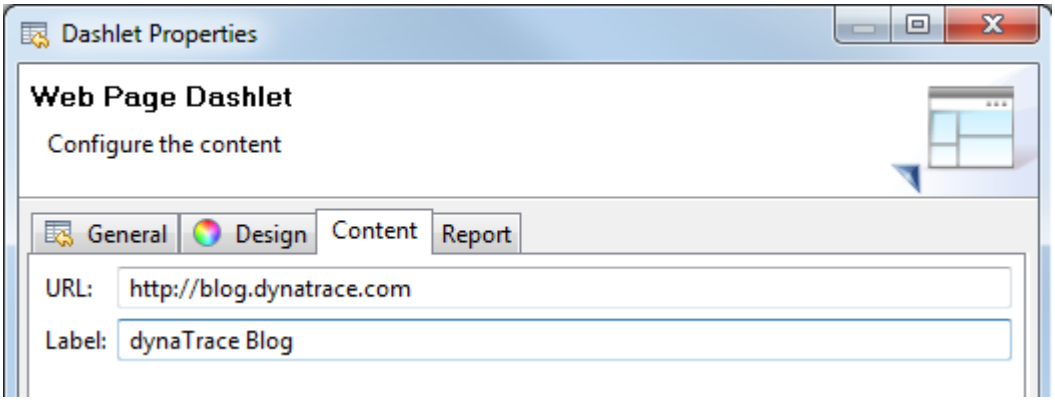
1. Right-click inside the dashlet, and select **Dashlet properties** from the context menu. You can also use the Alt+Enter hotkey.
2. Add your measure in the **Measure** field of the **Content** tab. Click **Browse**, and select the required measure.
3. Type your text in the **Text** edit box from the **Content** tab.
4. Use the customization options to change text and background images as well as the details of the measure.


Dashlet Properties
Text and Measurement Dashlet
General
Source
Design
Text Style
Background
Size
Content
Dashboard Links
Reporting
Content
Text:
Visit count: {0}
Measure:
Crash Rate by App Version - Visit Count
Browse...
Aggregation:
Average
show only last value
show units
show as integer
Hint
Specify text and a measurement. The placeholder {0} will be replaced with the last value of the selected measurement.
OK
Cancel

Web Page dashlet



Web Page Dashlet



Use the **Web Page** dashlet to specify and request the URL of a website to include on your dashboard. The dashlet does not refresh automatically. Use a meta-refresh tag in the website to refresh the content.

Charting

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Chart dashlet

Dual Axis Chart dashlet

Dynamic Measure Matrix dashlet

Incident Chart dashlet

Layer Breakdown dashlet

Time zones

The **Charting** node underneath a System Profile provides quick access to some preconfigured charting templates.

See FAQ - Measures and Charting for more information.

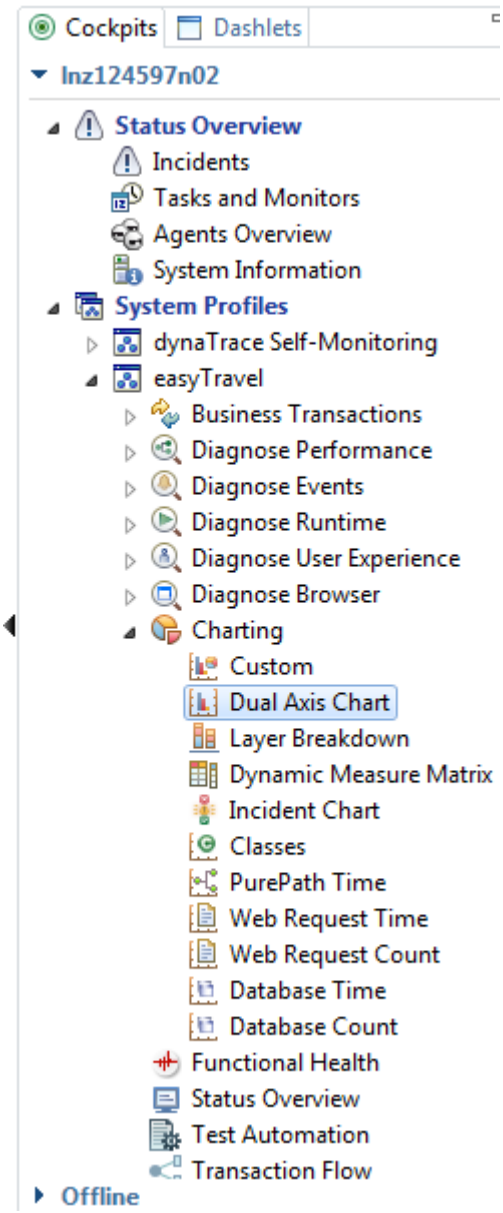


Chart Template	Description and Content
Custom	Creates a generic empty **Chart Dashlet** dashlet for this System Profile
Dual Axis Chart Dashlet	Creates a chart of two values with two independent axes, scales, and scale units
Layer Breakdown	Shows the **Layer Breakdown Dashlet** dashlet for this System Profile
Dynamic Measure Matrix	Creates an empty **Dynamic Measure Matrix Dashlet** dashlet for this System Profile
Incident Chart	Creates an empty Incident Chart for this System Profile

Classes	Displays the number of loaded and unloaded classes per JVM
PurePath Time	Displays PurePath Times (PurePath CPU Time and PurePath Time)
Web Request Time	Web Request timing measures for this System Profile
Web Request Count	Web Request count measures for this System Profile
Database Time	Displays a chart of the time spent in the database layer
Database Count	Number of database calls made

Chart dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Add series

Configure series

Chart types

Heat field

Chart splitting modes

Use the Chart dashlet to visualize collected Measures.
See FAQ - Measures and Charting for information about chart types, use, characteristics, and behavior.

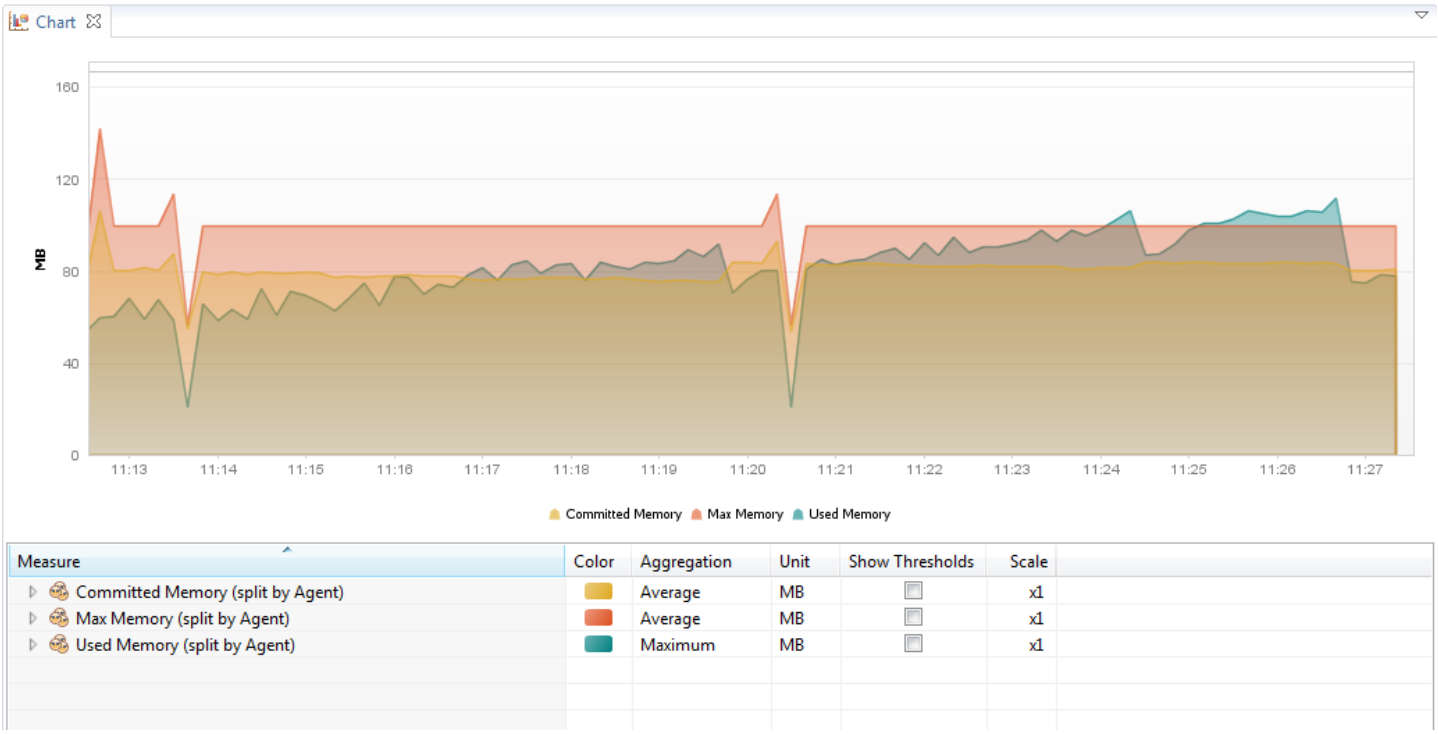
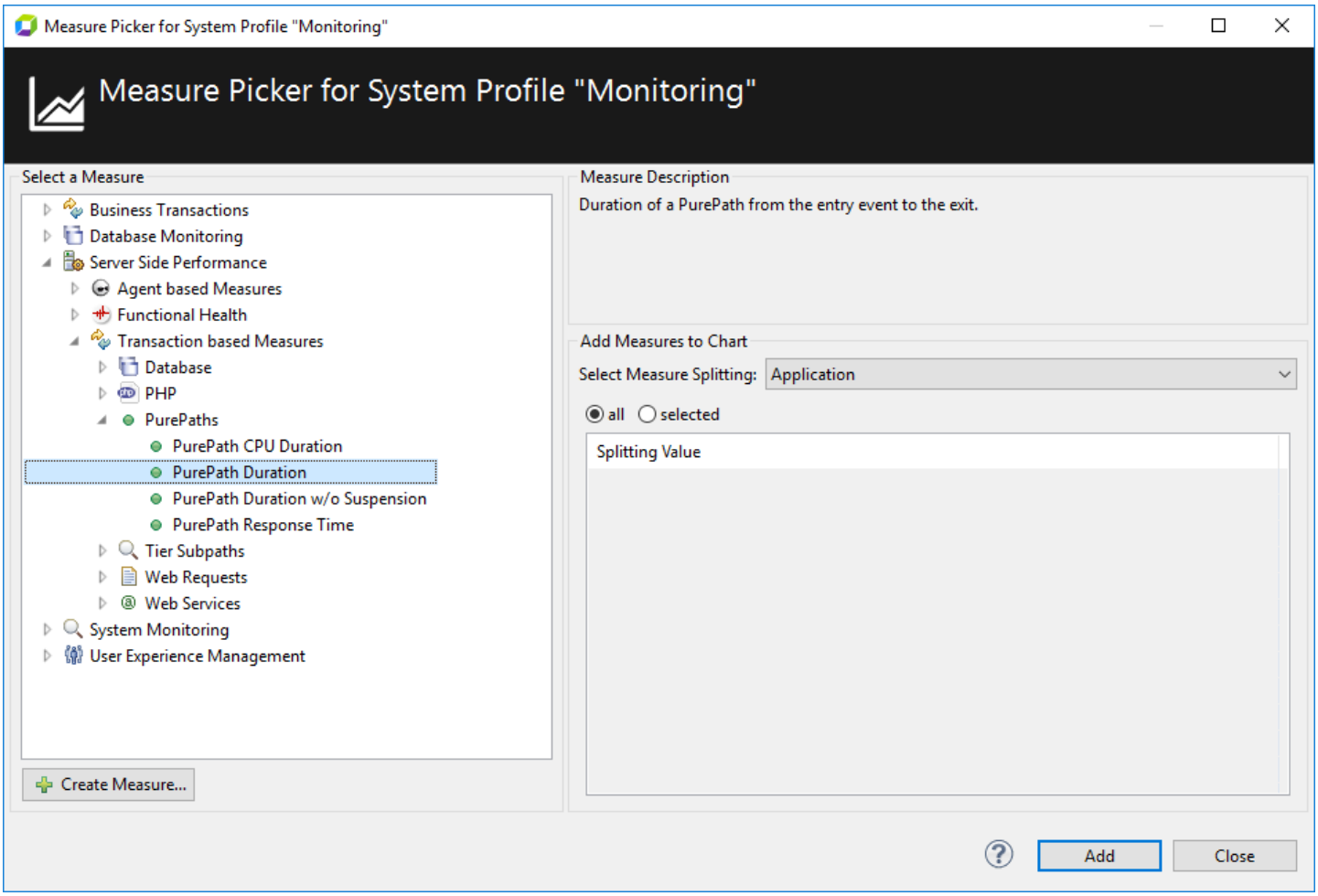


Chart dashlet

Add series

Use the **Measure Picker** dialog box to select a series to represent in a chart. To access the dialog box, right click the chart, and select **Add Series** from the context menu, or click **Add Series** in an empty chart.



Measure Picker dialog box

Configure series

To configure the chart series, right click the Chart dashlet and select **Configure Series**. Click the **Series** tab in the **Dashlet Properties dialog box**.

You can override the default chart-specific settings of each series with the following settings:

Setting	Description

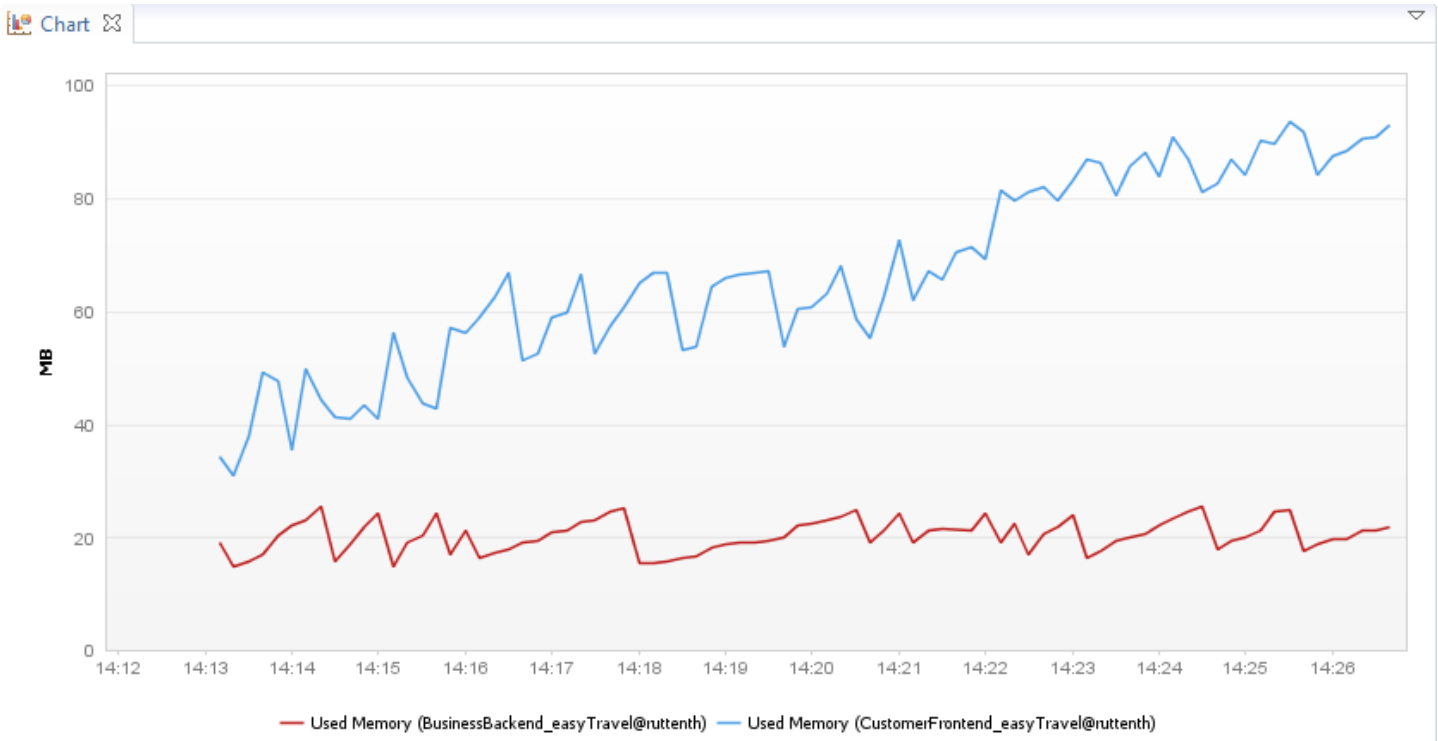
Color	Set the color for the series representation in chart.
Scale	Define a scale from 10^{-10} to 10^{10}.
Aggregation	Define the measurements aggregation.
Unit	Defines the measurements units

Chart types

The various chart types are described below.

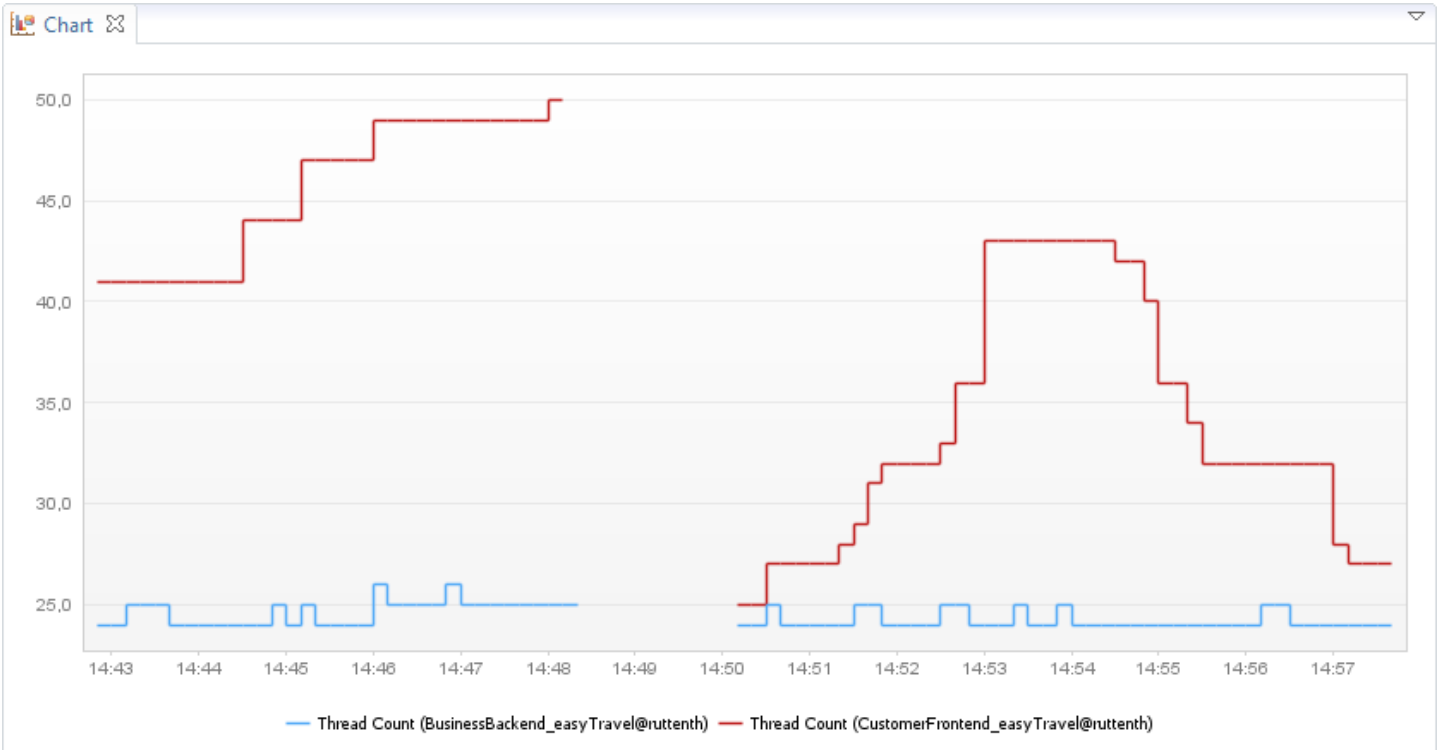
Line chart

- Represents any kind of Measures.
- Similar to an Area chart but supports a higher number of Measures.



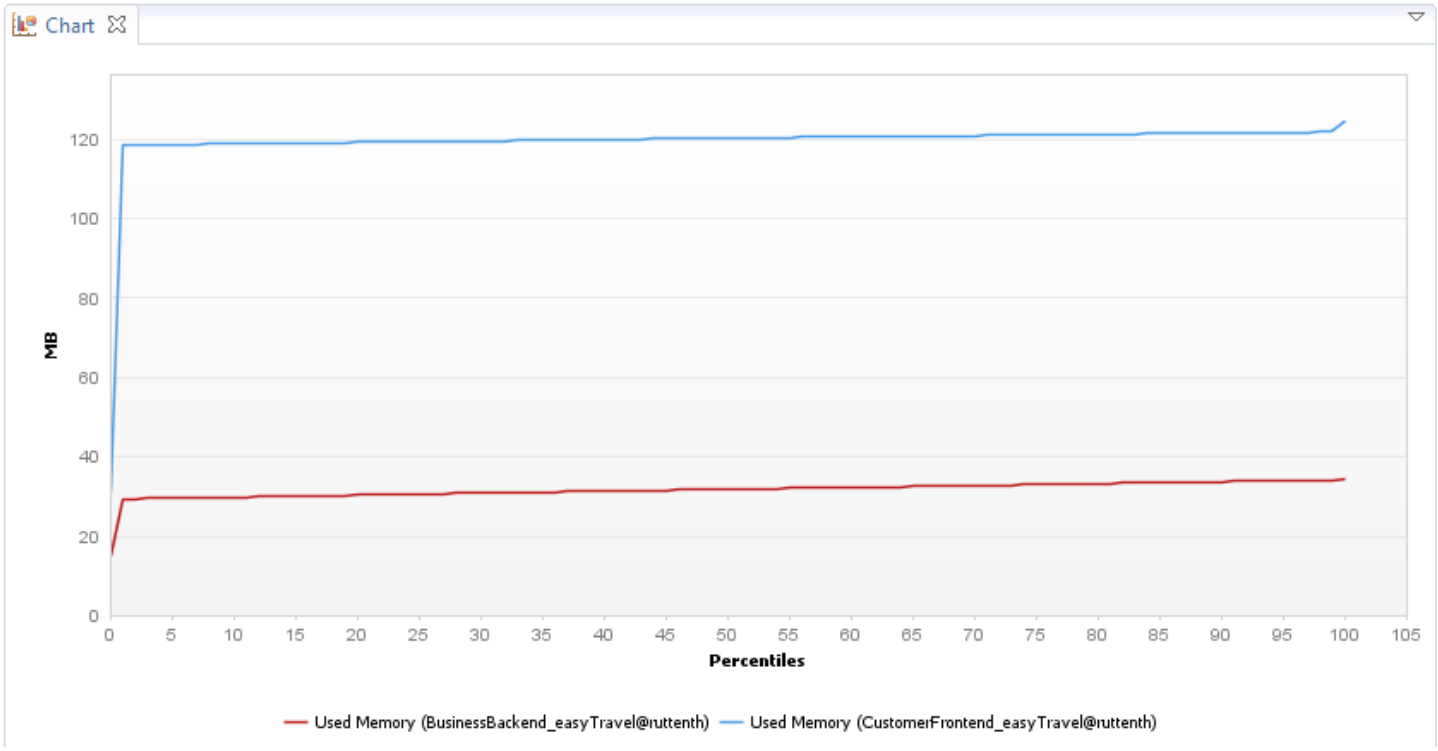
Step Chart

- Renders its values with discrete steps.
- Useful to display count Measures (threads).
- Unlike Line charts, the intervals are not connected if data is unavailable for successive intervals.



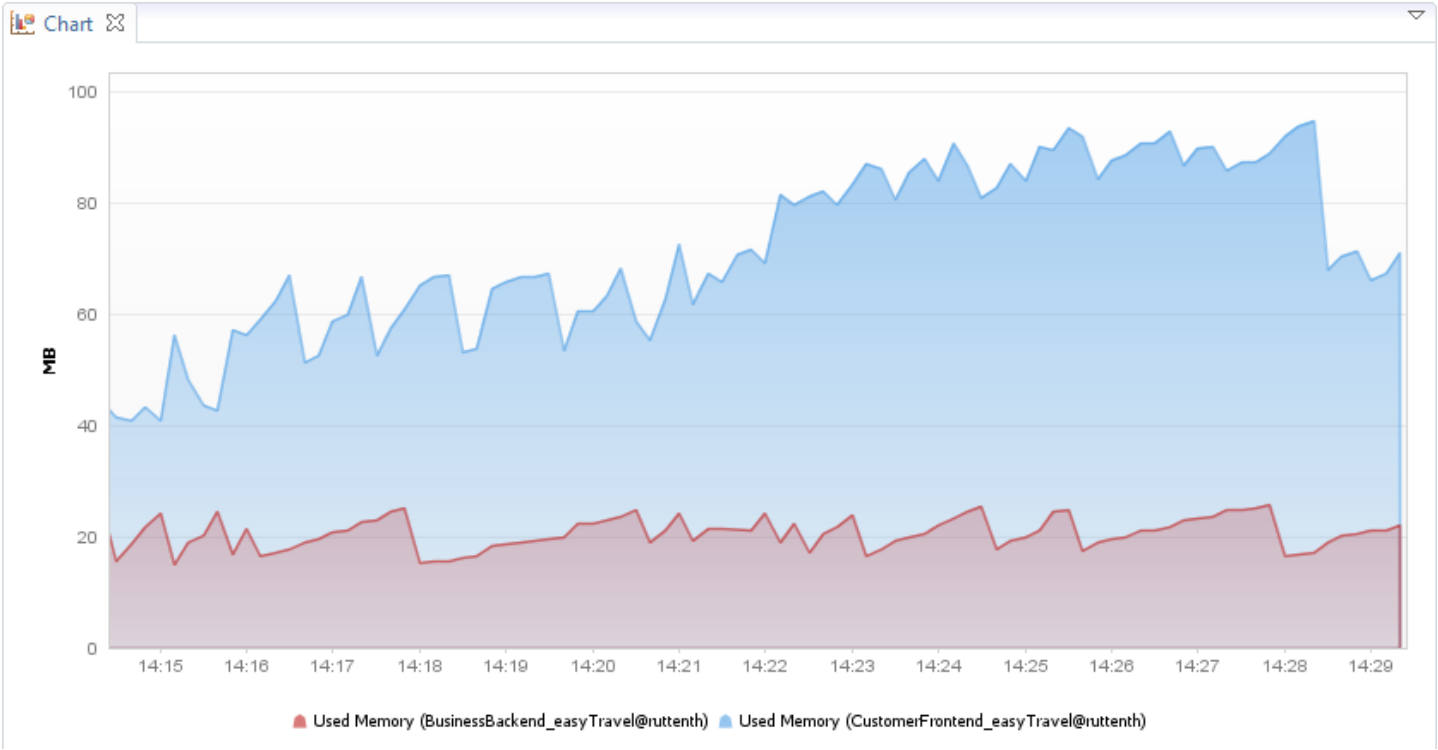
Percentile chart

- Plots percentiles from 0 to 100 for a time frame.
- Chart time frame is one hour, regardless of configured time frame (based on one-hour percentile aggregation).
- If the chart has many Measures, use a logarithmic y-axis to increase readability.



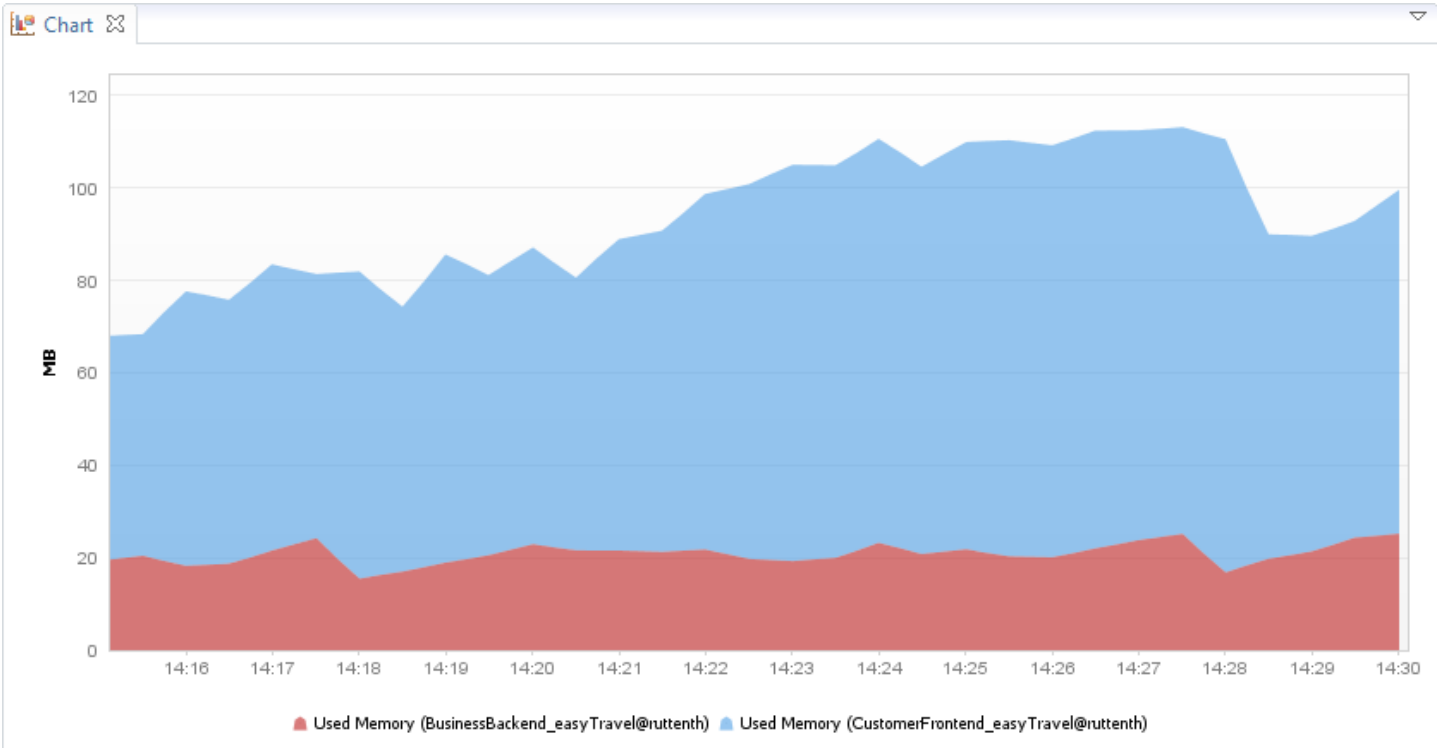
Area chart

- To plot an area, you must have two measurements per Measure in the configured time frame.
- Multiple Measures are plotted in separate layers, therefore the chart cannot process numerous Measures.



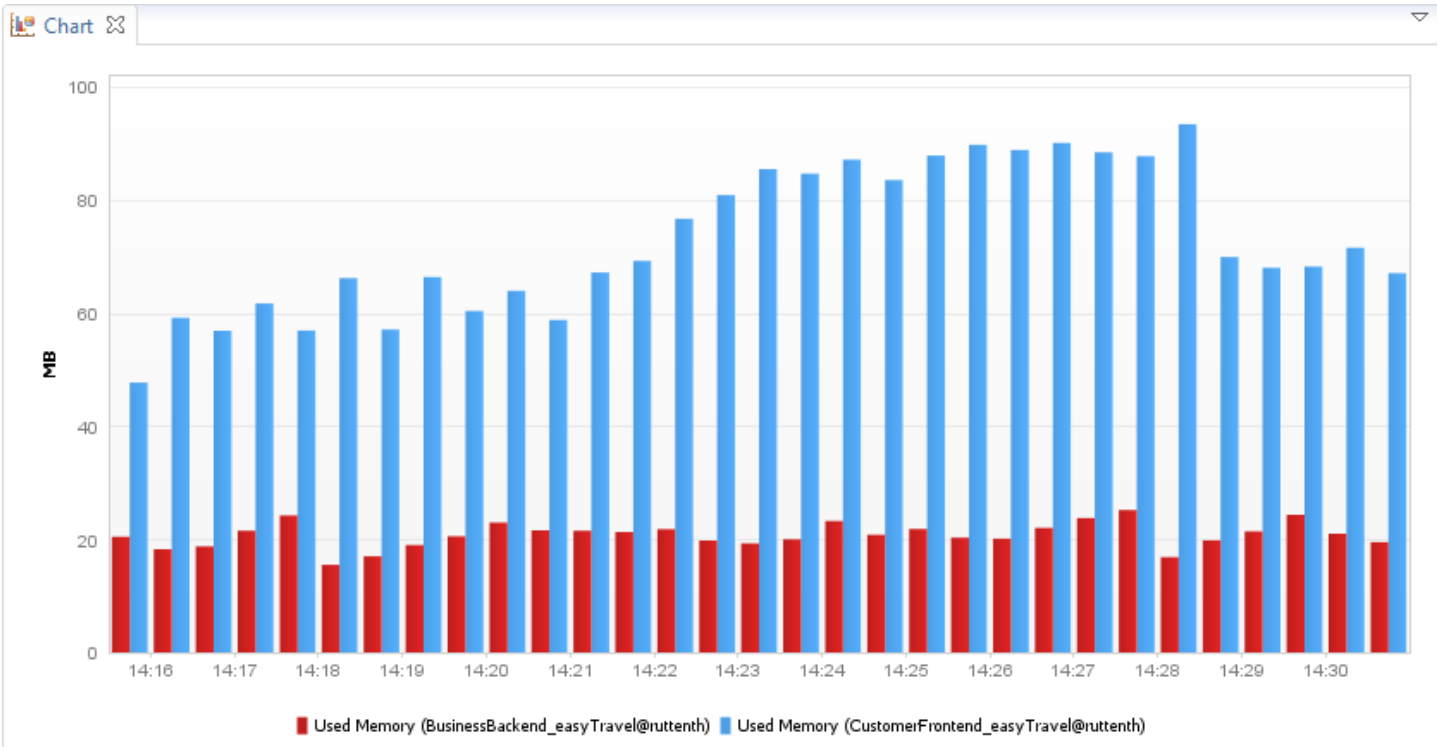
Stacked area chart

- Values are stacked for each time stamp.
- To plot an area, you must have two measurements per Measure for the configured time frame.



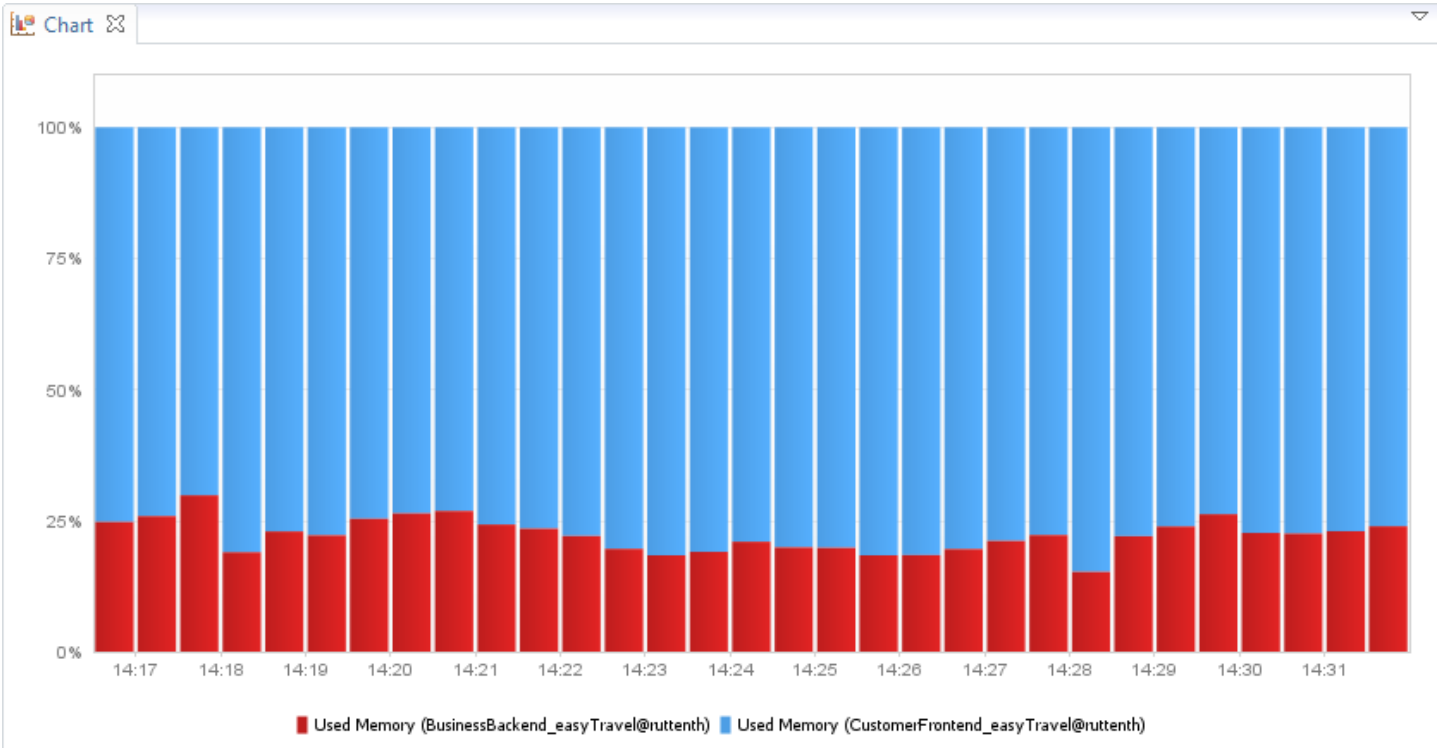
Clustered bar chart

- Measures are represented by individual bars for each aggregated time stamp.
- Increase the chart resolution to support numerous Measures.



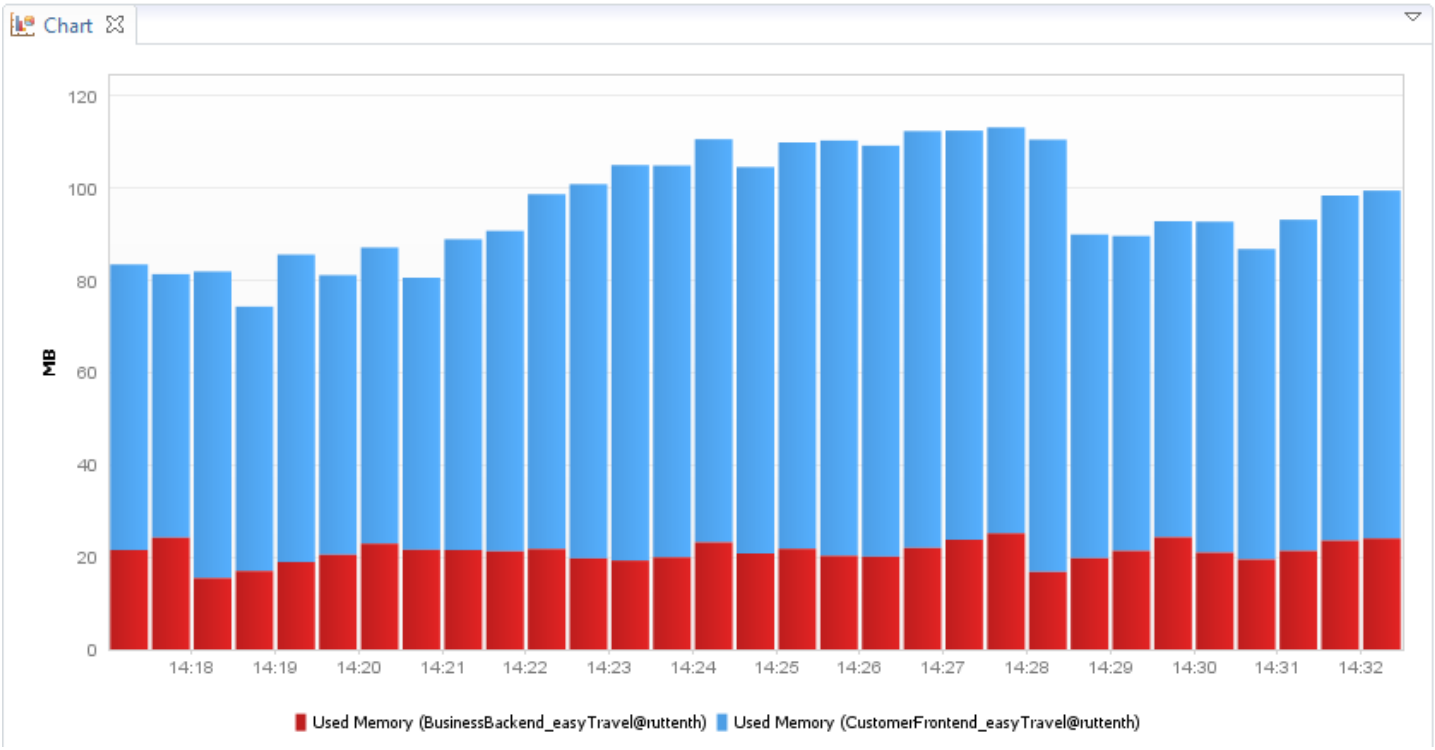
Stacked percentage bar chart

Configure all Measures to use the same unit and aggregation.



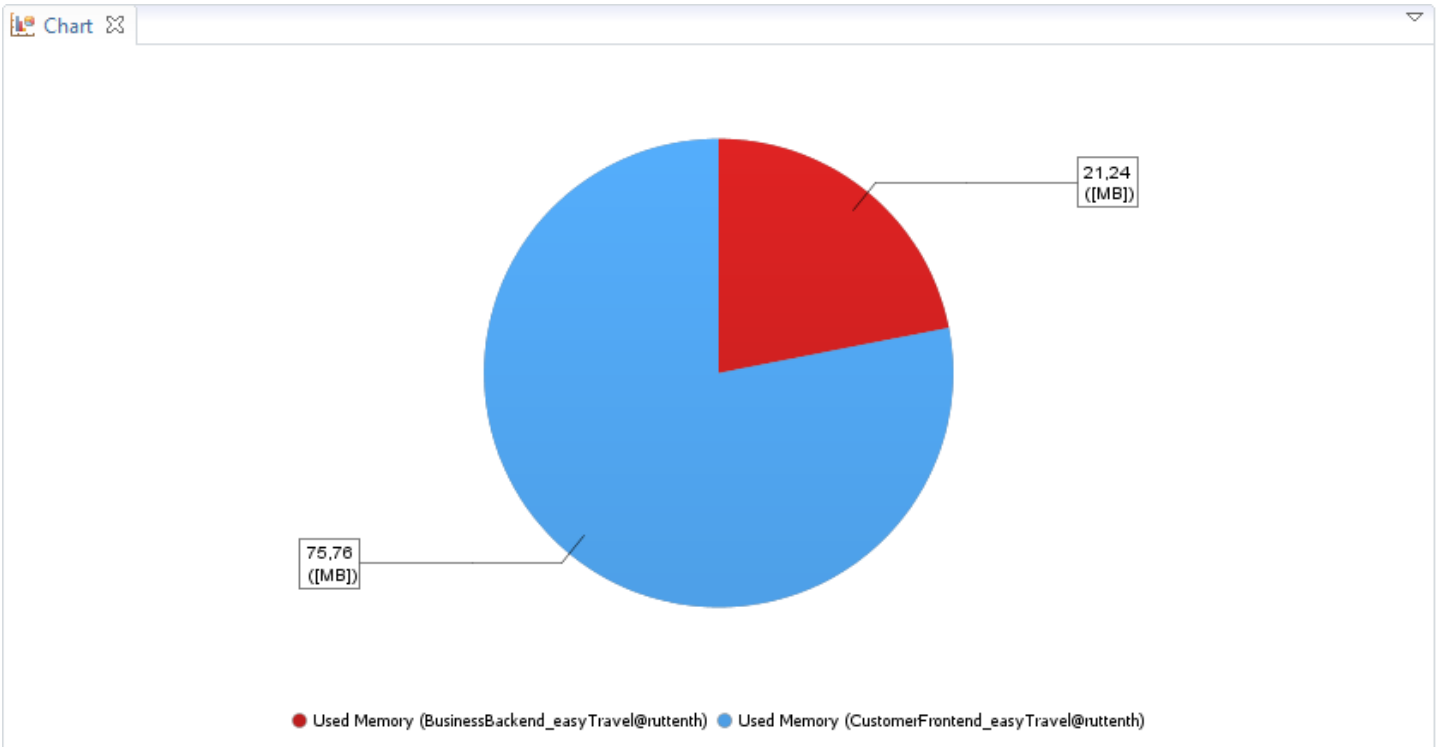
Stacked bar chart

- Measures are stacked for each aggregated time stamp, depending on the chart resolution.
- Configure all Measures to use the same unit and aggregation so they are comparable.



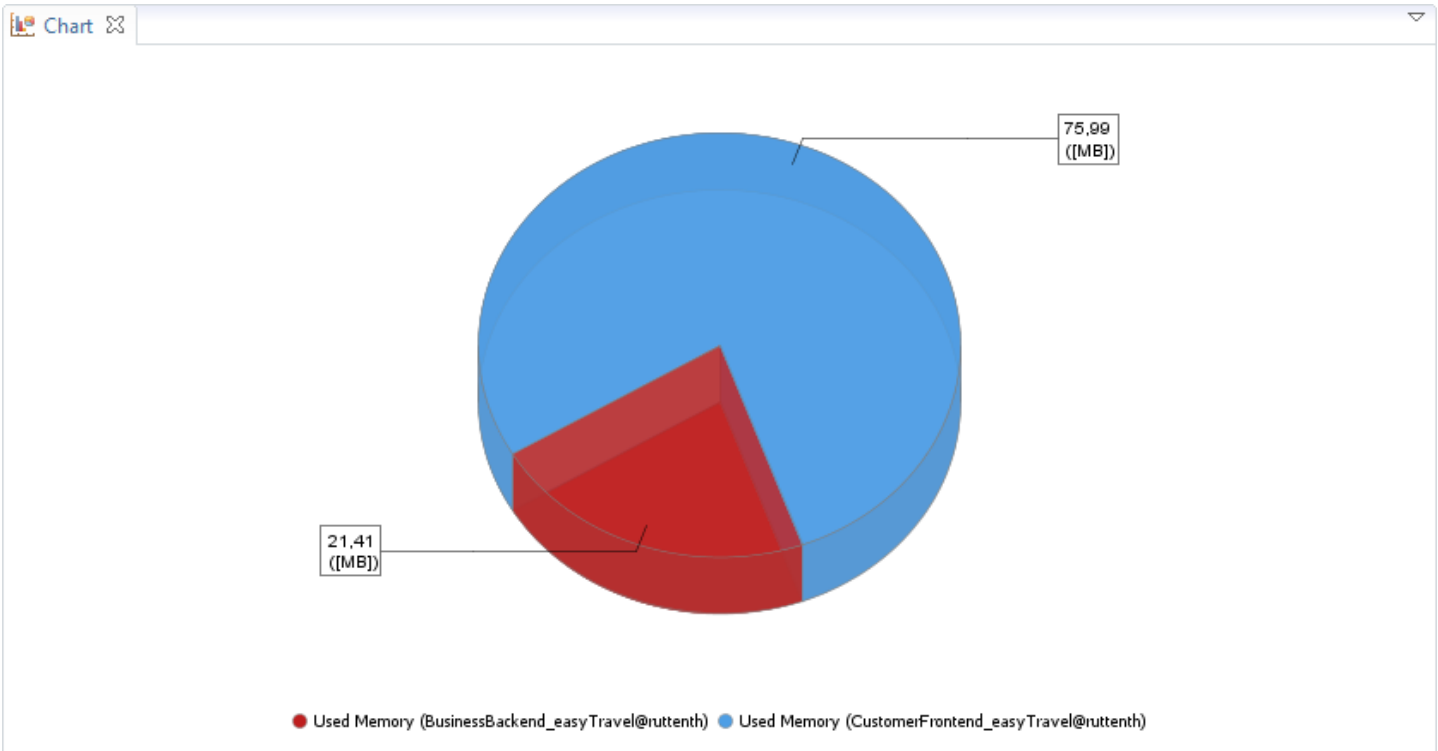
Pie chart

- The value shown depends on aggregation and the time frame.
- For more information about aggregation, see the Meter chart.



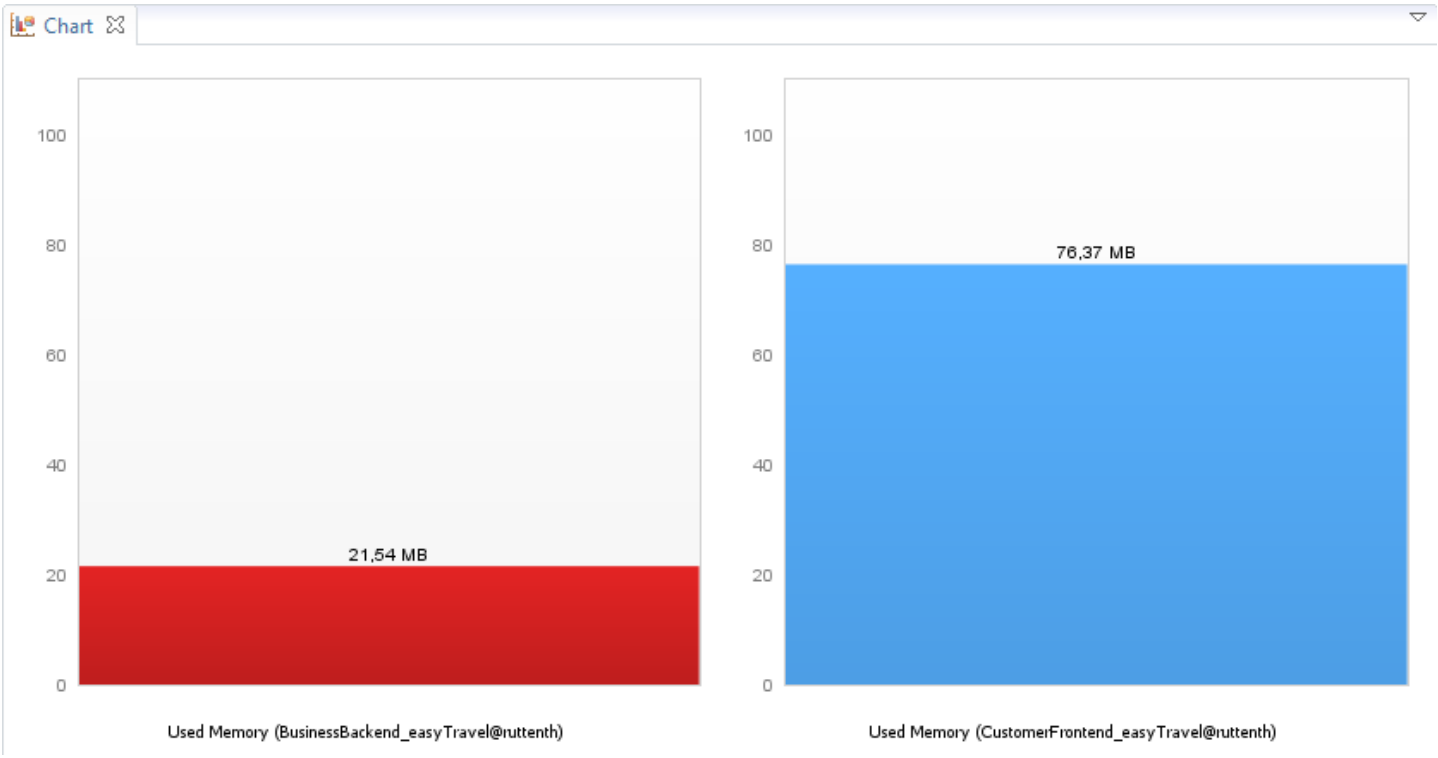
Pie chart 3D

- The value shown depends on aggregation and the time frame.
- For more information about aggregation see the Meter chart.



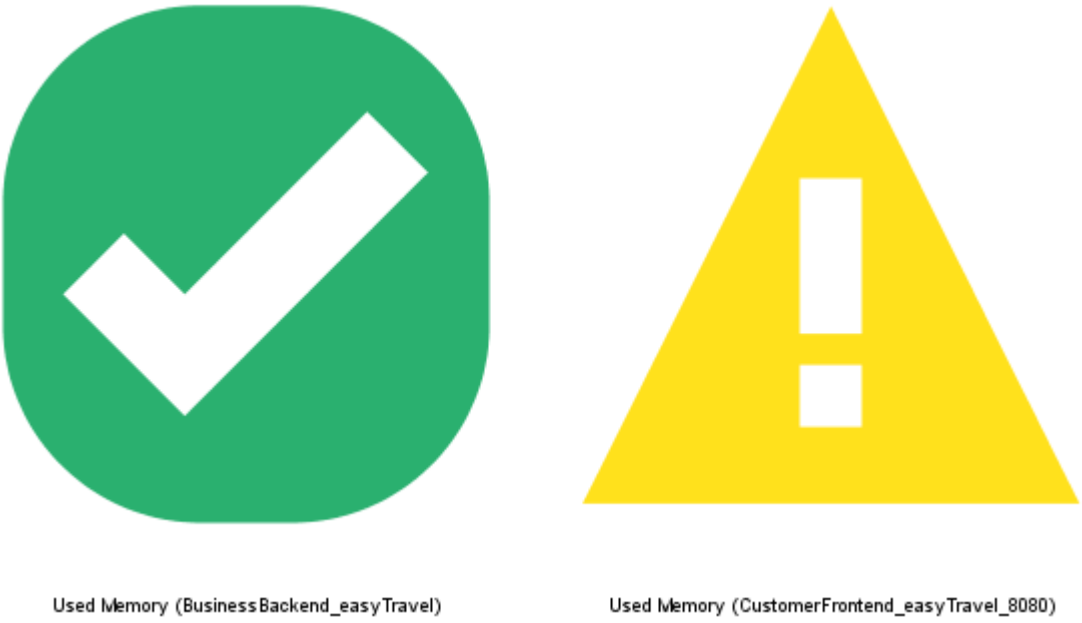
Meter chart

- Shows the aggregated value of a Measure for the time frame.
- The following explains the value calculation which depends on the aggregation:
  - Avg: Average value of available measurements within time frame.
  - Max: Maximum value within time frame.
  - Min: Minimum value within time frame.
  - Sum: Sum value of available measurements within time frame.
  - Count: Amount of measurements within time frame.
- Resolution affects the displayed value.
- Set a fixed axis lower and upper bound to disable the auto calculation of the y-axis.



Traffic light chart

- Charts the threshold violation of a Measure.
- Represents the most severe threshold violation within a time frame.
- Threshold violation state is calculated against the configured aggregation.
- Resolution effects how the threshold state is displayed.
- You must define a proper threshold for the displayed Measure.
- To analyze why a threshold is violated, hover over the chart to see the tooltip, or switch to a Line chart.



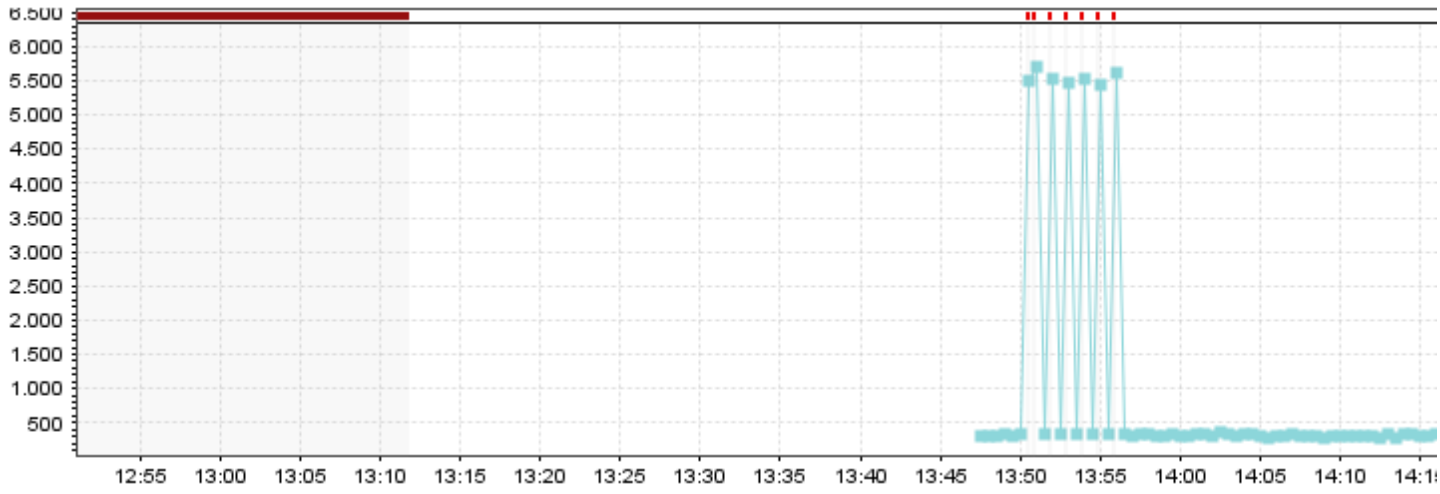
Heat field

The heat field represents incidents that are triggered on AppMon Server.
Use the default settings for the following:

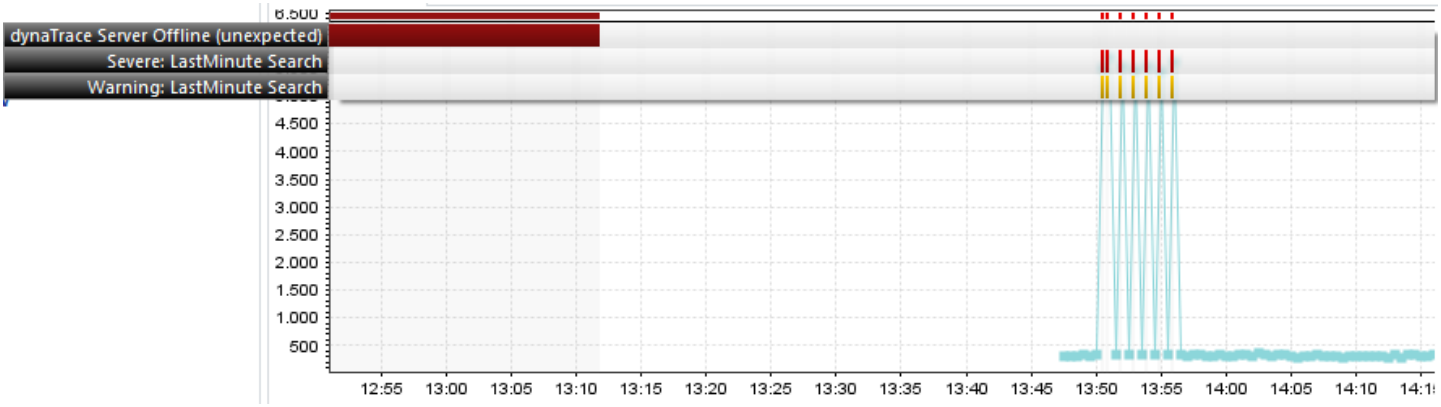
- Self-monitoring incident rules
- Incident rules that are based on charted Measures
- User defined custom events (manually added)

Usage

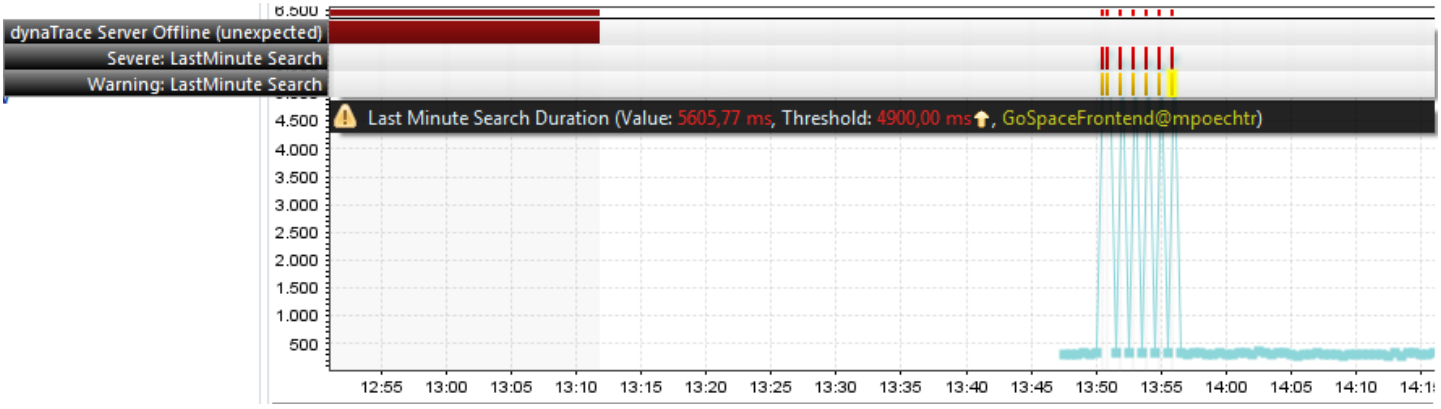
Select individual incidents from the expanded heat field for more information. Double-click on an incident to drill down into the incident rule dashboard. Click the context menu for more drilldowns.



Hover above the heat field to automatically expand the field.

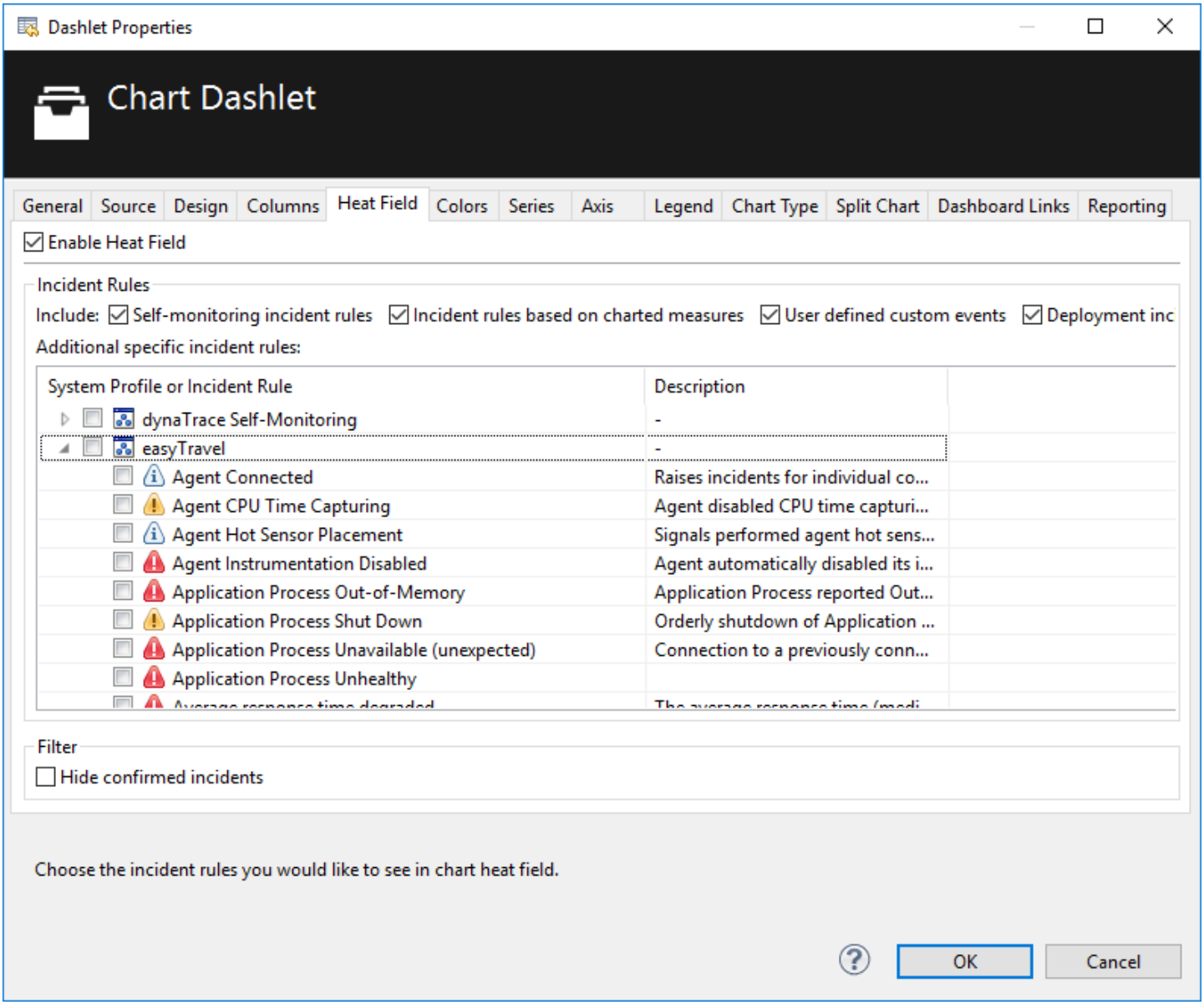


The heat field is enabled by default and appears at the top of the chart.



Configuration

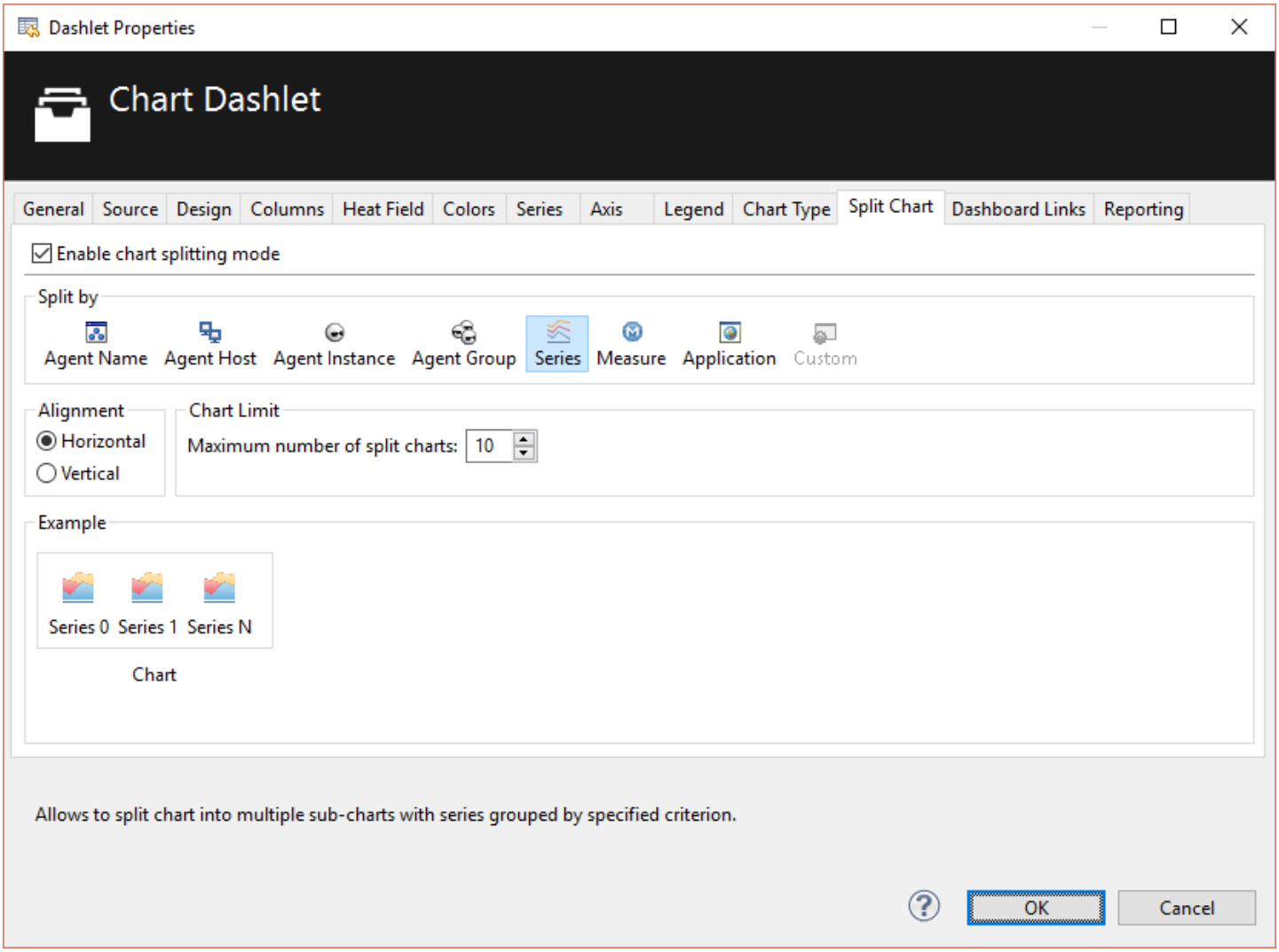
To configure the chart heat field, right click the dashlet and select **Dashlet Properties > Heat Field tab**.



Heat Field tab of a Chart Dashlet

Chart splitting modes

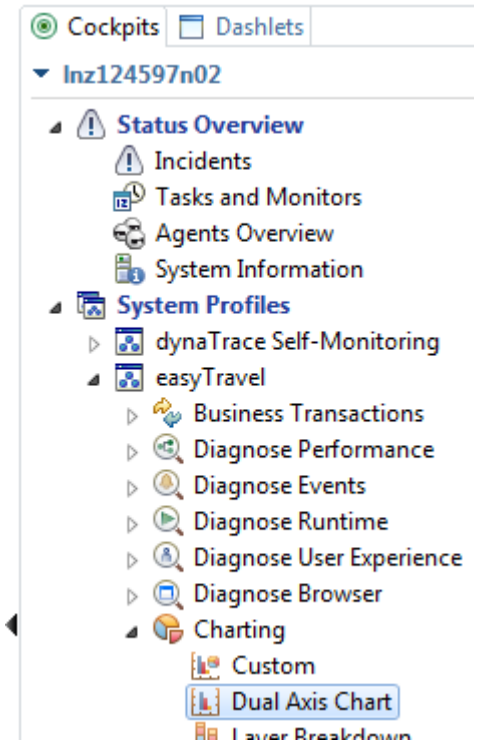
By default all measures that are added to a chart are drawn as separate series within a single chart, except for the meter chart which is split by series by default and cannot be displayed without a splitting mode. To show different splittings in different charts, chart splitting settings have to be adjusted under **Dashlet Properties > Split Chart**.



The available splittings options are Agent Name, Agent Host, Agent Instance, Agent Group, Series, Measure, Application and Custom. Depending on the measure there might not be all options available. In order to avoid a massive amount of separate charts being shown, there is a configurable limit at which point all charts will be drawn in a single chart even though chart splitting is enabled.

One more thing to note is that in case a split mode different from the splitting of the added measure is applied, this splitting will not be replaced by the chosen split mode, but rather the split mode will be added to the measure splitting. For example, adding a measure that is split by application to a chart and then choosing agent name as chart splitting mode will result in different charts for each combination of agent name and application that values exist for.

Dual Axis Chart dashlet



Cockpit Directory with Dual Axis Chart

To open a Dual Axis chart, double-click an item in the Cockpit, and then click **Sidebar > AppMon Server name > System Profile > System Profile name > Charting > Dual Axis Chart.**
The Dual Axis chart shows two values with two independent axes, scales, and scale units.

Dynamic Measure Matrix dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

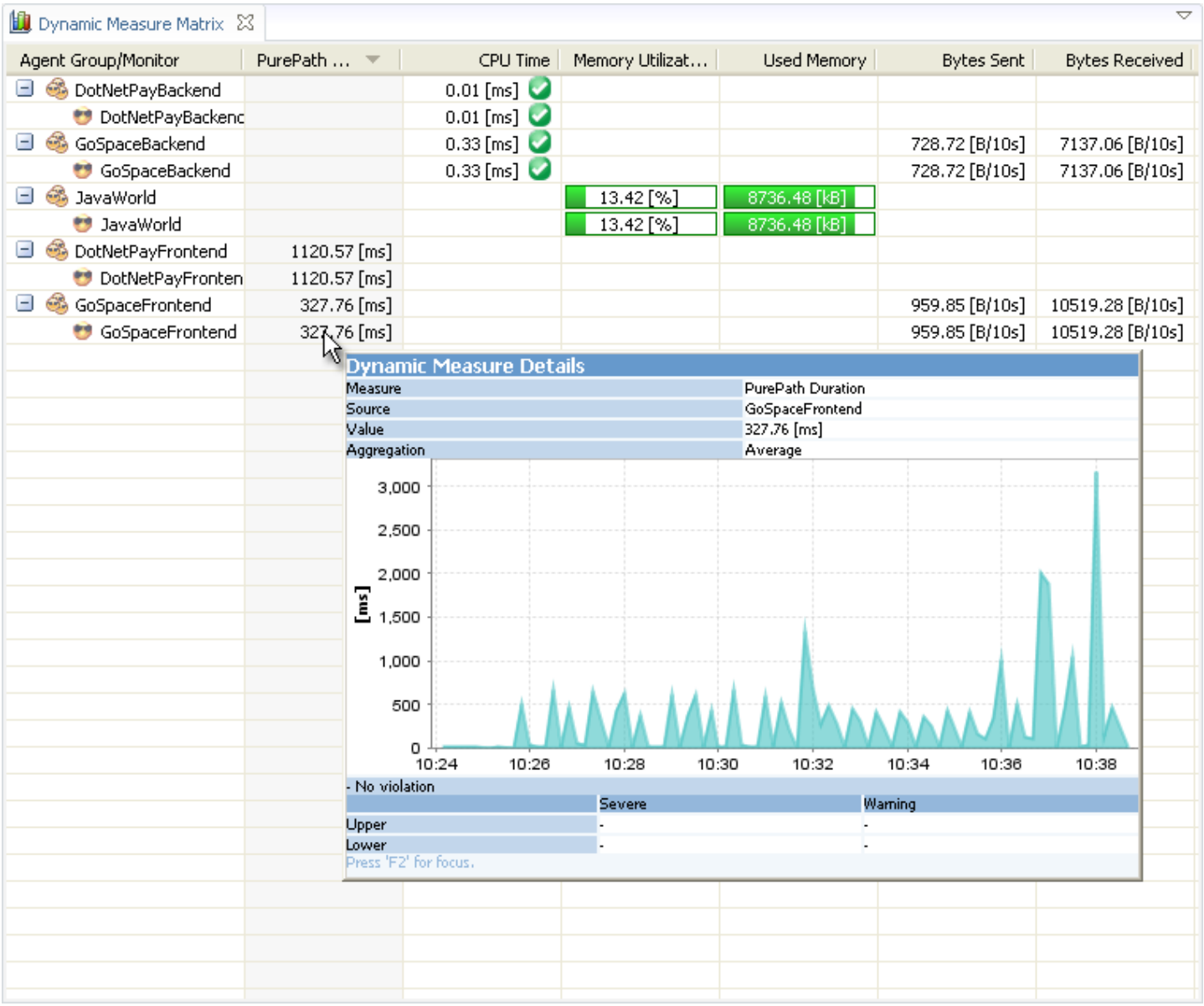
Configuration

Column visualization

Remarks

Use the **Dynamic Measure Matrix** dashlet to display aggregated dynamic Measures per agent group and agent.

Each agent group and agent is presented in a row. Each added Measure is exposed in a separate column.
Hover over a cell and the tooltip shows a chart representation.



Dynamic Measure Matrix Dashlet

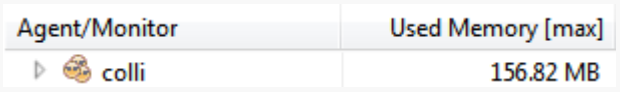
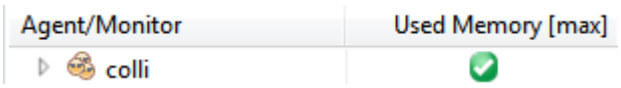
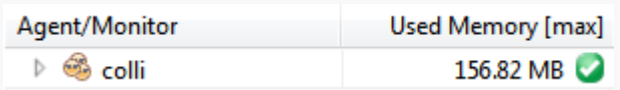
Configuration

Use the **Measures properties** dialog box to add and remove Measures, as well as change the associated unit and aggregation.

Open the context menu and click **Configure Measures** to open the **Measures properties** dialog box.

Column visualization

Use the **Visualization** sub context menu to change the column visualization to one of the following:

Visualization	Description	Example
Value	Plain text measure value	
Threshold	Traffic light that indicates threshold violations	
Value+Threshold violation	Combination of measure value and threshold violation indicator	
Bar	A logarithmic scaled bar that changes background color when threshold are violated	

Remarks

To change the threshold of a measure, click **Configure Measures > Select Measure > Edit**.
The change affects only the column from which the context menu is invoked.

The measure value bars next to agents can appear red although the average value on the bar didn't breach the threshold. This can be because one or more of the agents underlying applications breached the threshold, so although the overall average is not breaching, one or more apps will change the color (please expand the agent node to inspect the individual applications).

Incident Chart dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Configuration

Visualization

Incidents table

Related pages

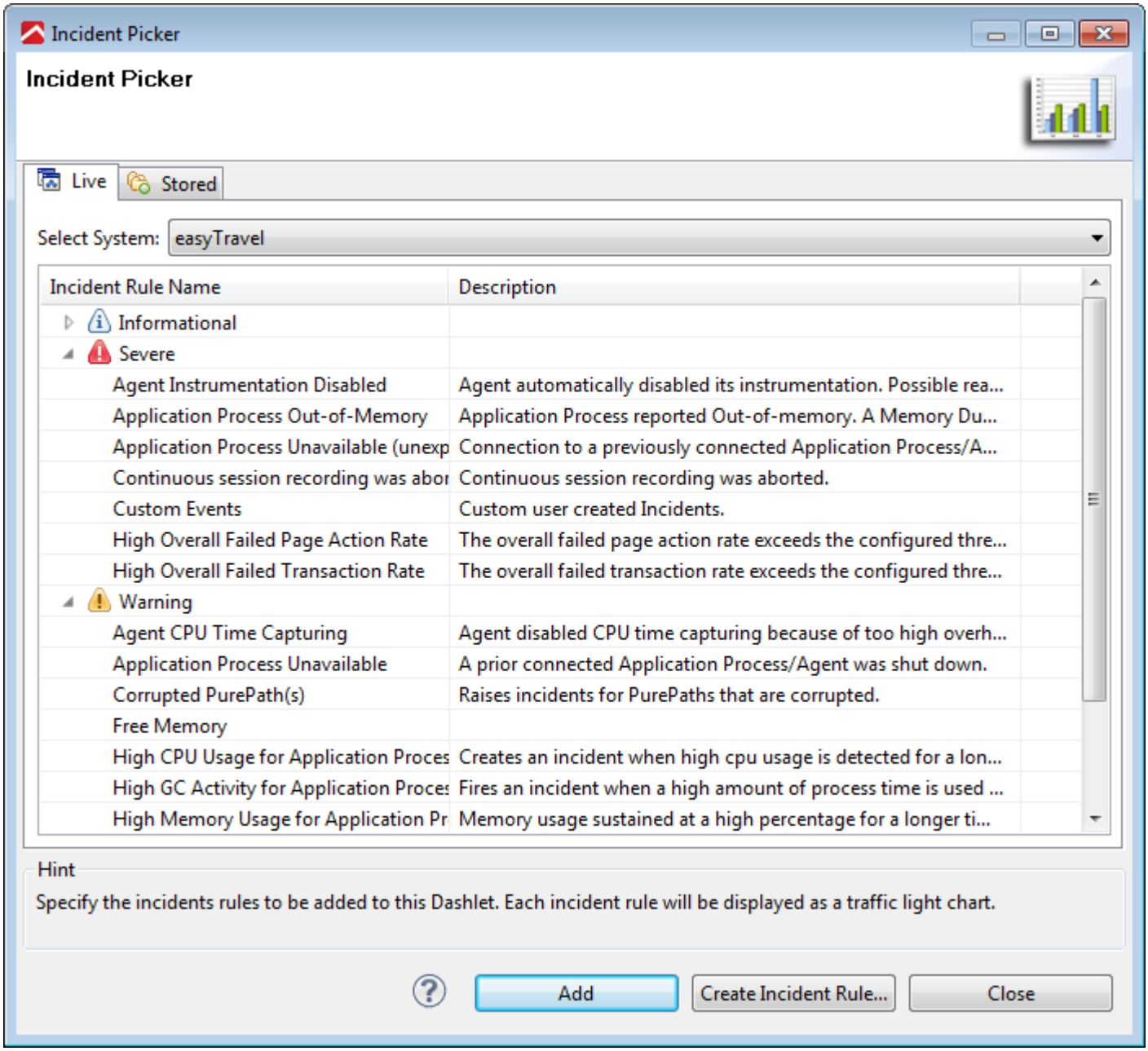
Incidents Configuration

The **Incident Chart** dashlet visualizes the violation of selected incident rules during the configured time frame.

Configuration

Add Incident Rules

To add Incident rules open the dashlet's context menu item and click **Add Incident Rule**.You can also click **Add Incident Rule** on an empty dashlet. Click the **Incident Picker** dialog box to choose the desired incident rules from live, stored, or offline sessions; or create new incident rules.



Incident Picker

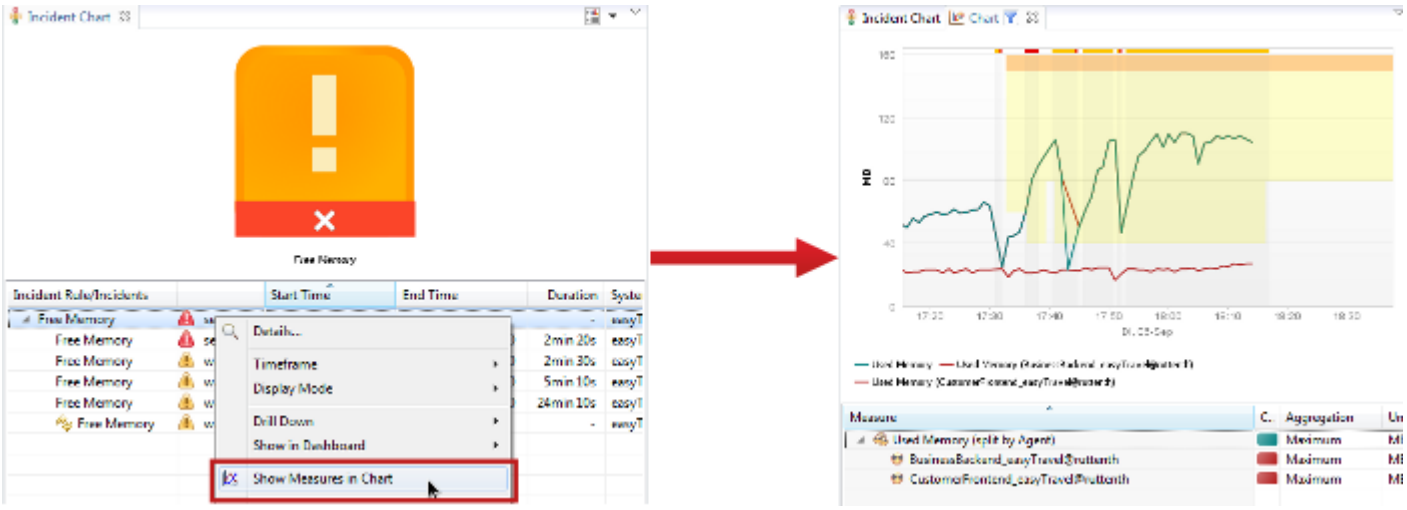
Edit Incident Rules

To edit Incident rules, open the Incident rule context menu in a chart, or open the Incident rule entry in the chart table, and click **Edit Incident Rule**. See Configuring Incident Rules for more information.

Remove Incident Rules

Open the Incident rule context menu, or open the Incident rule entry in the chart table, and click **Remove**.

Show violating measures in a Chart dashlet



Click **Show Measures in Chart** to visualize the selected incidents' Measures with visible thresholds in a new **Chart Dashlet**.

> **Open in new dashboard**
>
> Hold **Shift + Modifier key** and invoke the context menu to open the **Chart** dashlet in a new dashboard.

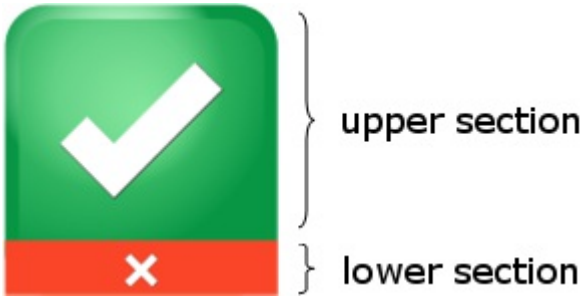
Visualization

The Incident chart shows the incident occurrence. This is not an information incident chart. Log and show informational incidents in the chart heat field and the incidents overview. See Heat Field and Incidents Overview Dashlet for more information.



Two sections inform about the incident state:

- The **upper section** answers the question, “Is there an incident currently active?”
  - You see the actual incident state.
- The **lower section** answers the question, “Did an incident happen in the past?”
  - You see the incident history within the time frame.

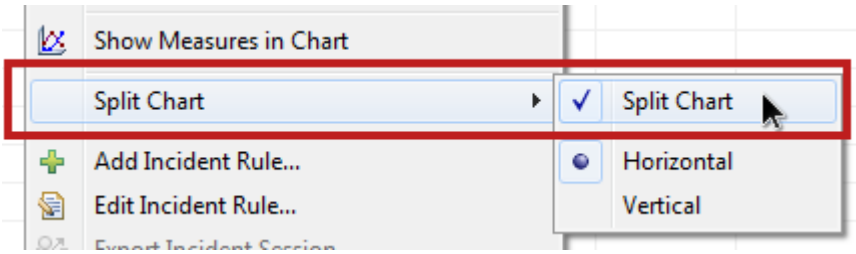


Examples

Example 1:		A non-incident occurred within the time frame.
Example 2:		A severe incident is triggered and occurred within the time frame.
Example 3:		No incident occurred.

Split chart mode

By default, the Split Chart mode is enabled and an incident traffic light visualizes each incident rule you add. To disable splitting, open the context menu, click **Split Chart**, and clear the **Split Chart** setting in the sub-menu.



When you disable this, all incident rules get aggregated into a single traffic light.

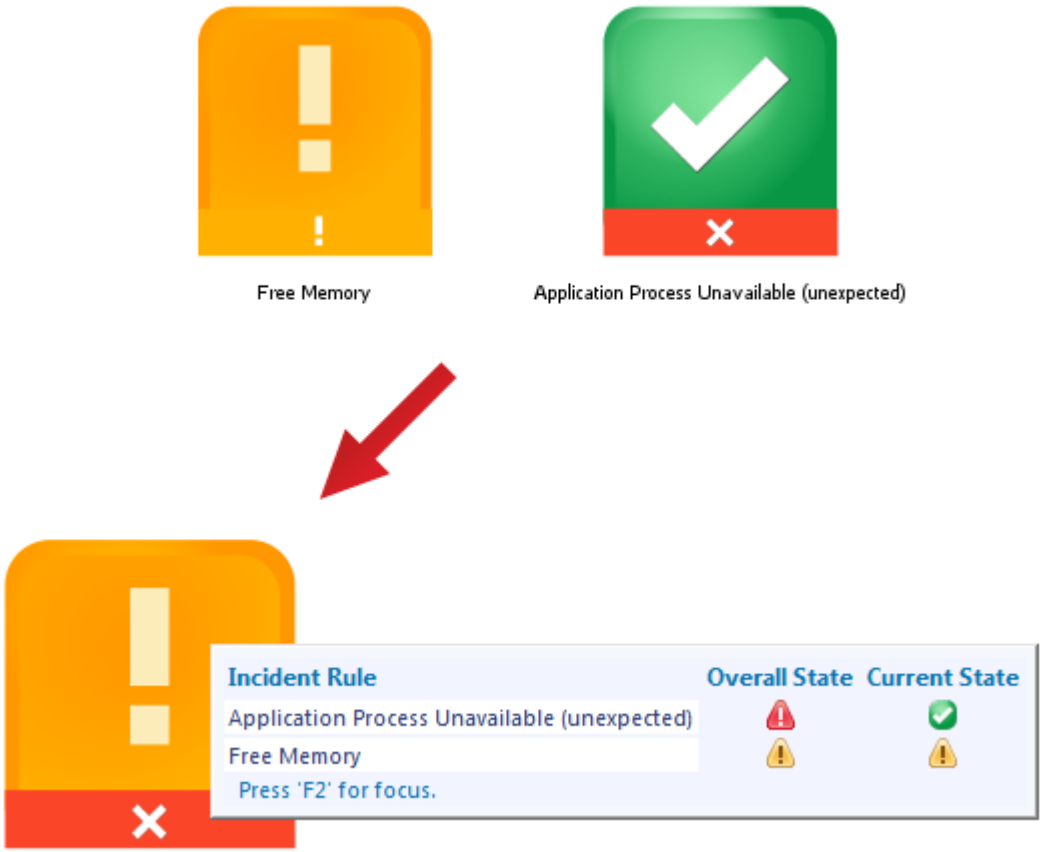


Chart Aggregation with Split Chart Disabled. The Tooltip Shows Details about Aggregated Incident Rules.

Incidents table

Column	Description
Incident Rule/Incidents	Name of the incident rule or incident
Severity	Icon that represents the violation status of incident rule or incident
Start Time	Start time of the incident interval
End Time	End time of the incident interval
Duration	Formatted duration of the incident interval

System Profile	System Profile of the incident rule
Message	Short description of incident rule violation
Description	Detail description of incident rule violation

Rows of pending incidents are highlighted by the

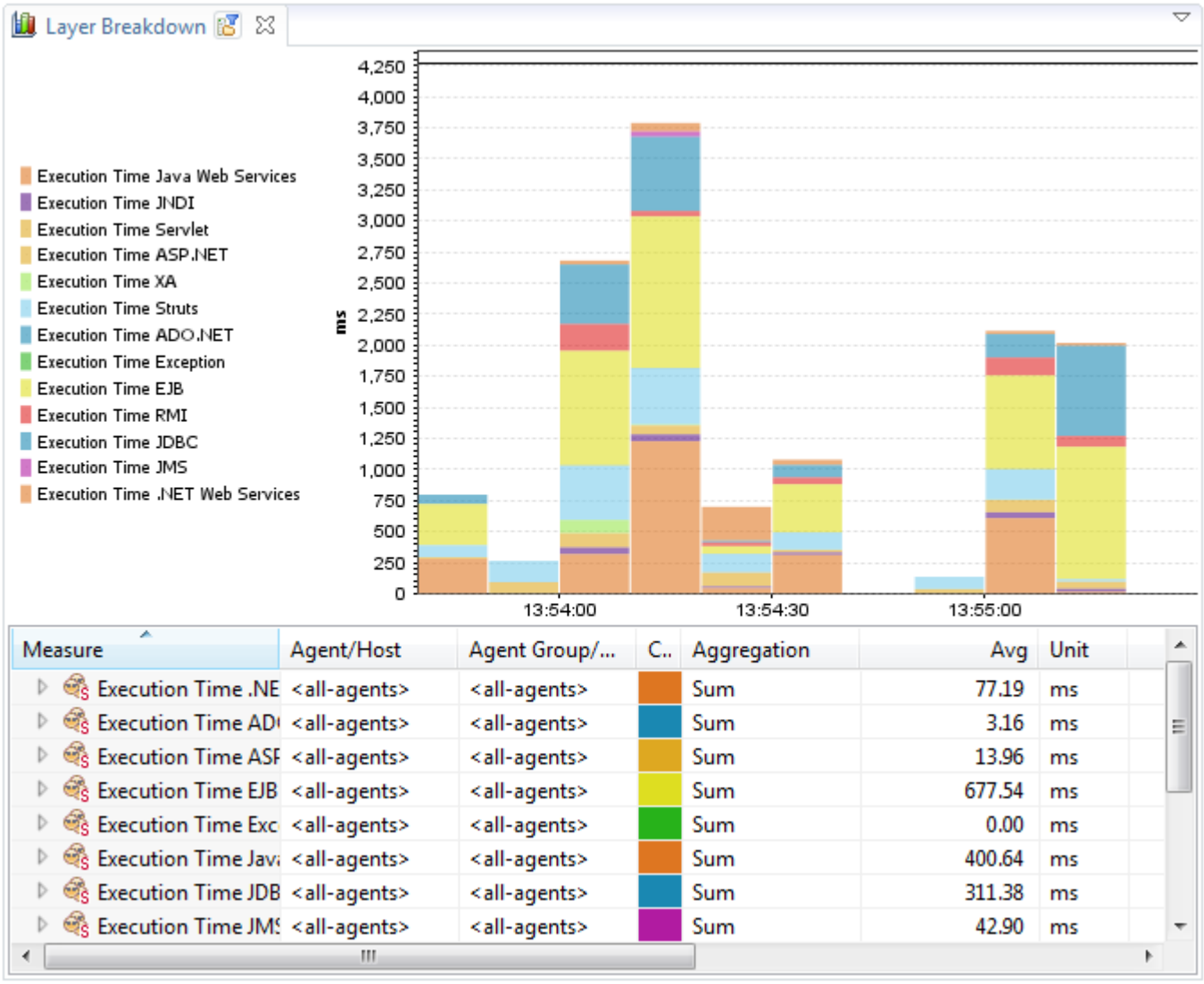
icon.

Table structure modes

The following are table structure modes:

- Flat mode: No grouping in table. Shows only incidents.
- Group incidents by incident rule
- Group incidents by severity: Informational, warning, and severe.
- Group incidents by state: Pending and closed.

Layer Breakdown dashlet



Layer Breakdown Dashlet

Use the **Layer Breakdown** dashlet to get a quick overview of the application total execution or CPU time, and this number's distribution over the application API layers.

Time zones

Resolution 1d: All data for one day aggregates into a single data point with a 12:00 am time stamp.
Time Frame: The time frame *today* shows all data for the time series from 12:00 am until now (2:15 pm) in the AppMon Server. The aggregate value shifts by a six-hour time difference as the AppMon Client visualizes it. As a workaround, you can adjust the client machine time zone to be the same as the server machine time zone.

AppMon Server location	Linz, AT (GMT+01:00) (GMT = Greenwich Mean Time = UTC)
AppMon Client location	Boston, US (GMT-05:00)
Time frame	today
Current Time Linz	19-Nov 2009 - 2:15 pm
Current Time Boston	19-Nov 2009 - 8:15 am

Example

Time Frame	Description
Today, This Week, This Month	AppMon Client uses its own time zone to visualize a time range. However, data aggregates using AppMon Server time.
Last X Time Range	Data shifts by time zone difference.

Possible pitfalls with time frames

AppMon Server interprets time frames and aggregates time series. In AppMon Client, the time stamps shift via an offset when AppMon Client and server time zones differ.

AppMon charting does not handle time zones, and the first day of the week is fixed as Monday. This may result in unexpected behavior when people in different time zones visualize charts.

Charts time zone handling

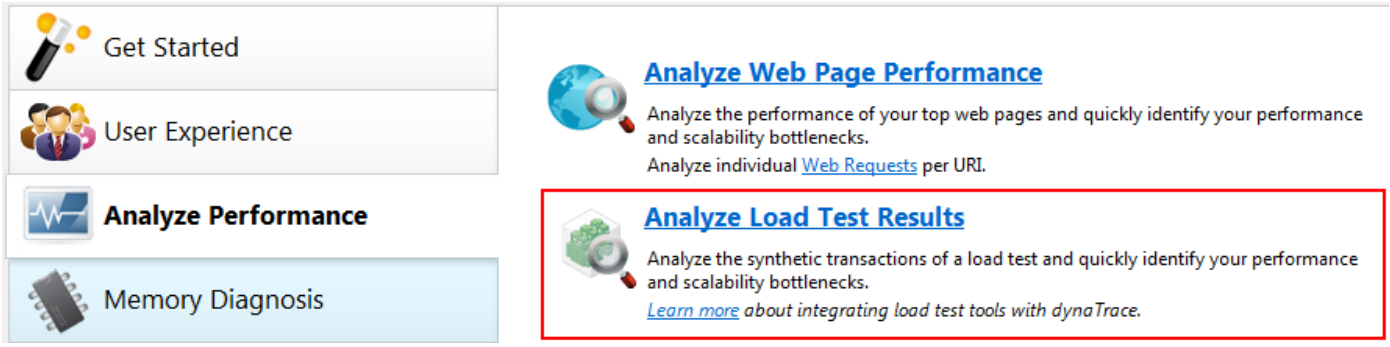
Charts time zone handling

Load Test Overview dashboard

3. Analyze the load test result.
   The top-left of the dashboard shows the **Tagged Web Requests** dashlet. All other dashlets on that dashboard are linked with that dashlet. You must select a test name (you can also multi-select). Data from the name or names you select displays in the charts and the dashlets such as Transaction Flow Dashlet, Database Dashlet, and Response Time Hotspots Dashlet. Use this to analyze the root cause of tests that run slow.

2. Select a stored session or System Profile. Typically, you have a stored session for a previously run load test. The new dashboard that opens up prompts you for the data source. Select a System Profile if you want to analyze the live session or select a stored session.

1. Launch the dashboard from the Start Center.



When you use the Load Testing Integration to run load tests, you can analyze the results that start in the **Tagged Web Requests** dashlet or use the **Load Test Overview** dashboard.

Web Page Performance Analysis dashboard

Displays the collected web request information to help diagnose and fine tune software components that handle the application's web requests like servlets and JSPs. Data shown in this tab is the same data shown in the Web Requests dashlet. See Web Requests dashlet for information on performance metrics used and analysis options.

Web Requests

Data shown in this tab is the same data shown in the **Errors** dashlet. See Errors dashlet for more information.

Shows an overview of all errors detected within a specified time frame. This tab includes an Error overview pane that shows your configured error rules, and an Error details pane that shows details for the selected error category.

Errors

Data shown in this tab is the same data shown in the **Database** dashlet. See Database dashlet for more information.

The **Database** tab provides an overview and in-depth information for database connection pool usage and SQL statement executions. The tab includes three additional tabs, each with a table that provides execution statistics. You can select which statistics to display by right-clicking any column head to display the context menu and selecting or clearing the columns.

Database

Data shown in this tab is the same data shown in the **Response Time Hotspots** dashlet. See Response Time Hotspots dashlet for more information.

The **Response Time Hotspots** tab provides a high-level view of a session to give you a quick impression of where the time is spent. You can filter Business Transactions to see how similar transactions are distributed regarding response time and where most of the time is contributed, by tier and by API.

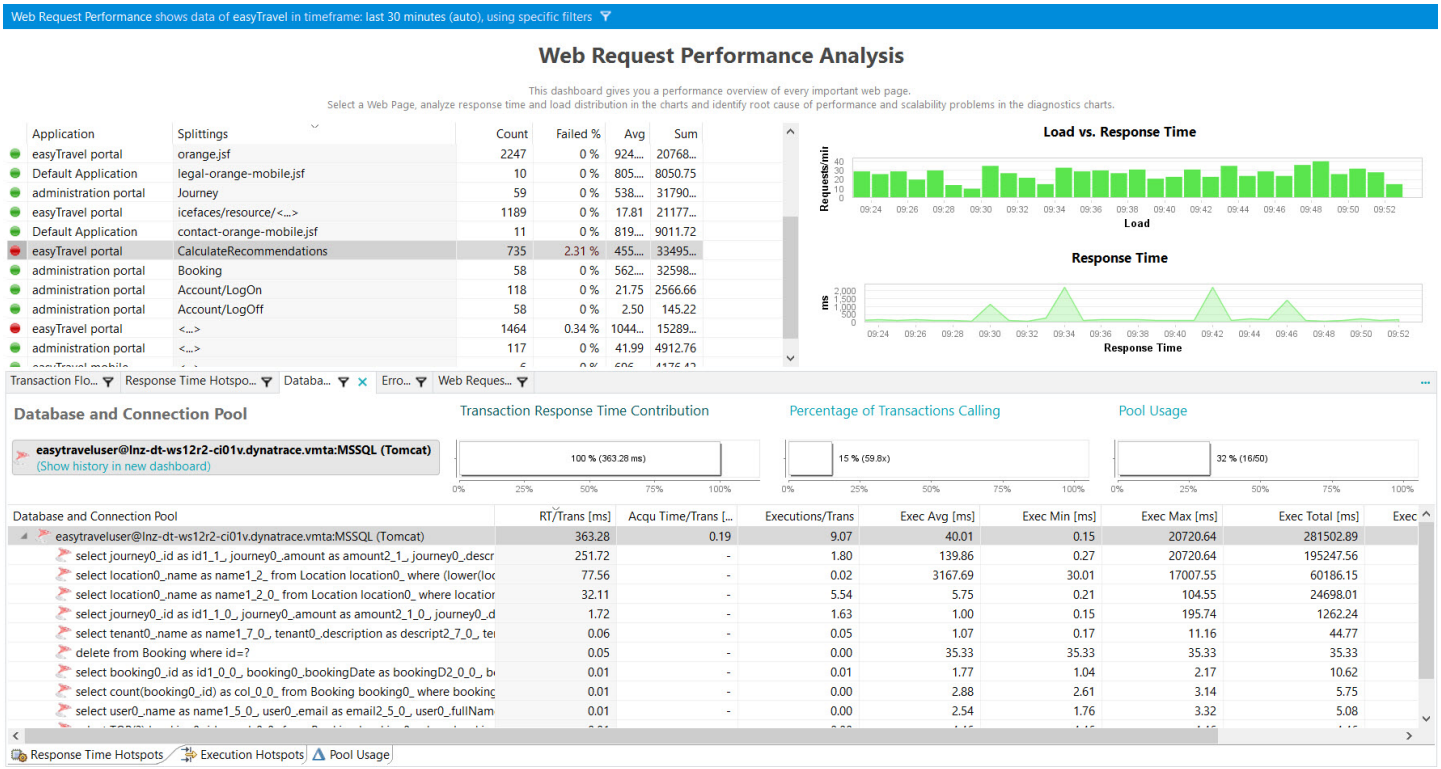
Response Time Hotspots

Data shown in this tab is the same as the data shown in the Transaction Flow Dashlet. See Transaction Flow dashlet for information on how to read the different views, understanding inter-tier communication data, and other information.

Provides different visualization modes that for different transaction detail views and levels. The Topology View mode is the default mode. Use this to analyze the basic operation. Use the Transaction Response Time view to analyze transactions.

Transaction Flow

Web Performance Analysis Dashboard



Select an application split to show the corresponding data in the **Transaction Flow, Response Time Hotspots, Database, Errors, and Web Requests** tabs, and in the lLoad vs. Response Time and Response Time tiles in the dashboard.

The master dashlet is the fixed Web Page Requests business transaction.

Use the following detail view to select an application split, analyze response time, load distribution, and identify performance and scalability issue root causes in the diagnostic charts.

To access the **Web Page Performance Analysis** dashboard, click **Start Center > Analyze**

Performance, then click the **Analyze the performance** link under **Analyze Web Request Performance**.

The **Web Request Performance Analysis** dashboard provides a performance overview of every important Web page.

APIs

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

System Profile - APIs

AppMon uses APIs to designate grouping criteria for classes and packages. Typical criteria are:

- Basic technologies such as JDBC and Servlets
- Frameworks, for example Hibernate
- Customer Application Boundaries or Subsystems such as BusinessLogic and EventProcessing.

AppMon includes a predefined set of APIs (for example, Servlet, JDBC, and JMS) that are applied to all nodes (methods) that are part of a PurePath. Nodes may have more than one API (for example, both Servlet and WebService), but for analysis reasons AppMon always chooses one API as the most significant one.

If a node does not belong to the predefined set of APIs, AppMon generates an API name from the package name of the node.

You can define APIs for selected packages through the System Profile. See System Profile - APIs for more information.

Business Transactions

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Visualization of Business Transactions

Business Transactions on stored sessions

Examples

Related pages

Business Transaction configuration

Business Transaction management walkthroughs

Business Transaction page of the Monitoring dashboard

Preventing excessive split results

Real time Business Transactions feed

Splitting options

Watch and learn

What are Business Transactions?

Advanced Business Transactions

Creating server-side Business Transactions

Creating User Action Business Transactions

Creating visit-based Business Transactions

Business Transaction Management (BTM) helps you manage IT based on your business. It tracks

technical data within business transactions across applications. End users benefit through good service quality and improved application effectiveness. BTM bridges the gap between an application's technical and business aspects.

A *business transaction* is a set of automated interactions between IT systems that execute a business task or business process. In AppMon, a business transaction is a categorization of PurePaths defined by filter or grouping parameters. When a PurePath matches those parameters, one or more evaluation metrics are calculated to create the result set of the business transaction.

You can use Business Transactions to track special transactions such as *login* and *buy* in the application under diagnosis.

For example, you can track the number and response time of a login action for a specific user group for the application. The Business Transaction consists of a filter measure that checks for the occurrence for a specific user tag, such as a tagged web request measure. The default response time and count are selected by default. The splitting measure is **application**, which is also the default.

To learn how to create and configure a business transaction, see Business Transaction configuration and Business Transactions page.

Visualization of Business Transactions

The metrics associated with Business Transactions can be used for charting. See Charting for detailed information about charting of measures.

Evaluation of Business Transactions generates or updates entries in the **Business Transaction** view, which is described in Business Transactions Dashlet. An entry is created the first time the filter evaluation of a Business Transaction is passed. Subsequent positive filter evaluations increase the count of the entry and update the displayed PurePath-related timing information.

A new entry is generated for new occurrences of measurement values of measures used for splitting, assuming that the filter criteria are fulfilled. Recurring values result in an update of the entry representing the splitting criteria. Evaluation functions also generate or update entries within the **Business Transaction** view.

Business Transactions on stored sessions

Business Transactions and the underlying measures can be reconfigured in stored and offline sessions. PurePath-related measures can be added, new thresholds can be configured, Business Transactions can be changed or created on new measures, etc.

After configuration changes are made to the System Profile, all PurePaths of the stored session have to be analyzed again to reflect the configuration changes that result in new measurements for charting and Business Transaction values. Select **Process Configuration Changes** from the context menu. The new measurements and Business Transactions are stored to the analyzed session.

Use this feature to analyze stored session data from different perspectives by using new or changed business cases.

Examples

In the resulting measurements for the examples, **boldfaced** values are processed by the Time Series Center.

Logging count

A PurePath contains different logging calls that contribute to a configured Logging Count measure.

```
+ doFoo - 10ms - MyApp
  + log -      - Log4J
  + ...
  + log -      - Log4J
  + log -      - Log4J
```

Resulting measurement:
count = 3, **sum = 3**, avg = 1, **min = 1**, **max = 1**

Web requests

A single request results in multiple nodes with attachments. For web requests and web services, *recursion detection* filters nested calls. In this example, the `doFilter` calls the `doGet`… that will be suppressed.

```
+ doFilter - 10ms - Servlet (index.htm)
  + doGet  -  8ms - Servlet (index.htm)
```

Resulting measurement:
count = 1, **sum = 10**, avg = 10, **min = 10**, **max = 10**

Business Transaction configuration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

General

Calculate Business Transaction performance

Filter

Results

Splitting

Base Line settings

Measure configuration example

Related pages

Business Transactions

Real time Business Transactions feed

Splitting options

Preventing excessive split results

You can create business transactions via the System Profile, or at the **Business Transactions** page of the **Monitoring** dashboard. This page describes editing in the System profile. To read about the **Business Transactions** page, refer to Business Transaction page of the Monitoring dashboard.

To configure a Business Transaction:

1. Open the System Profile editor, and select the **Business Transactions** item.
2. Click **Create** to create a new Business Transaction, or select a Business Transaction from the list and click **Edit**.

3. Enter information and select settings as needed, then click **OK** to save the settings. See below for parameters description

General

This section contains general settings of the business transaction:

- **Name**: The name of the business transaction. The name must be unique within a System Profile and cannot be empty.
- **Description**: Short description of the business transaction.
- **Store results in Performance Warehouse**: Enables storage of the captured transactions in the Performance Warehouse.
- **Export results via HTTP**: Enables real time Business Transaction feed to external systems for further storage and analysis. When activated, you can include the following additional data to the stream:
  - **PurePaths reference**
  - **Performance data**

Calculate Business Transaction performance

Select the entity for business transaction capturing.

Filter

Use filters to restrict the PurePaths, user actions, and visits for which the Business Transaction should be evaluated. For example, you can add a specific filter that checks for the occurrence of a specific tag on a servlet request.

The aggregation type defines how multiple values received in the same interval are aggregated.

Multiple filters can be concatenated logically. When the filter is evaluated, no operator precedence (`AND` stronger than `OR`) is applied:

- If the first `FALSE` condition is followed by an `AND` concatenation, then the complete expression evaluates to `FALSE`.

- If the first `FALSE` condition occurs after an `AND` concatenation, then the complete expression evaluates to `FALSE`.
- If the first `TRUE` condition is followed by an `OR` concatenation, then the complete expression evaluates to `TRUE`.

For example:

- `true AND false OR true` – evaluates to false
- `true OR false AND true` – evaluates to true
- `true OR false AND false` – evaluates to true
- `false AND true` – evaluates to false

Threshold configuration

Since filters are based on measures, you need to configure thresholds in the measures you are assigning for filtration.

For example:

- To include all PurePaths that last longer than three seconds, set the *lower* threshold for the PurePath duration filter to `3` seconds.
- To include all PurePaths that last less than three seconds, set the *upper* threshold for the PurePath duration filter to `3` seconds.

You can use measures that provide numeric values as underlying measures for Business Transactions without any special configuration. The only constraint is that you must define a threshold for measures that are used for filtering. Measures that deal with string values require a more sophisticated configuration. See Measure configuration example section below.

> **Note**
>
> The threshold settings of measurements are used both by the Business Transactions filter and by incidents. It is recommended that you create separate instances of measures to use in Business Transactions.

Results

You can add additional related metrics to your business transaction. Add the required measures in this pane. Each measure added to the results produces values if and only if the filter criteria from the previous pane are matched. If the filter criteria are not fullfilled, no metrics are measured. The aggregation type defines how multiple values received in the same interval are aggregated.

Splitting

The splitting can be used to segment Business Transactions according to the splitting criterion. By default, all newly created Business Transactions are split by application. The maximum number of splittings per Business Transaction is set to 5,000 by default. You can add more than one splitting criterion, e.g. split by user and by URI.

Base Line settings

Click **Violation Detection Settings** to configure the maximum deviations for the base line evaluation. In these are exceeded, a base line violation incident triggers. See Base line and smart alerting for more information.

Measure configuration example

A Business Transaction is required that filters login transactions of the user `demouser`. Login validation is performed by the method `authenticate`, which requires string parameters for username and password with the first parameter providing the username.

1. Create a measure, based on the `authenticate` method.

    1. Right-click your System Profile, and select **Edit System Profile > Measures**.
    2. Click **Create Measure**.
    3. Search for **Methods - Argument Value** metric, and select it.
    4. In the **Analyze and evaluate calls to the following methods** pane, select the **Manually specify matching pattern** checkbox.
    5. In the Class field, click the magnifying glass icon, to open the **Class Browser**.
    6. In the Class Pattern field, select the **contains** criterion, and type `login` as the search pattern.
    7. In the **Agent** field, click **Pick**, and select the **Customer Frontend** Agent in the **Agent Browser**.

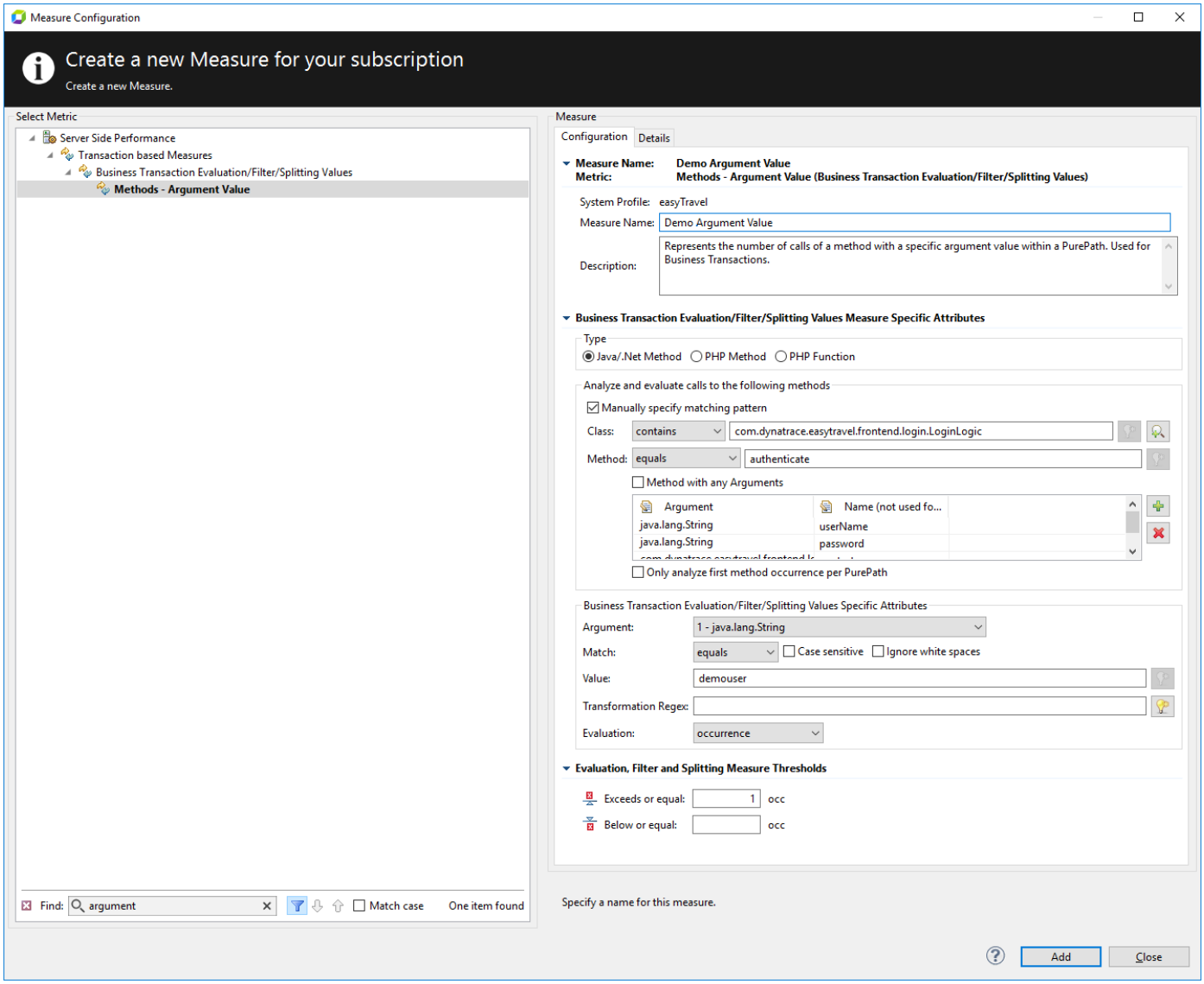
8. In the result table, expand the `com.dynatrace.easytravel.frontend.login.LoginLogic` node, and select the `authenticate` method.

2. In the **Business Transaction Evaluation/Filter/Splitting Values Specific Atttributes** pane, configure the measure to evaluate the occurrence of method calls where the value of the first parameter is `demouser`.

   1. From the **Argument** list, select **1 - java.lang.String**.
   2. From the **Match** list, select **equals**.
   3. In the **Value** editbox, type `demouser`.
   4. From the **Evaluation** list, select **occurrence**.

3. Define a threshold for the measure so it can be used for the filter. Set the **Exceed of equal** threshold as `1`, which causes a threshold violation if a method call with the parameter `demouser` occurs once within a PurePath. This is the behavior required to filter login transactions by the user `demouser`.

   Threshold violations do not automatically trigger incidents, so you don’t have to worry about it.



Configured Measure

4. Set the configured measure as a *filter* of a Business Transaction.

Business Transaction page of the Monitoring dashboard

On this page

Filter

Actions

Related pages

Business Transactions

Real time Business Transactions feed

Splitting options

Preventing excessive split results

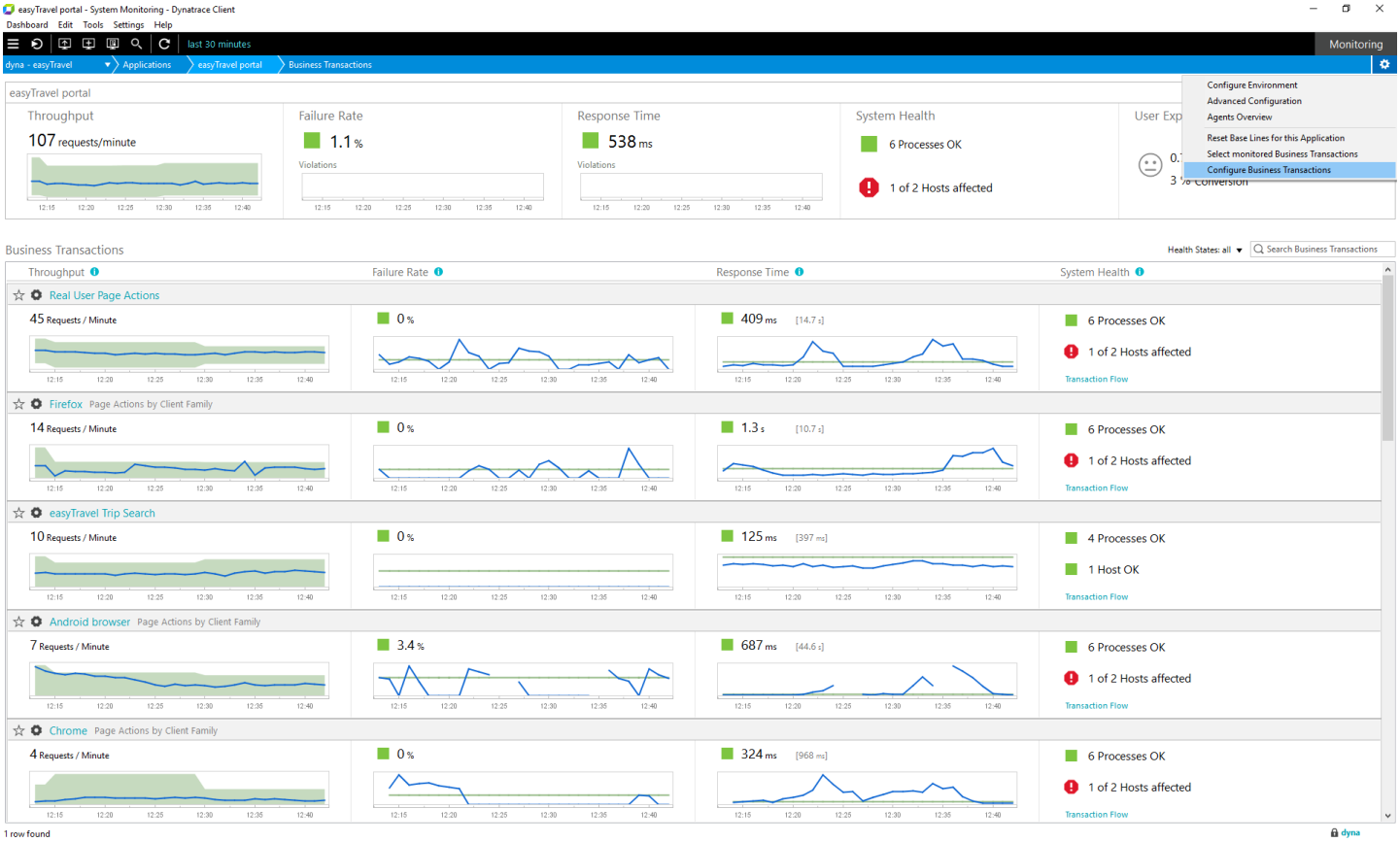
You can create business transactions via the System Profile, or an the **Business Transactions** page of the **Monitoring** dashboard. This page describes the **Business Transactions** page. To learn about the System Profile approach, refer to Business Transaction configuration.

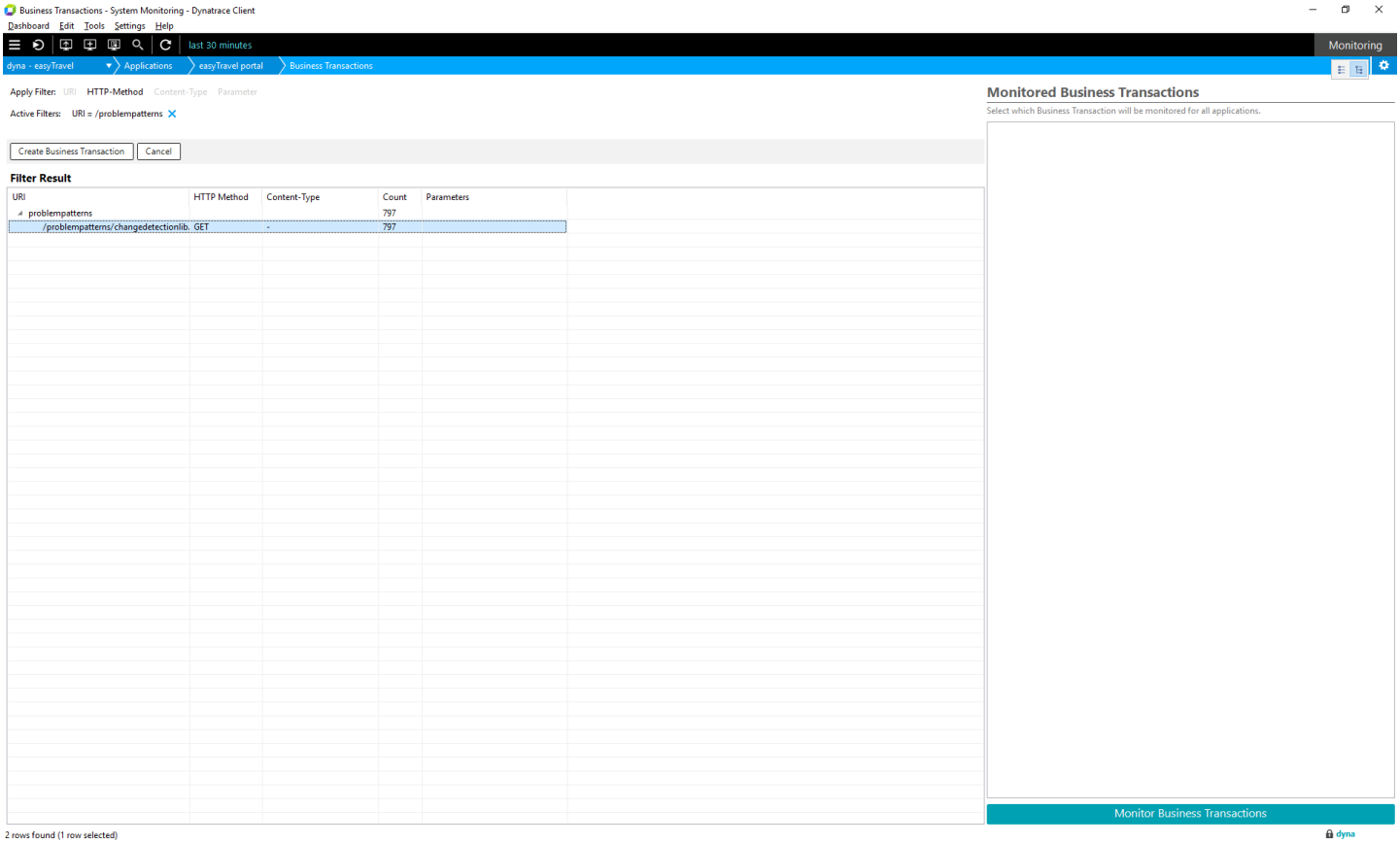
To open the **Business Transactions** page select one of your applications and select **Configure Business Transaction** item from its gear menu.

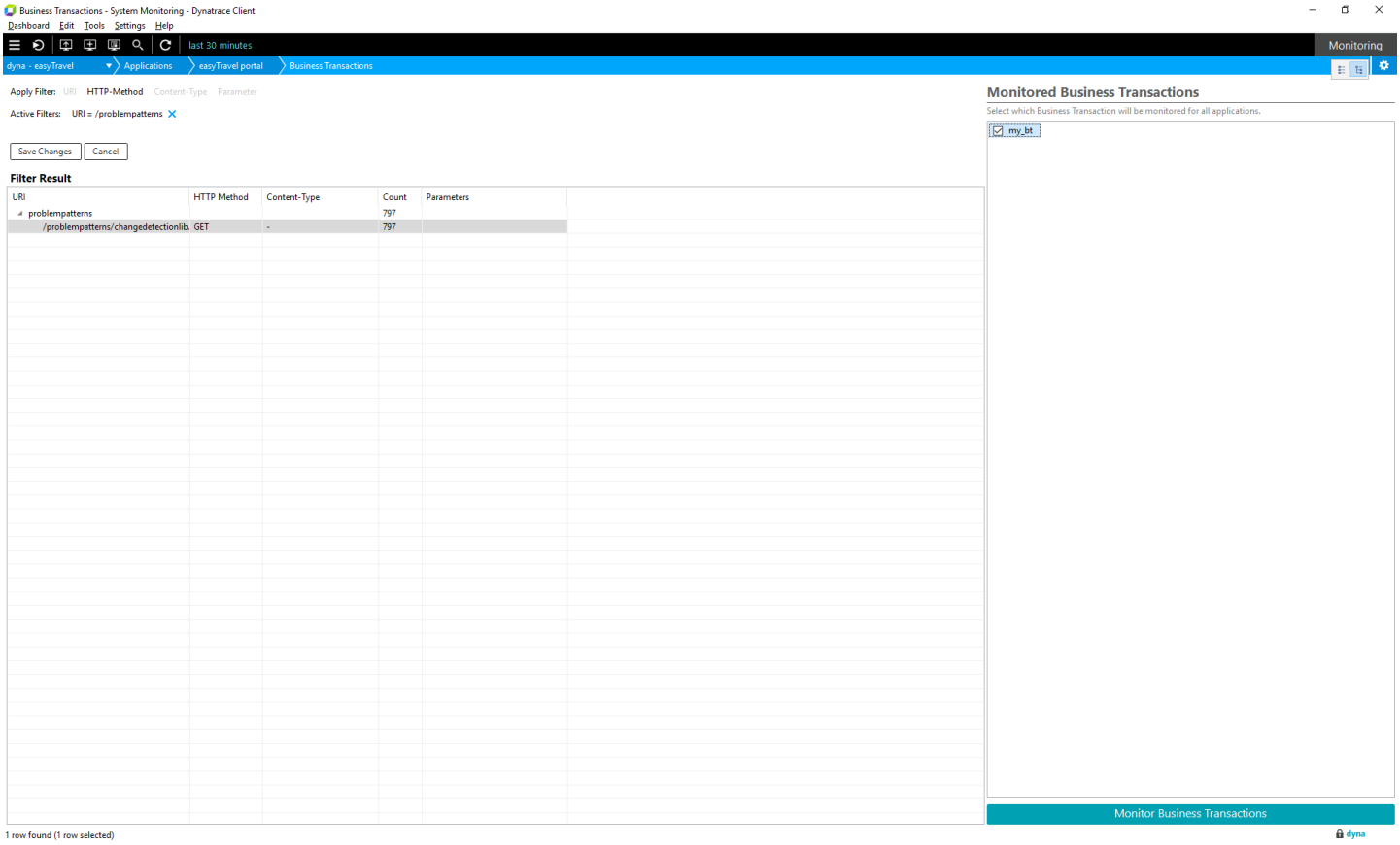
By preparing different Filters (URI, HTTP-Method, Content-Type and Parameter) a Business Transaction can be created very easily. For example HTTP GET was selected as a filter here and it is shown in the **Filter Result** section.

The active filters can be found in the **Filter Breadcrumbs** section and deleted one by one.

After all appropriate filters have been applied, a Business Transaction can be created via the **Create Business Transaction** button.


easyTravel portal - System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
last 30 minutes
Monitoring
dyna - easyTravel
Applications
easyTravel portal
Business Transactions
easyTravel portal
Throughput
107 requests/minute
Failure Rate
1.1 %
Violations
Response Time
538 ms
Violations
System Health
6 Processes OK
1 of 2 Hosts affected
Configure Environment
Advanced Configuration
Agents Overview
Reset Base Lines for this Application
Select monitored Business Transactions
Configure Business Transactions
Business Transactions
Health States: all
Search Business Transactions
Throughput
Failure Rate
Response Time
System Health
Real User Page Actions
45 Requests / Minute
0 %
409 ms [14.7 s]
6 Processes OK
1 of 2 Hosts affected
Transaction Flow
Firefox Page Actions by Client Family
14 Requests / Minute
0 %
1.3 s [10.7 s]
6 Processes OK
1 of 2 Hosts affected
Transaction Flow
easyTravel Trip Search
10 Requests / Minute
0 %
125 ms [397 ms]
4 Processes OK
1 Host OK
Transaction Flow
Android browser Page Actions by Client Family
7 Requests / Minute
3.4 %
687 ms [44.6 s]
6 Processes OK
1 of 2 Hosts affected
Transaction Flow
Chrome Page Actions by Client Family
4 Requests / Minute
0 %
324 ms [968 ms]
6 Processes OK
1 of 2 Hosts affected
Transaction Flow
1 row found
dyna


Business Transactions - System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
last 30 minutes
Monitoring
dyna - easyTravel
Applications
easyTravel portal
Business Transactions
Apply Filter: URI HTTP-Method Content-Type Parameter
Active Filters: URI = /problempatterns
Create Business Transaction
Cancel
Filter Result
URI
HTTP Method
Content-Type
Count
Parameters
problempatterns
797
/problempatterns/changedetectionlib.
GET
-
797
Monitored Business Transactions
Select which Business Transaction will be monitored for all applications.
Monitor Business Transactions
2 rows found (1 row selected)
dyna



Filter

URI-, HTTP-method- and content-type-filter are unique filters and can only be created once for each Business Transaction. The parameter-filter might be applied multiple times for different parameters. To enable the content-type filter the content-type response header needs to be captured by the Web Server Sensor. To enable the parameter filter the capturing of the request parameter has to be added to the Web Server Sensor, Servlet Sensor or ASP.NET Sensor.

Actions

Create a Business Transaction

Click the **Create Business Transaction** button to create a Business Transaction when at least one filter is active.

Edit an existing Business Transaction

When a Business Transaction is selected in the **Monitored Business Transaction** section, its active filters and the filtering results are shown. Filters can be deleted and new filters can be applied.

Delete a Business Transaction

Delete a Business Transaction by using the context menu of a BT in the **Monitored Business Transaction** section.

Monitor Business Transaction

Select a Business Transaction in the **Monitored Business Transaction** section, to make it appear at the Application Overview Dashboard. Click **Monitor Business Transactions** or the application breadcrumb to return to the application page and see the results of the newly created Business Transaction.

Preventing excessive split results

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Causes of measure explosion

Solutions

Related pages

Preventing excessive split results - FAQ

Business Transactions

Real time Business Transactions feed

Splitting options

Business Transaction configuration

Watch and learn

Preventing excessive split results

Excessive splitting results, also known as *measure explosion*, can cause the Performance Warehouse to fill up, so additional data cannot be written to the Performance Warehouse. AppMon detects misconfigured Business Transactions and disables creating measures for them, to prevent excessive data. A disabled Business Transaction has the

mark.

When a Business Transaction is disabled:

- No new charting data is generated for the Business Transaction. No new data is generated for existing splitting values.
- No new splitting entries are generated in the Business Transaction Hotspots Dashlet dashlet. Existing splitting values are not updated.
- No Incidents are triggered based on result measures of the Business Transaction.

All other diagnostic dashboards operate as normal, without any limitations.

Causes of measure explosion

A reasonably designed Business Transaction can have thousands or even tens of thousands of splits. For example:

- The World Map dashboard may have 200 to 300 regions times three satisfaction levels times the number of applications.
- If the captured URI for the metric Web Requests - URI Pattern Value contains a `Session-ID`, a very large number of unique URIs may be collected.

> **Note**
>
> A common method to transmit the `Session-ID` is to put it into the query `http://www.example.com/index?sid=`**<Session-ID>** or into the URI path `http://www.example.com/`**<Session-ID>**`/index`).

The Business Transaction configuration includes options for restricting splitting. The default is **No restriction**. When no splitting restriction is defined and the split results contain a measure like the example described above, the Business Transaction is vulnerable to a measure explosion.

Solutions

The following techniques can help to prevent a measure explosion, while preserving the intended goals of a Business Transaction.

Disable results storage in the Performance Warehouse

If the Business Transaction is not intended for charting, you can disable the results storage in the Performance Warehouse. In most cases, charting data is not needed. Charting a Business Transaction split by a URI with a `Session-ID` in particular has no purpose. To do so, un-select the **Store results in the Performance Warehouse** checkbox in the **Business Transaction Editor** dialog box.

Change splitting options

If you need to use the Business Transaction for charting, try the following solutions for measure explosion:

Reconfigure the Business Transaction

You may be able to subscribe another Split Result measure. This is possible if the measure explosion was an effect of misconfiguring the Business Transaction.

If possible, remove the explosion-triggering part from the splitting value

For all metrics that may produce unpredictably huge numbers of splitting values, it is possible to define a regular expression for transformation. When you use a regular expression, you should be able to remove the explosion-triggering part. However, using a regular expression is prone to error. It also slows down real time analysis and cannot be recommended for high-load scenarios.

Apply a splitting restriction

You can configure splitting options to restrict the number of splitting values in the Business Transaction configuration. See Reset Splittings for details.

Measure Explosion Prevention

The Measure Explosion Prevention maintains a record of the number of splitting values every Business Transaction produces. For the splittings held in Server memory, the exact numbers are constantly updated. In the Performance Warehouse, the numbers are updated once per hour.

If the number of splittings in the Server memory or in the Performance Warehouse exceeds the default limit of 50,000, the Measure Explosion Prevention disables creating measures for the affected Business Transaction.

Preventing excessive split results - FAQ

On this page

My Business Transaction creates a bounded set of splitting values, but creating measures is disabled by the Measure Explosion Prevention

I created my Business Transaction with a release earlier than AppMon 5. Is this Business Transaction affected by the splitting limit?

My Business Transaction exceeded the default limit of splittings. I performed the steps to override the default limit but now the AppMon server does not start anymore. How do I fix this problem?

Related pages

Business Transactions

Real time Business Transactions feed

Splitting options

Business Transaction configuration

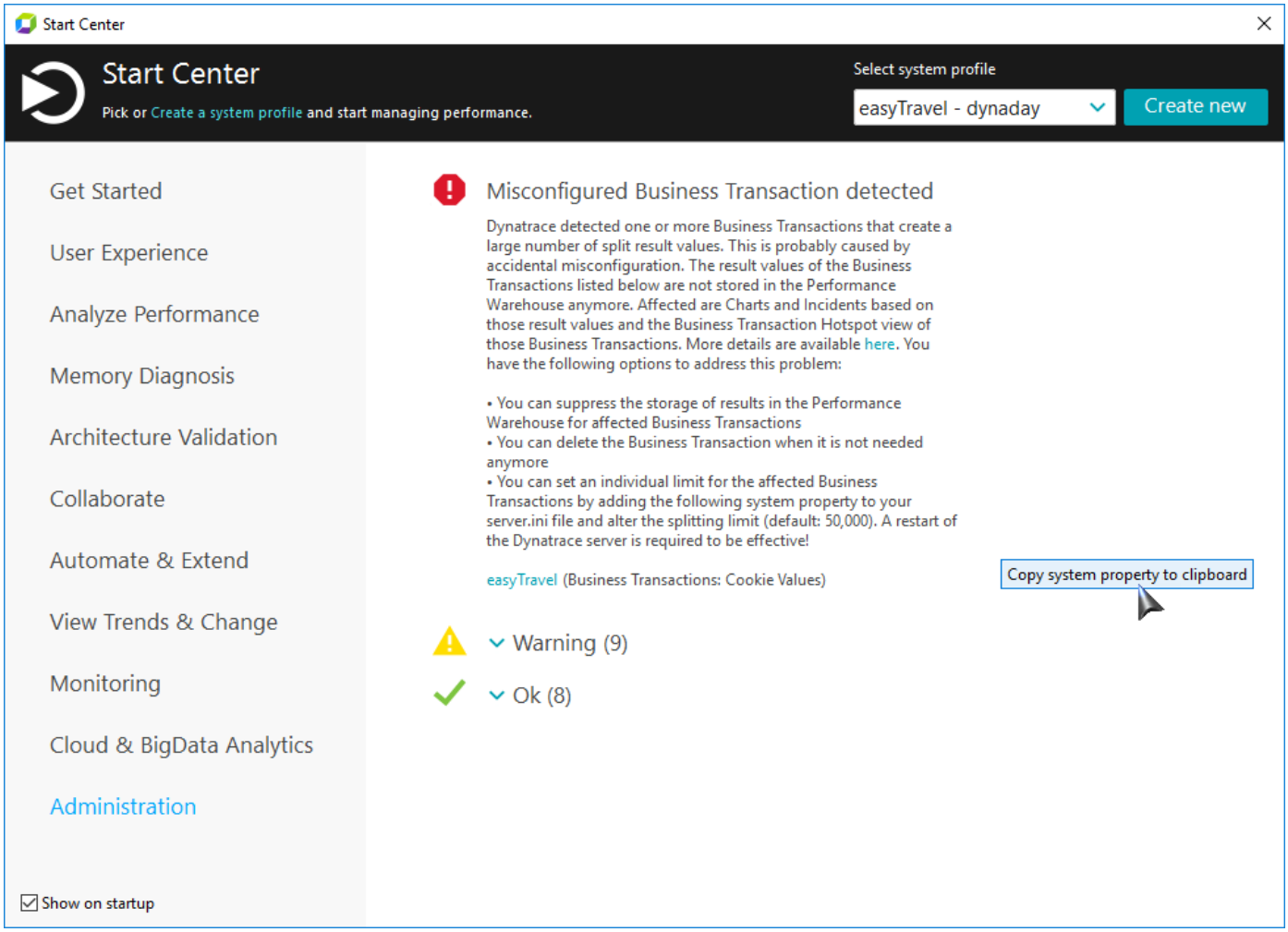
Watch and learn

Preventing excessive split results

My Business Transaction creates a bounded set of splitting values, but creating measures is disabled by the Measure Explosion Prevention

You can set a different limit for each system property. Depending on the scaling of your Performance Warehouse, a higher limit may be manageable. To set an individual limit for the amount of splittings for your Business Transactions, perform the following steps;

1. Click the **Copy system property to clipboard** button located in the Start Centers Administration section, as shown.



2. Open the `dtserver.ini` file, located in the home directory of the AppMon Server, in your preferred text editor.

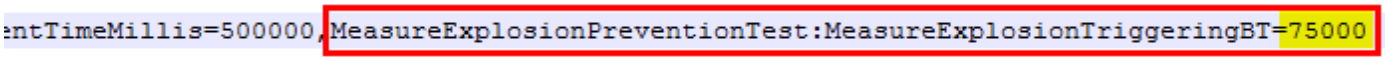
3. Paste the following code above the `-server` section in `dtserver.ini`.

```
-Dcom.dynatrace.diagnostics.server.OverruleMaxSplittingsBTs=MeasureExplosionPreventior
```

```
-Duser.country=US
-Duser.language=en
-Dcom.dynatrace.diagnostics.server.OverruleMaxSplittingsBTs=MeasureExplosionPreventionTest:ManySplittingsCurrentTimeMillis=50000,Me
-server
```

4. Adapt the individual splitting limits to your desired upper bound. If a Business Transaction exceeds the specified limit, it will be disabled.

```
...
-Dcom.dynatrace.diagnostics.server.OverruleMaxSplittingsBTs=MeasureExplosionPreventior
-server
```



5. Save and close the `dtserver.ini` file.
6. Restart the AppMon Server.

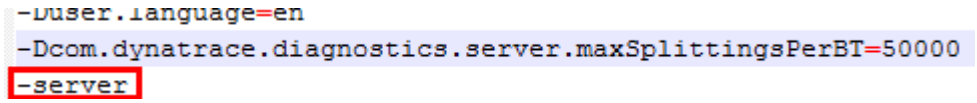
Contact support if you need further help.

I created my Business Transaction with a release earlier than AppMon 5. Is this Business Transaction affected by the splitting limit?

The Measure Explosion Prevention is active for all Business Transactions. The default limit is 50000. To set a new global splitting limit perform the following steps.

1. Open the `dtserver.ini` file, located in the home directory of the AppMon Server, in your text editor.
2. Paste the following line above the `-server` section in `dtserver.ini`.

```
-Dcom.dynatrace.diagnostics.server.maxSplittingsPerBT=50000
```



3. Adapt the limit to your needs. For example, you might increase it to 75000.

```
...
-Dcom.dynatrace.diagnostics.server.maxSplittingsPerBT=75000
-server
```

```
-Duser.language=en
-Dcom.dynatrace.diagnostics.server.maxSplittingsPerBT=75000
-server
```

4. Save and close the `dtserver.ini` file.

5. Restart the AppMon Server.

Contact support if you need further help.

My Business Transaction exceeded the default limit of splittings. I performed the steps to override the default limit but now the AppMon server does not start anymore. How do I fix this problem?

If the name of your Business Transaction or System Profile contains characters that are not defined in the ISO-8859-1 character set, change the name to eliminate the invalid characters: don't use accented characters or special characters.

You can also change the splitting limit for **all** Business Transactions instead of changing individual settings.

Use the following code in your `dtserver.ini` to change all Business Transactions:

```
-Dcom.dynatrace.diagnostics.server.maxSplittingsPerBT=<maxNumberOfSplittings>
```

Instead of the code for individual settings:

```
-Dcom.dynatrace.diagnostics.server.OverruleMaxSplittingsBTs=<SystemProfileName>:<Business
```

For details, see the procedures above.

Splitting options

Looking for the 6.5 version of this page? Open in 6.5 documentation

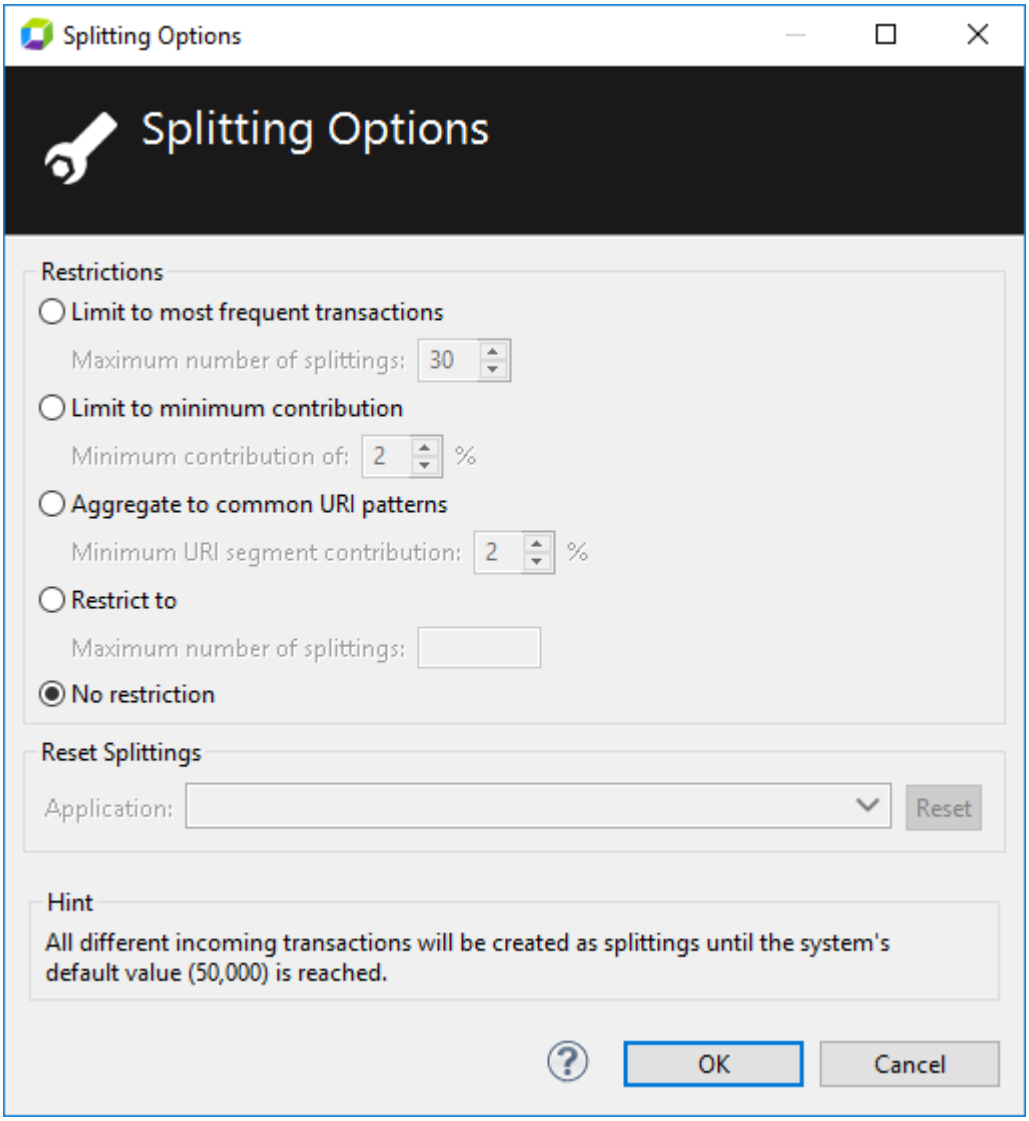
On this page

Restrictions

Reset splittings

Use the **Splitting Options** dialog box to define how to limit the number of splitting values for a Business Transaction. Setting a maximum number of splittings helps to keep the results comprehensible and protect your AppMon environment from overload. To access the dialog box, in the the **Business Transaction Editor** dialog box, click **Splitting Options**.

You can also reset the current set of splitting values for any application configured in your System Profile.



Splitting options dialog

Restrictions

When you configure a splitting restriction, the splitting values of an application are determined based on an initial set of transactions. This set is one of the following, depending on the load:

- All transactions in an observation period of one minute.
- The first 500 transactions that are collected, if fewer than 500 transactions occur within one minute.

With a sufficient load, there can be more than 500 transactions in the set. The final splitting values are calculated by taking the contribution of every splitting value of a transaction into account. Splitting values with a small contribution are mapped to a remainder splitting value.

Limit to most frequent transactions

From the initial set of transactions, only the *X* most frequently observed splitting values establish a splitting. Other splitting values are mapped to the remainder splitting value `<...>`. The upper limit, *X*, can be 5 through 100.

Limit to minimum contribution

From the initial set of transactions, the observed splitting values with a contribution of *X* or larger establish a splitting. Other splitting values are mapped to the remainder splitting value `<...>`. The minimum contribution, *X*, can be 1% through 50%.

Aggregate to common URI patterns

The observed splitting values in the initial set of transactions are assumed to contain URIs. The segments are delimited by the slash (`/`) character. The minimum contribution of a segment can be 1% through 10%. Segment contributions that are less than the defined minimum are mapped to the `<...>` segment.

For example, the initial set of transactions contains 500 items. For this Business Transaction, the minimum segment contribution is 1%. So at least five transactions of the same segment must occur to get a distinct splitting.

The table illustrates the results of five transactions: four involving `images/**ex1**/` and one involving `images/**ex2**/`.

URI	Resulting Splitting Values	Reason
`images/**ex1**/`	(none)	
`images/**ex1**/`	(none)	
`images/**ex1**/`	(none)	
`images/**ex1**/`	(none)	Segment `ex1` occurs only four times and therefore does not reach the minimum contribution.
`images/**ex2**/`	images/**<...>**/	Segment `images` occurs five times and therefore reaches the minimum contribution.

Note

The splitting measure source can be a method argument or return value. This restriction can be applied to any input separated by slashes (`/`).

Restrict to

This setting specifies the maximum number of splittings for the Business Transaction. The value can be 100 through 50,000.

No restriction

When **No restriction** is selected, there is no observation period for limiting splitting values. Every distinct splitting value in a Business Transaction establishes a new splitting value or measure.

Note

If you select **No restrictions** for a URI splitting measure and your application uses a `sessionID` parameter, the number of splitting values can increase rapidly and fill up the Performance Warehouse. See Preventing Excessive Split Results for information on avoiding “measure explosion”.

Business Transactions created in AppMon 4.2 or earlier have the default setting of **No restrictions**. They are not automatically migrated to restrict splitting results.

For a new Business Transaction, the default is to split by the 30 most frequent transactions. When adding the splitting measures **Complete URI Path**, the restriction is set to **Aggregate to common URI patterns** with a minimum contribution of 5%.

Reset splittings

If a splitting restriction is active for a Business Transaction, you can reset the calculated splitting values separately for every configured application. After a reset, the splitting values determination process — the initial observation period — is started automatically for the application.

A reset can be helpful if an atypical set of transactions was recorded during the observation period. For example, the observation period included maintenance transactions, but you want to observe only user-generated transactions.

To perform a reset, select the required application from the list, and click **Reset**.

Real time Business Transactions feed

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Message format

Basic setup

Business Transaction-specific setup

Related pages

Business Transactions

Splitting options

Preventing excessive split results

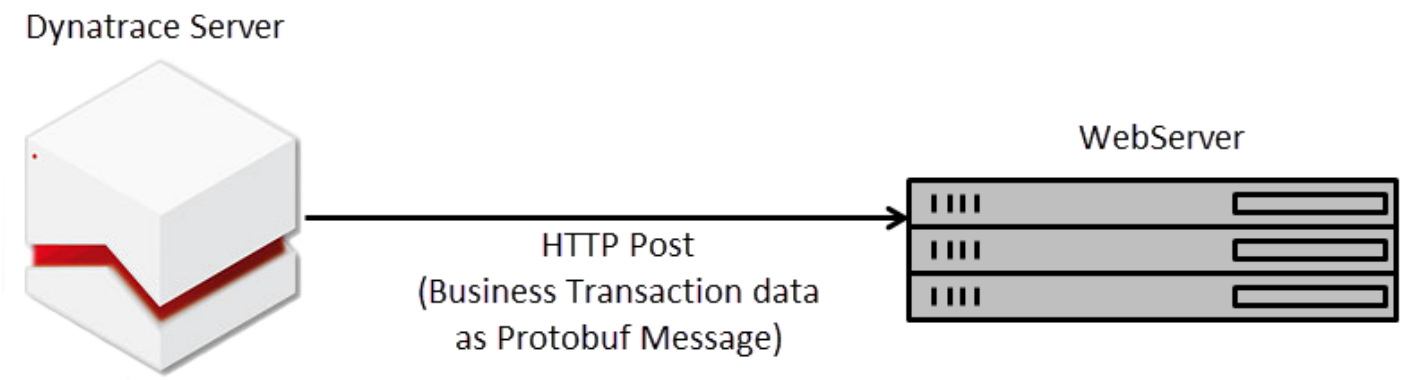
Watch and learn

Setting up Business Transactions feed

AppMon can send Business Transaction data in real time to external systems for further storage and analysis. The data feed contains Business Transaction data, including evaluation and splitting values, for single transactions. This means the Business Transaction feed does not aggregate the data, but streams out discrete transaction data.

For Business Transactions based on PurePaths and user actions, it's possible to include the PurePath reference and PurePath performance data. The exported data is encoded using Google Protobuf, resulting in an efficient, flexible and future proof interface.
See Big Data Business Transaction Bridge and Splunk Application for ready-to-use examples of how to integrate the Business Transaction Feed into your Big Data environment. Also, compare this to the PureLytics Stream to ElasticSearch and Kibana.

The following figure shows the data feed architecture.



Business Transaction Export

Message format

The Business Transactions are sent in the following Protobuf message format, using HTTP post requests. This format can be used to generate code for various programming languages such as Java, C, and C++.

```
syntax = "proto2";

option java_package = "com.dynatrace.diagnostics.core.realtime.export";

option optimize_for = SPEED;

package export.bt;

message BusinessTransactions {
        repeated BusinessTransaction businessTransactions = 1;          //exported busine
        optional int32 lostTransactions = 2;            //number of lost transactions, nc
        optional string serverName = 3;                         //name of dynatrace serve
}
message BusinessTransaction {
        required string name = 1;                                       //name of busines
        enum Type {                                                                     /
                PUREPATH = 0;
                USER_ACTION = 1;
                VISIT = 2;
```

```
                visit = 2;
        }
        optional Type type = 2;                                    //type of busines
        optional string application = 3;                        //application of business
        repeated string measureNames = 4;                       //result measure names
        repeated string dimensionNames = 5;                     //splitting measure names
        repeated BtOccurrence occurrences = 6;          //bt results for purepaths/user a
        optional string systemProfile = 7;                      //system profile the busi
}
message BtOccurrence {
        required int64 startTime = 1;                           //java time stamp; start
        optional int64 endTime = 2;                                //java time stamp
        optional string purePathId = 5;                         //id of purepath; e.g. "P
        repeated double values = 6;                                //values of resul
        repeated string dimensions = 7;                         //splittings
        optional string actionName = 8;                         //name of user action
        optional string url = 9;                                   //url of user act
        optional string query = 10;                             //not used currently
        optional int64 visitId = 11;                            //id of visit
        optional string visitTag = 12;                          //tag of visit
        optional double userExperienceIndex = 13;       //user experience index of visit
        optional bool converted = 14;                              //visit converted
        optional double responseTime = 15;                      //ms; response time of (u
        optional double cpuTime = 16;                           //ms; cpu time of (user a
        optional double syncTime = 17;                          //ms; sync time of (user
        optional double waitTime = 18;                          //ms; wait time of (user
        optional double suspensionTime = 19;            //ms; suspension time of (user ac
        optional double execTime = 20;                          //ms; execution time of (
        optional double duration = 21;                          //ms; duration of (user a
        optional bool failed = 22;                                 //purepath/user a

        optional int32 nrOfActions = 23;                        //number of user actions
        optional string clientFamily = 24;                      //client family of visit
        optional string clientIP = 25;                          //client ip address of vi
        optional string continent = 26;                         //continent of visit
        optional string country = 27;                           //country of visit
        optional string city = 28;                                 //city of visit
        optional int32 failedActions = 29;                      //number of failed user a
        optional int32 clientErrors = 30;                       //number of client errors
        optional bool exitActionFailed = 31;            //last action of visit failed
        optional bool bounce = 32;                                 //bounce visit (i
```

```
        optional string osFamily = 33;                          //os family of visit (e.g
        optional string osName = 34;                            //os name of visit (e.g.
        optional string connectionType = 35;                //connection type of visit
        repeated string convertedBy = 36;                       //names of visit conversi

        optional double clientTime = 37;                        //ms; client time of user
        optional double networkTime = 38;                       //ms; network time of use
        optional double serverTime = 39;                        //ms; server time of user
        optional int32 urlRedirectionTime = 40;             //ms; url redirection duration of
        optional int32 dnsTime = 41;                            //ms; dns duration of use
        optional int32 connectTime = 42;                        //ms; connect duration of
        optional int32 sslTime = 43;                            //ms; ssl duration of use
        optional int32 documentRequestTime = 44;            //ms; document request duration c
        optional int32 documentResponseTime = 45;           //ms; document response duration
        optional int32 processingTime = 46;                     //ms; processing duration
}
```

Each message can include multiple different Business Transactions. The `BusinessTransaction` structure groups the occurrences for a Business Transaction. It contains the defined evaluation measure names and dimension names (splitting measure names). Each `BtOccurrence` instance corresponds to a single PurePath, user action, or visit that was evaluated for the specific Business Transaction. Which values are set in the `BtOccurrence` depends on the type of the Business Transaction and the configuration:

	PurePath	User Action	Visit
startTime	✓	✓	✓
endTime	✗	✗	✓
duration	✓	✓	✗
failed	✓	✓	✗
purePathId	optional	optional	✗

values	✓	✓	✓
dimensions	✓	✓	✓
actionName	✗	✓	✗
url	✗	✓	✗
query	✗	✗	✗
visitId	✗	✓	✓
user	✗	✗	✓
apdex	✗	✗	✓
converted	✗	✗	✓
responseTime	optional	optional	✗
cpuTime	optional	optional	✗
syncTime	optional	optional	✗
waitTime	optional	optional	✗
suspensionTime	optional	optional	✗
execTime	optional	optional	✗

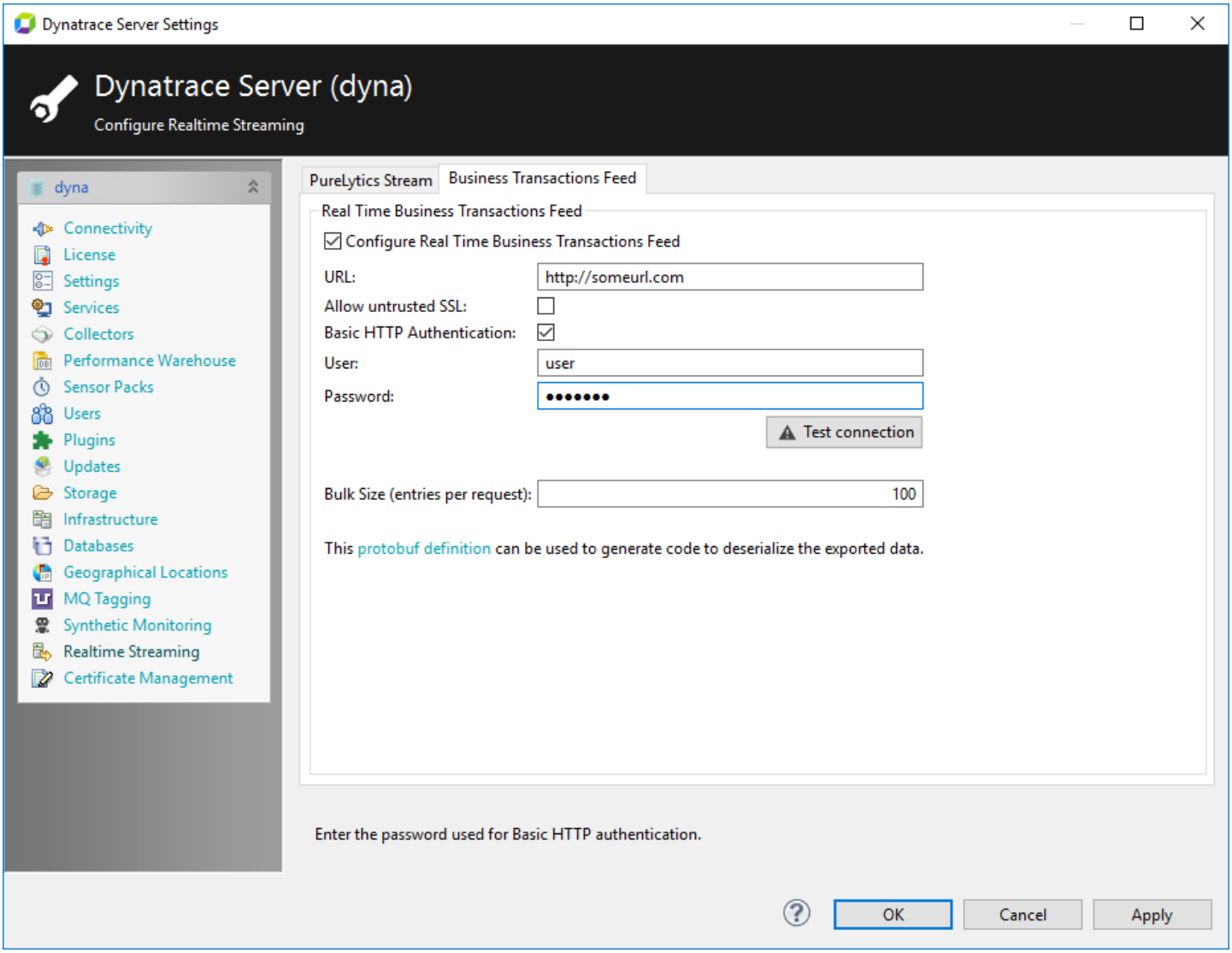
nrOfActions	✗	✗	✓
clientFamily	✗	✗	✓
clientIP	✗	✗	✓
continent	✗	✗	✓
country	✗	✗	✓
city	✗	✗	✓
failedActions	✗	✗	✓
clientErrors	✗	✓	✓
exitActionFailed	✗	✗	✓
bounce	✗	✗	✓
osFamily	✗	✗	✓
osName	✗	✗	✓
connectionType	✗	✗	✓
convertedBy	✗	✗	✓
clientTime	✗	✓	✗

networkTime	✗	✓	✗
serverTime	✗	✓	✗
urlRedirectionTime	✗	✓	✗
dnsTime	✗	✓	✗
connectTime	✗	✓	✗
sslTime	✗	✓	✗
documentRequestTime	✗	✓	✗
documentResponseTime	✗	✓	✗
processingTime	✗	✓	✗

The configuration of the Business Transaction feed is split into a basic setup that configures the global export settings, and a Business Transaction-specific configuration that allows the export to be enabled for each Business Transaction.

Basic setup

Perform the basic setup for the Business Transaction export in the **Server Settings > Realtime Streaming > Business Transactions Feed**.



Configure Business Transactions Feed

Select **Enable Real Time Business Transactions Feed**, then provide the required information for the feed.

- **URL**: Specify the URL to which the Business Transaction data will be posted.
- **Allow untrusted SSL**: Export to servers without checking the certificate.
- **HTTP Authentication**: Enable basic authentication
- **Test connection**: Validate the connection settings.
- **Bulk Size**: Set the number of Business Transaction results to be included in one request. There is a timeout of one minute, so a request might actually contain fewer Business Transaction results than specified here.

Business Transaction-specific setup

Enable data export for a specific Business Transaction through the **Business Transaction Editor**. Go to **System Profile Preferences > Business Transactions**, then select the Business Transaction and click **Edit**; or click **Create** for a new Business Transaction.

Select **Export Results via HTTP**. Two additional check boxes appear. Select the additional data to export: **PurePath reference** and/or **Performance data** (e.g. CPU or suspension time).

Business Transactions created for multidimensional analysis can contain many different splittings or split measures that result in many “groups”, possibly with single values, e.g. a transaction ID. In this situation, make sure **Store results in Performance Warehouse** is not selected. The data would be unsuitable for charting and would only burden your Performance Warehouse unnecessarily. Also remove any splitting option restrictions, to collect all values.

Business Transaction management walkthroughs

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related topics

Create a splitting Business Transaction

Follow your transactions with Transaction Flow

How to use Business Transactions

The following walkthroughs are available:

- How to use Business Transactions
- Follow your transactions with Transaction Flow
- Create a splitting Business Transaction

How to use Business Transactions

On this page

The main features of Business Transactions

Goal of this walkthrough

Creating a basic Business Transaction for web requests

Add additional return values

Conclusion

Related pages

Create a splitting Business Transaction

Follow your transactions with Transaction Flow

Watch and learn

What are Business Transactions?

Business Transaction Management Webinar

Creating server-side Business Transactions

Creating User Action Business Transactions

Creating visit-based Business Transactions

The main features of Business Transactions

The following is a list of the significant business transactions features:

- Operates on server-side PurePaths
- Evaluates user actions
- Filters and splits

- Limits split values to significant values only with Smart Splitting
- Baselines default results response time count and failure rate
- Includes numerous default business transactions
- Includes configurable measures storage in performance warehouse

Goal of this walkthrough

This walkthrough demonstrates some of the above features. You'll create new business transactions based on the default transactions. These transactions use filters to get meaningful and representative information from all web requests. You'll also group by various means and, finally, create a few charts for your measures.

Scenario

Use the easyTravel demo application to learn about various AppMon features and functions. Or you can use any standard web application because this scenario works with simple web requests. The walkthrough includes the following steps:

1. Create a basic business transaction for web requests.
2. Add additional return values.
3. Understand synthetic vs. real web requests.

Requirements

Run the demo application with injected agents and connect to a System Profile. If you use your own application, associate and activate the necessary web server Sensor.

Creating a basic Business Transaction for web requests

Enable the web request sensor in your System Profile to see the single requests to your application server. Your application may request single resources such as style sheets and images. Typically, you won't need this static content, or requests from bots or spiders, since they distort the necessary data.

Follow these steps to create a business transaction that filters out these requests:

1. Right click your System Profile and select **Edit System Profile > Business Transaction**.
2. Click **Create** to create a new business transaction.

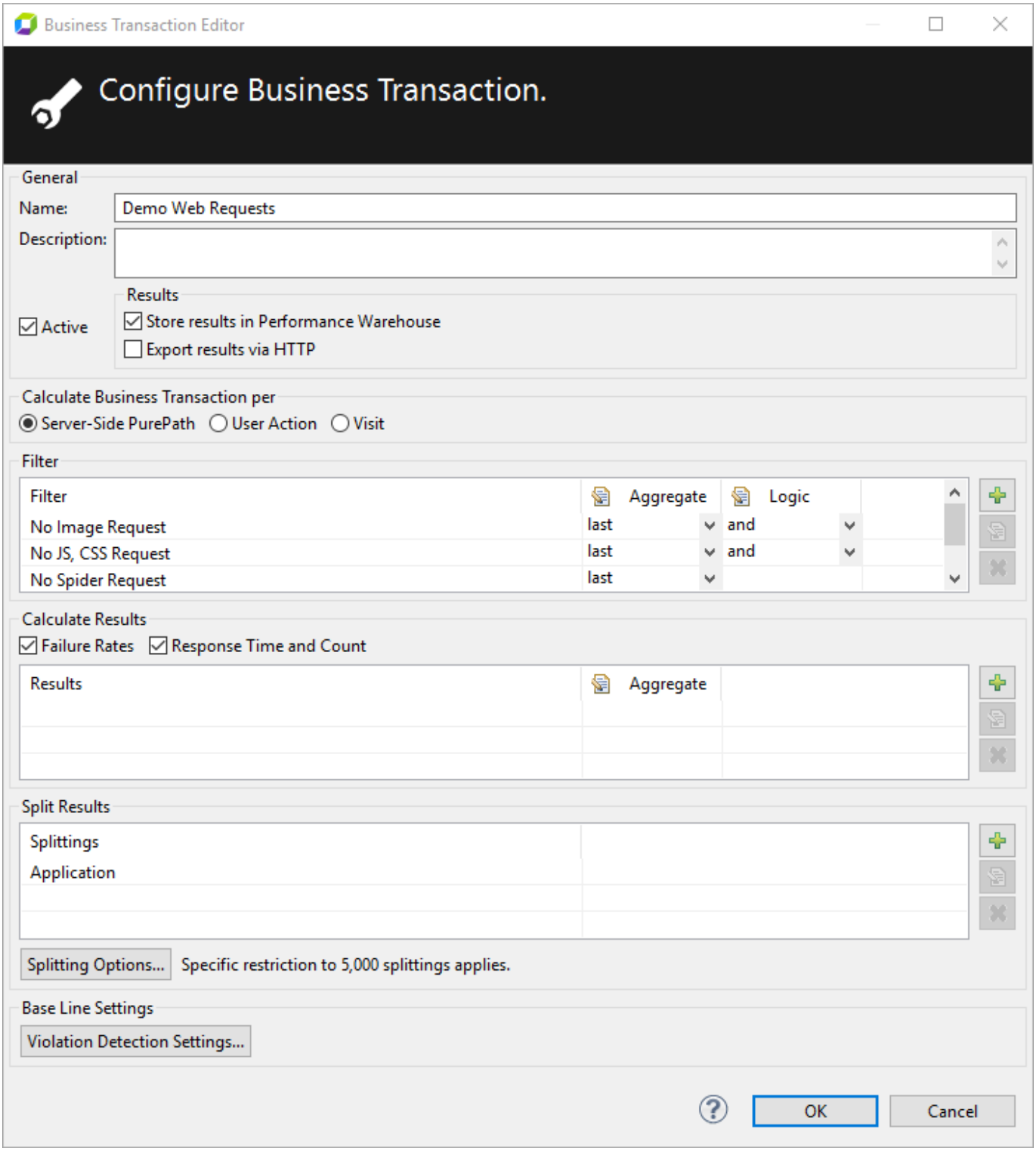
3. Give it a name, such as **Demo Web Requests**.
4. To return only relevant requests, add the following predefined filters from the **Transaction Based Measures > Web Requests** group:
   - **No JS, CSS Requests:** This uses a negation regular expression for URIs containing `.js` and `.css` files.
   - **No Image Requests**: This uses a negation regular expression for URIs containing `.jpg`, `.jpeg`, `.gif`, `.png`, and `.ico` files.
5. Add an advanced filter from the Business Transaction Evaluation/Filter/Splitting Values group: **No Spider Request:** This looks at the web request's header field *User Agent* and uses a regular expression.

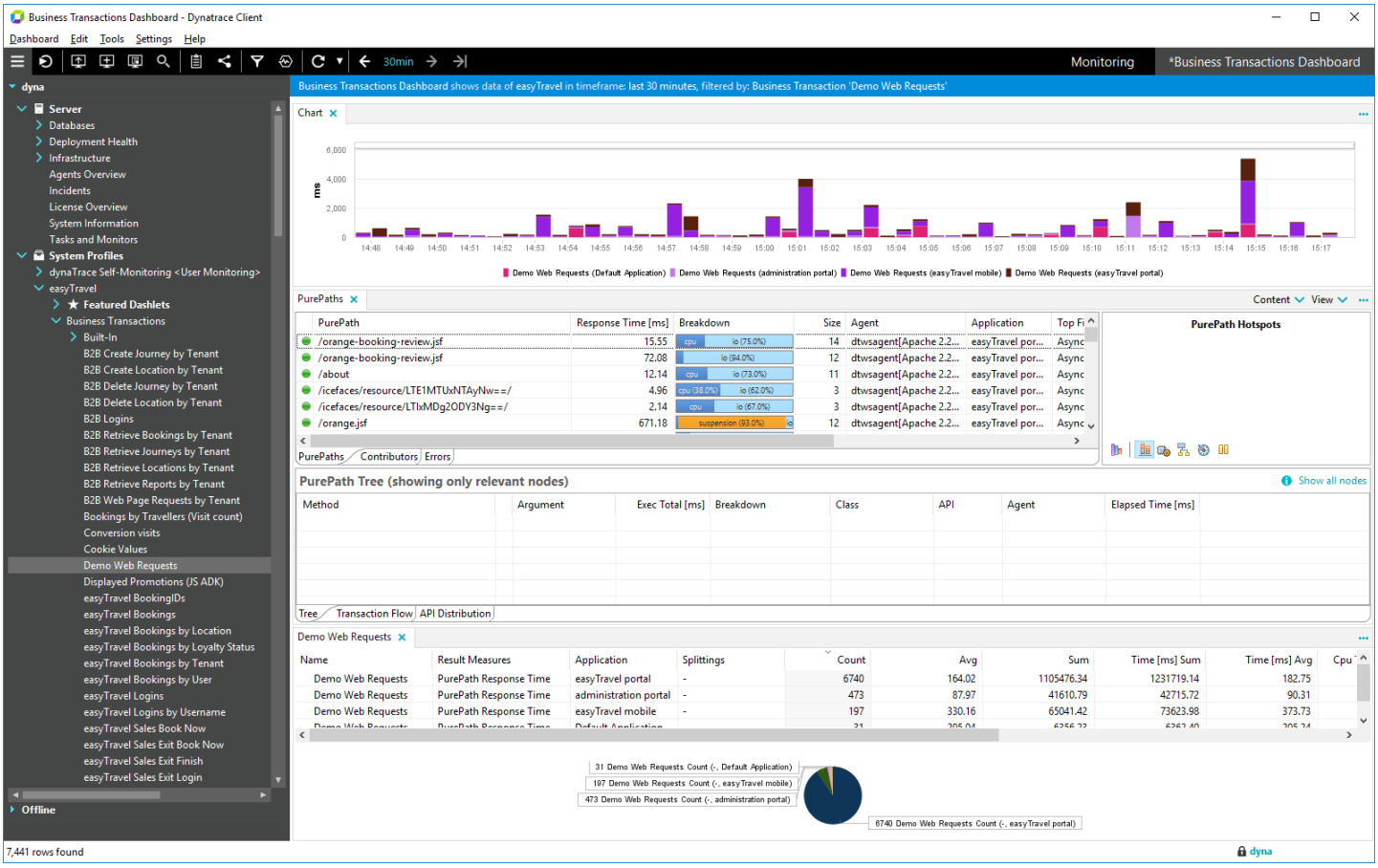
> **Note**
>
> The No Spider Request filter's regular expression may not exclude each bot and spider. If your site gets crawled by bots or spiders, adapt the filter or create additional filters. See http://www.user-agents.org for a list of user agent strings.

The result is the average response time (the PurePath duration) for the web requests. Use this business transaction to configure further, and chart this measure for an overview.

The image below shows the business transaction configuration and a dashboard that includes the following:

- The overall web requests that match the **Demo Web Requests** business transaction.
- The matching PurePaths.
- A simple chart with the average response time over time.


Business Transaction Editor
Configure Business Transaction.
General
Name:
Demo Web Requests
Description:
Active
Results
Store results in Performance Warehouse
Export results via HTTP
Calculate Business Transaction per
Server-Side PurePath
User Action
Visit
Filter
Filter
Aggregate
Logic
No Image Request
last
and
No JS, CSS Request
last
and
No Spider Request
last
Calculate Results
Failure Rates
Response Time and Count
Results
Aggregate
Split Results
Splittings
Application
Splitting Options...
Specific restriction to 5,000 splittings applies.
Base Line Settings
Violation Detection Settings...
OK
Cancel



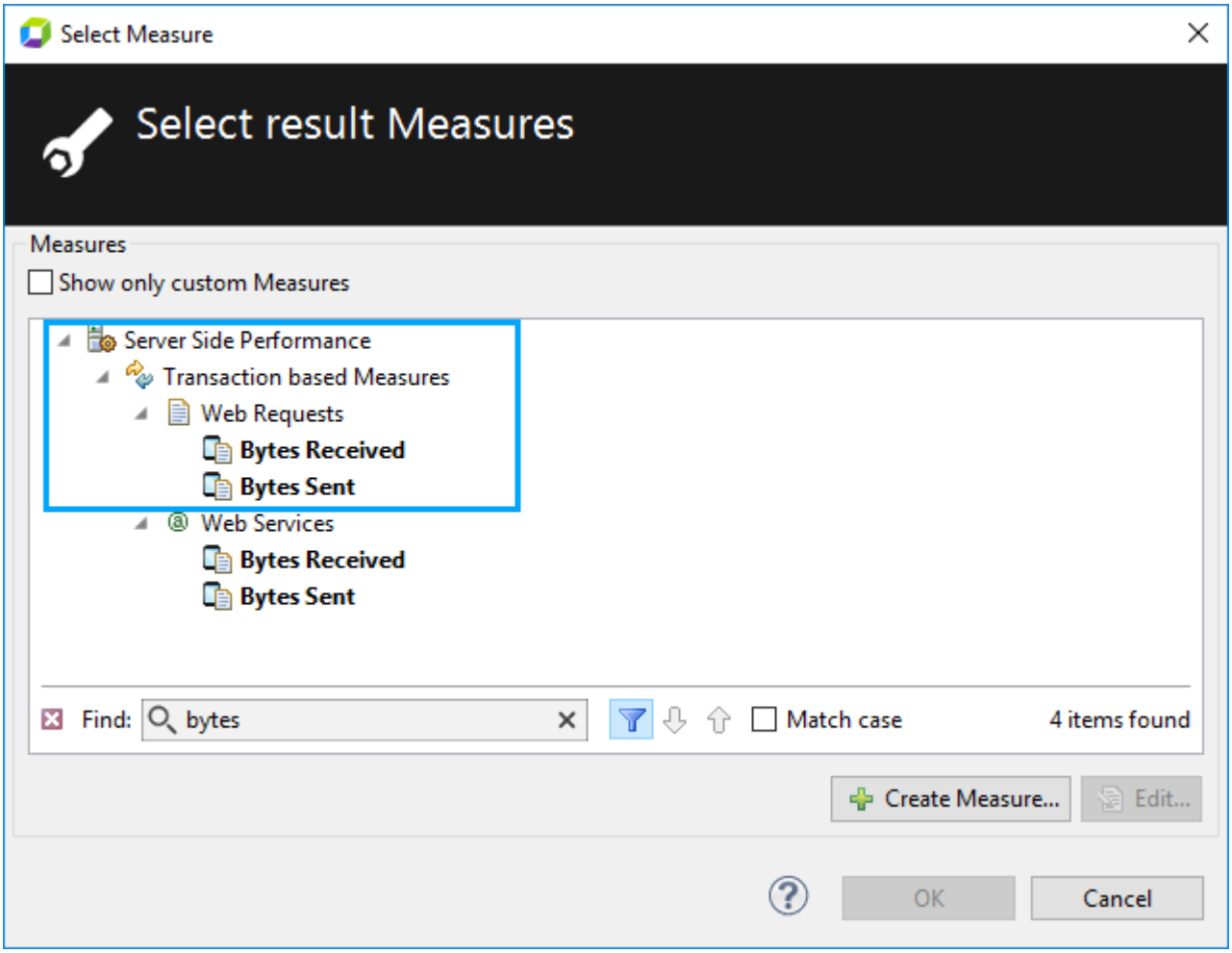
The average response time varies because there are different requests with different execution and response times. You should group and separate (split) different web requests.

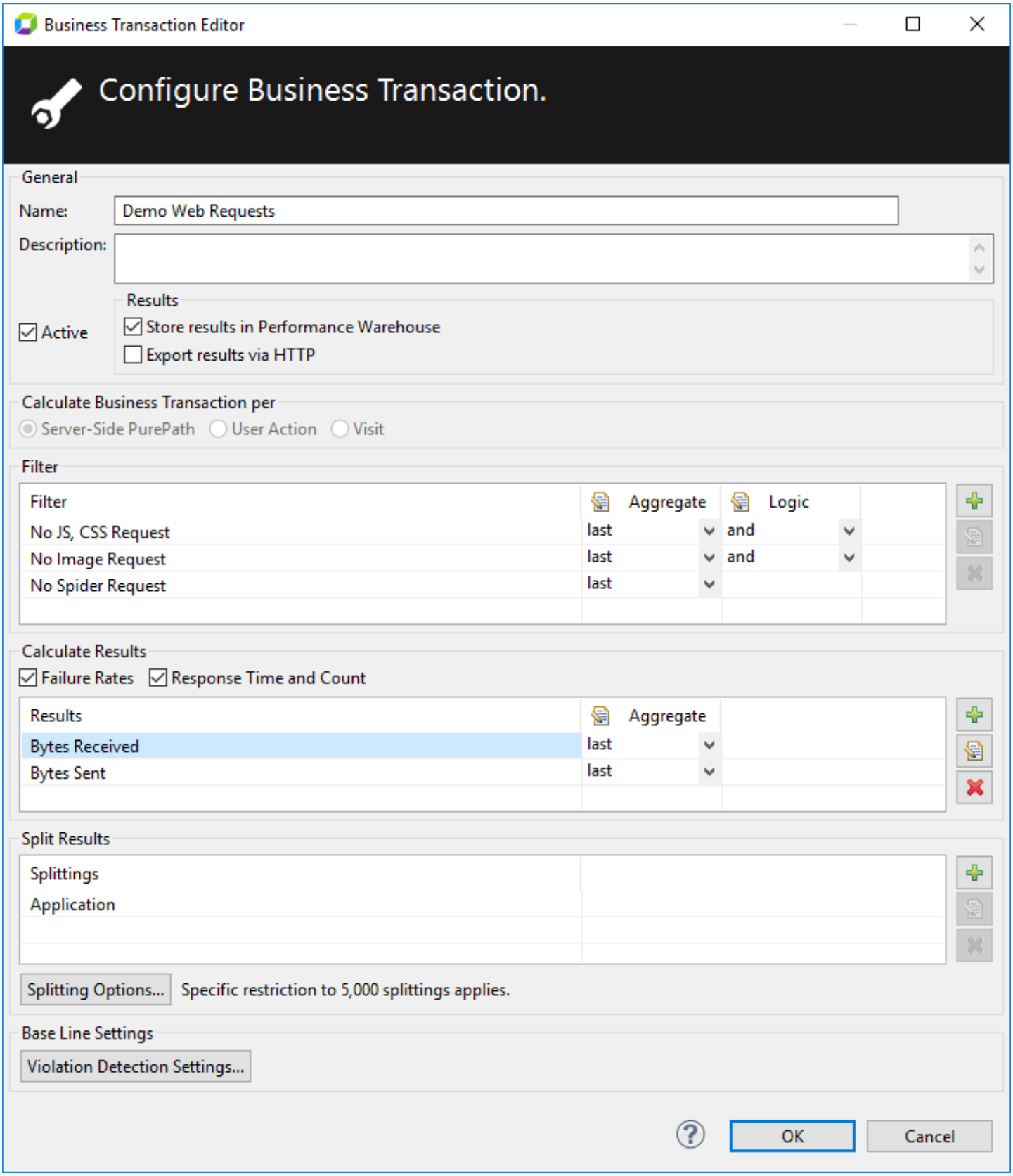
Add additional return values

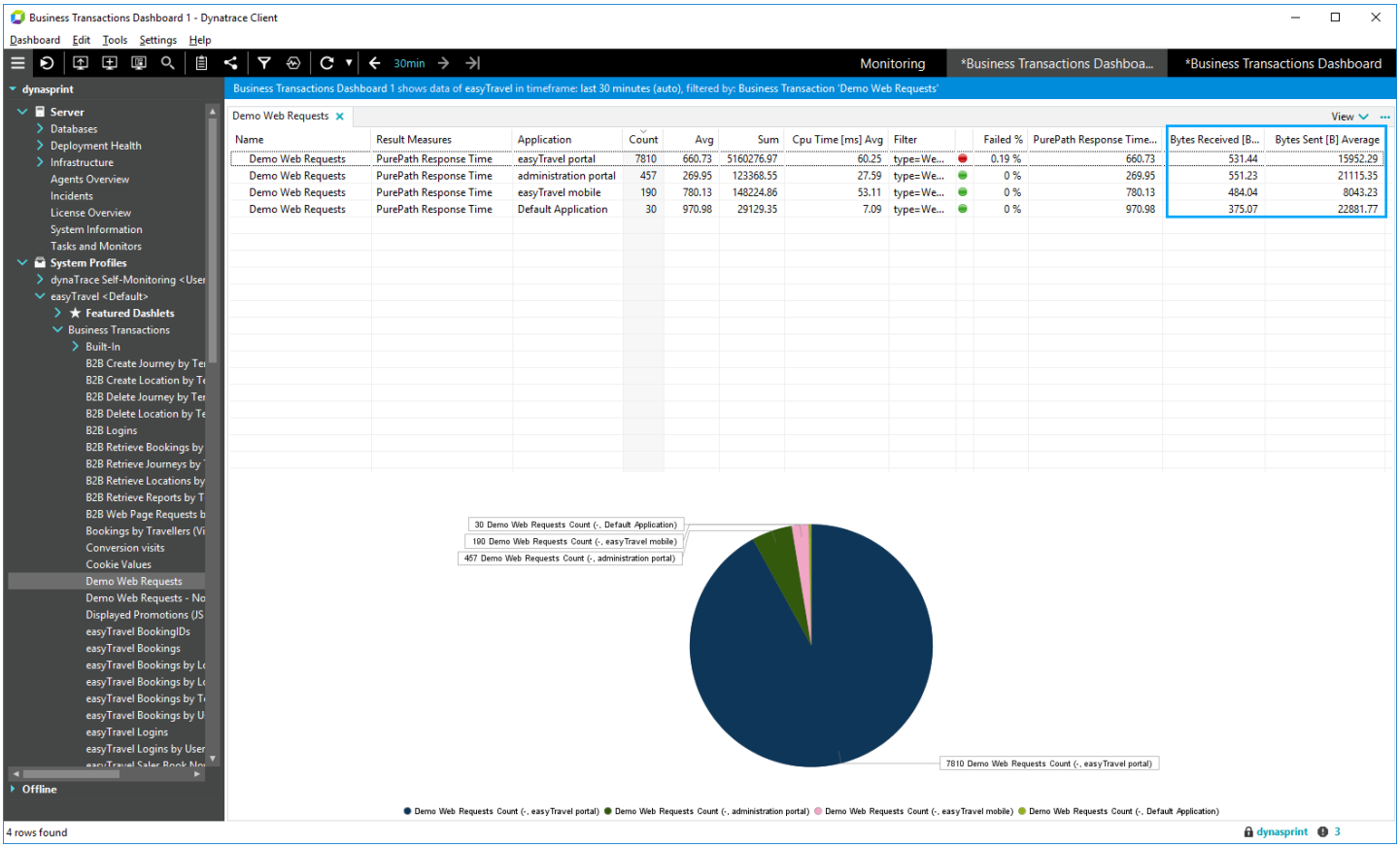
In addition, you may want to analyze other metrics, like response size or the request size. With AppMon you can get them from the same business transaction. Follow these steps:

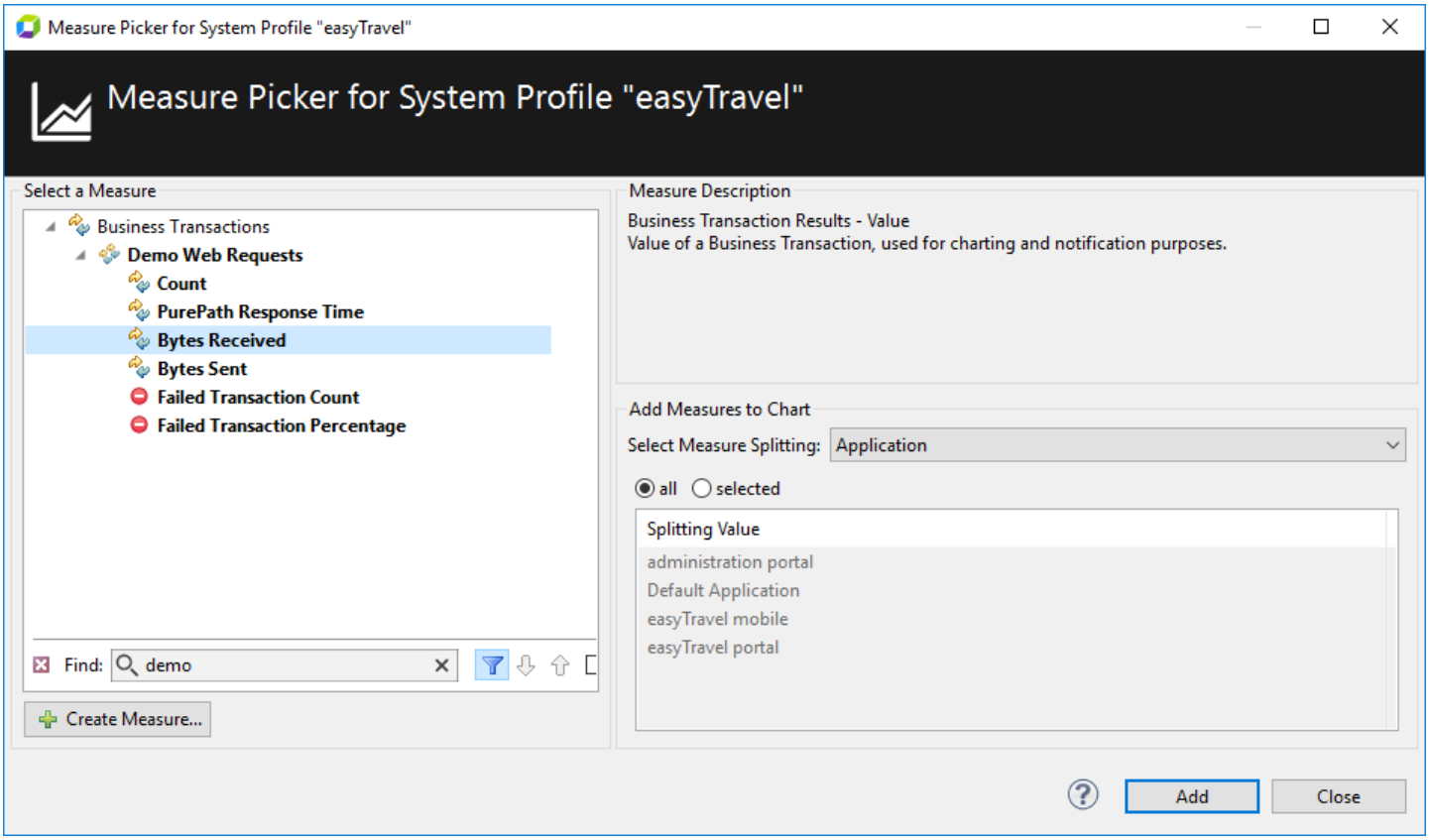
1. Open the System Profile editor, and select sensor, responsible for web requests. In our example it is **Servlets** from the **Customer Frontend (Java)** Agent group.
2. Make sure that the **Measure request/response sizes** checkbox is selected.
3. Switch to the **Business Transactions** item, and select the **Demo Web Requests** business transaction for editing.
4. In the **Calculate Results** pane, click **+** to add measures.
5. In the **Select Measure** dialog box, find and select the **Bytes Received** and **Bytes Sent** measures from the **Server Side Performance > Transaction Based Measures > Web Requests** group.
6. Save the changes to the business transaction.
7. In the Cockpit, expand the **Business Transactions** node of the System Profile (easyTravel in this walkthrough), and double click the **Demo Web Requests**.

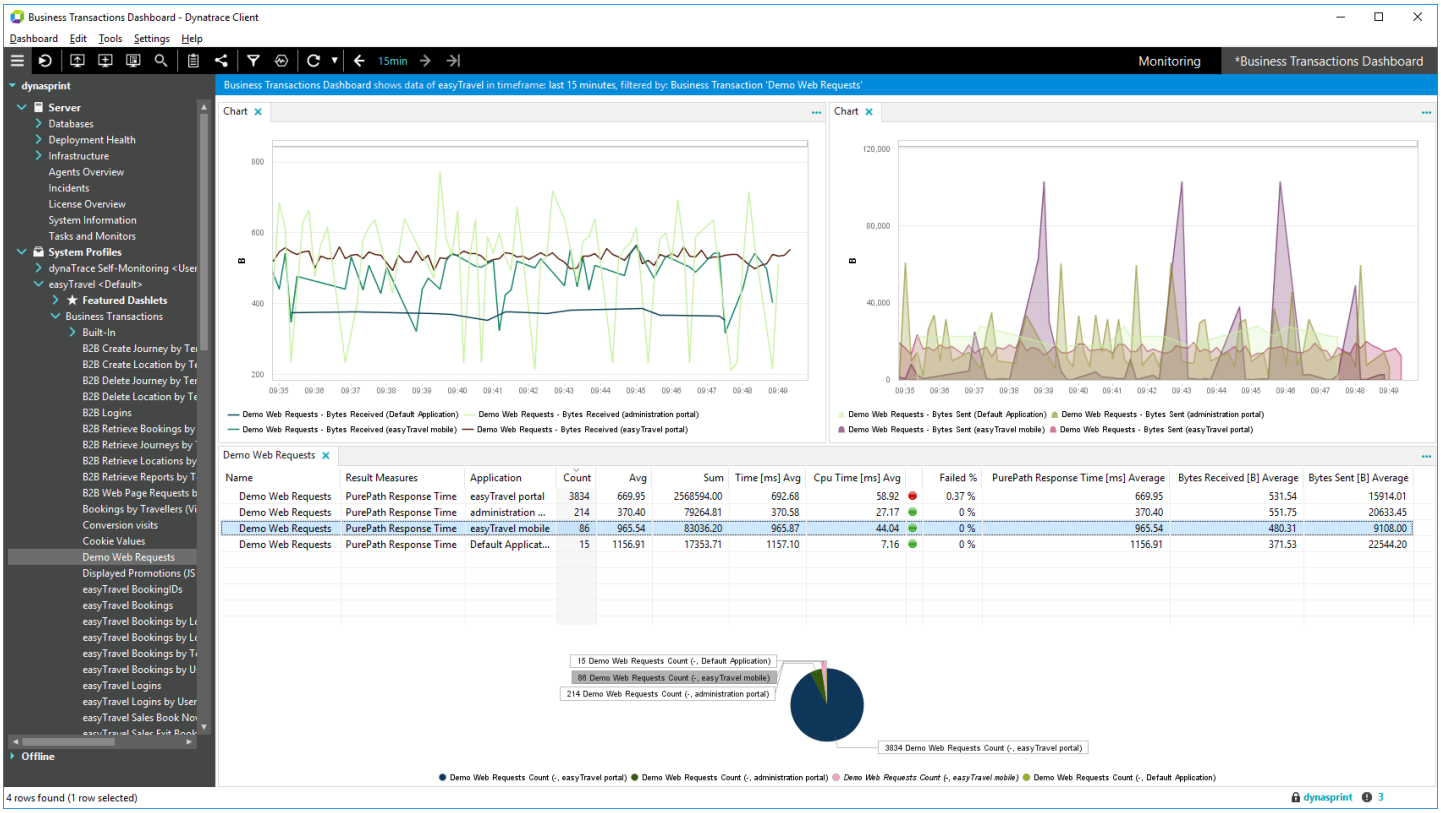
8. In the Business Transactions dashlet, note two new columns with the newly added measures. You can use them, for instance, in charts.
9. Add a new **Chart** dashlet.
0. Click **Add Series**, find and select **Bytes Received** measure of the **Business Transactions > Demo Web Requests** group for it.
1. Repeat for the **Bytes Received** measure.
2. Experiment with charts, to get the most valuable and comfortable for you appearance. See Chart dashlet for more information.




Business Transaction Editor
Configure Business Transaction.
General
Name:
Demo Web Requests
Description:
Active
Results
Store results in Performance Warehouse
Export results via HTTP
Calculate Business Transaction per
Server-Side PurePath
User Action
Visit
Filter
Filter
Aggregate
Logic
No JS, CSS Request
last
and
No Image Request
last
and
No Spider Request
last
Calculate Results
Failure Rates
Response Time and Count
Results
Aggregate
Bytes Received
last
Bytes Sent
last
Split Results
Splittings
Application
Splitting Options...
Specific restriction to 5,000 splittings applies.
Base Line Settings
Violation Detection Settings...
OK
Cancel


Business Transactions Dashboard 1 - Dynatrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*Business Transactions Dashboa...
*Business Transactions Dashboard
dynasprint
Business Transactions Dashboard 1 shows data of easyTravel in timeframe: last 30 minutes (auto), filtered by: Business Transaction 'Demo Web Requests'
Demo Web Requests
View
Name | Result Measures | Application | Count | Avg | Sum | Cpu Time [ms] Avg | Filter | Failed % | PurePath Response Time... | Bytes Received [B... | Bytes Sent [B] Average
Demo Web Requests | PurePath Response Time | easyTravel portal | 7810 | 660.73 | 5160276.97 | 60.25 | type=We... | 0.19 % | 660.73 | 531.44 | 15952.29
Demo Web Requests | PurePath Response Time | administration portal | 457 | 269.95 | 123368.55 | 27.59 | type=We... | 0 % | 269.95 | 551.23 | 21115.35
Demo Web Requests | PurePath Response Time | easyTravel mobile | 190 | 780.13 | 148224.86 | 53.11 | type=We... | 0 % | 780.13 | 484.04 | 8043.23
Demo Web Requests | PurePath Response Time | Default Application | 30 | 970.98 | 29129.35 | 7.09 | type=We... | 0 % | 970.98 | 375.07 | 22881.77
30 Demo Web Requests Count (-, Default Application)
190 Demo Web Requests Count (-, easyTravel mobile)
457 Demo Web Requests Count (-, administration portal)
7810 Demo Web Requests Count (-, easyTravel portal)
Demo Web Requests Count (-, easyTravel portal)
Demo Web Requests Count (-, administration portal)
Demo Web Requests Count (-, easyTravel mobile)
Demo Web Requests Count (-, Default Application)
4 rows found
dynasprint


Measure Picker for System Profile "easyTravel"
Measure Picker for System Profile "easyTravel"
Select a Measure
Business Transactions
Demo Web Requests
Count
PurePath Response Time
Bytes Received
Bytes Sent
Failed Transaction Count
Failed Transaction Percentage
Find: demo
Create Measure...
Measure Description
Business Transaction Results - Value
Value of a Business Transaction, used for charting and notification purposes.
Add Measures to Chart
Select Measure Splitting: Application
all
selected
Splitting Value
administration portal
Default Application
easyTravel mobile
easyTravel portal
Add
Close



Synthetic vs real web request

Monitor or load test tools, and not users, to generate synthetic web requests. Use these to test, or check availability, of a service to get the following:

- Metrics from an application (a server status page that contains metrics).
- Performance data for a certain function.

You can filter out those requests from your production monitoring charts, or treat them separately. They use the same data and parameters every time. This makes them unsuitable for data-driven behavior, and therefore, they aren't very representative.

These synthetic requests contain information to distinguish them from *real* requests. If you use AppMon's built-in web transaction monitors, it adds an HTTP header field labeled `dynatrace`. This makes it easy to create a filter or grouping. Also, this header is used when you integrate with test tools.

AppMon includes the following two predefined measures, which you can use as filters to include or exclude these requests:

- **No Synthetic Dynatrace Request**
- **Synthetic Dynatrace Request**

The **Tagging** setting in this filter only refers to the AppMon header. When you add the **No Synthetic Dynatrace Request** filter to the business transaction you can remove all of the requests generated by your own web transaction monitor. This lets you see only the real requests.

Now, you can create two copies of the previously used business transaction. You can add a different filter to each copy, and separate synthetic and real web requests.

1. Open the System Profile editor, and select the **Business Transactions** item.
2. Make a copy of the **Demo Web Requests** business transaction, and select it for editing.
3. Change the name to the **Demo Web Requests - Synth**
4. In the **Filters** pane, add the **Synthetic Dynatrace Request** measures.
5. Save your changes.
6. Repeat for the **No Synthetic Dynatrace Requests** measures.
7. Put the new business transactions on one dashboard. After you create a few manual requests, you can see a difference between the synthetic and real requests.

Conclusion

AppMon has many features to easily manage business transactions. Multiple return values are a great way to get more than one metric from the same underlying Purepaths. Use regular expressions to filter and group, and get meaningful splitting values.

You can also use business transactions for the following:

- Filter PurePaths by missing method calls. You use a negative filter that throws incidents if a method that is expected to be called isn't called.
- User actions and visits.
- Add multiple grouping values, like grouping by URI and by HTTP header value/parameters.

Expressions

Too many complex regular expressions can cause an excessive load on your system. Also, if you split by a Measure and generate many unique values, such as a session ID, it can add an excessive load on your performance warehouse.

Follow your transactions with Transaction Flow

On this page

Your application landscape from the bird’s eye view

Goal of this walkthrough

The topology view

The transaction response time view

Drill down from the Transaction Flow

Conclusion

Related pages

How to use Business Transactions

Create a splitting Business Transaction

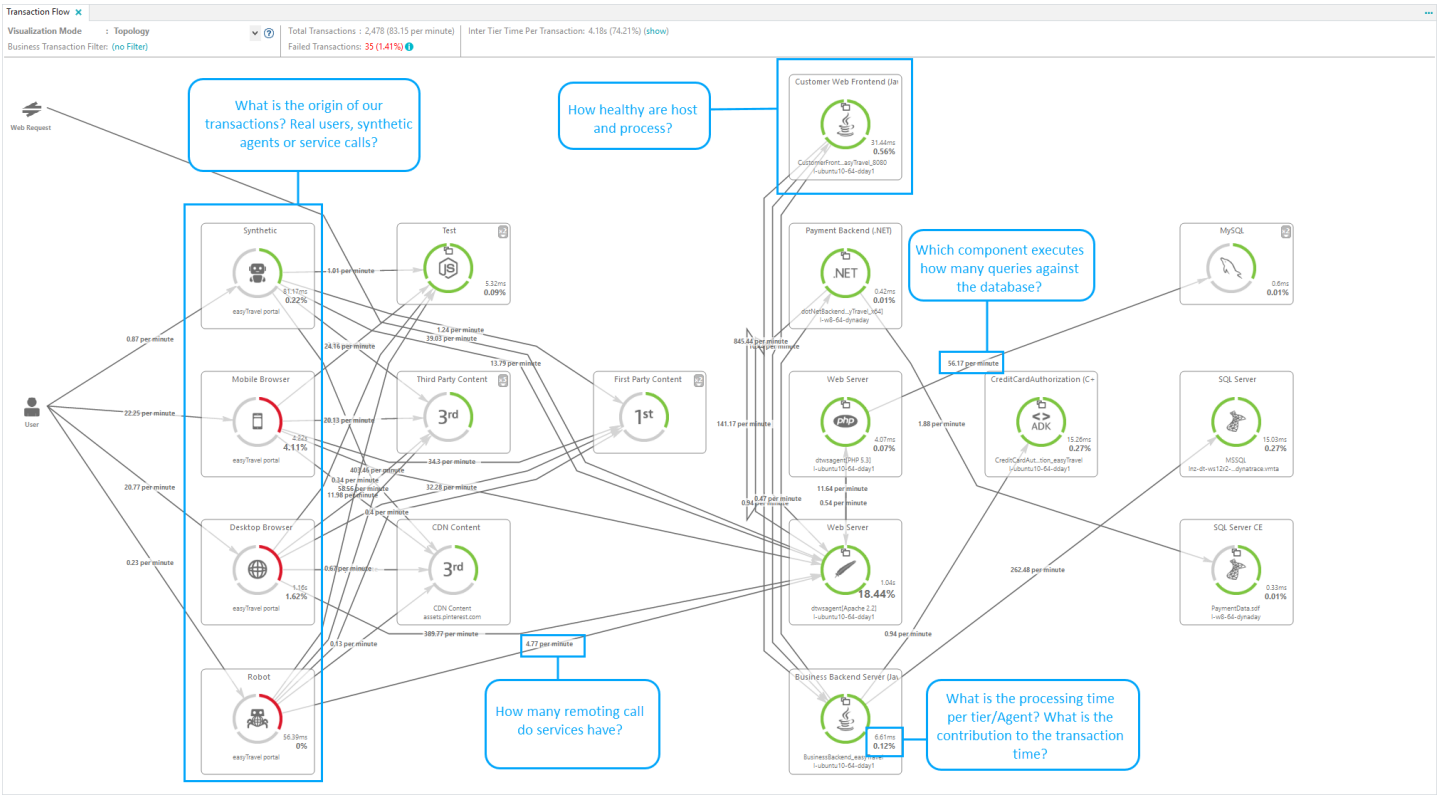
Transaction Flow dashlet

Your application landscape from the bird’s eye view

For a quick and easy way to see what’s going on in your application environment, use the **Transaction Flow** dashlet to see the following:

- Aggregated view of all application components.
- Application dependencies.
- Communication paths between single tiers and key indicators to locate problems.

The **Transaction Flow** dashlet displays diagrams to help you understand the information flow. It’s a great starting point for deeper analysis and drill-downs.



The Transaction Flow Dashlet

Goal of this walkthrough

This walkthrough explains how to use the **Transaction Flow** dashlet. It also discusses its two operation modes: Topology view and Transaction Response Time view.

Scenario

Use the easyTravel demo application to learn about various AppMon features and functions.

Requirements

If AppMon is running, you are ready to go. If you have configured business transactions, you can use these.

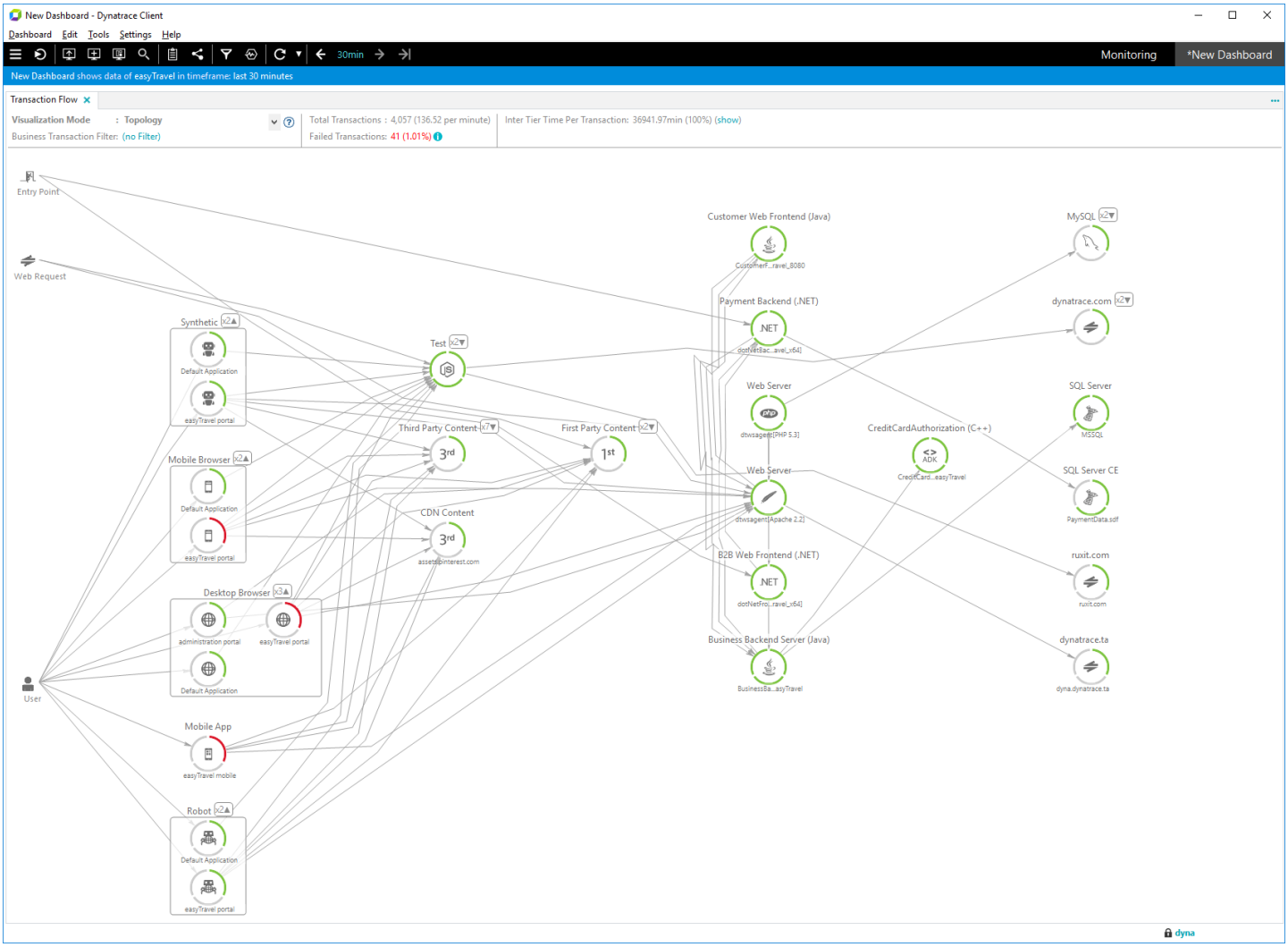
The topology view

When you open the **Transaction Flow** dashlet it starts in Topology visualization mode. This is a great way to transition to deeper analysis.

Know your application landscape

You may know where your application components run, and how they work together. You may understand the dependencies of components and services, and which protocols AppMon uses to communicate. But do you know what happens in real time? Do you know how many calls AppMon makes, and how Inter Tier Time affects your user's response time?

Use the **Transaction Flow** dashlet to understand a complex application environment. You can even skip the architecture, documentation, and source code analysis.



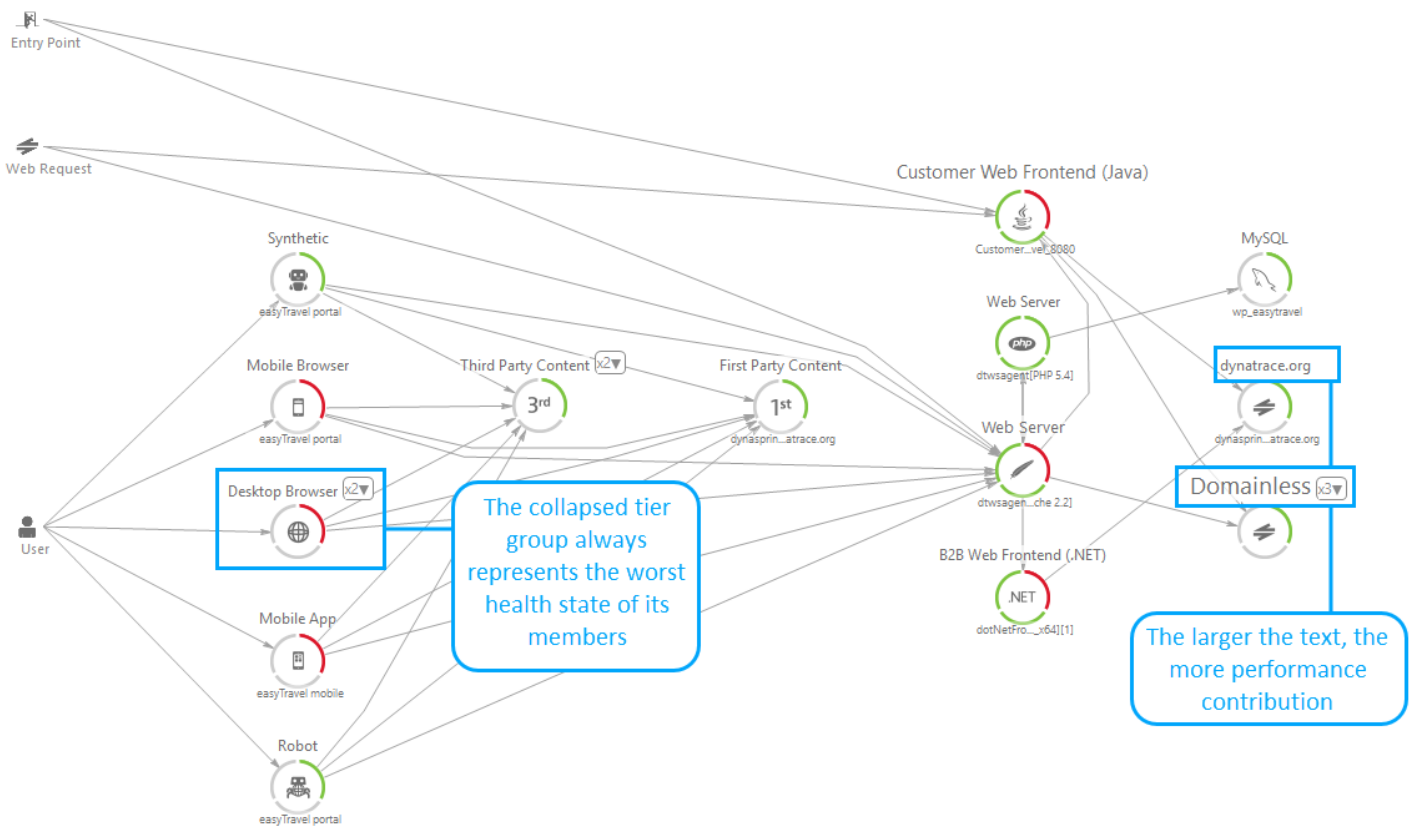
Transaction Flow Diagram of easyTravel

The demo setup includes the following servers:

- Web server (Apache)
- 3 third party content servers
- B2B Web front end and payment back end (.Net)
- Customer Web front end and business back end (Java)
- Credit card authorization (C++)
- MySQL and SQL Server CE DBs

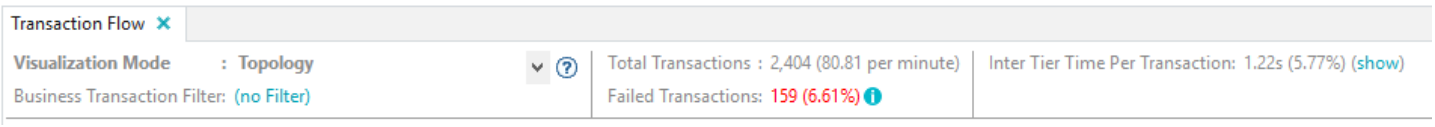
Does everything run smoothly?

The **Transaction Flow** dashlet displays the flow of very large environments. It uses small circles or large boxes to represent tiers. The tier name tag size represents the node's performance / response time contribution. To find hotspots, look for nodes with larger text. The collapsed tier group always represents the worst health state of its members.



Transaction Flow with Hotspots and Collapsed Tier Groups

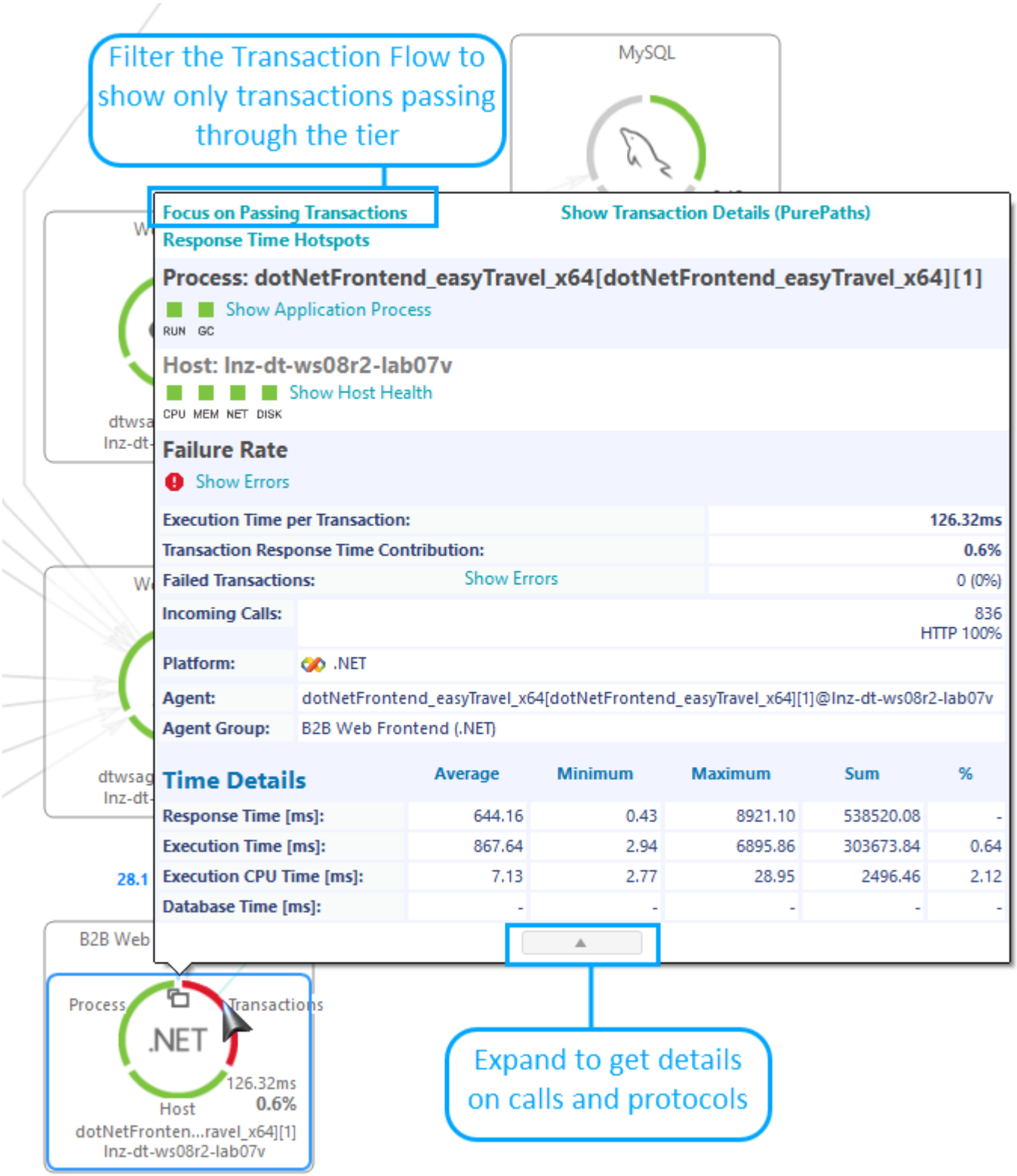
Look at the information in the header of the dashlet:



Note that the average Failed Transactions number is very high. What does this mean, and how does this affect the application? It may be due to network related problems, or poorly performing queues. Where do these failed transactions come from?

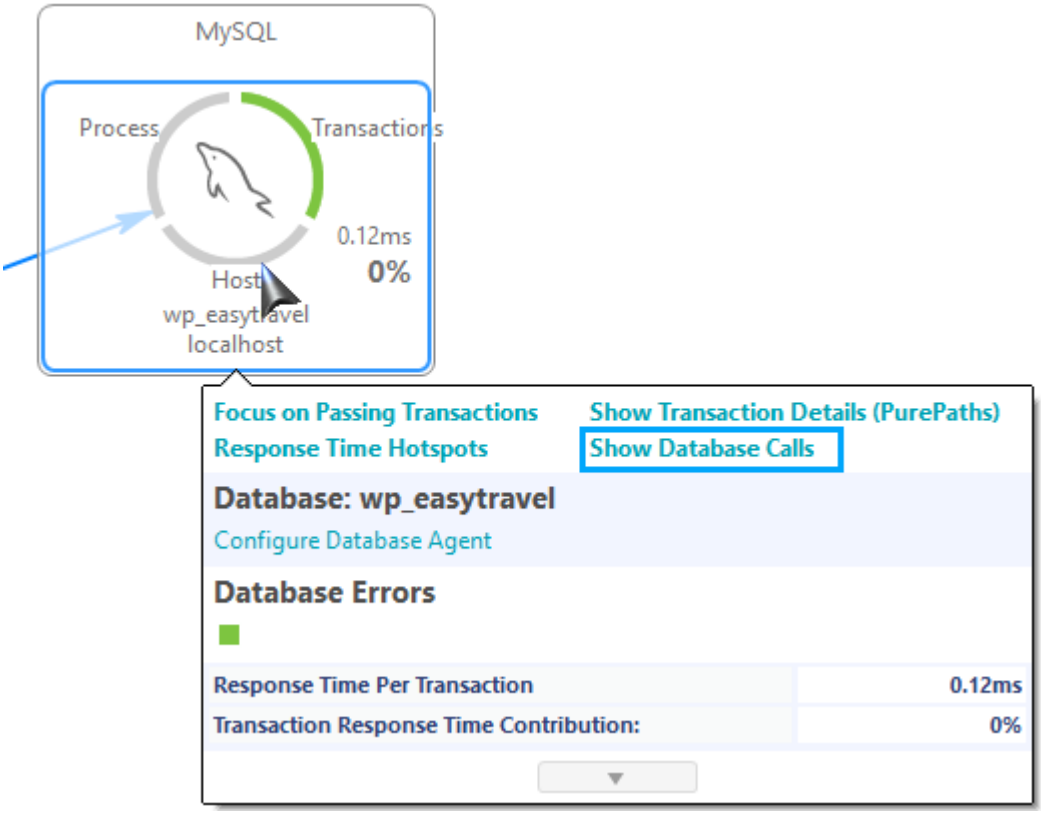
Focus on passing transactions

Inter Tier Time comes from transactions that pass through one particular tier. Some PurePaths take different routes. To analyze a specific tier, hover over the Agent and select **Focus on Passing Transactions**. This opens a new **Transaction Flow** for paths that pass through that node.



Focus on Passing Transactions

How many database calls?



Show Database Calls

From each node in the dashlet, you can do the following:

- Hover over the **DB Tier**.
- Hover over the tier that makes calls to the DB.
- Go to the **Visits** dashlet from the **Users** node.
- Go to the **Database** dashlet.

Hover over the different elements, or right-click, to explore the options.

The transaction response time view

The **Transaction Response Time View** visualization mode displays the following:

- The response time of certain transactions passing through the application
- Passing transactions and response time per node
- Inter-Tier communication

Use this vital information to analyze the following and determine the root cause of poor end user experience:

- Performance of single transaction types
- How single components contribute to the overall transaction response time

Drill down from the Transaction Flow

Note that for the same type of requests, one frontend server performs poorly and the other performs fine. What is the root cause of this poor performance?

Drill down from the **Transaction Flow** to analyze this further. If you're in Operations, you might drill down from one Agent to a **Host Performance** dashboard. On this dashboard check to see if there is a problem with the machine that is running the frontend server. If you're interested in the number of database statements that AppMon generates for certain requests, drill down to the Database Dashlet. If the machine is performing well, drill down from the **Agent Group** to the Layer Breakdown Dashlet. Compare the application behavior on the two different hosts. Dive deeper and go straight to the PurePath Comparison Dashlet. You may discover bad source code.

Conclusion

The **Transaction Flow** dashlet provides a general overview and helps you isolate the source of resource bottlenecks or problems. You can also use it to examine arbitrary business transactions and pinpoint problems for single transaction types. With a few clicks, you can locate the root cause for these problems.
The **Transaction Flow** dashlet and associated components provide clear analysis, so you don't have to browse through tables to link and manually aggregate data.

Create a splitting Business Transaction

On this page

Goal of this walkthrough

Configure the Servlet Sensor

Create the measure to use for splitting.

Create a splitting Business Transaction

See results

Related pages

How to use Business Transactions

Follow your transactions with Transaction Flow

Watch and learn

What are Business Transactions?

Business Transaction Management Webinar

Creating server-side Business Transactions

Creating User Action Business Transactions

Creating visit-based Business Transactions

Goal of this walkthrough

This walkthrough shows how to configure a Business Transaction that splits web requests by user session attribute `loginBean.getUserContext().getUserName()`.

Scenario

Use the easyTravel demo application to learn about various AppMon features and functions. Or you can use any standard web application because this scenario works with simple web requests. The walkthrough includes the following steps:

1. Configure the servlet sensor properties.
2. Create the measure to use for splitting.
3. Create the splitting Business Transaction.
4. See results.

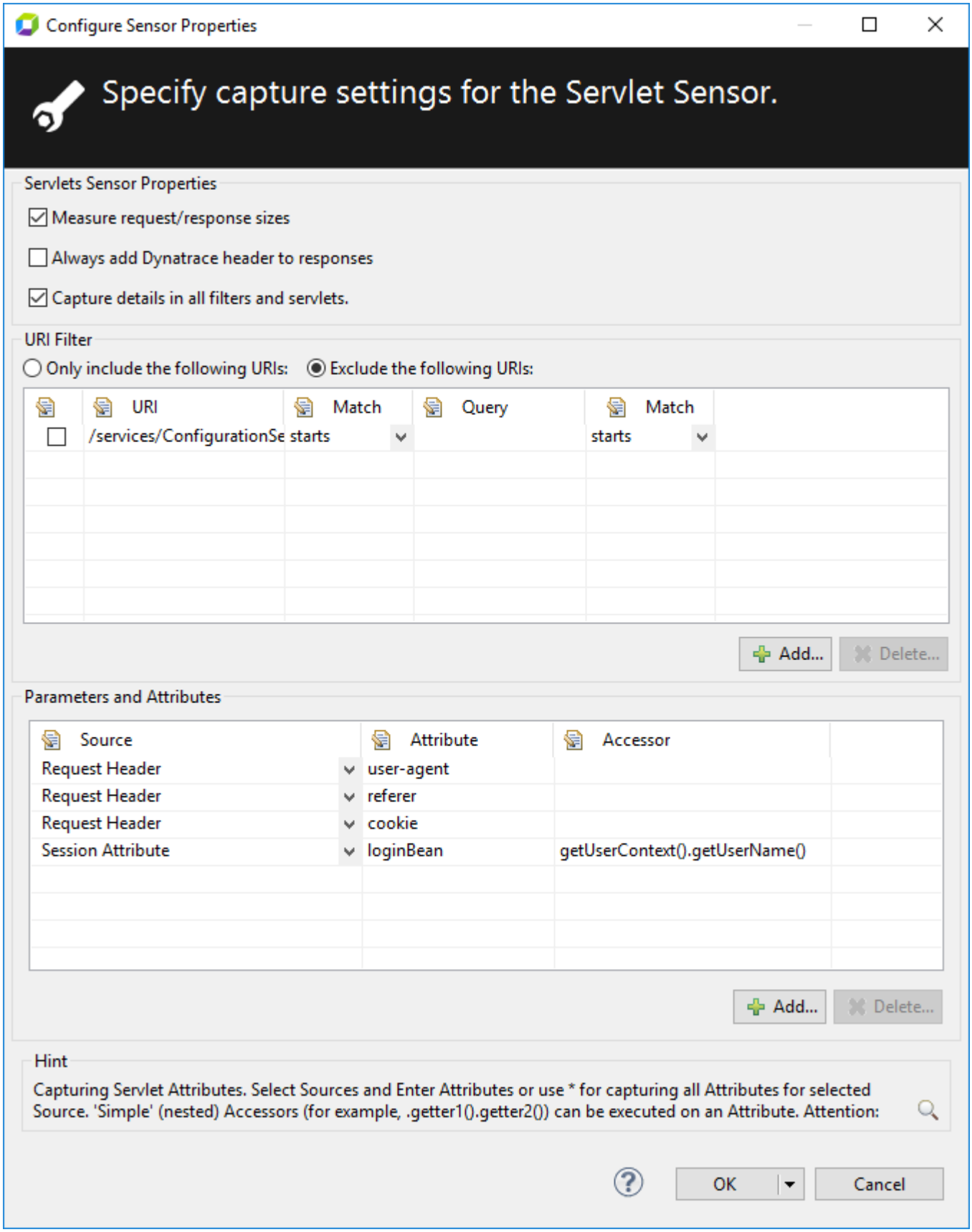
Requirements

Run the demo application with injected agents and connect to a System Profile. If you use your own application, associate and activate the necessary web server Sensor. Please note, that this walkthrough may use naming bases on easyTravel demo app.

Configure the Servlet Sensor

To define Business Transactions based on a web request session attribute, you must enable capture of this parameter in the **Configure Sensor Properties** dialog box.

1. Open the System Profile editor, and select the **Customer Web Frontend (Java) > Sensor Configuration** item.
2. Double click the **Servlets** sensor for editing.
3. In the **Parameters and Attributes** pane, click **Add**, and select the **Session Attribute** in the **Source** column of the newly create row.
   You can enter a wildcard (*) to capture all parameters. However, this can result in capturing more data than you need and thus more overhead than necessary. A better approach is to enter action to capture the specific parameter.
4. In the **Attribute** column, type `loginBean`.
5. In the **Accessor** column, type `getUserContext().getUserName()`.
6. Save the new measure.

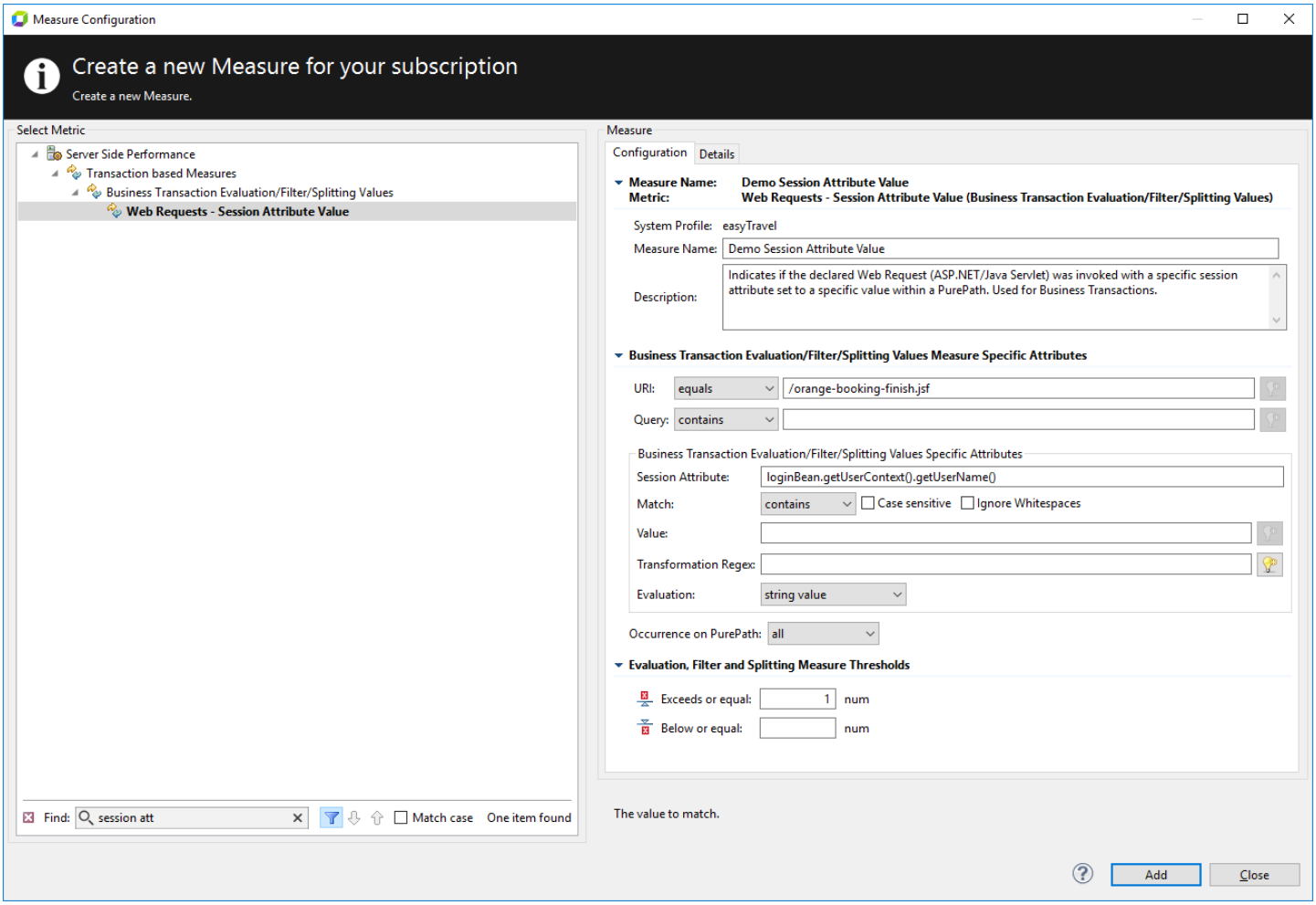


Configured Servlets sensor

Create the measure to use for splitting.

Now you need a measure which will serve as the input for the Business Transaction.

1. In the **System Profile Preferences** dialog box, select the **Measures** item.
2. Click **Create Measure** to create a new measure.
3. Find and select the **Web Requests - Session Attribute Values** metric in the **Server Side Performance > Transaction based Measures > Business Transaction Evaluation/Filter/Splitting Values** group.
4. Give it a name, such as **Demo Session Attribute Value**.
5. In the **URI** field of the **Business Transaction Evaluation/Filter/Splitting Values Specific Attributes** pane, select **equals** from the list, and type `/orange-booking-finish.jsf`.
6. In the **Session Attribute** field of the **Business Transaction Evaluation/Filter/Grouping Values Specific Attributes** pane, type `loginBean.getUserContext().getUserName()`, and select **contains** from the **Match** list.

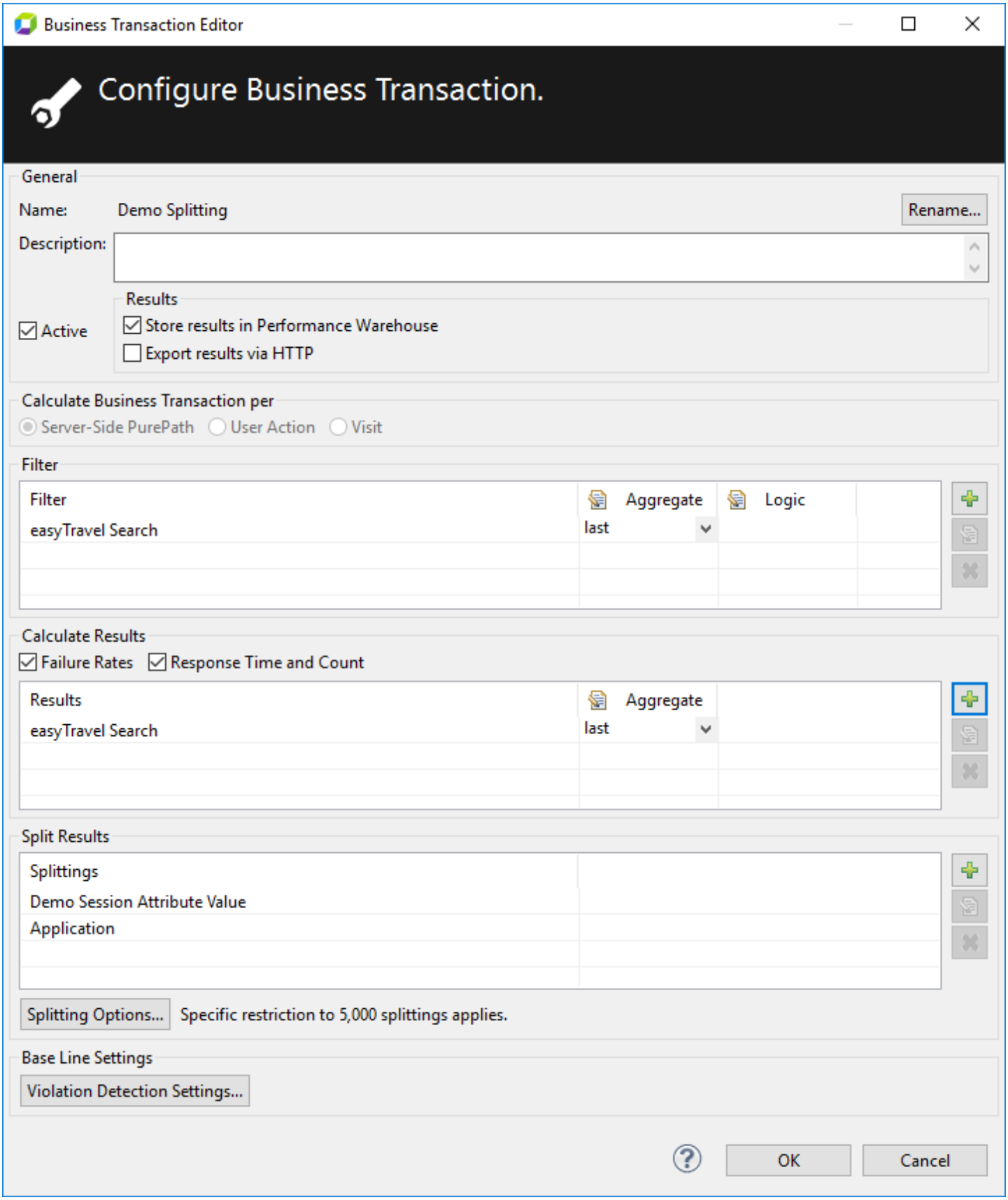


Configured measure

Create a splitting Business Transaction

Now you can create splitting Business Transaction.

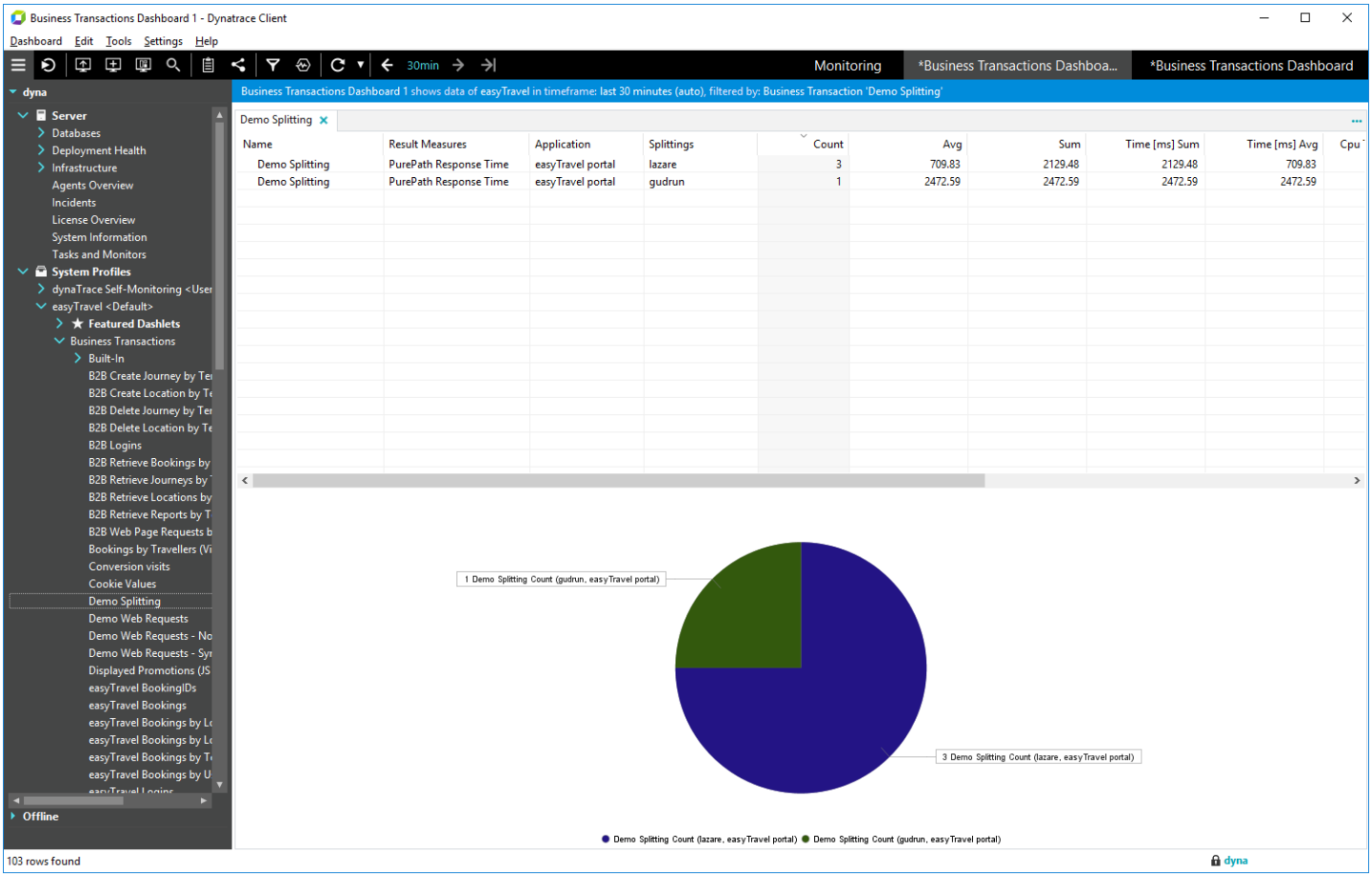
1. In the **System Profile Preferences** dialog box, select the **Business Transactions** item.
2. Click **Create** to create a new business transaction.
3. Give it a name, such as **Demo Splitting**.
4. In the **Filter** pane, click **+** to add a filter measure.
5. Find and select the **easyTravel Search** measure, to get web requests for our business transactions.
6. In the **Split Results** pane, click **+** to add a splitting measure.
7. Find and select the **Demo Session Attribute Value** measure.
8. Save the new business transaction.



Configured Business Transaction

See results

Now you can use your business transaction. In the Cockpit, expand the **Business Transactions** node of the System Profile (easyTravel in this walkthrough), and double click the **Demo Splitting**. If you have just started easyTravel, you may have to wait while it produces some traffic, and new PurePaths are collected and analyzed.



Business Transaction dashboard

Problem pattern detection

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Problem pattern categories

Threshold Adjustment

Problem Pattern Detection Webinar

Related pages

PurePaths dashlet

PurePath Comparison dashlet

PurePath explained

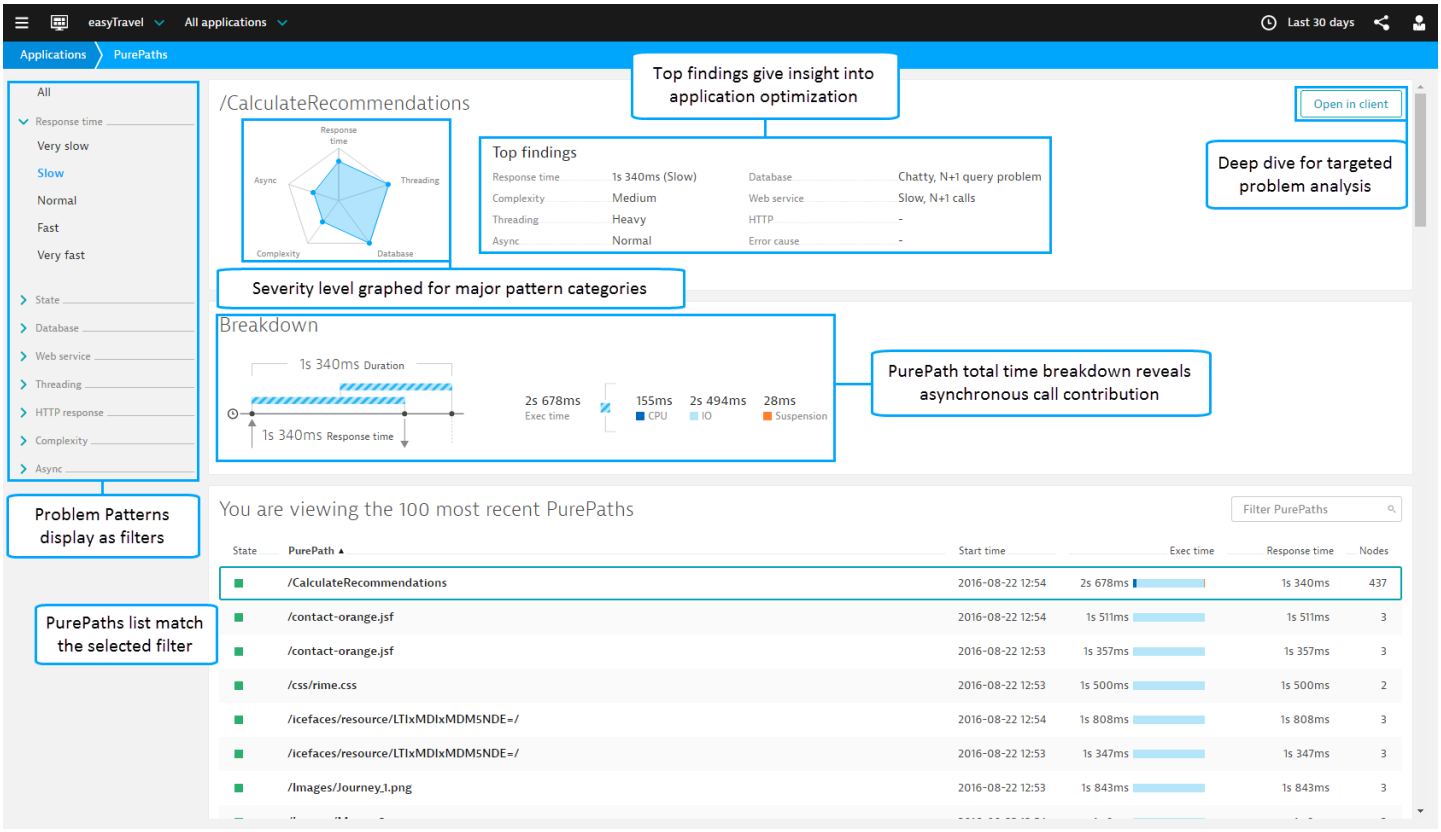
Error analysis

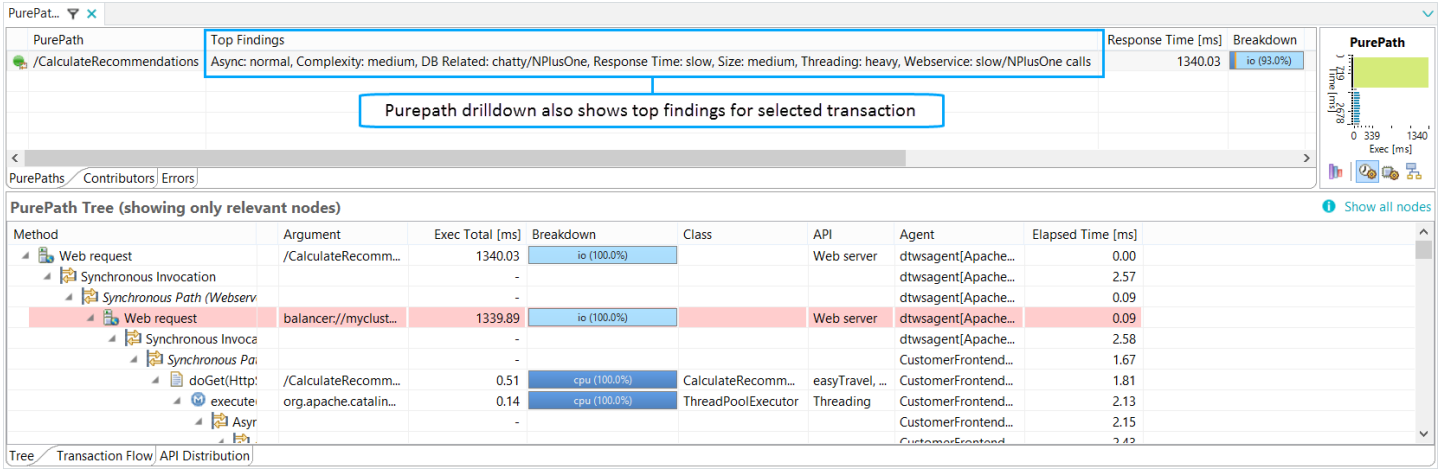
Overview

Many applications suffer from the same or similar performance, scalability and architectural issues. Using AppMon to detect and analyze them over time reveals common patterns that can be categorized as types of performance issues.

AppMon 6.5 and later analyzes every PurePath against these automatically detected common patterns. The AppMon Web PurePath view lets you easily find PurePaths within a particular pattern, and shows related performance patterns. Patterns show as PurePath categories with customizable buckets indicating the performance issue severity for that category.

Categories display also as top findings, which let you see potential transaction problem areas at a glance, giving insight into resolving performance issues and finding optimization areas in the application. These display in both AppMon Web and in the AppMon Client, so the insights are readily available as you perform analysis.





Purepaths in the AppMon Web PurePath view cover all transactions, which are tagged with these common problem pattern categories:

- Response Time
- State
- Database

- Web Service
- Threading
- HTTP Response
- Complexity
- Async

Problem pattern categories

Response time

Default Bucket Thresholds: Very Fast (<100ms); Fast (100 – 500ms); Normal, (500 – 1000ms); Slow (1-4s); Very Slow (> 4s)

This analysis looks at speed of PurePaths. Among other ways to analyze using this category, you can analyze very slow transaction by comparing them with the same requests on the same URL that went really fast, since AppMon captures all transactions all the time. This lets you quickly spot slow transactions and compare them method by method with similar fast transactions.

State

Default Bucket Thresholds: No Errors (transaction has zero errors); Detected Errors (one or more non-failing errors detected); Transaction Failed (one of more errors led to a transaction failure)

This category focuses on detected errors in transactions. An *error* is an unexpected event that happens while a user interacts with a system. Common errors include resources that cannot be found on a web server, HTTP 500 response codes sent by a web service, or application exceptions. This category lets you get straight to root cause analysis by selecting a transaction and stepping through the errors that were detected on method calls of the transaction.

Database

Default Bucket Thresholds: DB Chatty (> 100 SQL Executions); Single Long SQL (1 SQL taking > 80% of SQL Total); N+1 Query Problem (Same SQL executed > 10 times); Overall DB time too high (JDBC/ADO.NET/PDO time > 80% of Total Exec Time).

Database access is the most common application problem pattern. AppMon analyzes database access and marks your PurePaths that show strange patterns. Even if response time of the application is fast, this category can reveal whether a pattern has been added that can cause problems with increased traffic on the system or a larger database. This type of analysis also only works if you always capture every single transaction with all details.

You can also use this data to compare the same business transaction that shows a bad access pattern with a transaction that does not. A classic example is a bad search query that causes lots of database queries. For that to know you need to have the good and the bad transaction available and compare them.

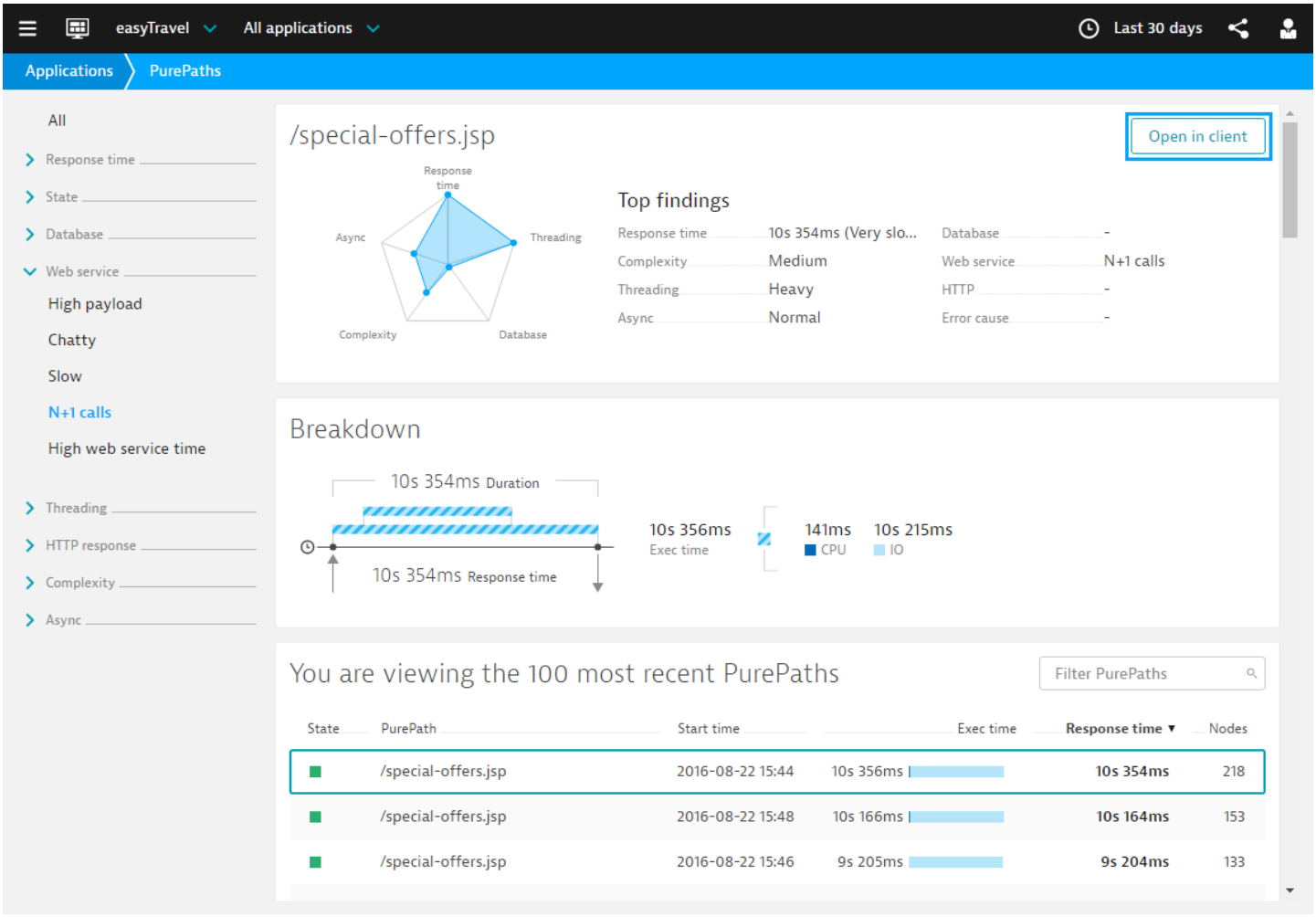
Web service

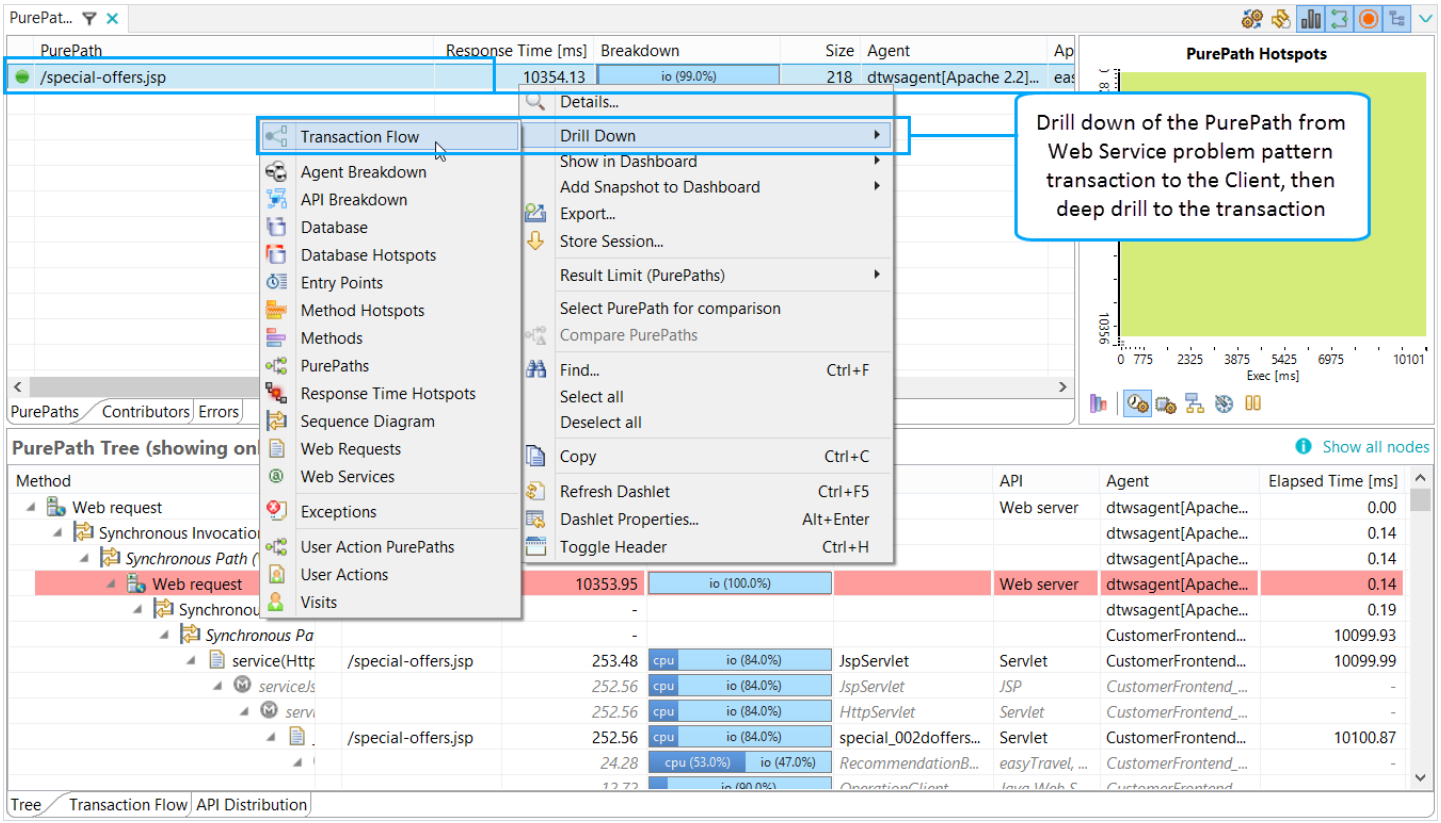
Default Bucket Thresholds: High payload (Request + Response Size > 500k); Chatty (> 50 Service Calls per Request); Slow (1 Service Call > 80% of Total WS Time); N+1 Pattern (same Service call more than once); High Web Service Time (> 80% Total Time)

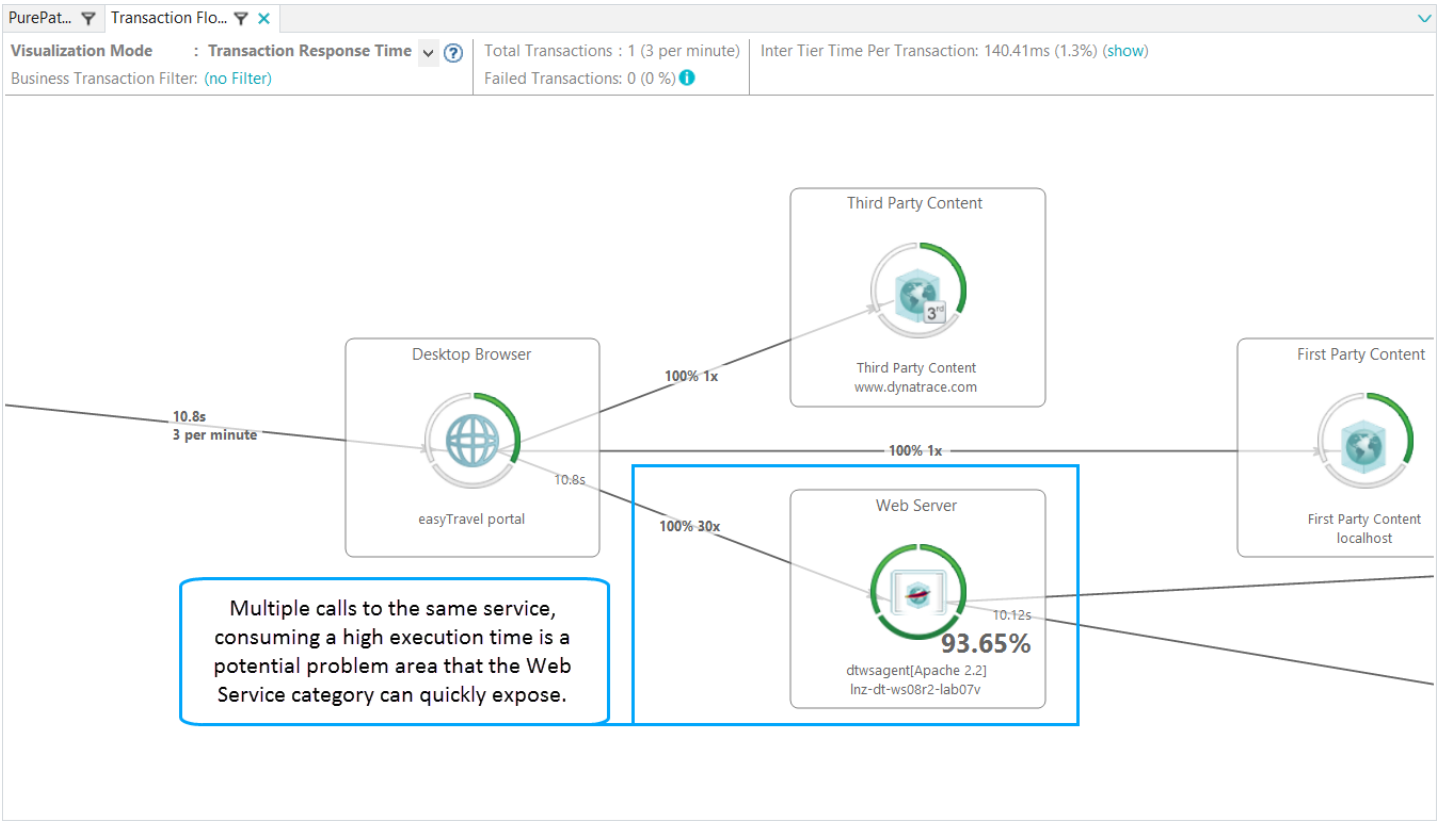
Services have always been used by applications, even more so with the popularity of MicroServices and many frameworks that make it easy to write and consume services. This practice potentially creates a lot of bad web service access patterns.

For example, a PurePath that shows a high number of the same service calls that consume more than 80% of the total execution time can potentially indicate a problem.

This example shows how the Web Service category quickly identifies PurePaths with potential problematic web service access. The PurePath /special-offers.jsp tops the list of recent Purepaths in the N+1 calls sub-category. The Transaction Flow drilldown shows 30 calls to the same service, taking over 93% of the execution time, indicating a problem PurePath.


easyTravel
All applications
Last 30 days
Applications
PurePaths
All
Response time
State
Database
Web service
High payload
Chatty
Slow
N+1 calls
High web service time
Threading
HTTP response
Complexity
Async
/special-offers.jsp
Open in client
Response time
Async
Threading
Complexity
Database
Top findings
Response time 10s 354ms (Very slo...
Complexity Medium
Threading Heavy
Async Normal
Database -
Web service N+1 calls
HTTP -
Error cause -
Breakdown
10s 354ms Duration
10s 354ms Response time
10s 356ms
Exec time
141ms
CPU
10s 215ms
IO
You are viewing the 100 most recent PurePaths
Filter PurePaths
State PurePath Start time Exec time Response time Nodes
/special-offers.jsp 2016-08-22 15:44 10s 356ms 10s 354ms 218
/special-offers.jsp 2016-08-22 15:48 10s 166ms 10s 164ms 153
/special-offers.jsp 2016-08-22 15:46 9s 205ms 9s 204ms 133


PurePath
Response Time [ms]
Breakdown
Size
Agent
/special-offers.jsp
10354.13
io (99.0%)
218
dtwsagent[Apache 2.2]...
Transaction Flow
Agent Breakdown
API Breakdown
Database
Database Hotspots
Entry Points
Method Hotspots
Methods
PurePaths
Response Time Hotspots
Sequence Diagram
Web Requests
Web Services
Exceptions
User Action PurePaths
User Actions
Visits
Details...
Drill Down
Show in Dashboard
Add Snapshot to Dashboard
Export...
Store Session...
Result Limit (PurePaths)
Select PurePath for comparison
Compare PurePaths
Find... Ctrl+F
Select all
Deselect all
Copy Ctrl+C
Refresh Dashlet Ctrl+F5
Dashlet Properties... Alt+Enter
Toggle Header Ctrl+H
Drill down of the PurePath from Web Service problem pattern transaction to the Client, then deep drill to the transaction
PurePath Hotspots
Exec [ms]
PurePaths
Contributors
Errors
PurePath Tree (showing on
Show all nodes
Method
API
Agent
Elapsed Time [ms]
Web request
Web server
Synchronous Invocatio
Synchronous Path (
Web request
10353.95
io (100.0%)
Synchronou
Synchronous Pa
service(Http
/special-offers.jsp
253.48
cpu
io (84.0%)
JspServlet
Servlet
serviceJs
252.56
JspServlet
JSP
servi
252.56
HttpServlet
Servlet
/special-offers.jsp
252.56
special_002doffers...
24.28
cpu (53.0%)
io (47.0%)
RecommendationB...
easyTravel, ...
Tree
Transaction Flow
API Distribution



Threading

Default Bucket Thresholds: Heavy (> 5 Threads per Path); Acceptable (< 5 Threads per Path); Normal (<=2 Threads per Path)

Many transactions use a lot of background threads to parallelize work. Transactions using a high number of threads have a high potential for inefficient coding and poor synchronization. Also, the more threads involved, the more resources consumed on the App Server, which can directly affect system complexity.

Use this category to focus on the total number of threads involved in processing a single transaction.

HTTP response

Default Bucket Thresholds: 4xx, 5xx

This category helps you focus on requests that return a certain HTTP Status Code and analyze failing requests that return an HTTP 5xx. This category also helps to more easily reveal transactions with bad redirects or authentication issues.

Complexity

Default Bucket Thresholds: Complex (> 10 Agents or > 1000 Nodes); Medium (< 10 Agents or < 1000 Nodes); Normal (< 3 Agents or < 100 Nodes)

Sorting PurePaths by size (number of nodes) typically lets you quickly see the more complex transactions. These are typically transactions with a greater likelihood for issues.

Use this category to understand the impact that the number of tiers have on a transaction's efficiency for larger sized Purepaths. A transaction that spawns more than 10 tiers usually is a highly complex transaction where the potential for problems rises. The Complexity category helps you streamline analysis of these PurePaths.

Async

Default Bucket Thresholds: Heavy (Duration > 10x of Response Time); Normal (has async activity); No Async

Response Time is good to optimize, but can result in initially fast responding transactions that end up running indefinitely in the background. For that reason, AppMon also captures PurePath duration, which includes all asynchronous activity triggered by the initial request. An unusually long Duration as compared to the Response Time is usually a sign of potential issues with the transaction.

Threshold Adjustment

AppMon comes with predefined thresholds that can be adjusted for each System Profile. If needed, you can adjust the thresholds in the applicable System Profile configuration `xml` file, located in `<DT_HOME>/server/conf/profiles`.

The following graphic shows where to adjust your preferred thresholds.

```
      <defaultpathcategorizationconfig enabled="true">
        <webservicecategory enabled="true">
                                                    ount="-1" totalcallcount="-1" singlewebservicetimethreshold="-1.0" hightimethreshold="-1.0"
name="Hi                                            ion" requestresponsesize="500.0" pathtagposition="WebService_Payload" />
                                                    ount="-1" totalcallcount="-1" singlewebservicetimethreshold="0.8" hightimethreshold="-1.0"
name="Sl                                            uestresponsesize="-1.0" pathtagposition="WebService_Slow" />
          <webservicetag samewebservicecallcount="-1" totalcallcount="-1" singlewebservicetimethreshold="-1.0" hightimethreshold="0.8"
name="High Web Service Time" description="No description" requestresponsesize="-1.0" pathtagposition="WebService_HighTime" />
          <webservicetag samewebservicecallcount="-1" totalcallcount="20" singlewebservicetimethreshold="-1.0" hightimethreshold="-1.0"
name="Chatty" description="No description" requestresponsesize="-1.0" pathtagposition="WebService_Chatty" />
          <webservicetag samewebservicecallcount="2" totalcallcount="-1" singlewebservicetimethreshold="-1.0" hightimethreshold="-1.0"
name="NPlusOne" description="No description" requestresponsesize="-1.0" pathtagposition="WebService_NPlusOneCalls" />
        </webservicecategory>
        <sizecategory enabled="true">
          <sizetag name="Very Large" description="No description" minsize="1024" pathtagposition="Size_VeryLarge" />
          <sizetag name="Medium" description="No description" minsize="100" pathtagposition="Size_Medium" />
          <sizetag name="Normal" description="No description" minsize="0" pathtagposition="Size_Normal" />
          <sizetag name="Large" description="No description" minsize="500" pathtagposition="Size_Large" />
        </sizecategory>
        <threadingcategory enabled="true">
          <threadingtag name="Heavy" description="No de                    ion="Threading_Heavy" minthreads="6" />
          <threadingtag name="Normal" description="No d                    tion="Threading_Normal" minthreads="1" />
          <threadingtag name="Acceptable" description="No description" pathtagposition="Threading_Acceptable" minthreads="3" />
        </threadingcategory>
```

Each category is configureable.

Customize to your own preferences!

Note

You must restart the AppMon server for any changes made to a System Profile configuration file to take effect.

Problem Pattern Detection Webinar



Error analysis

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Detecting errors

Monitoring functional health

Error root cause analysis

Related pages

System Profile - Error Detection

Problem pattern detection

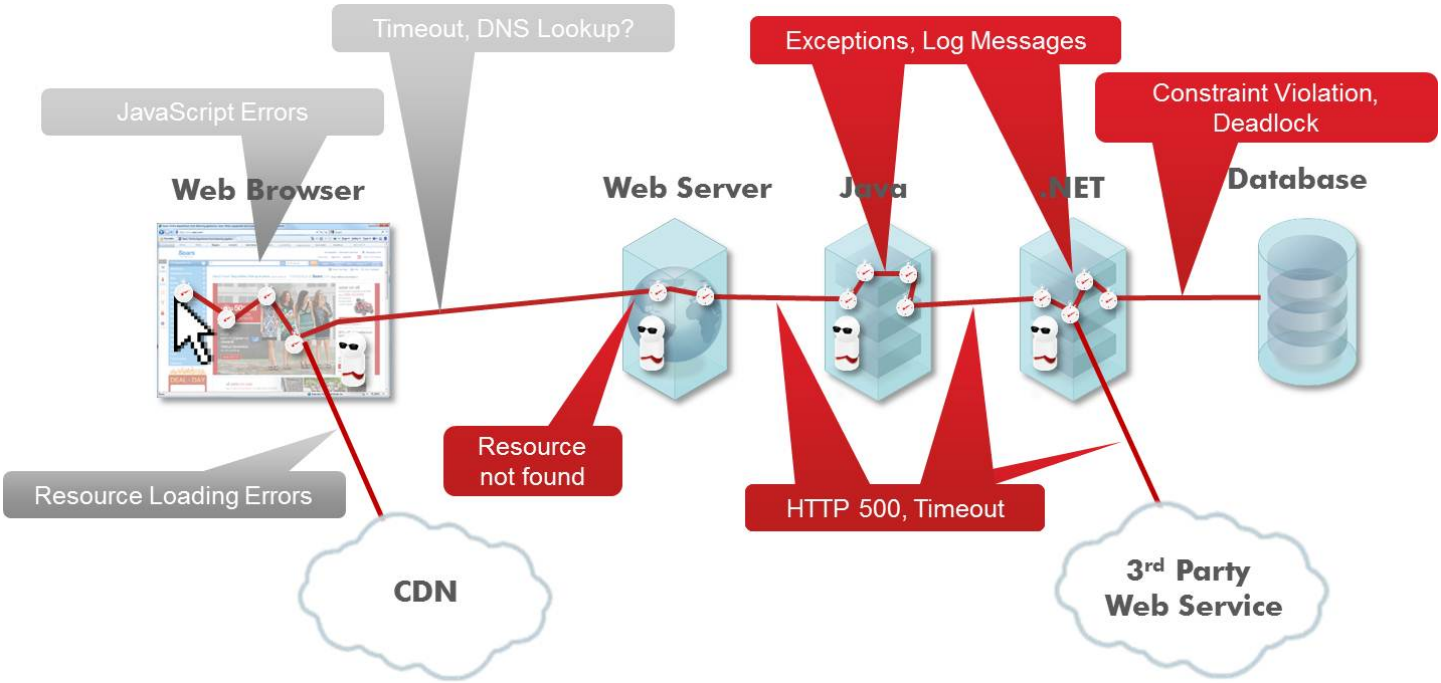
Watch and learn

Error analysis

Overview

An *error* can be any kind of unexpected event that happens while a user interacts with a system. Some examples of errors that can occur are resources that cannot be found on a web server, HTTP 500 response codes sent by a web service, or exceptions that occur within an application.

The figure below illustrates some errors and the parts of the system architecture they may affect.



Detecting errors

AppMon detects errors by defining rules based on captured HTTP response codes, Java/.NET exceptions, Java/.NET log messages, or JavaScript errors (browser errors). Each error detection rule further defines the consequences of the detected error. A detected error has one of the following impacts:

	The error is simply detected and does not have further consequences.
	The error indicates that the whole transaction (PurePath) where the error was detected has failed.
	The error indicates that the whole PageAction (including the transaction) to which the PurePath belongs has failed.

The following are examples of how rules can be defined:

- All HTTP 5*xx* response codes that are detected on a transaction entry point indicate errors that cause the whole PageAction to be marked as failed. PurePaths with HTTP 5*xx* response codes are also automatically grouped as an HTTP response sub-category in the Web UI as part of problem pattern detection. See Problem Pattern Detection and Memory Diagnostics for more information.
- All exceptions that happen on database calls indicate errors.

- When a log message with a certain text is captured, the transaction (PurePath) where this log message has occurred is marked as failed.

The AppMon installation includes a predefined set of error detection rules that can be easily extended by custom rules. See the System Profile - Error Detection for details on creating or revising error detection rules.

Monitoring functional health

AppMon evaluates error detection rules on all PurePaths that arrive at the AppMon Server, and applies the defined consequences. Transactions and user actions are marked as failed depending on the detected errors. AppMon presents the detected errors either *actively*, by raising incidents based on failed transaction rates or failed user action rates, or *passively*, by presenting error information in various dashlets and charts.

For alerting and charting based on failure rates, AppMon provides out-of-the-box measures subscribed as **Server Side Performance - Functional Health**. These measures are:

- Failed Transaction Count
- Failed Transaction Percentage
- Failed User Action Count
- Failed User Action Percentage

These measures reflect the overall count or percentage of failed transactions and user actions. In addition to the overall failed rates, each Business Transaction can track its own failure rate and count. Two auto-subscribed measures for each Business Transaction are configured to calculate its failure rates, one for the failed percentage and one for the failed count.

Error notifications

AppMon offers alerting for failed transactions and user action, either overall or for Business Transactions.

Out-of-the-box alerting based on the overall failure rates is enabled by default for a failed transaction rate of more than 3% within 5 minutes and for a failed user action rate of more than 3% within 5 minutes. These defaults can be changed in the Error Detection configuration of the System Profile.

For more advanced alerting scenarios, you can create an incident rule based on the measures for failed transactions user actions that are available for a Business Transaction. You can define incident rules to raise incidents if the failure rate of the Business Transaction exceeds a specified threshold. For example, an incident may be raised when more than 1% of all purchase transactions fail within a time range of 15 minutes.

Functional Health dashlet

It is essential to monitor the current functional health of a system and compare it with the past health, particularly after deploying a new release or patch. The starting point for monitoring errors is the Functional Health Dashlet dashlet, which provides:

- An overview of the failing transactions within a certain timeframe compared with another timeframe.
- An overview of the error hotspots that have been detected, which is the starting point for analyzing the root cause of the most common errors.

Custom error charts

All failed transaction count or rate measures, and all failed user action count or rate measures, can be added to charts. You can chart the overall failed count and rate measures as well as the failed count and rate measures for Business Transactions.

Error root cause analysis

After identifying errors, you need to find their cause.

The Errors Dashlet dashlet gives an overview of all errors that have occurred. From this dashlet, you can drill down to the transactions where the errors were detected. You can select the transaction to step through the errors that were detected on method calls of the transaction.

Another approach to root cause analysis is to use analysis dashlets such as Transaction Flow Dashlet to identify components that are causing errors.

Many other dashlets provide error information. For details about the information provided by these dashlets, see the following topics:

- Errors Dashlet
- Transaction Flow Dashlet

- Business Transactions Dashlet
- PurePaths Dashlet
- Entry Points Dashlet
- Web Requests Dashlet
- Web Services Dashlet
- Database Dashlet
- Remoting Dashlet
- Messaging Dashlet

Using sessions

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Import and export sessions

Live sessions

Offline sessions

Process configuration changes - re-analyze stored or offline session

Session comparison

Session encryption

Stored sessions

Watch and learn

Working with sessions

A *session* is a set of diagnosis data related to a specific System Profile.

Examples of a session are:

- **PurePath Session**: Combined information about PurePaths and Time Series within a given time period.
- **Memory Dump**: Analysis data about the number, size, and class of allocated objects and their references.
- **Thread Dump**: Collection of data for thread analysis, including CPU time information.
- **Sampling**: Statistical data for all threads, which can help to find Rules and entry points.

Sessions types

Depending on the location where a session is stored, there are three types of sessions.

- Live Sessions
- Stored Sessions
- Offline Sessions

Live sessions

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Stored sessions

Offline sessions

Using sessions

A *live session* is the most recent 5GB of diagnostic data for a System Profile and is stored on an AppMon Server. Each System Profile has one (and only one) live session. All PurePath sessions refer to live session data.

Live sessions are continuously persisted and streamed to disk at 10-second intervals if **Continuous Transaction Store** is enabled in **Server > Storage**.

Live sessions can be explicitly recorded and stored. Right-click the desired System Profile and select **Cockpit > Session Storage > Start / Stop Recording** or **Cockpit > Session Storage > Store Session**. Stored Sessions are available through **Cockpit > Session Storage** and accessible via the AppMon Server.

Stored sessions

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Creating stored sessions

Move session storage to a new disk or directory

Stored session states

Live sessions

Offline sessions

Using sessions

A *stored session* is a memory dump, a thread dump, a sampling, or a former PurePath session that is stored on the hard drive. Stored PurePath sessions also include time series data. Stored sessions can be located via an AppMon Server at an AppMon Client in the **Session Storage** tree in the client Cockpit.

Exported sessions are stored sessions zipped up in a single `.dts` file. Multiple stored sessions can correspond to one System Profile.

Store session context menu - explicit storage

You can manually store a session to disk.

To save PurePaths or Time series data of a specified time frame to extra files, select **Store Session** or **Export Session** from the System Profile's **Session Storage** context menu.

- **Store Session** saves session components in files (`.dtsf`) in a `YYYYMMDDHHMMSS_n.session` directory in the `sessions\stored` folder specified in Server settings.

- **Export Session** zips up stored session files (`.dtsf`) of a `YYYYMMDDHHMMSS_n.session` directory into a single `.dts` session archive to the path specified, from where it can easily be distributed.

Make available offline

When you right-click a stored session and click **Make Available Offline**, the session is exported and immediately imported as an offline session into your Client, accessible without an AppMon Server.

Creating stored sessions

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Stored sessions

Using sessions

You can create a stored session using any of the following methods:

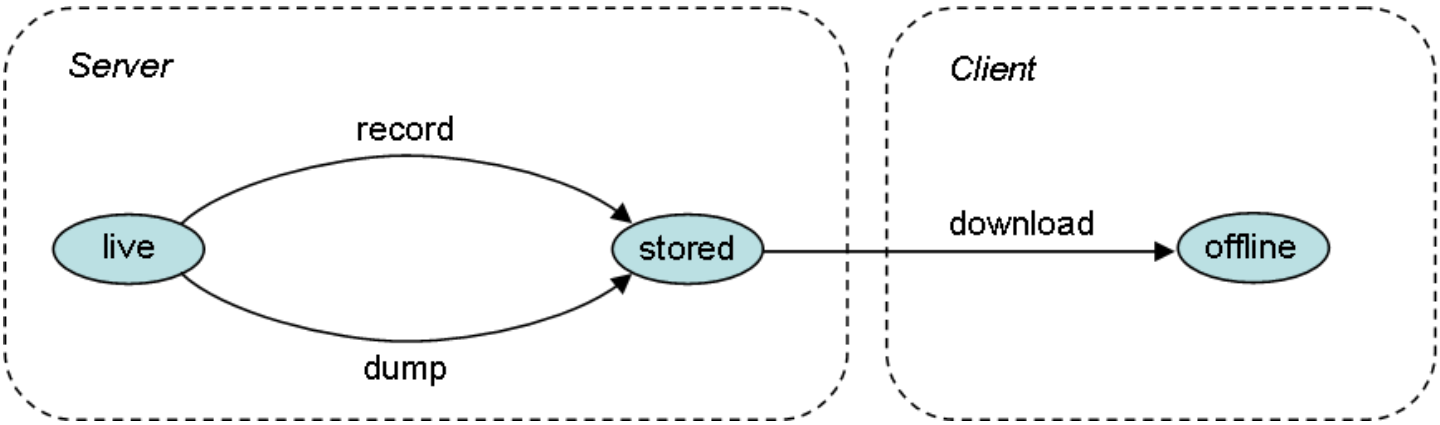
- **Recording**: You can create a stored session by recording a PurePath session or a sampling.
  - To record a PurePath session manually, select **Session Storage > Start Recording** from the live session's context menu, then **Session Storage > Stop Recording** to end the recording.
  - To capture samplings, select **Start Sampling** and **Stop Sampling** within the *Sampling* dashlet.
- **Dumping**: *Dumping* is the process of creating a snapshot of the live PurePath session, of an System Under Diagnosis (SUD)'s memory, or of an SUD's thread state.
  - To transfer the live PurePath session to a stored session, select **Session Storage > Store Session** from the live session's context menu.
  - You can create a memory dump or a thread dump through the Total Memory, Selective Memory, or Threads Dashlet dashlet.
- **Download**: You can download a stored session from an AppMon Server to an AppMon Client by using the command **Make Available Offline** on the context menu of a stored session.

Each stored session contains a copy of the System Profile that was valid when the session was stored. You can view and edit the System Profile of stored sessions, although some changes may have no effect (e.g. disabling or (un-)placing sensors).

Note

When a System Profile of a stored session is manipulated, a backup of the original System Profile is created to avoid loss of information about the System Profile's state at the time of session storage.

The following figure illustrates the sequence of creating, storing, and downloading a session.



Session usage

Move session storage to a new disk or directory

The end of the file may have some errors about why the server couldn't initialize the location you picked.

If the new location has been reverted, check the `Server.0.0.log` in the System Information dashlet. **Server Name > Support Information > Log Files > Server.0.0.log**.

If sessions are not found on restart, check the *Storage* link in the settings again to make sure the new location is set.

Troubleshooting

If you have no data in your session store or don't intend to keep it, simply follow steps 1-4 and when prompted to restart, choose Yes.

If you have no existing data

1. Open the client and navigate to the AppMon Server settings (**Settings > Dynatrace Server**).
2. Click "Storage" to configure.
3. Enter the new location of the session store (Absolute path or relative to the `{install}/server/` directory).
4. Click **Apply**. When prompted to restart the server, choose Later.
5. Stop the AppMon Server either manually or through the client (right-click System Profiles > Shutdown Server…).
6. Move the contents of the original directory (for example, `{install}/server/sessions`) to the new directory (must be created).
7. Start the Server.
8. Verify the stored sessions still show in the Server's Session Storage on the client.

Instructions

The default location is in your installation directory `{install}/server/sessions/stored`. For production installations it is recommended to reconfigure this location to be an absolute path to a directory outside of the installation directory to eliminate the need to move the storage with each AppMon upgrade.

Stored session states

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Stored sessions

Offline sessions

Using sessions

Stored and offline sessions can be in different states within their life cycle.

Possible states for **stored or offline** sessions are:

- **In progress**: The session is currently being created. Only stored sessions can be *in progress*, because the creation only happens on the AppMon Server.
- **Finished**: The session is finished. It contains all expected data.
- **Incomplete**: The session is finished, but not all expected data might be available. Sessions in this state are fully readable, but some data might be missing. For example, the AppMon Server went offline before session recording was finished.
- **Corrupt**: The session did not finish successfully. Sessions in this state cannot be read and will be automatically deleted during the next server startup.

Offline sessions

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Live sessions

Stored sessions

Using sessions

If a stored session is saved on the client machine, it is also called an *offline session*. The session can be accessed without connecting to an AppMon Server.

Import and export sessions

Looking for the 6.5 version of this page? Open in 6.5 documentation

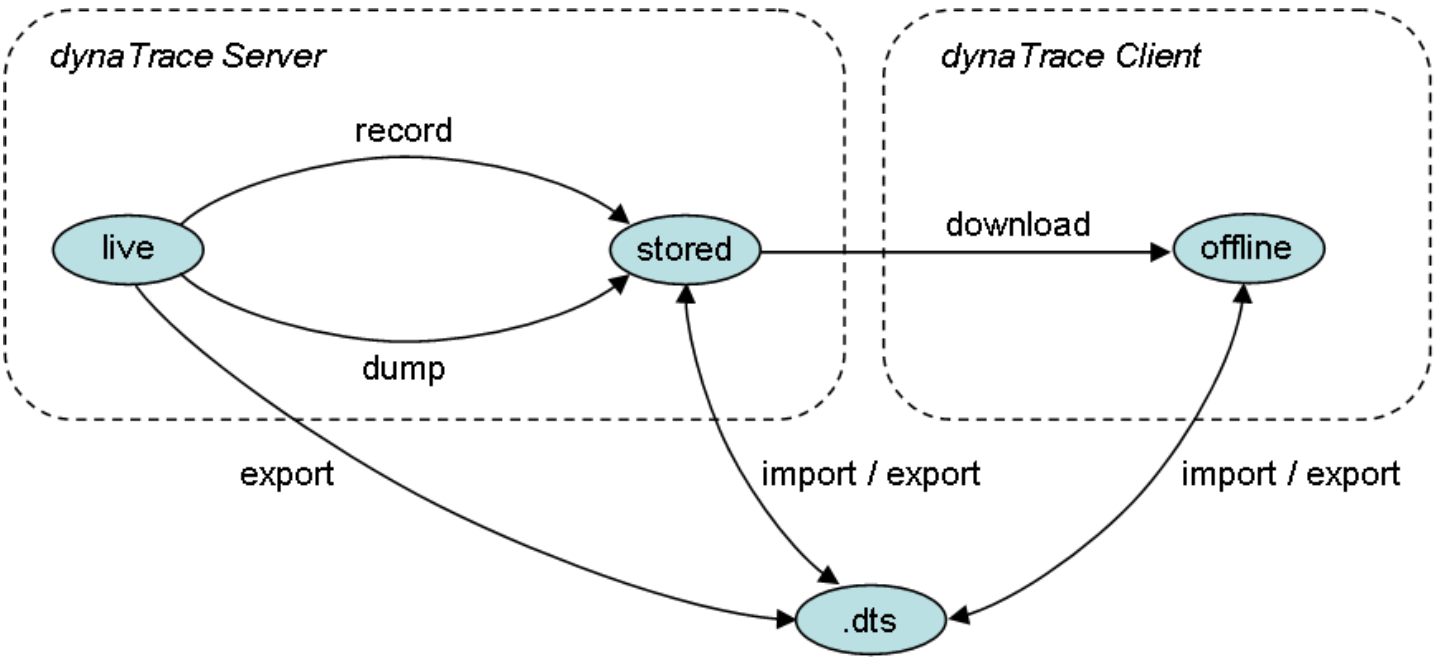
Related pages

Using sessions

Every session — live, stored, or offline — can be exported to a `.dts` archive (Dynatrace session file). A stored session can be restored by importing a `.dts` archive. You can import and export sessions by using the Cockpit Dashlet or the Session Browser Dashlet dashlet.

The **Export Session** command zips up the stored session files (`.dts`) at the specified location.

The **Import Session** command retrieves the session from the file system and makes it available in **Cockpit > Offline > Local Sessions**. The import command is also available through a button on the client toolbar and as an entry on the Client's **File** menu.



Importing and exporting sessions

Session comparison

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Usage

Overview

AppMon provides advanced session comparison functionality for diagnostic analyses.

The following analysis dashlets support comparison:

- Agent Breakdown Dashlet
- API Breakdown Dashlet
- Business Transactions Dashlet
- Components Dashlet
- Database Dashlet
- Entry Points Dashlet
- Memory Allocations Dashlet
- Messaging Dashlet
- Methods Dashlet
- Naming Services Dashlet
- Remoting Dashlet
- Tagged Web Requests Dashlet
- Web Requests Dashlet
- Web Services

A differential diagnostics analysis that is executed on two sessions, referred to as the *main* session and the *compare* session, is processed in the following way:

1. For each session, the corresponding regular diagnostics analysis is executed.
2. The results of these two analyses are matched against each other.

3. All matched and all unmatched records form the output of the differential analysis.

The differential analysis is not based on session data directly, but acts as a layer above the regular analysis. This allows a comparison of sessions located on different AppMon Servers without any connection between them. The comparison of the two intermediate AppMon Server results is done on the client.

When two records are matched, their characteristic values are checked for equality. A record's *characteristic values* correspond to its static content (e.g. the API name of an API Breakdown analysis record), as opposed to the *variable content* (e.g. the timings).

While the static content of a differential result record is displayed unvaried, the variable content is presented in the following way:

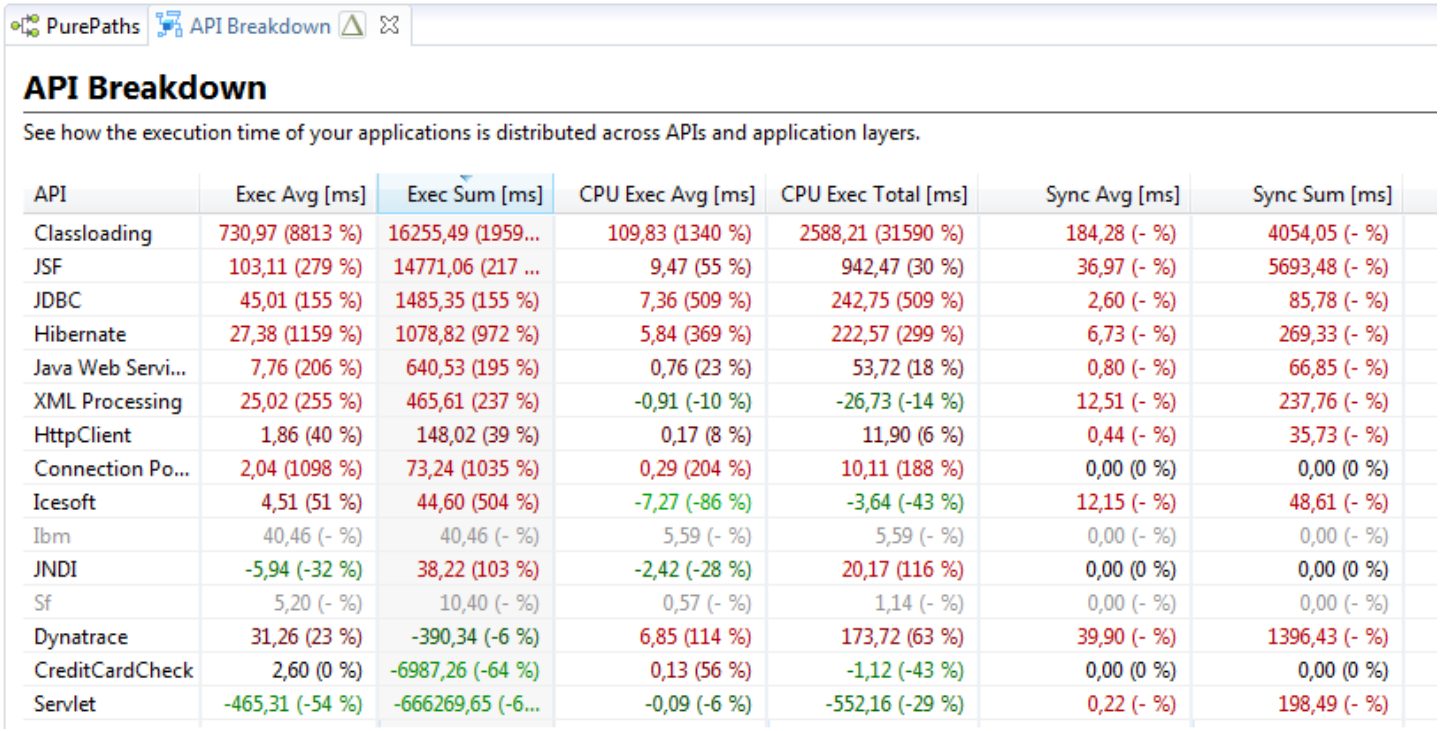
```
difference ( percentage % )
```

where:

```
difference = mainValue – compareValue
percentage = ( difference / compareValue ) * 100
```

Take note of the following:

- Comparing two empty values (-) yields an empty difference and value: `- (-%)`.
- If only one value is empty, it is treated as `0`, but no percentage is displayed: `xyz (-%)`.
- Sorting is based on the difference value, because this usually gives a better idea of the relevance of a given difference among all other values.
- Shading is based on the percentages, with negative values in green and positive values in red. The higher the values are, the more saturated the colors.
- If a record cannot be matched, it is shaded in gray and displays only the difference relative to `0`.



PurePaths API Breakdown

API Breakdown

See how the execution time of your applications is distributed across APIs and application layers.

API	Exec Avg [ms]	Exec Sum [ms]	CPU Exec Avg [ms]	CPU Exec Total [ms]	Sync Avg [ms]	Sync Sum [ms]
Classloading	730,97 (8813 %)	16255,49 (1959...	109,83 (1340 %)	2588,21 (31590 %)	184,28 (- %)	4054,05 (- %)
JSF	103,11 (279 %)	14771,06 (217 ...	9,47 (55 %)	942,47 (30 %)	36,97 (- %)	5693,48 (- %)
JDBC	45,01 (155 %)	1485,35 (155 %)	7,36 (509 %)	242,75 (509 %)	2,60 (- %)	85,78 (- %)
Hibernate	27,38 (1159 %)	1078,82 (972 %)	5,84 (369 %)	222,57 (299 %)	6,73 (- %)	269,33 (- %)
Java Web Servi...	7,76 (206 %)	640,53 (195 %)	0,76 (23 %)	53,72 (18 %)	0,80 (- %)	66,85 (- %)
XML Processing	25,02 (255 %)	465,61 (237 %)	-0,91 (-10 %)	-26,73 (-14 %)	12,51 (- %)	237,76 (- %)
HttpClient	1,86 (40 %)	148,02 (39 %)	0,17 (8 %)	11,90 (6 %)	0,44 (- %)	35,73 (- %)
Connection Po...	2,04 (1098 %)	73,24 (1035 %)	0,29 (204 %)	10,11 (188 %)	0,00 (0 %)	0,00 (0 %)
Icesoft	4,51 (51 %)	44,60 (504 %)	-7,27 (-86 %)	-3,64 (-43 %)	12,15 (- %)	48,61 (- %)
Ibm	40,46 (- %)	40,46 (- %)	5,59 (- %)	5,59 (- %)	0,00 (- %)	0,00 (- %)
JNDI	-5,94 (-32 %)	38,22 (103 %)	-2,42 (-28 %)	20,17 (116 %)	0,00 (0 %)	0,00 (0 %)
Sf	5,20 (- %)	10,40 (- %)	0,57 (- %)	1,14 (- %)	0,00 (- %)	0,00 (- %)
Dynatrace	31,26 (23 %)	-390,34 (-6 %)	6,85 (114 %)	173,72 (63 %)	39,90 (- %)	1396,43 (- %)
CreditCardCheck	2,60 (0 %)	-6987,26 (-64 %)	0,13 (56 %)	-1,12 (-43 %)	0,00 (0 %)	0,00 (0 %)
Servlet	-465,31 (-54 %)	-666269,65 (-6...	-0,09 (-6 %)	-552,16 (-29 %)	0,22 (- %)	198,49 (- %)

Example session comparison

Usage

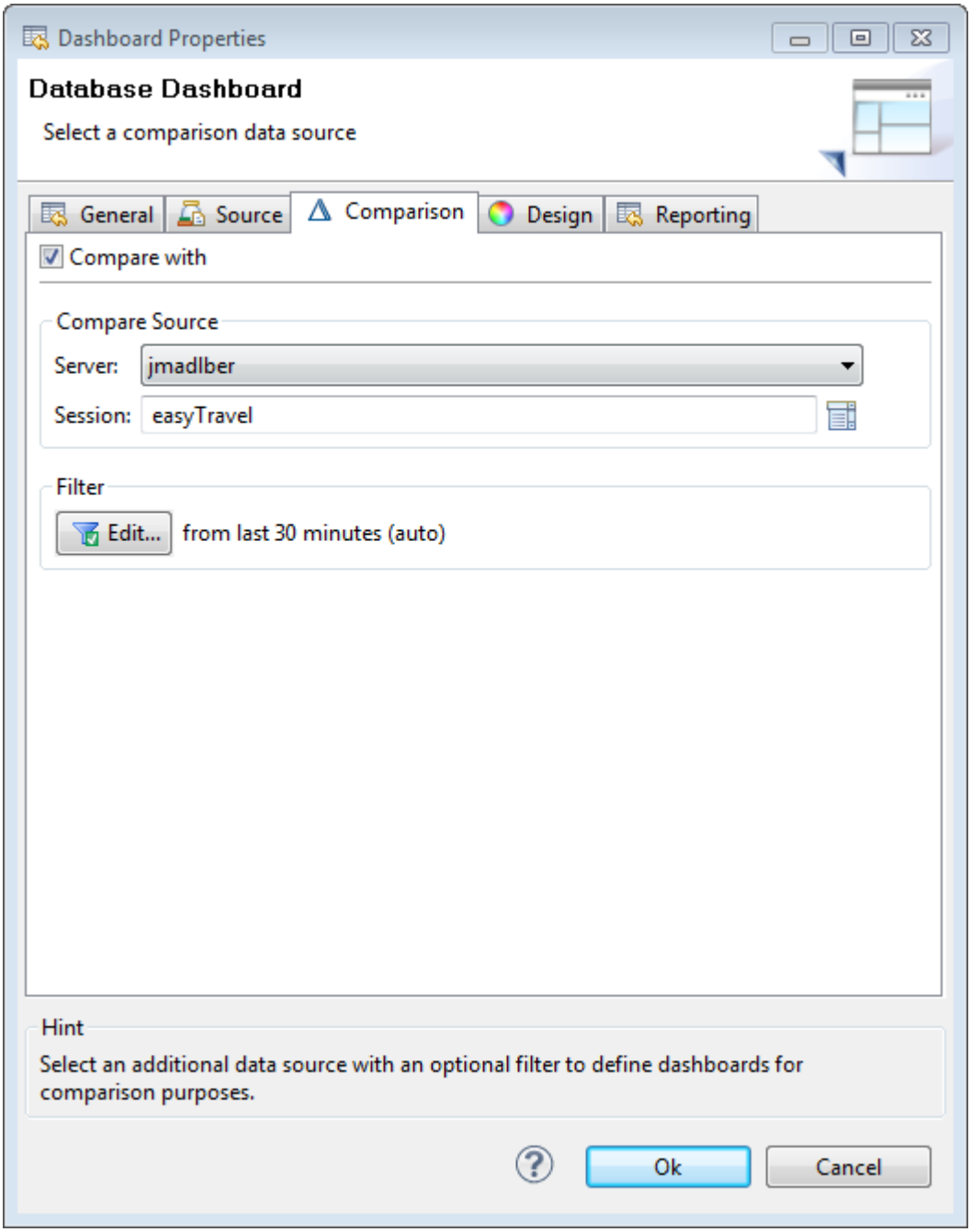
This section first explains how to configure the comparison settings, then describes the available comparison display variants.

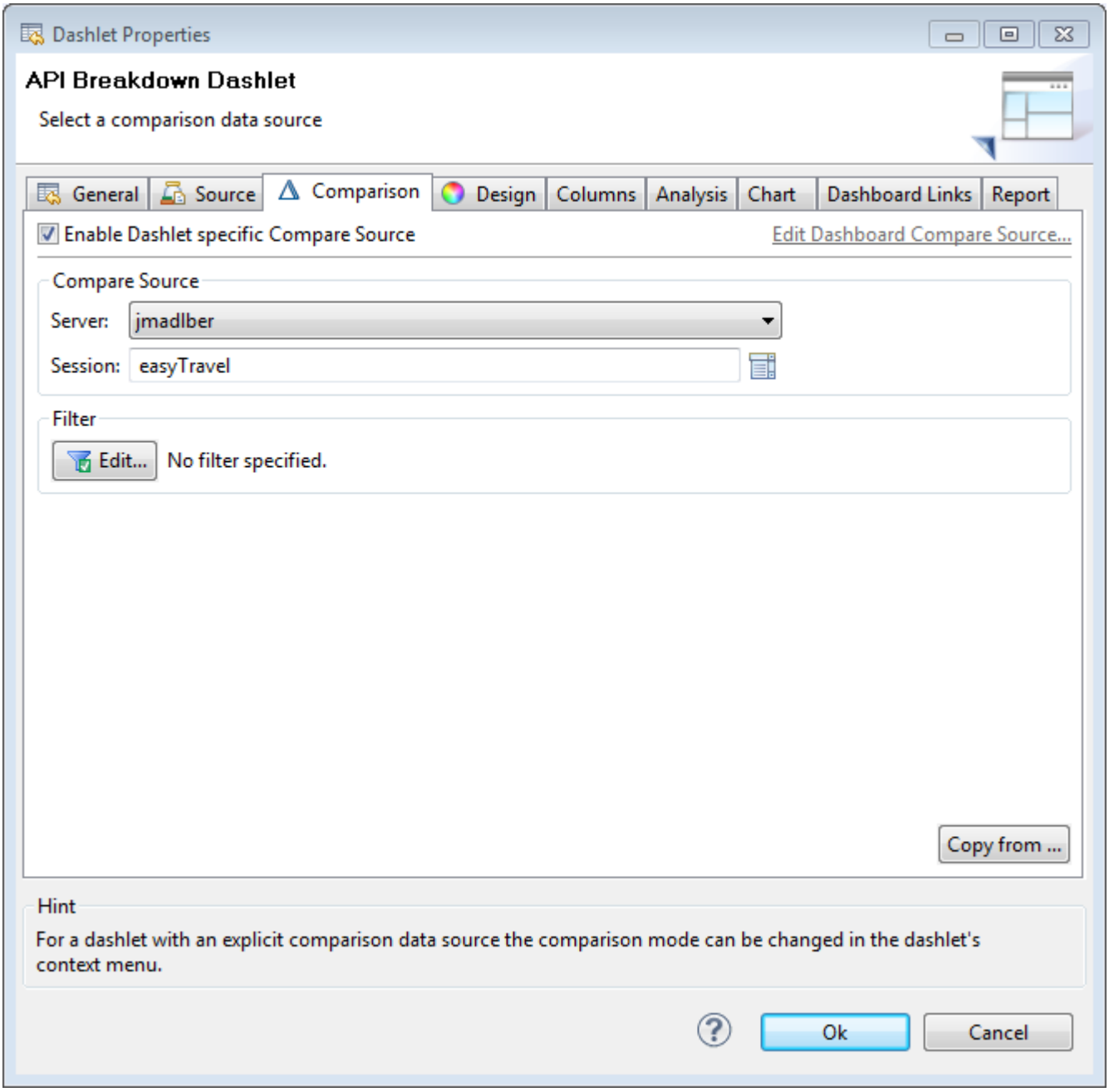
Configuration

Session comparison can be set globally for a dashboard in the dashboard's configuration. It can also be set for each dashlet, overriding the inherited dashboard configuration.

To configure the comparison, open the **Properties** dialog box for the dashboard or the dashlet, and select the **Comparison** tab. The comparison data source can be configured just like the main data source in the **Source** tab, but is completely independent from it. None of the filter criteria applied to the main data source, either dashlet-specific or inherited, are applied to the compare data. However, filters for the compare data can be configured just like the regular filters, and will also inherit any configuration from the dashboard's **compare** filter.

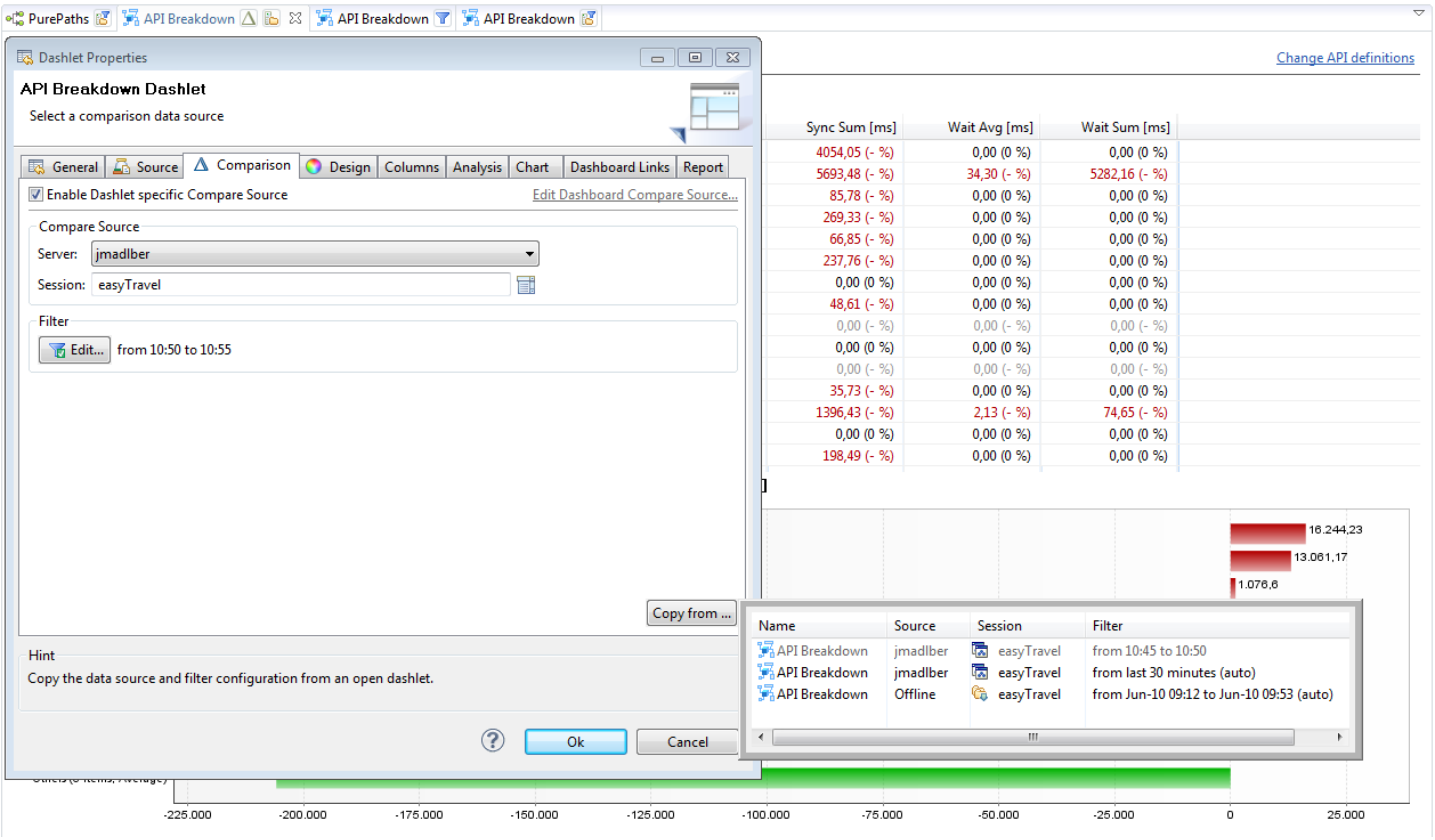
The following figures show the **Comparison** tabs for dashlet and dashboard.


Dashboard Properties
Database Dashboard
Select a comparison data source
General
Source
Comparison
Design
Reporting
Compare with
Compare Source
Server:
jmadlber
Session:
easyTravel
Filter
Edit...
from last 30 minutes (auto)
Hint
Select an additional data source with an optional filter to define dashboards for comparison purposes.
Ok
Cancel



You can copy the main data source settings including the filter setup from an open dashlet by clicking **Copy from** and selecting an entry.

The entry shaded in gray is the dashlet that is being configured.



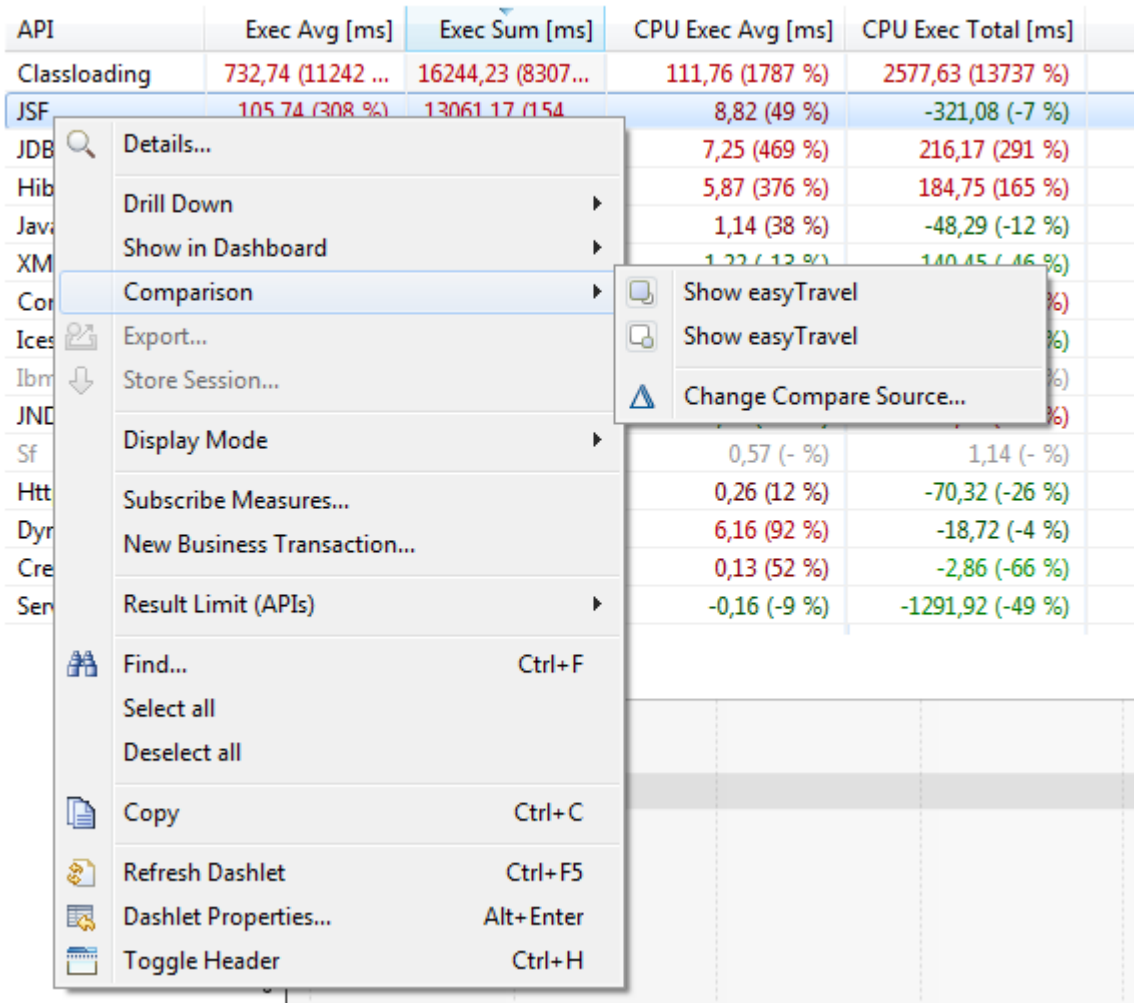
Copying Comparison Settings

Comparison display variants

All dashlets that support comparison can switch the data being displayed. Besides displaying the results of the differential analysis, a dashlet can show the results of the two regular analyses.

To switch the display, select **Comparison** from the context menu as shown below.

The currently displayed data has no further influence on the behavior of the dashlet. When you refresh the dashlet, the two primary sources will be analyzed again and the differences computed.



Context Menu - Comparison

Process configuration changes - re-analyze stored or offline session

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Stored sessions

Offline sessions

Using sessions

Stored and offline PurePath sessions can be adapted to changing needs. For example:

- The underlying measures and thresholds for your Business Transactions may be created or reconfigured.
- PurePath-related measures can be added.

After changes to the session's System Profile, all PurePaths have to be re-analyzed. Right-click **Session** and click **Process Configuration Changes**. The values for charting and Business Transactions may change.

When you use **Process Configuration Changes**, the values are stored back into the Session. By contrast, **Reanalyze all PurePaths** on the context menu for the **Business Transaction** dashlet only analyzes the PurePaths and tags them with the Business Transaction, but it does not persist any measures back to the stored session.

Process Configuration Changes does **not** affect system monitoring or time series data stored in the Performance Warehouse, such as CPU or memory usage. No values will be created if they were not recorded before.

Note

Visit-based business transactions cannot be reanalyzed.

Session encryption

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Encryption options on OSes

Related pages

Using sessions

Use file system or drive-level encryption for your session data if you need to protect them.

Additionally, you can hide strings in PurePaths that should be kept confidential when the data is captured.

Encryption options on OSes

OSes are different in what they offer regarding file-system-level encryption of your session data. As session data are quite large with high write rates it may be worthwhile to consider native encryption at the drive level (hardware permitting).

Windows

Windows (Ultimate, Enterprise, Server 2008 and up) BitLocker is an option. It needs a TPM though.

TrueCrypt was declared insecure by the developers themselves.

Linux

The dm-crypt subsystem is an option on Linux.

It's assumed that you have a partition available which can be completely erased. In this example we use /dev/sdb1 as identifier.

Make sure you have dm-crypt available in your kernel. Then install cryptsetup:

```
sudo apt-get install cryptsetup
```

Prepare you partition encryption:

```
sudo cryptsetup create -y dtsession /dev/sdb1
```

By confirming your passphrase twice you've completed setting up the partition-encryption. Next step is to create a filesystem. In this example we use ext4, but others should do as well:

```
sudo mkfs.ext4 /dev/mapper/dtsession
```

This will take a while, depending on how much space is available on the drive. Now the mountpoint has to be created and the partition has to be mounted:

```
sudo mkdir /mnt/dtsession
sudo mount /dev/mapper/dtsession /mnt/dtsession
sudo chown -R USER:<UserAccountDynatraceServerIsRunningUnder> /mnt/dtsession
```

Finally the AppMon Server must be configured to use the newly created session storage. In the AppMon Client, select **Settings > Dynatrace Server… > Storage vertical tab** and put the path in the **Stored Sessions Directory** field (in this example: `/mnt/dtsession/sessions/stored`). You must restart the AppMon Server to use the new storage location.

Amazon EC2

Encryption can be enabled when creating a new EBS volume. In the AppMon Client, select **Settings > Dynatrace Server… > Storage vertical tab** and put the EBS path in the **Stored Sessions Directory** field.

Baseline and smart alerting explained

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Significant measurement

Baseline calculations

Violations and alerting

Related pages

Application Overview dashboard

Baseline chart

Business Transaction details

Error analysis

AppMon provides automatic, *smart* baselines. The smart alerting mechanism reduces alerts on false positives, which can occur with traditional statistical approaches that look only at averages and violations based on standard deviation.

A baseline is calculated:

- For every splitting value of the Business Transactions that are selected for the Application Overview.
- For response time (median and slowest 10%), failure rate, and throughput. See Business Transaction Details.

Significant measurement

AppMon employs statistical methods to calculate expected application behavior from historical data and to compare current application behavior against the expected behavior.

Measurements for throughput, failure rate, and response time are regularly collected. Over time, the sample size increases to the point where a measurement gains statistical significance. Intermediate measurements are not statistically significant, but they still reflect the current behavior of the application.

The charts for throughput, failure rate, and response times show both significant and insignificant measurements.

The following screenshot shows that two significant measurements lead to a violation that also triggers a built-in incident. Settings for the Business Transaction and Business Transaction Splitting value allow you to adjust that default behavior.



Significant Measurement

Baseline calculations

Baselines are calculated based on different statistical approaches for response time, failure rate, and throughput.

Note

When you analyze chart data, take into account that if the number of data points becomes too high for the chart “real estate”, data is aggregated automatically. Keeping the one-minute resolution over a large time period would result in indistinguishable point clumps and excessive memory consumption.

Response time

The Response Time depends on the Business Transaction selected in the Application Overview. For a Server-side Business Transaction, it is the response time for the Server-side PurePath. If it is a user-action-based Business Transaction, it is the response time of the whole user action.

AppMon does not use averages or standard deviation to calculate baselines for response time. Instead, baselines are…

- Calculated for the 50th percentile (median) and 90th percentile (slowest 10 %) of the actual Business Transaction response time.
- Updated every 5 minutes for the first day; every day at midnight through the previous 7 days (for as many days as there is data until AppMon has 7 days of data).

Violations are identified if at least two significant measurements are above the threshold.

As the baseline is calculated every day using the data of the past 7 days, it automatically adapts to changes in your application.



Factors that affect the response time baseline

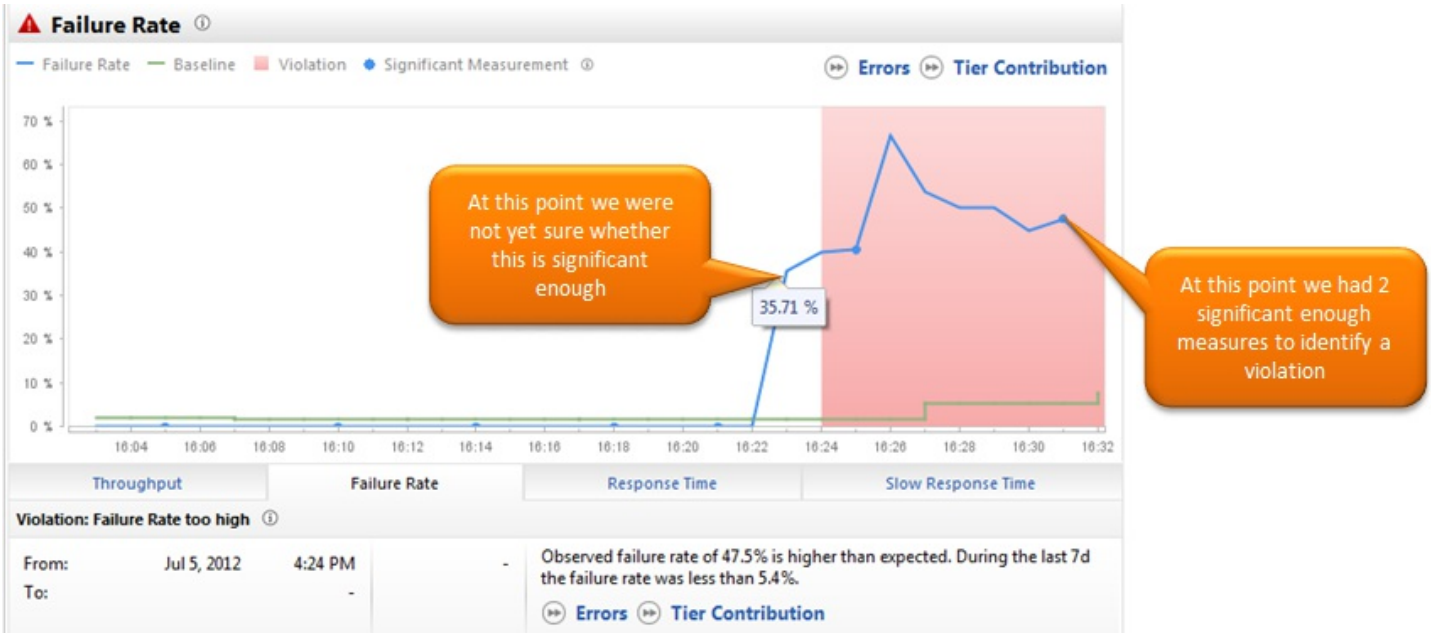
Failure rate

The failure rate is calculated based on the detected errors that identify a transaction or user action as failed. For details, see Error Analysis.

The baseline for failure rate uses a binomial distribution.

Violation detection is based on *significant measurements*. For example:

- One failed out of five requests is a 20% failure rate, but this may not be significant during a low-traffic time range such as late night.
- 100 failed out of 1000 requests — a 10% failure rate — during a high-load time range is significant, so an alert is issued.



Significant failure rate

Throughput

Throughput is the number of transactions that belong to the specific Business Transaction splitting value, such as the number of processed web requests for the URL `http://myapp.com/search.jsf`.

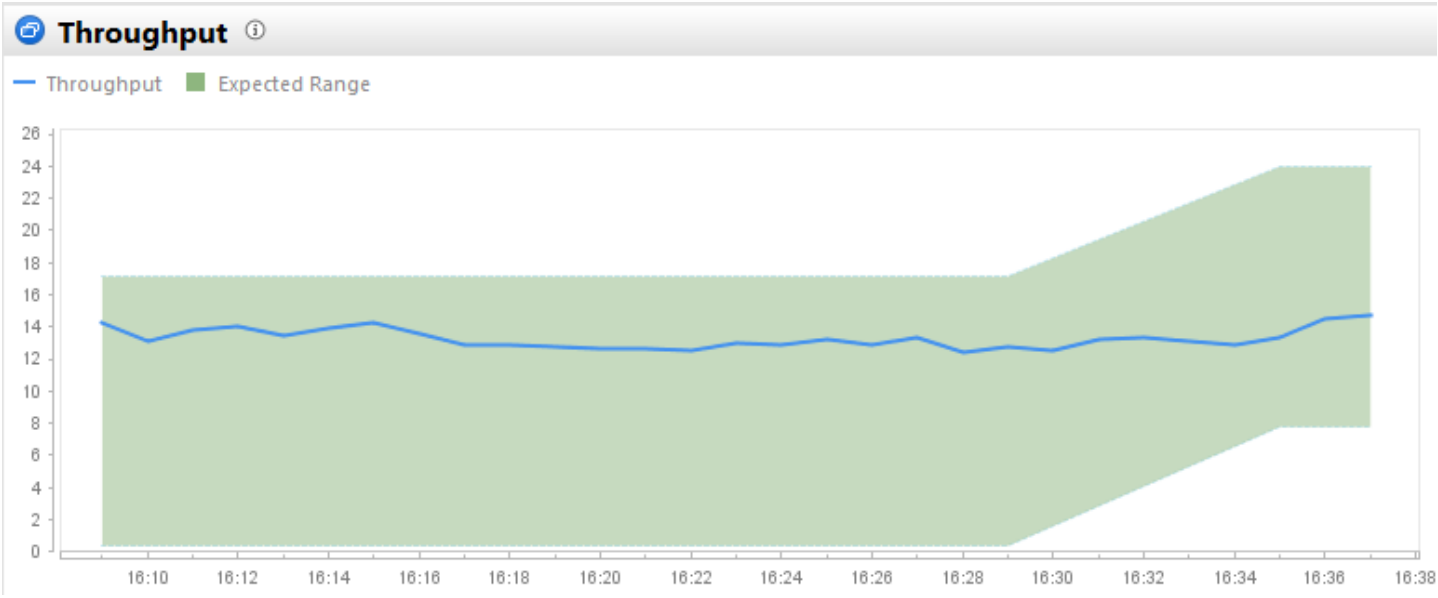
Business transactions with high throughput have significant measurements more frequently compared to business transactions with lower throughput. This is because it takes shorter intervals to gain a sample size large enough for statistical significance.

For the throughput baseline, AppMon calculates the expected range based on the historical data from the same time frame one week ago.

The initial phase of baselining, when AppMon does not have data from one week ago, uses different time frames, depending on how much data is available:

- The same 15 minute interval 7 days ago
- The same 15 minutes interval 1 day ago
- The same 15 minutes interval 1 hour ago
- The previous 15 minutes for the first hour when baselining started

Alerts are not issued for violations of the expected range for Throughput. You must check whether throughput is within the same range as for the same time frame from one week ago.



Throughput baseline

Throughput influences the measurements for failure rate and response time. If throughput is high, there are more data points for calculating the failure rate and response time and therefore more significant measurements (lower statistical spread). For low throughput, the statistical spread is higher, so the failure rate and response time measurements become less significant.

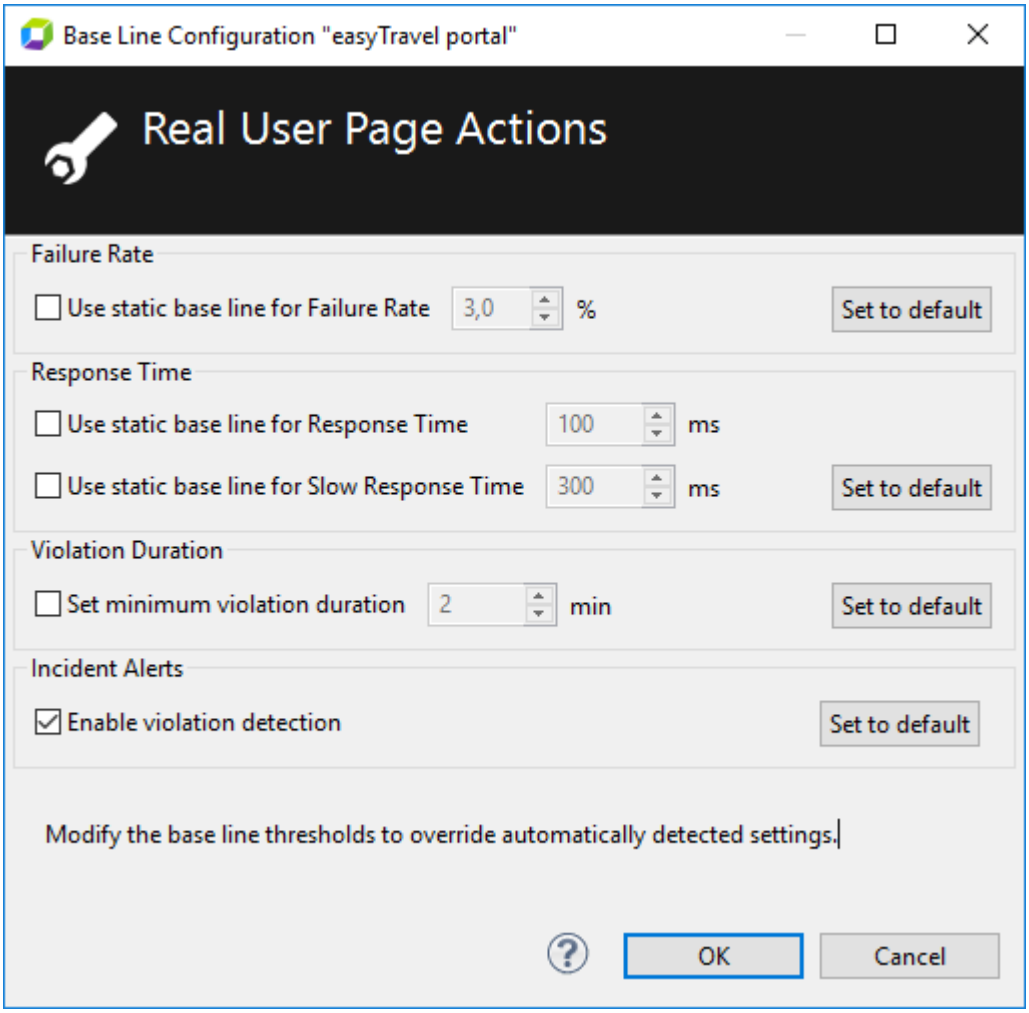
Violations and alerting

AppMon issues alerts for baseline violations for response time and failure rate.

By default violations are detected when two or more significant measurements violate the baseline. A violation ends when at least one of the violating significant measurements falls below the baseline.

These settings can be changed for Business Transactions and for splitting values.

You can set absolute and relative thresholds for violations in the **Configure Business Transaction** dialog box. You may need to scroll down to see the button. See Business Transaction configuration for more information on configuring Business Transactions.



Base line configuration

Incidents and alerting

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Configuring Incident Rules

Conditions

Actions

Linked dashboard

Related pages

Incidents dashlet

Server measures and server incidents

Incident downtimes

Built-in incidents

Watch and learn

Managing incidents and alerts

After subscribing to measures and configuring thresholds for the measures, create incident rules to define whom to notify when the thresholds are exceeded and how to act on violations.

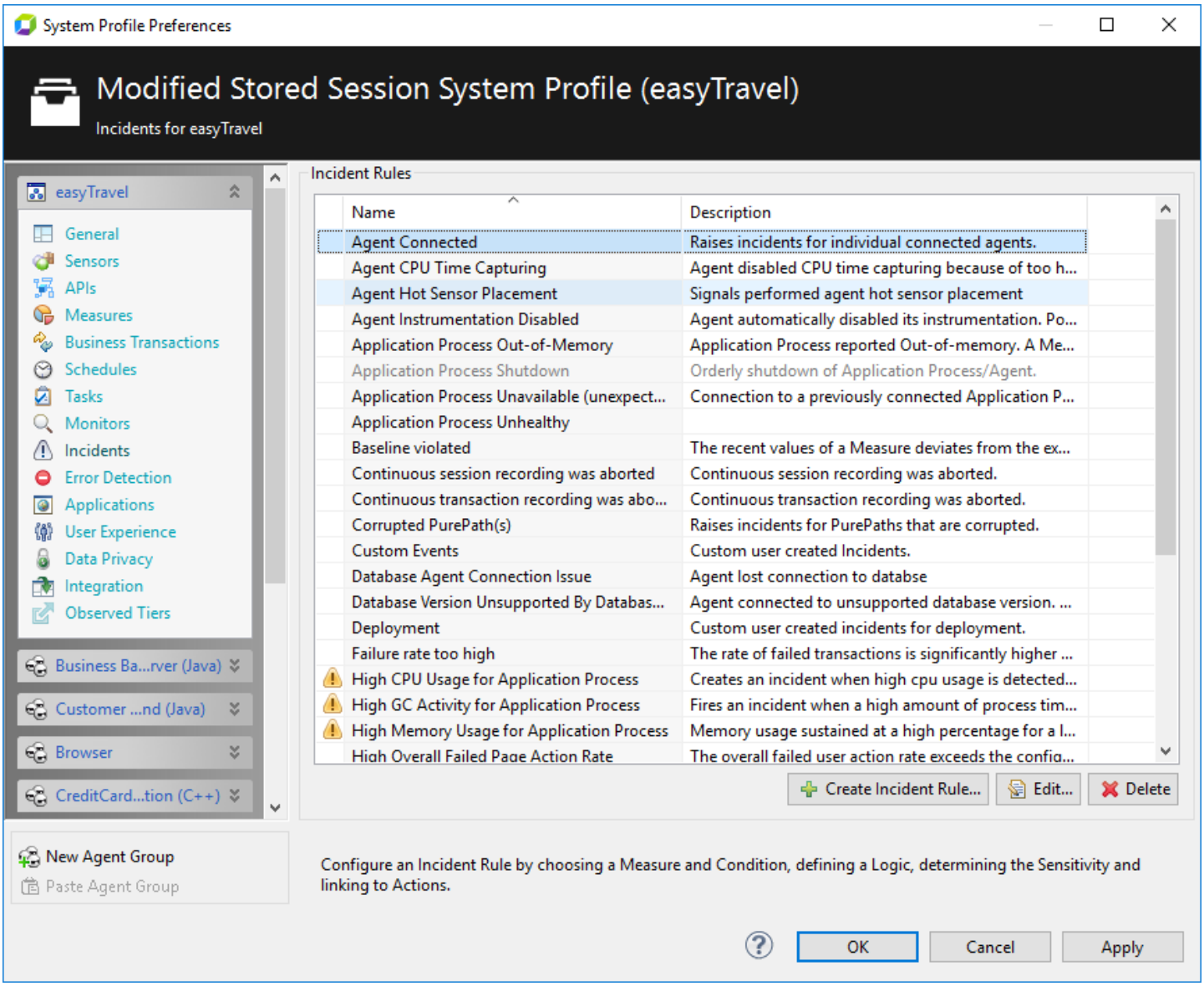
An *incident rule* in AppMon is a mapping between threshold violations of measures and the actions taken. Incident rules can be based on multiple measures and different severity levels of threshold violations to trigger notifications. Threshold checks are done every ten seconds. The threshold may be for cyclic measures like CPU and memory that are sampled every ten seconds, or for transactions.

If all the thresholds defined in the incident rule are violated, this is shown in the Incidents dashlet, even if no actions have been configured for the incident rule.

Incident mains are grouped by incident. Incidents occurring in close temporal proximity are grouped together in one mail and sent separately to each subscriber.

Configuring Incident Rules

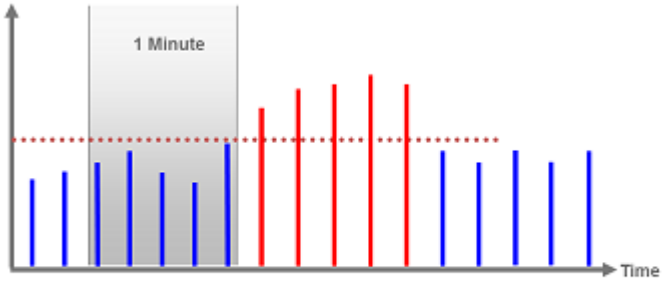
Rules are configured in the System Profile. Right-click the System Profile, and select **Edit System Profile > Incidents**.



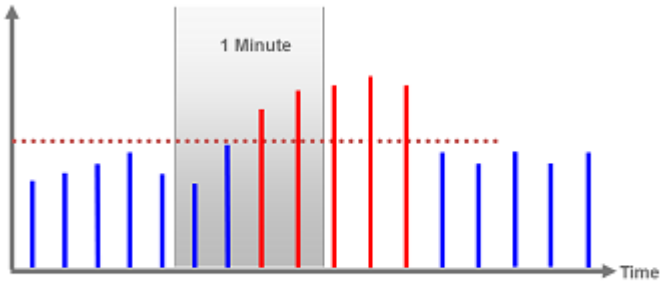
Incident Rules pane

Use the **Create Incident Rule** dialog box to define the rule properties.

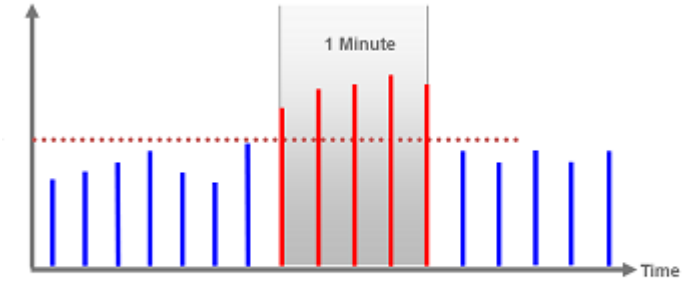
- **Name**: Give the rule a unique name that describes its purpose.
- **Description**: Optionally, provide a description of the rule, e.g. the measures associated with it and the action that occurs if the rule is violated.
- **Evaluation Timeframe**: Select the duration to be used to evaluate if the defined conditions have been met.
  For example, if you select a timeframe of one minute and use a PurePath duration measure with an average aggregation as input for this incident rule, the average PurePath duration of the last minute is calculated and checked for violation every 10 seconds. Measures remain in memory for one hour. The following figures illustrate how the timeframe is used to check for incidents.



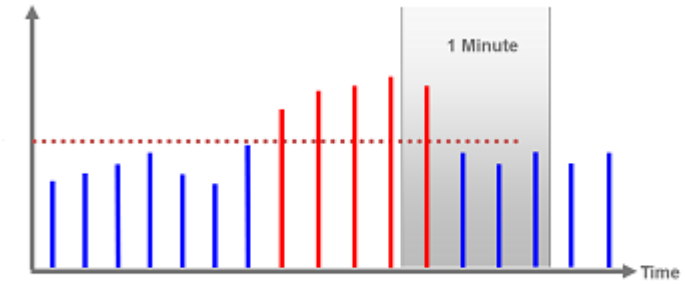
1.6 sec average



1.9 sec average



2.5 sec average
Incident condition fulfilled



2.0 sec average
Incident condition still fulfilled

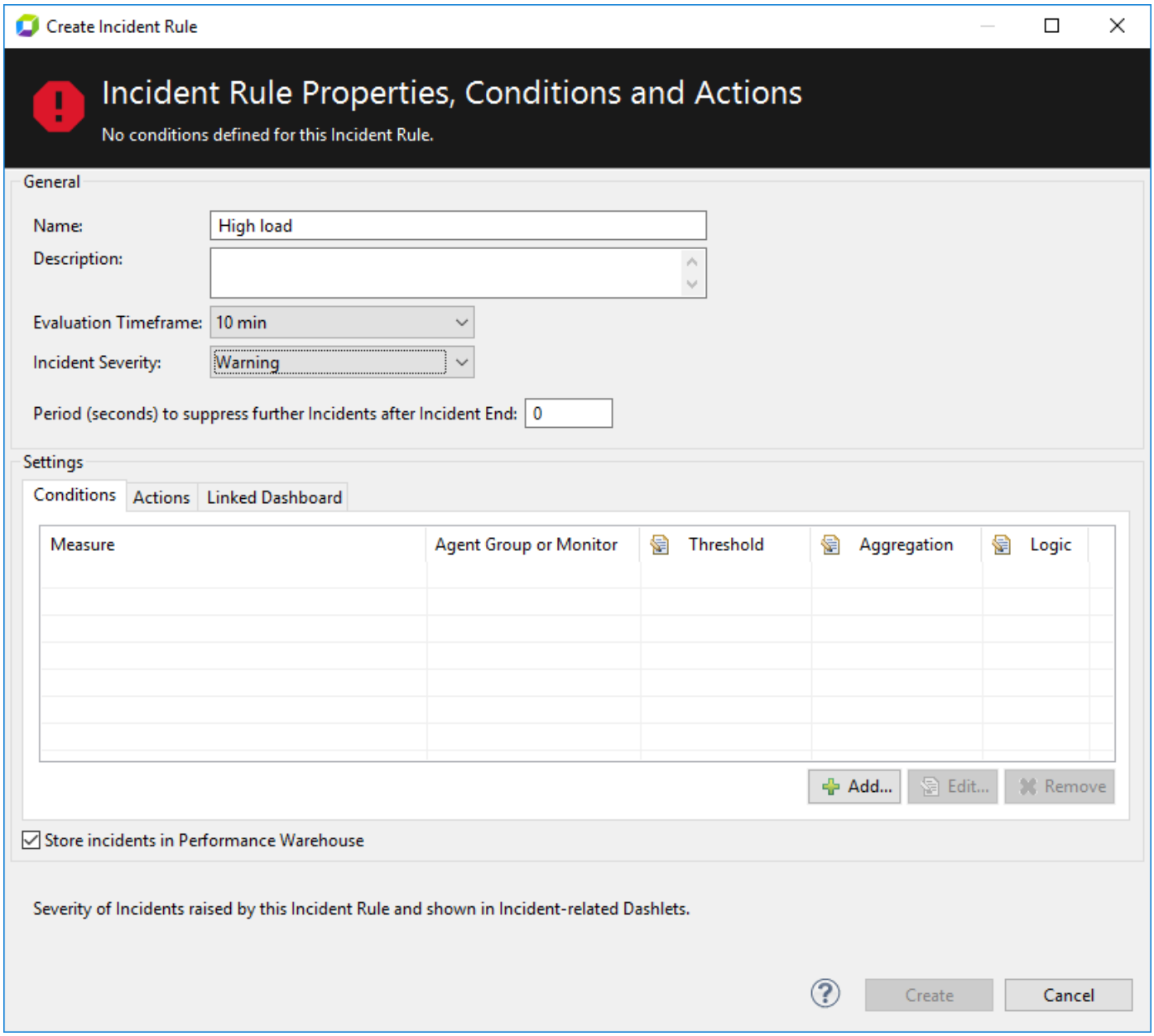
Incidents are closed only when their condition hasn't been met for a minute, to better deal with measurements oscillating around their thresholds. Once this timespan has passed without further violation, incidents are closed with the closing time set to fit actual measurements.

- **Incident Severity**: Choose the severity of incidents that will be shown in incident-related dashlets: **Informational**, **Warning**, or **Severe**. In general you should select the level that requires a response to notifications.
- **Incident Suppression**: Configure the incident delay in seconds. Incidents are suppressed during the configured number of seconds after an occurrence, to avoid sending redundant notifications.

Click to view an example of incident rule creation

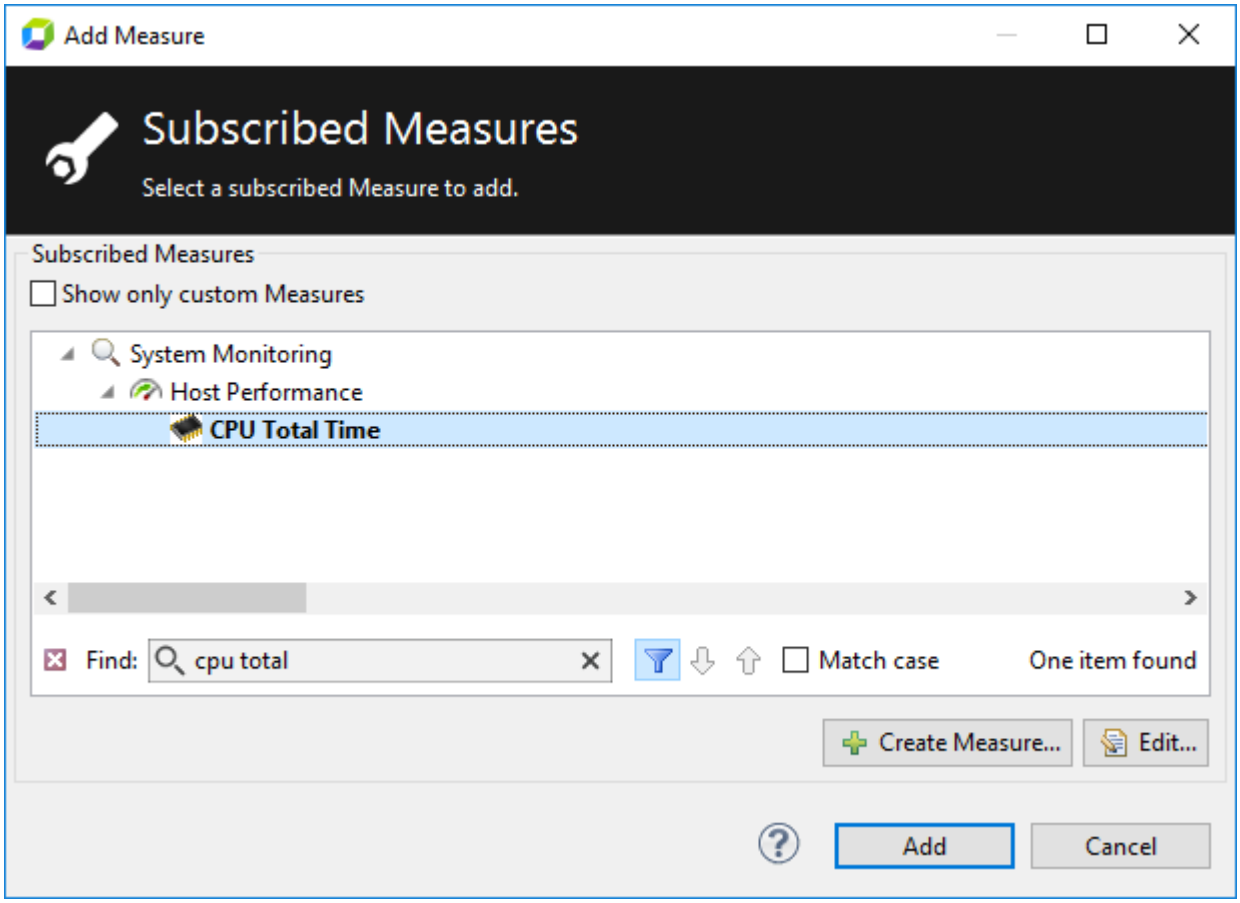
1. Right click your System Profile, and select **Edit System Profile > Incidents**.
2. Click **Create Incident Rule**.

3. Specify the name, evaluation timeframe, and severity of the incident.

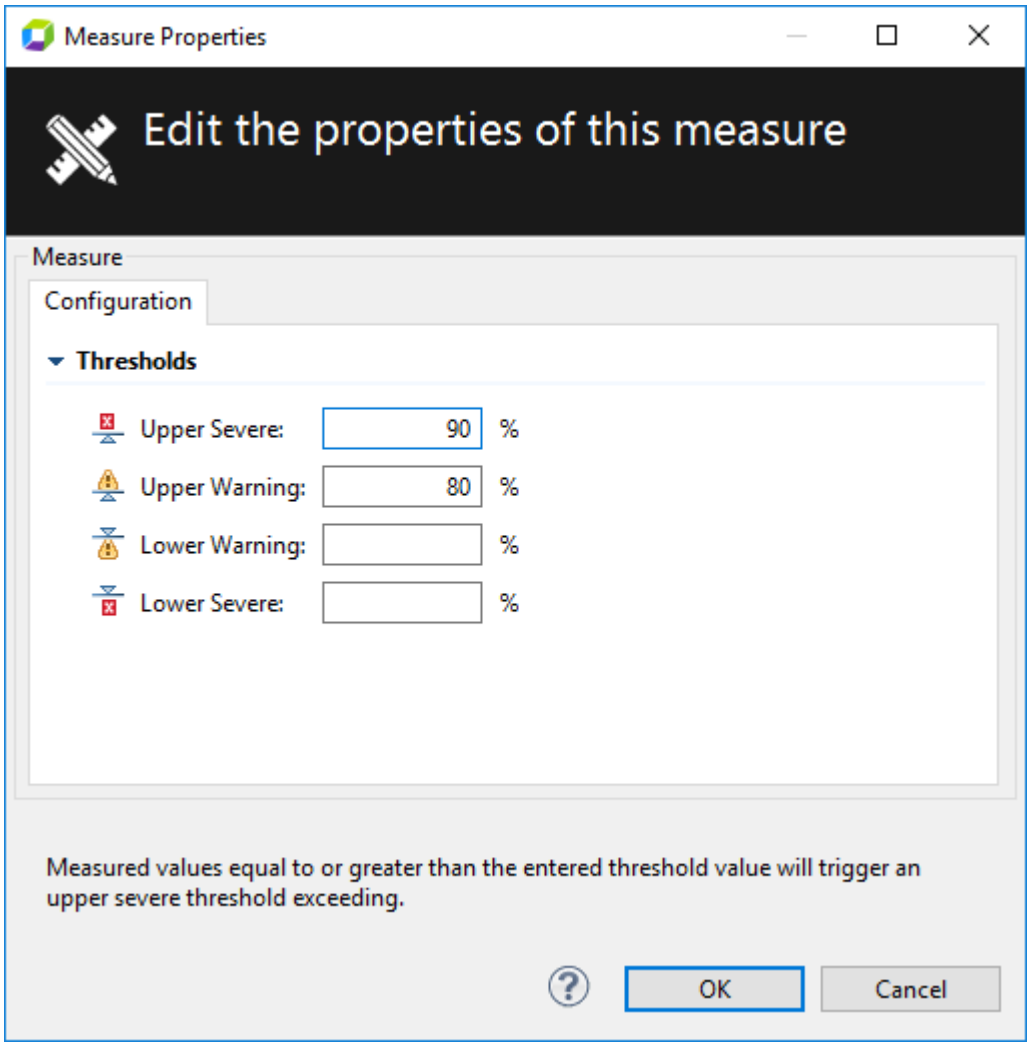


4. Under the **Conditions** table, click **Add** to add a measure to evaluate.

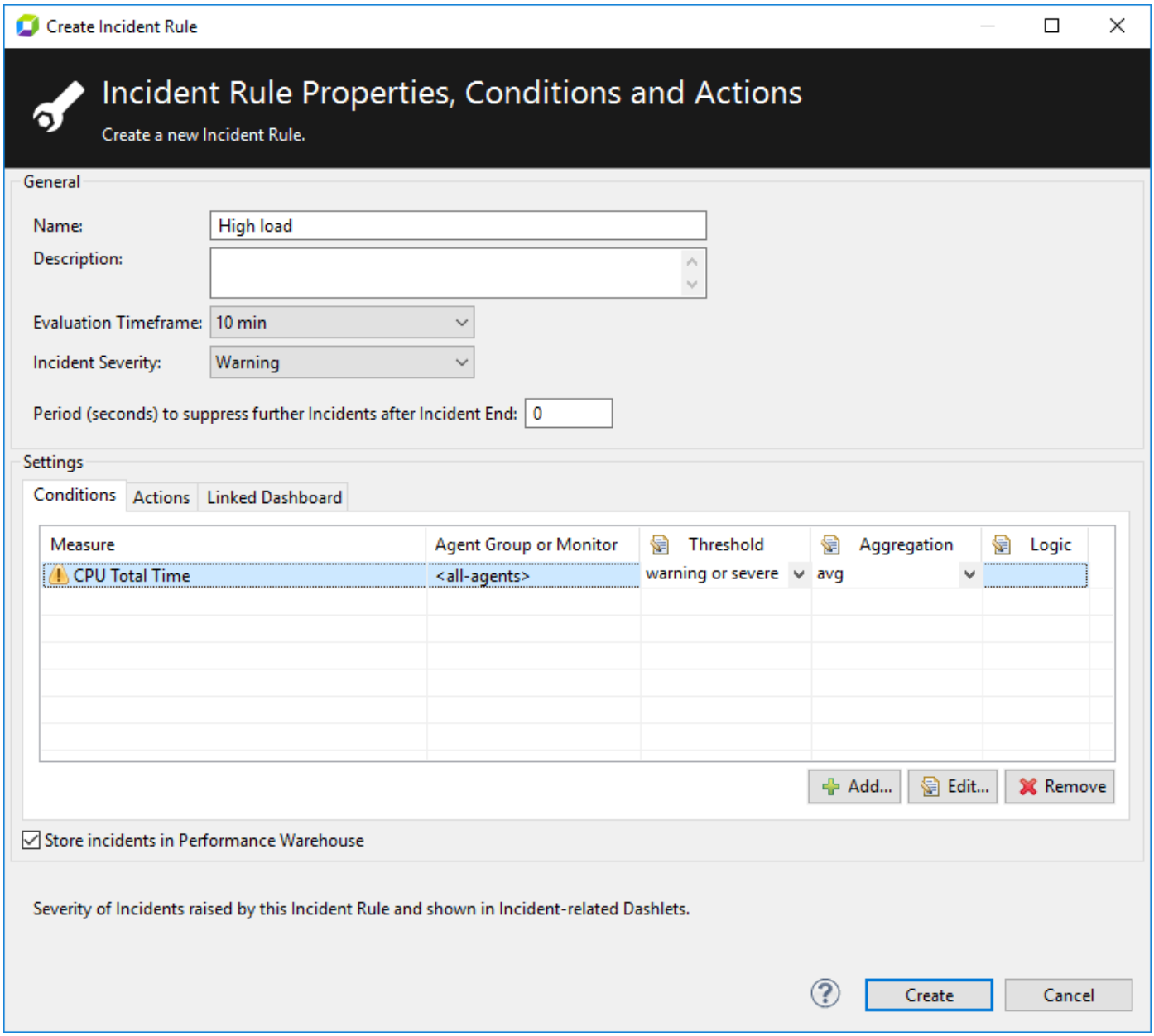
5. In the **Add Measure** dialog, find the required measure and click **Add**. In this example the **CPU Total Time** measure is used.



6. Double click the newly added measure to configure its thresholds.

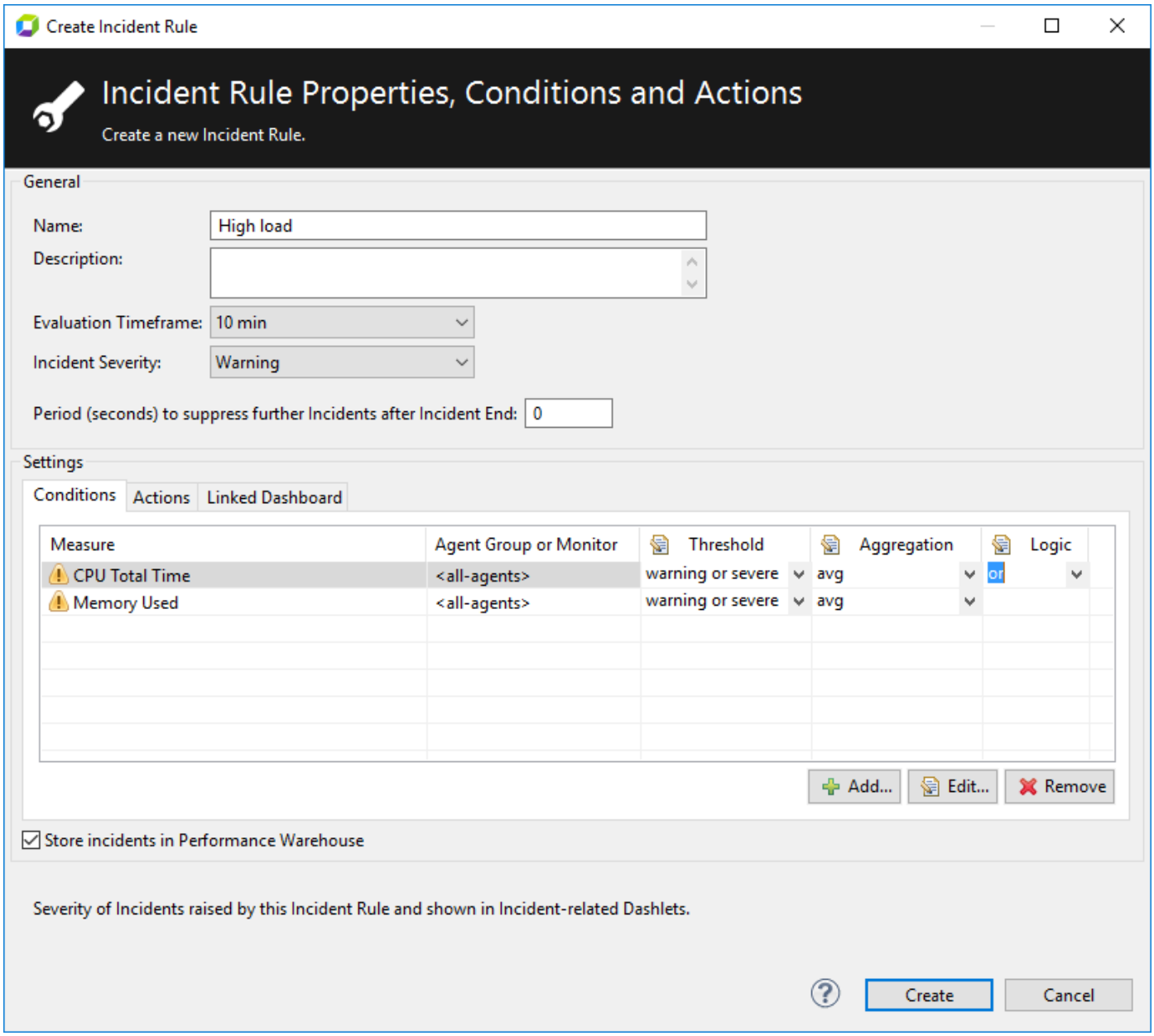


7. In the **Threshold** column, define the usage of the threshold. In this example it is **warning of severe**.

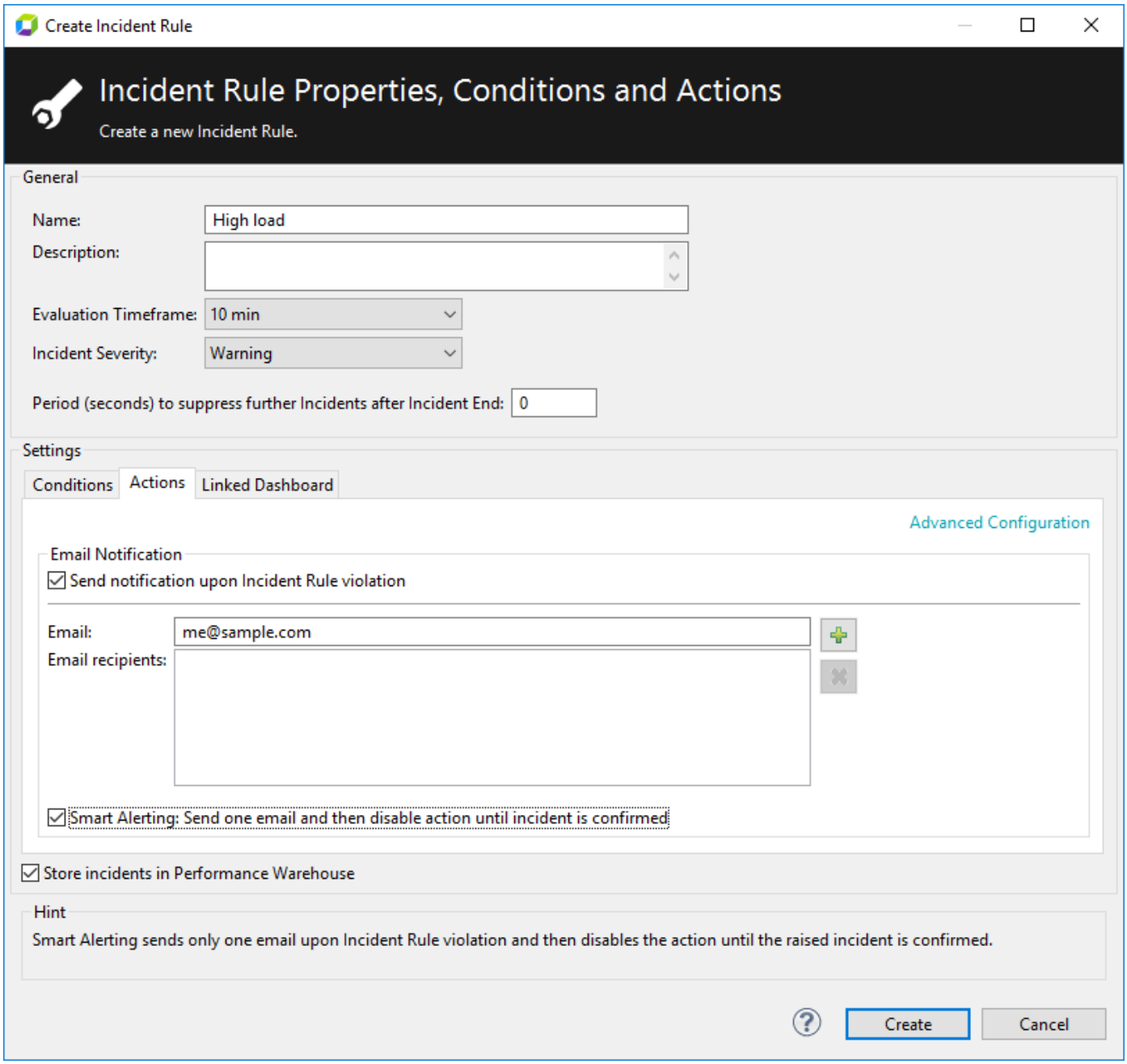


8. Repeat steps 4 to 7 to add another measure. In this example it is **Memory Used**.

9. In the **Logic** column, select the logic for incident evaluation. In this example it is **or**.



0. At the **Actions** tab, define the action on the incident. In this example it is notification via email.



1. Click **Create** to create the incident rule.

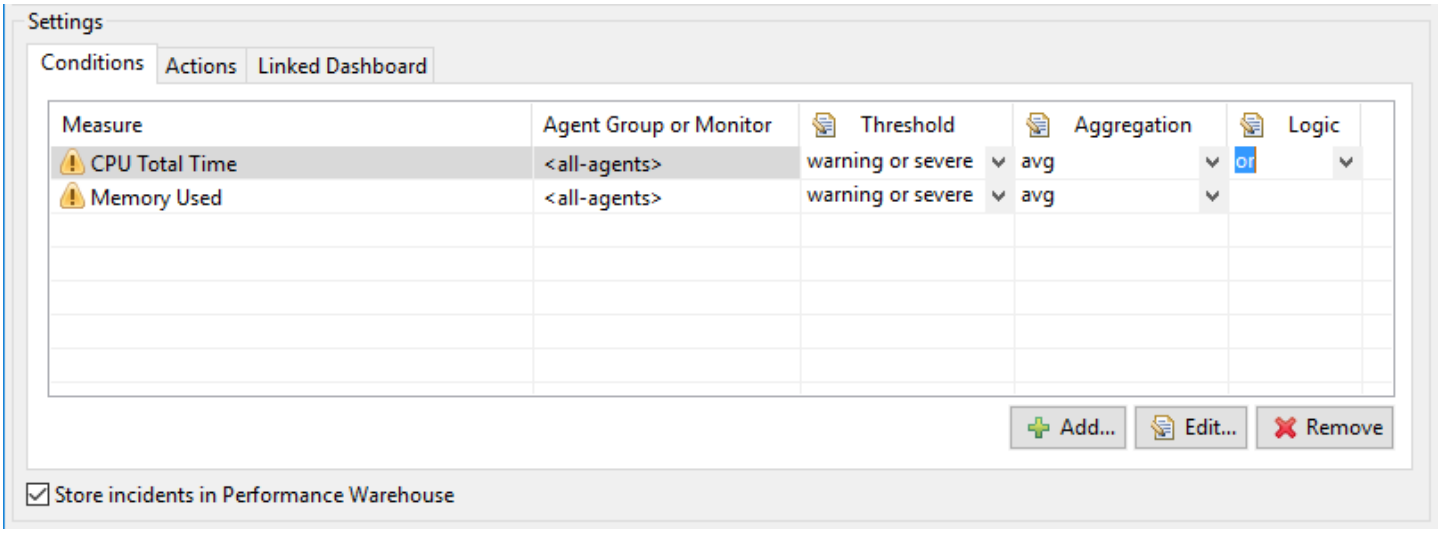
Conditions

An incident rule has one or more conditions based on **Warning** and **Severe** thresholds and an aggregation for the measure(s). Logical operators are used for multiple conditions. If the specified measures exceed the threshold, the actions assigned to the incident rule are executed.

The condition includes the following details:

- **Agent Group/Monitor**: The agent group or monitor for which the measure was configured.

- **Threshold**: The threshold type — **no threshold**, **warning or severe**, or **severe** — for a condition that triggers an incident.
  - **no threshold**: Condition is ignored when the incident rule is evaluated.
  - **warning or severe**: A warning triggers when the Warning threshold is exceeded. No additional incident will occur if the Severe threshold is exceeded. If you want an incident to be thrown if a Severe condition occurs after a Warning condition, you need to define a separate incident rule with the **severe** threshold.
  - **severe**: An incident triggers when the **severe** threshold is exceeded.
- **Aggregate**: The aggregate value that is used for the evaluation timeframe: **avg** (average), **count**, **last**, **max** (maximum), **min** (minimum), **sum**, or **first**.
  A measure can also occur multiple times per PurePath.
- **Logic**: The logical operator to combine multiple conditions: **and**, **or**. You can select an operator only if at least two measures are listed.



Conditions table

When you add a new measure to the conditions of an incident rule, its threshold state is initialized with the most severe level that is defined for the measure. If a value is only defined for the **warning** threshold, the incident rule threshold is initialized to **warning or severe**. If the **severe** threshold or both **warning** and **severe** thresholds are set, the incident rule threshold is initialized to **severe**.

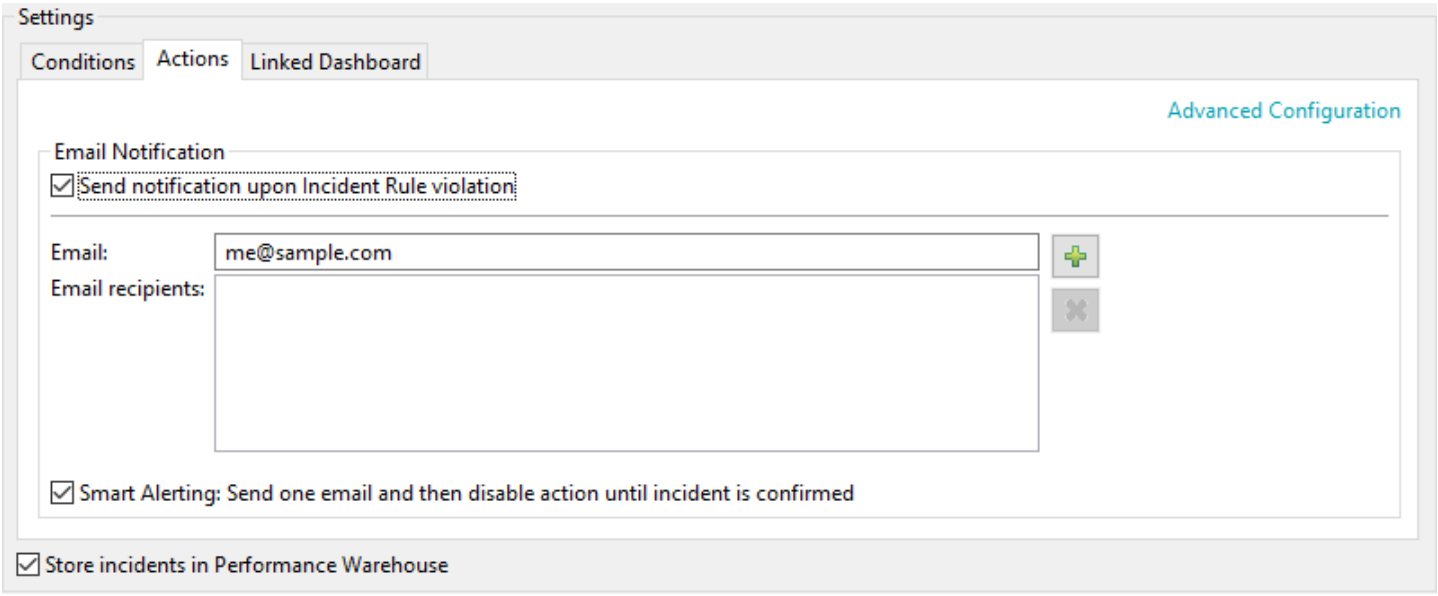
Changed threshold settings are effective for **all** incident rules using this measure. If different threshold settings are required, create an appropriately named copy of the measure with the desired threshold values.

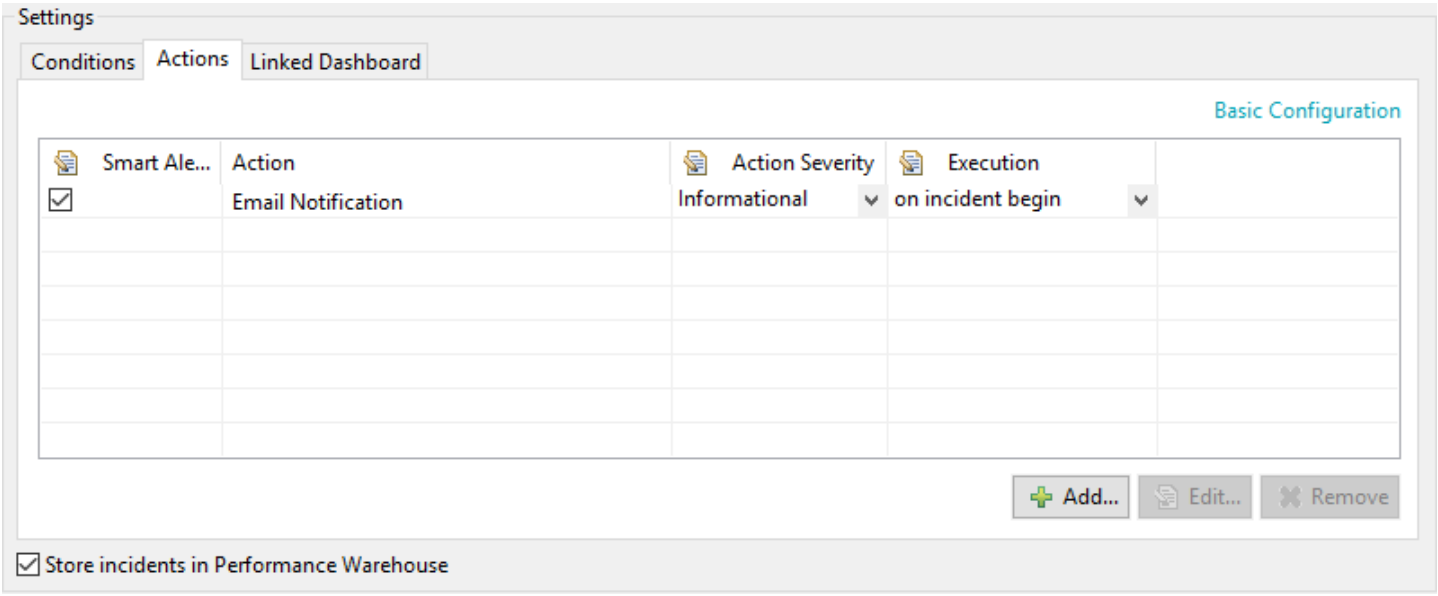
Actions

In the Basic Configuration mode, you can only configure email notification for incidents. To refine the incident response, click **Advanced Configuration** in the top left corner of the **Actions** pane.

To add an action to the advanced configuration, click **Add**.

The following figures show the screens you use to define advanced actions.





When you add an action to the advanced **Actions** table, you can configure it with the following settings:

- **Smart Alerting**: Selecting this option triggers the defined action just once if multiple incidents occur, until the raised incident is confirmed by a user. See Incidents Overview Dashlet for more information.
- **Action Severity**: A severity level — **Informational**, **Warning**, **Severe** — can be set for each action. This severity level is forwarded to the Plugins that execute the action.

- **Execution**: Specify when the action should be executed in relation to the incident. The action can be triggered when the incident is raised, when the incident is ended, or every time an incident is raised or ended.

The following actions are provided by default:

- **Activate Configuration Action**: Change the active configuration of the System Profile, for example to turn on Sensors in response to the incident.
- **CPU Sampling Action**: Start CPU Sampling Dashlet for the Agent that triggered the incident. See Sampling Based Diagnostics for details.
- **Email Notification Action**: Send notifications via SMTP to configured users.
- **Memory Dump Action**: Create a Selective Memory Snapshot or Selective Total Snapshot for the Agent that triggered the incident. See Incident based Memory Diagnostics for details.
- **SNMP Action**: Send notifications via SNMP to configured target hosts. See SNMP Action Plugin, below, for more information.
- **Store Session Action** Create a stored PurePath session of the System Profile affected by the incident. See Usage of Sessions for details.
- **Thread Dump Action** Create a thread dump for the Agent that triggered the incident. See Thread Diagnostics for details.

AppMon provides a Plugin interface and a Plugin development SDK (see Developing a User Plugin) for quick and easy development of user-defined action Plugins.

SNMP action plugin

To use the SNMP action plugin:

- Provide a MIB file to the SNMP administrator for the system that is the target of your SNMP trap action. The administrator compiles the file for the system. The MIB file provides the interface between the SNMP trap and your target system.
- Enable the SNMP Action Plugin in the Server settings: **Settings > Dynatrace Server > Plugins > SNMP Action Plugin**. Once enabled, the SNMP action appears in the list of incident actions.

The current MIB file is here: dynatrace-trap-MIB-v2. The MIB provides these values for the two supported traps:

```
dynaTraceIncidentStart TRAP-TYPE
ENTERPRISE dynaTrace
VARIABLES (name, message, description, severity, violation, server, systemprofile, startt
DESCRIPTION "Indicates that a new dynaTrace incident has begun."
::= 1
```

```
dynaTraceIncidentEnd TRAP-TYPE
ENTERPRISE dynaTrace
VARIABLES (name, message, description, severity, violation, server, systemprofile, startt
DESCRIPTION "Indicates that a dynaTrace incident has ended."
::= 2
```

Linked dashboard

Incident rules include the option to link a dashboard to the incident rule. This dashboard is used for reporting the incident via email. It can be opened through the **incidents** drilldown menu. You can only select dashboards stored on the server where the System Profile is stored. The default for new incident rules is the **Incident** dashboard, which is deployed with every AppMon Server.

For baseline incident rules, the dashboard configuration includes an option to open the affected splitting in the Application Overview. For host incidents, the configuration includes an option to open the Infrastructure Monitoring Dashboards displaying the affected host.

The table lists the host and baseline incident rules.

Host Incident Rules	Baseline Incident Rules
Application Process Unhealthy	Failure Rate Too High
Host CPU Unhealthy	Response Time Degraded for Slow Requests
Host Disk Unhealthy	Response Time Degraded
Host Memory Unhealthy	

Host Network Unhealthy	Host monitoring has detected a potential problem in the network (more than 90% of bandwidth of a network interface are in use)

Server measures and server incidents

Related pages

Incidents dashlet

Incident downtimes

Built-in incidents

Watch and learn

Managing incidents and alerts

Configure measures and incidents in the Self-Monitoring System Profile. For example, you may want to monitor the amount of free heap memory or the number of garbage collections.

You can enable self-monitoring via **Settings > Dynatrace Server > Settings > Enable Self-Monitoring**.

This page describes the default Server incident rules and provides links to additional information.

Incident Rule Name	Description	Further Information
Agent Instrumentation Disabled	This incident is raised for each Agent that disables its instrumentation. This may happen in the following cases: • The Agent did not get a connection to a Collector or Server at startup. For example, the Collector or Server was not running when the Agent was started). • The instrumentation overhead is too high (the average instrumentation time exceeds a defined limit). For example, there is too much load on the Collector during the Agent's startup. • An Agent could not be mapped to a System Profile. For example, the Agent's name is misspelled or the System Profile is disabled. • An Agent from an unsupported release tries to connect to the Server.	Set up Agents

Ambiguous Agent Mapping	An Agent is mapped to more than one System Profile. The profile used for mapping depends on the alphanumerical order of profiles. This arbitrary order may cause undesired results. It is recommended that you disable superfluous profiles or adapt the Agent's name or Agent mapping configuration in the affected profiles to guarantee a unique mapping profile.	System Profile Set up Agents
Collector not Connected (unexpected)	Connection to an AppMon Collector was lost, perhaps because of network problems or an error in the Collector.	Server Administration
Conflicting Agent JVM Options	A conflicting Agent was detected in the JVM startup settings.	Set up Agents
AppMon Server Offline	The AppMon Server terminated normally.	Set up the AppMon Server
AppMon Server Offline (unexpected)	The AppMon Server terminated abnormally.	Set up the AppMon Server
File Session Management	An incident occurred related to session storage. For example, the disk quota was exceeded or corrupted sessions were detected.	Performance Warehouse Administration
License	An incident occurred related to licensing. For example, an expired license was detected.	Licensing
Low Collector Memory	If a Collector runs low on memory, it tries to prevent an Out-Of-Memory exception by first refusing to accept further instrumented Agents and then by disabling instrumentation for already connected Agents. This incident is raised for each Agent that was accepted only in uninstrumented mode and each Agent for which instrumentation was disabled. Recommended solutions are to increase the Collector's heap size, reduce the number of connected Agents, or reduce the depth of instrumentation per Agent.	Set up Collectors
Low AppMon Server Memory	The AppMon Server detected that it is running low on non-heap memory such as the **PermSize** memory. It is recommended to increase the **PermSize** settings for the AppMon Server.	Set up the AppMon Server
Scheduled Task Execution Failed	The execution of a Plugin failed.	Manage and Develop User Plugins
Performance Warehouse	This incident is triggered by events related to the Performance Warehouse.	Performance Warehouse Administration
Performance Warehouse Clean-Up (Repository Clean-Up)	This incident is triggered when the Performance Warehouse cleanup task lasts too long.	Performance Warehouse Administration
Performance Warehouse Offline	A connection to the Performance Warehouse could not be established.	Performance Warehouse Administration

Performance Warehouse - Switching Temp Tables	This incident is triggered when the time required for switching the temp tables is violated.	Performance Warehouse Administration
Performance Warehouse - Writing Measurements/Percentiles Time	These incidents are triggered if the writing task for the repository exceeded either its warning or severe threshold. Review the **Performance Warehouse Write Health** dashboards for details.	Performance Warehouse Administration
Performance Warehouse Event Summary	An incident occurs after the cleanup job if either an important event or an exception occurs during a Performance Warehouse operation.	Performance Warehouse Administration

Incident downtimes

Related pages

Incidents dashlet

Server measures and server incidents

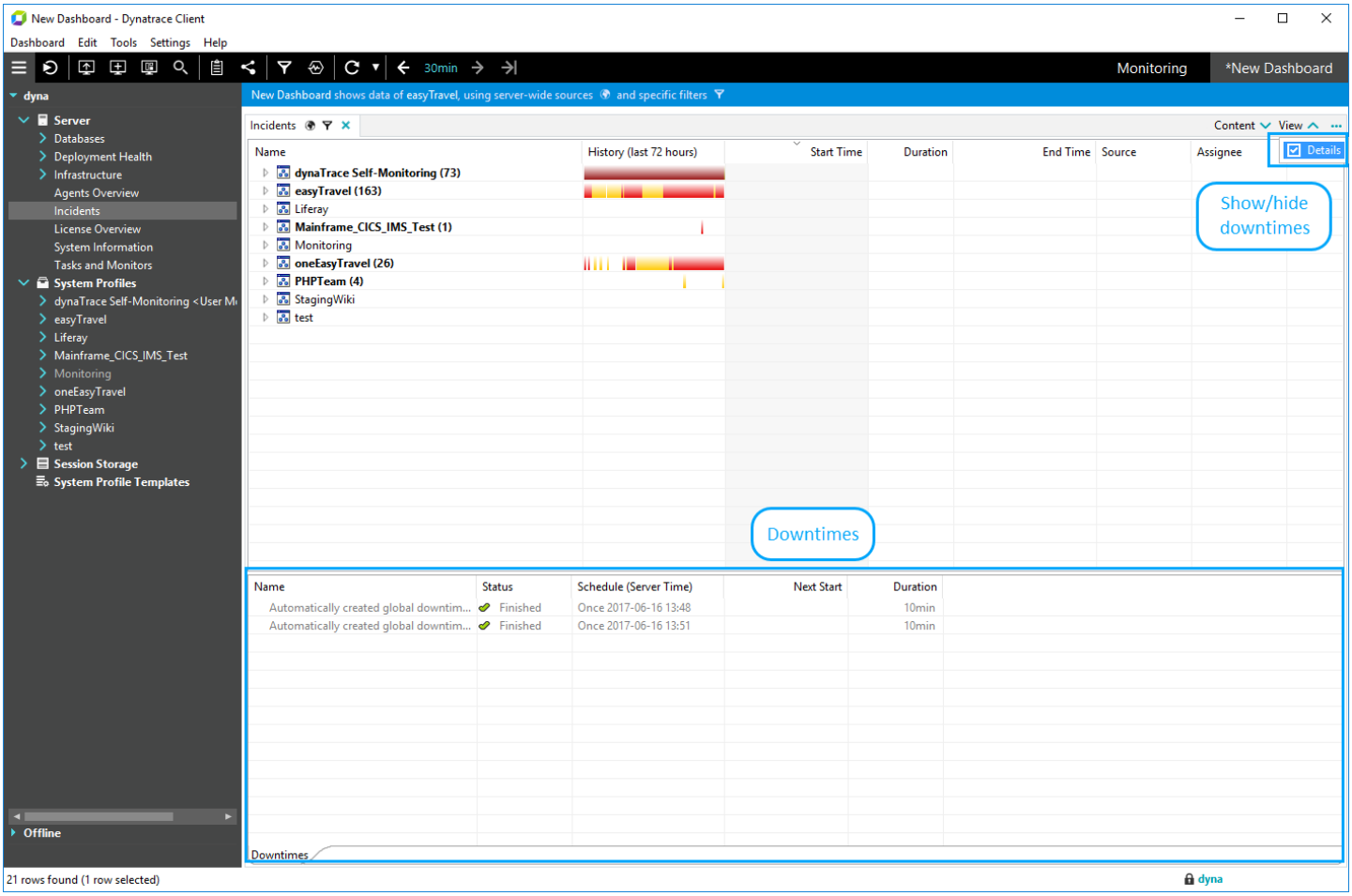
Built-in incidents

Watch and learn

Managing incidents and alerts

For planned actions such as system backups, database mirroring, and server restarts, you can suppress the firing of incidents by scheduling downtimes.

In the **Incidents** dashlet, show or hide the **Downtimes** pane by clicking **View > Details**, as indicated in the figure.



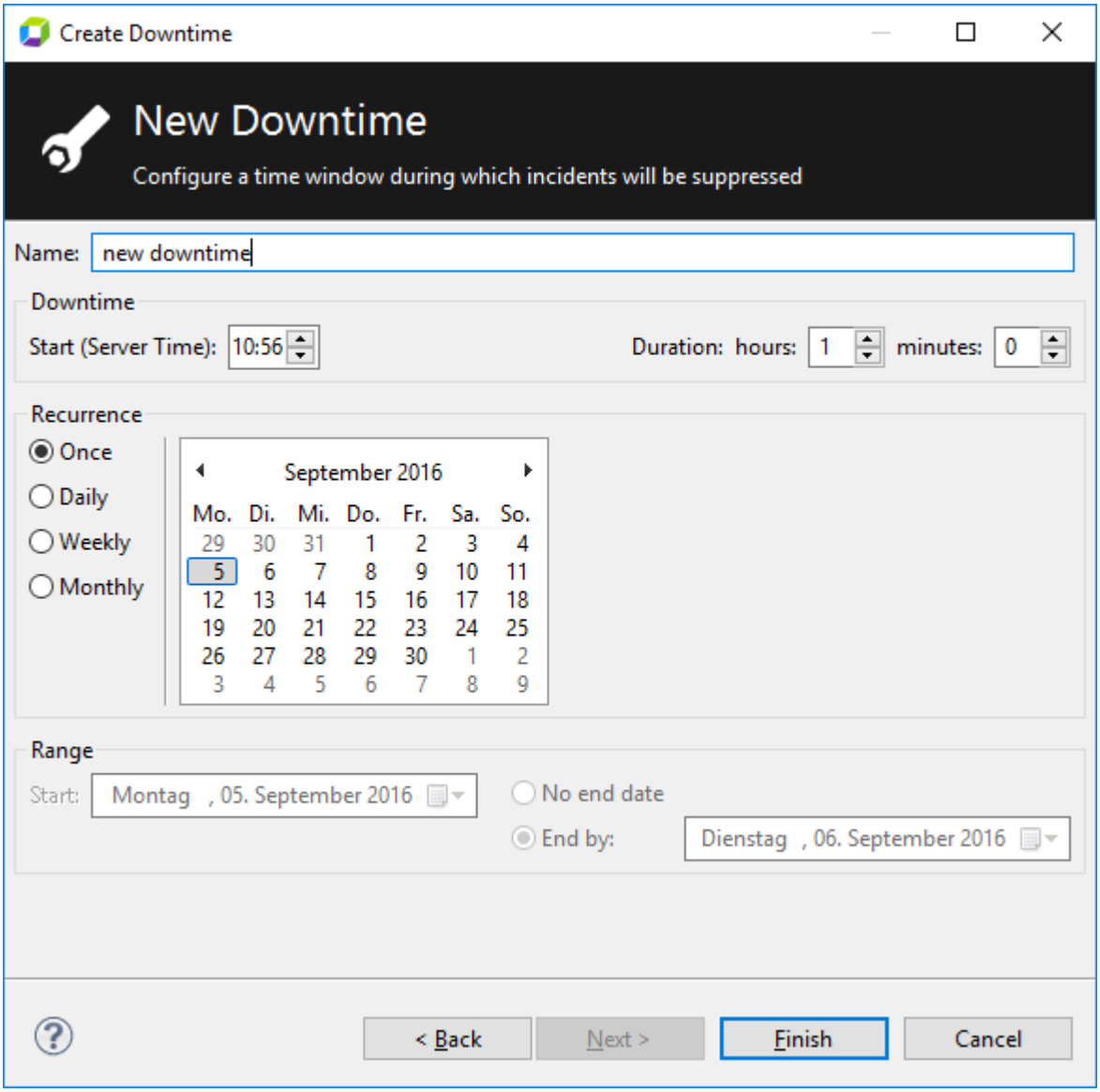
Incidents dashlet

Creating a downtime

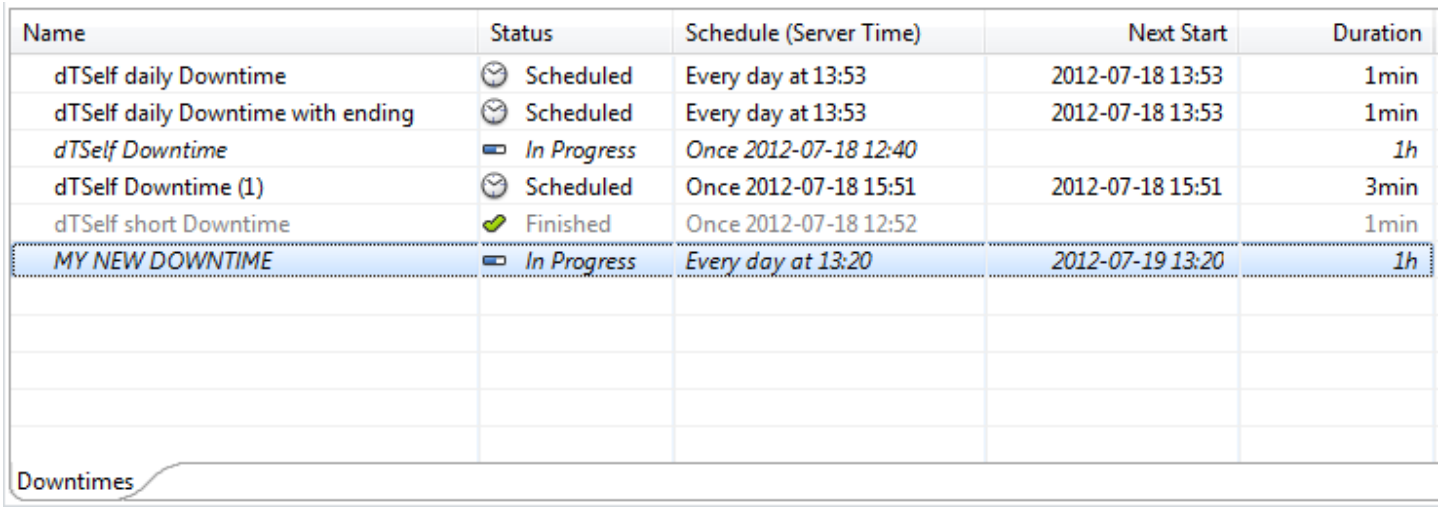
You can open the **Create Downtime** wizard through the **Create downtime** item of the context menu for the **Incident** dashlet's upper (incident rules) or lower (downtimes) pane:

- Right-click anywhere in the **Downtimes** pane to display the **first** page of the wizard. Select one or more System Profiles that will be affected by the downtime. The Admin user can select all System Profiles.
- When you right-click in the upper pane, the wizard's first page is skipped. Right-click an incident to create downtime for the associated incident rule. Right-click in the pane's empty space to create downtime for the active System Profile.

In the wizard's second page, define the downtime's start time (server time), duration, and recurrence including starting and end dates.



All created downtimes are listed in the **Downtimes** pane. The pane contains the information described in the table below.



Name	Status	Schedule (Server Time)	Next Start	Duration
dTSelf daily Downtime	Scheduled	Every day at 13:53	2012-07-18 13:53	1min
dTSelf daily Downtime with ending	Scheduled	Every day at 13:53	2012-07-18 13:53	1min
dTSelf Downtime	*In Progress*	*Once 2012-07-18 12:40*		*1h*
dTSelf Downtime (1)	Scheduled	Once 2012-07-18 15:51	2012-07-18 15:51	3min
dTSelf short Downtime	Finished	Once 2012-07-18 12:52		1min
MY NEW DOWNTIME	*In Progress*	*Every day at 13:20*	*2012-07-19 13:20*	*1h*

Downtimes

Downtimes pane

Column	Description
Name	The downtime's name.
Status	**Scheduled**: The downtime is scheduled and active at the specified time in the future. **In Progress**: The downtime is currently active. **Finished**: The downtime has finished and has not yet been rescheduled.
Schedule (Server Time)	The schedule is listed as **<RECURRENCE> <TIME>**. One-time downtime – **Once YYYY-MM-DD hh:mm**. Recurring downtime – For example, daily: **Every day at hh:mm**.
Next Start	The next scheduled occurrence for downtimes that are scheduled or in progress.
Duration	The downtime's duration.

To remove all finished one-time downtimes, select **Clear Finished Downtimes** from the context menu for the **Downtimes** pane.

Note

Downtimes created for a normal System Profile do not affect the firing of host health incidents. New incidents of this type appear even during an active downtime. See Host Health monitoring for more information.

Host health incidents can only be suppressed globally for all System Profiles by creating a downtime for the special **Infrastructure** System Profile, which is only visible in the **Infrastructure Incidents** dashlet accessible from the **Status Overview** in the Cockpit sidebar.

Built-in incidents

Related pages

Incidents dashlet

Server measures and server incidents

Incident downtimes

Watch and learn

Managing incidents and alerts

The table provides an overview of the built-in incident rules in the Self-Monitoring and Custom System Profiles.

Severity	Name	Description	System Profile
Severe	Agent Instrumentation Disabled Reasons for that include: Collector has performance issues, Agent response time is very slow or has poor instrumentation performance.	It is not possible to instrument the Agent.	Self-Monitoring & Custom
Severe	Agent Capturing Disabled	The Agent did not send updates for an extended period of time and was disabled.	Self-Monitoring
Warning	Ambigous Agent Mapping	The Agent could be mapped to several System Profiles.	Self-Monitoring
Severe	Continuous transaction recording was aborted	The server is no longer recording PurePaths to disk, probably due to a problem such as out of disk space or file system error.	Self-Monitoring & Custom
Severe	Custom events	When you right click in a chart dashlet and select **Heat Field > Add Custom Events...** you can mark a Custom time period in the heat field to highlight a maintenance or other event in the chart.	Self-Monitoring & Custom
Severe	Collector not connected (unexpected)	Connection to a previously connected Collector was lost. This incident also appears in the Start Center.	Self-Monitoring

Warning	Conflicting Agent JVM Options	Raises incidents for conflicting JVM options	Self-Monitoring
Warning	Dynatrace Server Offline	Signals normal termination of Dynatrace Server.	Self-Monitoring
Severe	Dynatrace Server Offline (unexpected)	Signals normal termination of Dynatrace Server.	Self-Monitoring
Informational	AppMon Server Online	Signals abnormal termination of Dynatrace Server.	Self-Monitoring
Severe	File Session Management	Session storage-related incidents, e.g. disk quota exceeded, corrupted sessions detected.	Self-Monitoring
Severe	License	License-related incidents (e.g. an expiring license). This incident also appears in the Start Center.	Self-Monitoring
Severe	Low Collector Memory	Agent-related incidents caused by Collector heap memory, such as the Agent's instrumentation being disabled automatically to free some memory.	Self-Monitoring
Severe	Low AppMon Server Memory	Low (non-heap) server memory.	Self-Monitoring
Severe	Number of Measures	The number of measures reached the maximum of 300,000, which is likely caused by a misconfigured Business Transaction.	Self-Monitoring
Severe	Performance Warehouse	Triggered by Performance Warehouse related events.	Self-Monitoring
Severe	Performance Warehouse Offline	Connection to Performance Warehouse could not be established. This incident also appears in the Start Center.	Self-Monitoring
Severe	Performance Warehouse - Switching Temp Tables Time	Triggered when the insert and switch procedure of the measurement heap tables lasts longer than the default severe threshold.	Self-Monitoring
Severe	Performance Warehouse - Writing Measurements Time	Writing of measurements into heap tables takes longer than the severe threshold.	Self-Monitoring
Warning	Performance Warehouse - Writing Measurements Time	Writing of measurements into heap tables takes longer than the warning threshold.	Self-Monitoring
Severe	Performance Warehouse - Writing Percentiles Time	Writing of percentiles into the percentiles table takes longer than the severe threshold.	Self-Monitoring

Warning	Performance Warehouse - Writing Percentiles Time	Writing of percentiles into percentiles tables takes longer than the warning threshold	Self-Monitoring
Severe	Port Conflict	A port used for communication on the AppMon Server could not be opened.	Self-Monitoring
Informational	Repository Clean-Up	Triggered when writing time is slow.	Self-Monitoring
Warning	Scheduled Task Execution Failed	A scheduled task has failed.	Self-Monitoring
Informational	System Profile Changed	The system profile configuration has been changed.	Custom
Informational	Agent Connected	Raises incidents for individual connected agents.	Custom
Informational	Agent CPU Time Capturing	The Agent disabled CPU time capturing because overhead was too high. Adjust the capturing options in the Agent Group's Sensor configuration.	Custom
Informational	Agent Hot Sensor Placement	Triggered when the Agent is Hot Sensor placed.	Custom
Severe	Application Process Out-of-memory	Memory dump triggers if it is not disabled in the Agent Group settings.	Custom
Severe	Application Process Unavailable (unexpected)	A previously connected application process was shut down.	Custom
Severe	Continuous Transaction Recording was aborted		Custom
Warning	Corrupted PurePath(s)	Raises incidents for PurePaths that are corrupted.	Self-Monitoring & Custom
Warning	High CPU Usage for Application Process	High CPU usage detected for a longer period.	Custom
Warning	High GC Activity for Application Process	A large amount of process time used for garbage collection.	Custom
Warning	High Memory Usage for Application Process	Memory usage is at a high percentage for a long time period.	Custom

Severe	High Overall Failed User Action Rate	The user action rate exceeds the configured threshold.	Custom
Severe	High Overall Failed Transaction Rate	The failed transaction rate exceeds the configured threshold.	Custom
Severe	Response Time Degraded for Slow Requests	The response time of slow transactions has changed.	Custom
Informational	Session Cleared	Clear Session was executed.	Custom
Informational	Test Violation	The test result has changed significantly: degraded, improved, failed, or is volatile.	Custom
Warning	Timed Out PurePath(s)	Either a method has not returned or a whole child path did not finish within a certain time.	Custom
Warning	Truncated PurePath(s)	PurePath(s) have been truncated because their size exceeds the node limit.	Custom

Tasks

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Configuration

Task types

Related pages

System Profile - Tasks

Tasks and monitors (REST)

Overview

A task can be scheduled for repetitive execution on the server or on a connected collector. Arbitrary execution schedules can be defined and shared between different tasks and monitors. On execution, a task gets a set of configuration parameter values, and after execution the task provides an execution status containing data about the execution.

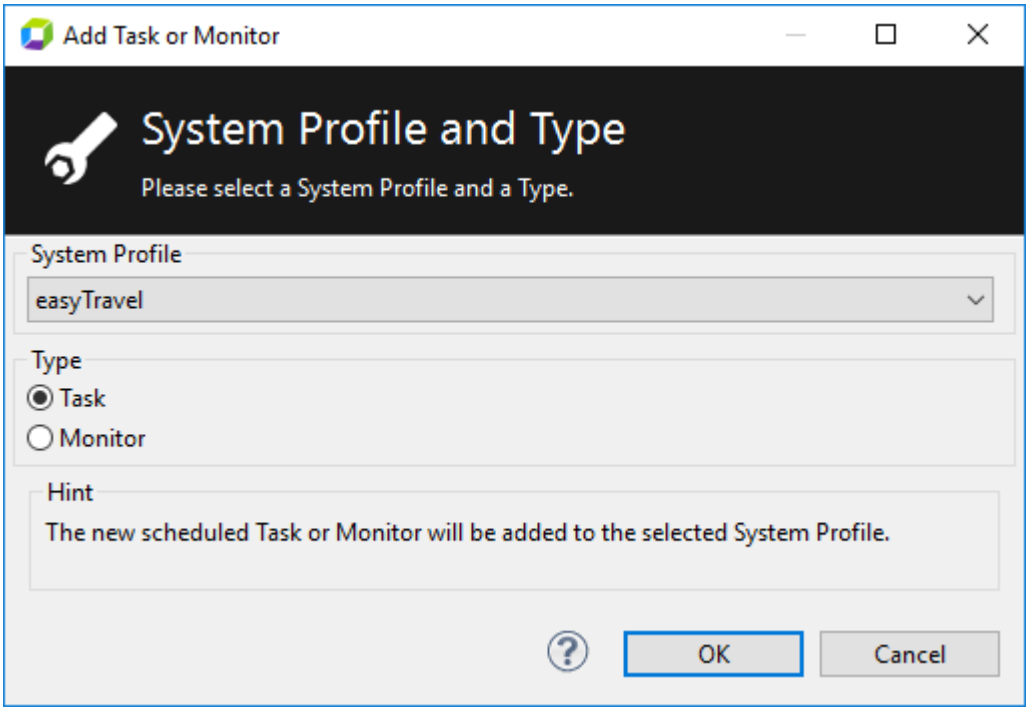
AppMon provides built-in tasks to automate repeating tasks like generating reports or storing sessions.

Configuration

You can create tasks through the System Profile or the **Tasks and Monitors** dashlet:

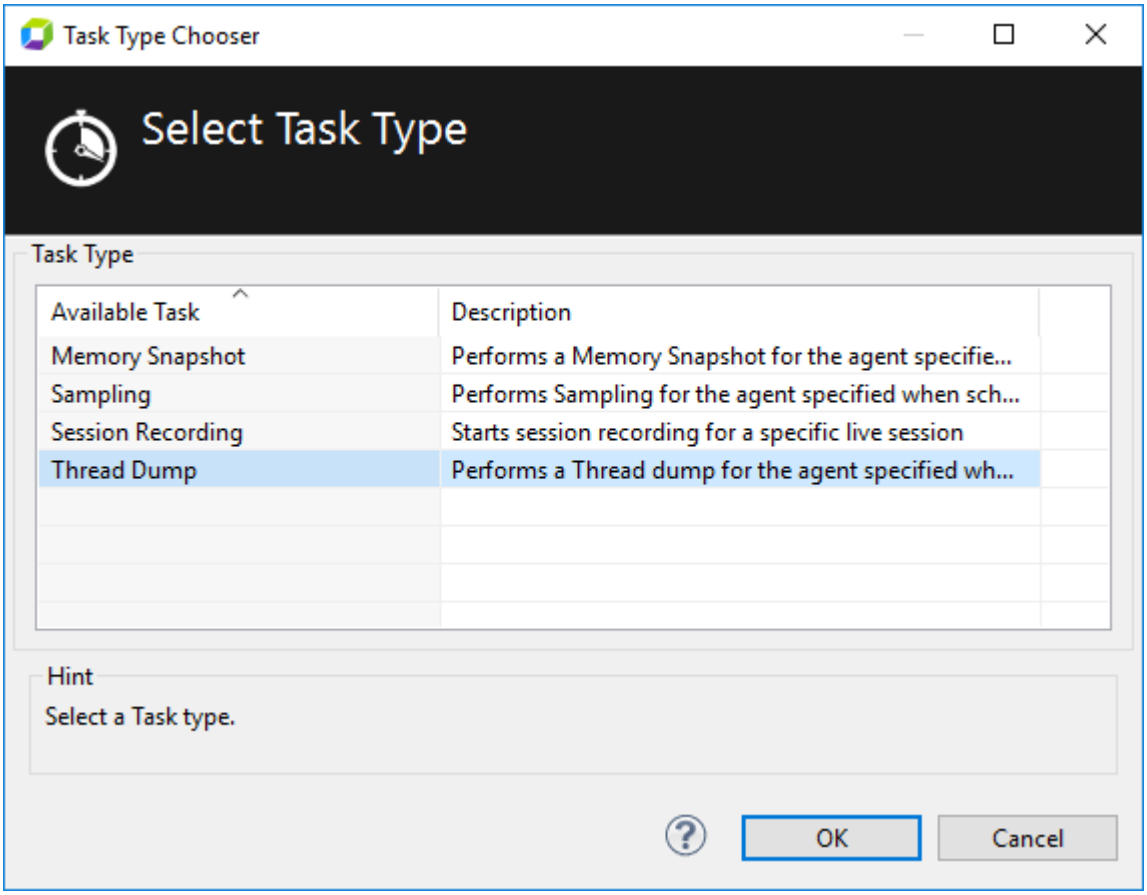
- In the **System Profile Preferences** dialog box, select **Tasks** and then click **Create**.
- In the **Tasks and Monitors** dashlet, right-click and select **Create** from the context menu.

When creating a task through the dashlet, you must first select a System Profile and specify that you are creating a **Task**.



Select System Profile for new task

When you click the **Create** button in the **System Profile Preferences** dialog box or click **OK** in the **Add Task/Monitor** dialog box, the **Task Type Chooser** appears.



Task type chooser

Select the type, then click **OK** to display the **Scheduled Task Editor**.

For each task type, the you must enter a unique **Name** and optionally a **Description**. The other settings are specific to the task type. These settings are described below in Task Types.

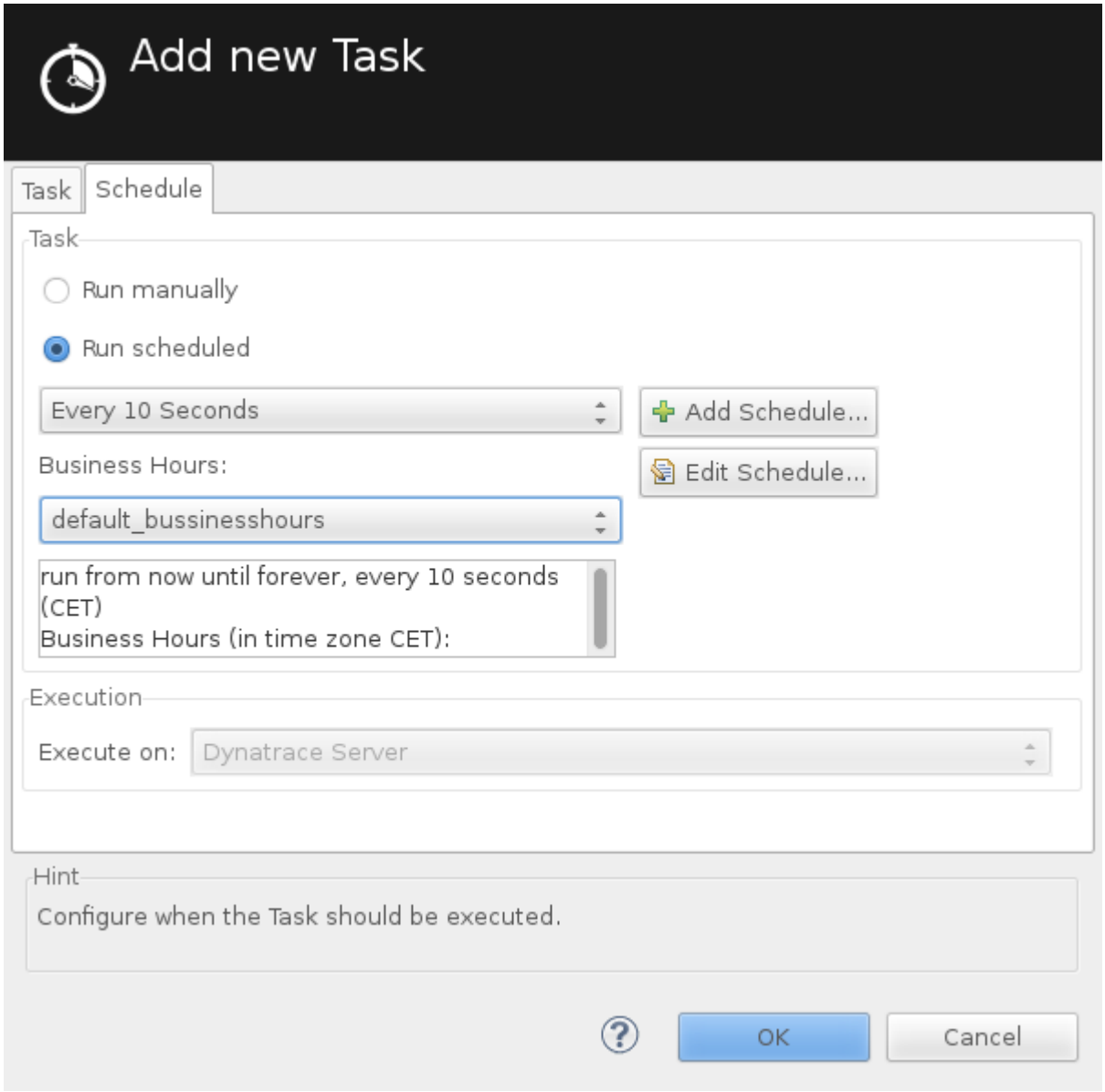
On the **Schedule** tab, define the task schedule and select the execution target.

Select whether the task should be run manually or run on a schedule.

If you select **Run scheduled**:

- Select the interval from the first list. The selections range from **Every 10 Seconds** through **Every Hour**.
- Click **Add** to use a preconfigured schedule or define new schedule, or click **Edit** to revise the schedule. For further information on schedules, see System Profile - Schedules.
- Optionally select **Business Hours** from the second list. For further information on Business Hours, see System Profile - Business Hours.

In the **Execute on** list, select the AppMon Server or an available Collector.



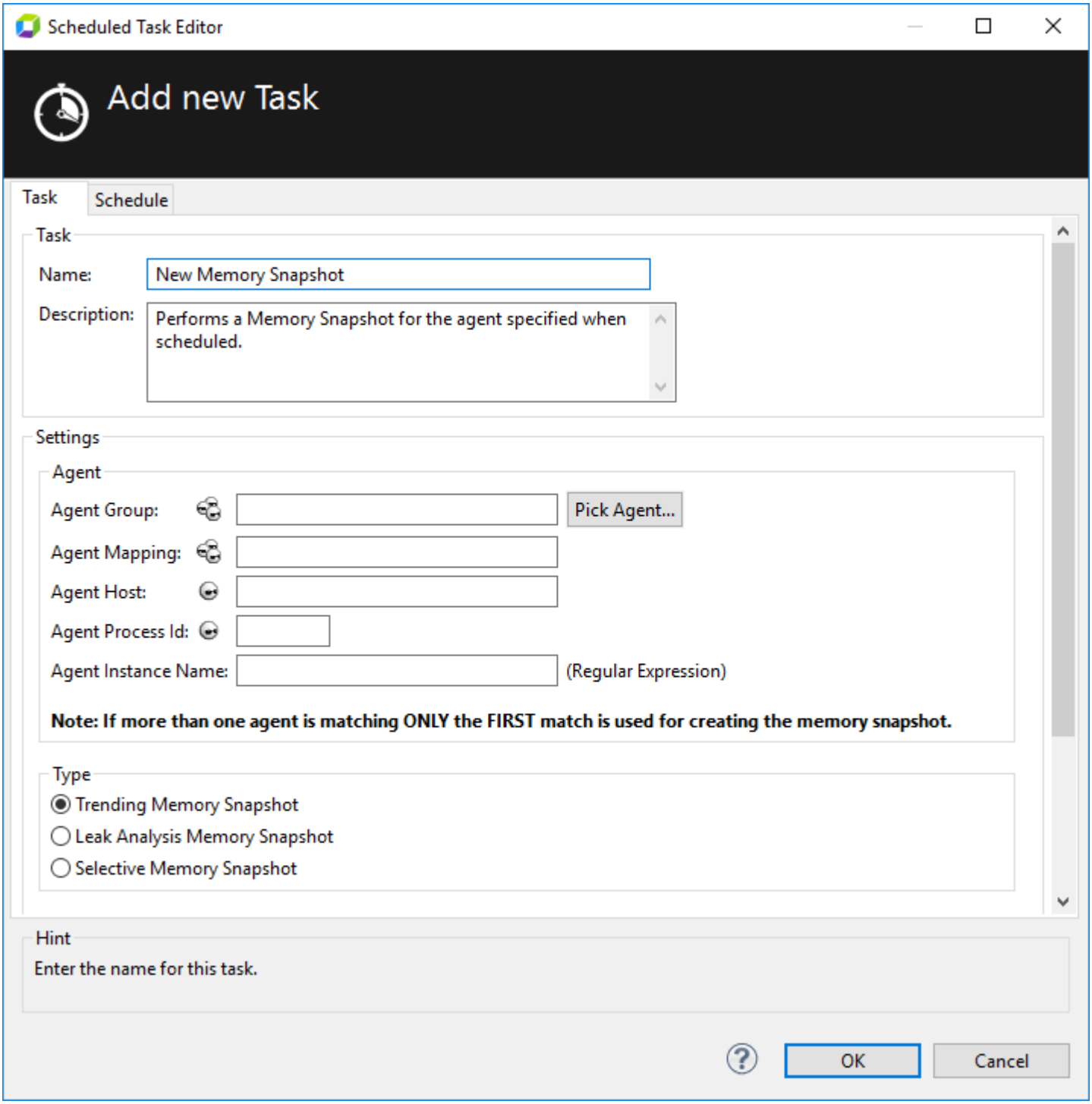
Scheduled task editor

Task types

This section describes the configuration options for the different task types.

Memory snapshot

The Memory Snapshot task creates a Memory Snapshot for a specified Agent.



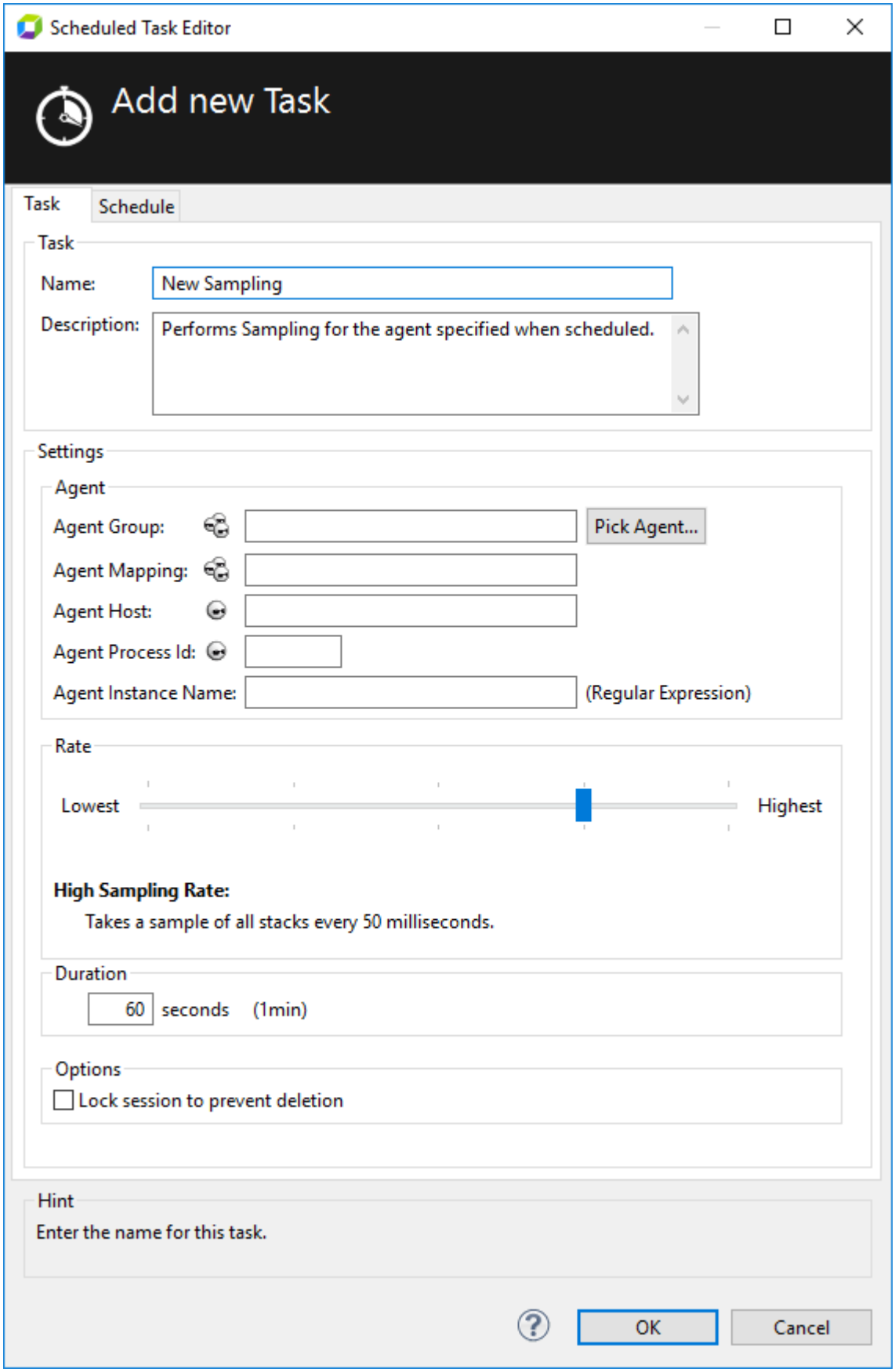
Memory Snapshot Task Configuration

Parameter Name	Description
Agent Group	To make sure the Agent Group name is entered correctly, click **Pick Agent** to select an Agent Group or Agent through the **Agent Browser**.
Agent Mapping	In the **Agent Browser**, you can select an Agent Group name or an Agent Mapping. If you select an Agent, the Agent Group is entered in the **Agent Group** field and the Agent Mapping is entered in this field. If you select an Agent Group, this field remains blank.

Agent Host	The Agent's host name.
Agent Process Id	The Agent's process ID.
Agent Instance Name	A regular expression defining the instance name for the Agent.
Type	Select one of the snapshot types: • **Trending Memory Snapshot** • **Leak Analysis Memory Snapshot** • **Selective Memory Snapshot** See Memory Diagnostics for information about the different snapshot types.
Lock session to prevent deletion	Select this option to prevent the session from being deleted during automated cleanups on low disk space.
Force Garbage Collection before creating snapshot	Force a Garbage Collector run before creating the Memory Snapshot.
Capture values of String objects	Include String values in the Memory Snapshot. This setting is only available for Leak Analysis Snapshots. It requires Java 6 or CLR 2.0.
Capture values of primitive fields	Include primitive field values in the Memory Snapshot. This setting is only available for Leak Analysis Snapshots. It requires Java 6 or CLR 2.0.
Automatically post-process memory snapshot	Enable or disable automatic post-processing after creating the dump.

Sampling

The Sampling task captures samples of all stacks of a specified Agent.

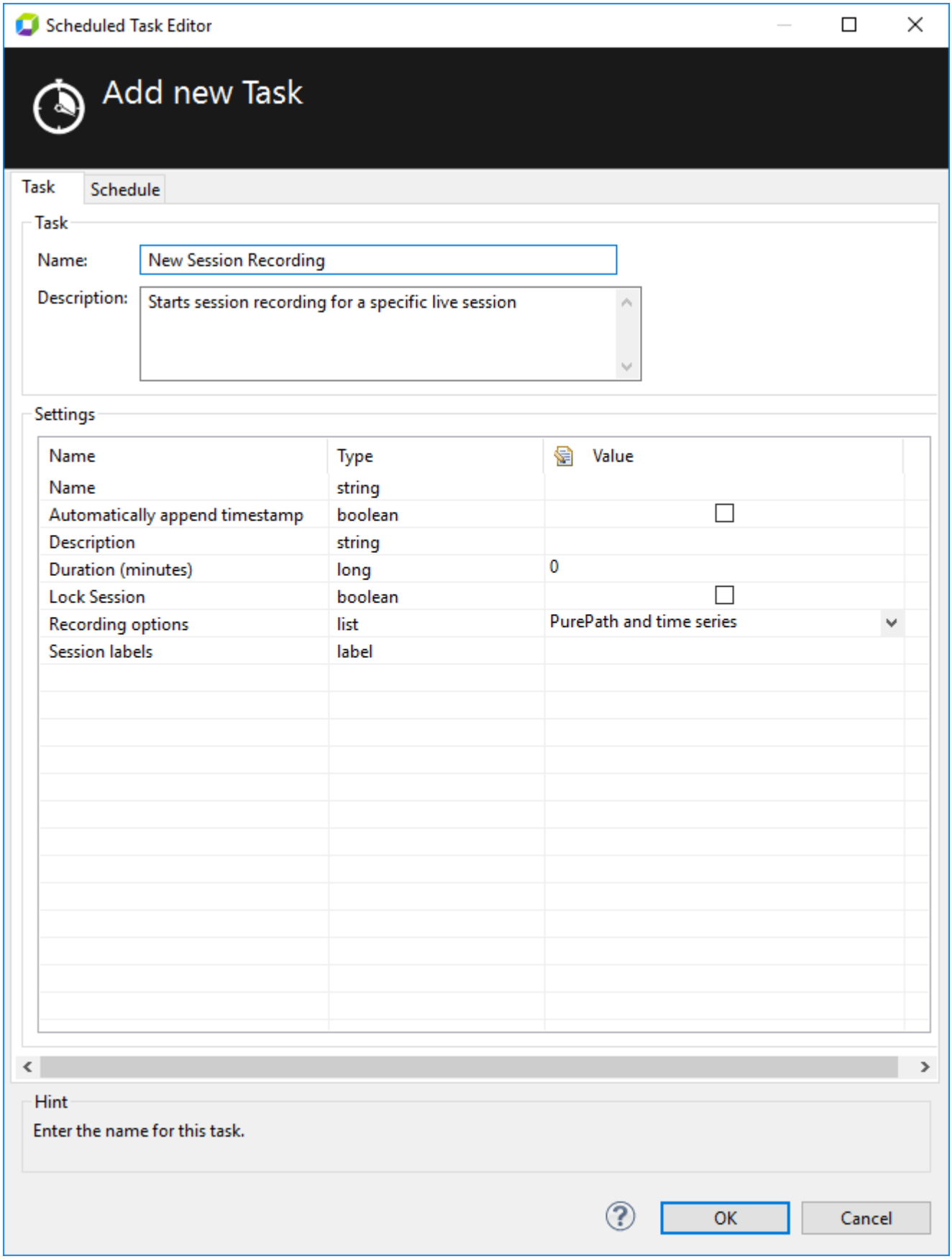


Sampling Task Configuration

Parameter Name	Description
Agent Group	To make sure the Agent Group name is entered correctly, click **Pick Agent** to select an Agent Group or Agent through the **Agent Browser**.
Agent Mapping	In the **Agent Browser**, you can select an Agent Group name or an Agent Mapping. If you select an Agent, the Agent Group is entered in the **Agent Group** field and the Agent Mapping is entered in this field. If you select an Agent Group, this field remains blank.
Agent Host	The Agent's host name.
Agent Process Id	The Agent's process ID.
Agent Instance Name	A regular expression defining the instance name for the Agent.
Rate	Move the slider to select a sampling rate from **Lowest** (every 1,000 milliseconds = 1 second) to **Highest** (every 10 milliseconds). The default is every 50 milliseconds. Higher sampling rates produce more accurate results but cause higher overhead.
Duration	The duration in seconds for how long the sampling should run. The default is 60 seconds.
Lock session to prevent deletion	Select this option to prevent the session from being deleted during automated cleanups on low disk space.

Session recording

The Session Recording Task starts session recording for a specified live session.

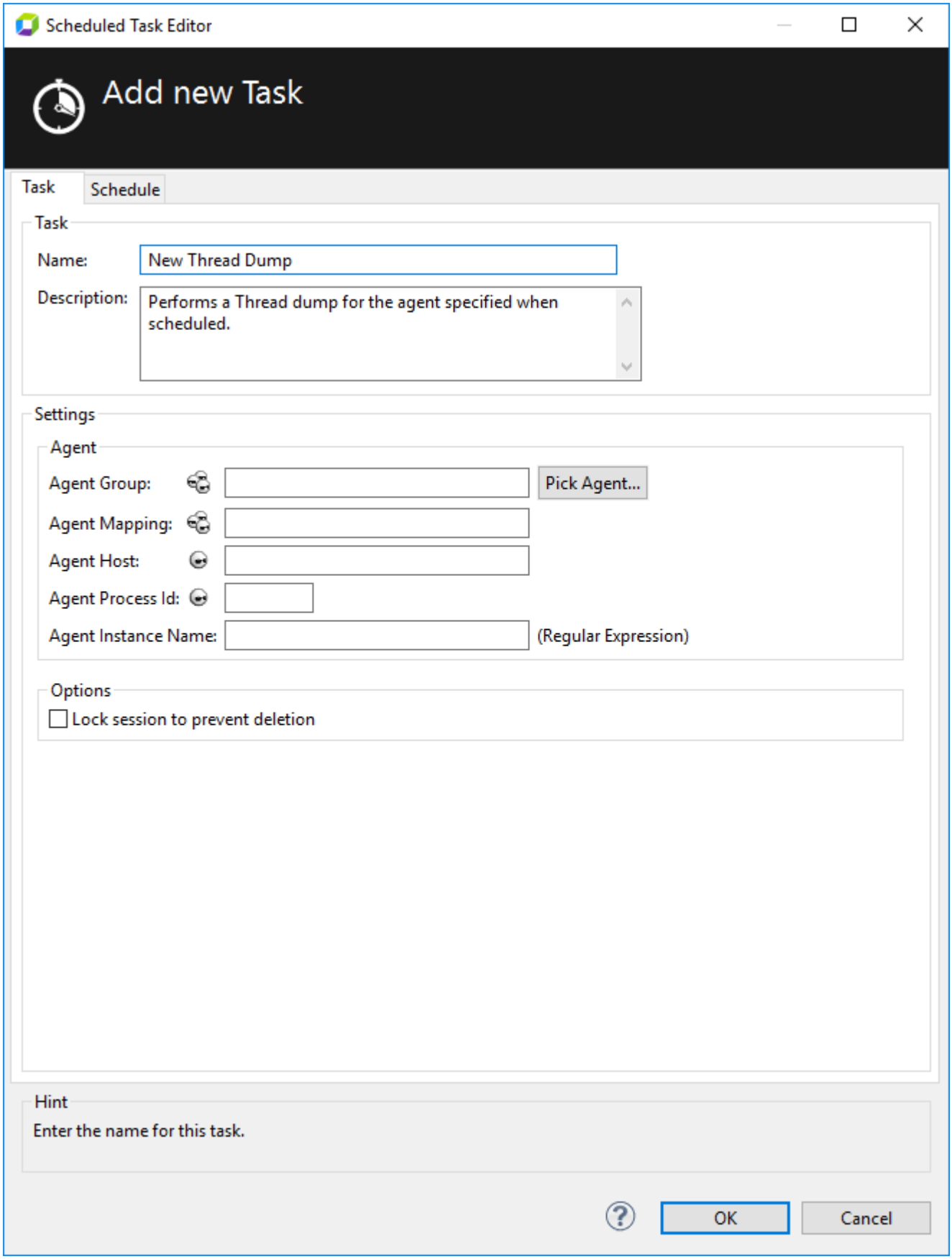


Session Recording Task Configuration

Parameter Name	Description
Name	Name for the recorded session file.
Automatically append timestamp	Select this option to append a timestamp file name.
Description	Description for the recorded session.
Duration (minutes)	How many minutes to continue recording the session.
Lock Session	Select this option to prevent the session from being deleted during automated cleanups on low disk space.
Recording options	Select the information to be stored in the session: **PurePath and time series**, **Only PurePaths marked as violated and time series**, only **Time** series.
Session labels	Specify the labels that will automatically be assigned to the created session. Clicking the **Value** field opens the **Edit Session Labels** dialog box through which you can create **New** labels and **Delete** labels.

Thread dump

The Thread Dump Task performs a thread dump for a specified Agent.



Thread Dump Task Configuration

Parameter Name	Description
Agent Group	To make sure the Agent Group name is entered correctly, click **Pick Agent** to select an Agent Group or Agent through the **Agent Browser**.
Agent Mapping	In the **Agent Browser**, you can select an Agent Group name or an Agent Mapping. If you select an Agent, the Agent Group is entered in the **Agent Group** field and the Agent Mapping is entered in this field. If you select an Agent Group, this field remains blank.
Agent Host	The Agent's host name.
Agent Process Id	The Agent's process ID.
Lock session to prevent deletion	Select this option to prevent the session from being deleted during automated cleanups on low disk space.

Runtime diagnostics

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Memory diagnostics

Sampling based diagnostics

Thread diagnostics

AppMon provides the following built-in features to analyze Java and .NET Runtime problems:

- Thread Diagnostics
- Memory Diagnostics
- Sampling Based Diagnostics

Thread diagnostics

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Grouping threads

Comparing thread dumps

Limitations

Overview

Thread dumps provide a snapshot of all JVM/CLR threads. They are a powerful way of finding deadlocks, idle or busy thread pools, thread leaks, and more.

This page describes how to use thread dumps to diagnose performance bottlenecks.

There are many advantages of thread dumps compared to JVM dumps:

- CPU time information is available.
- Multiple thread dumps can be compared.
- You can group threads by significant criteria.
- You can search for PurePaths where a specific thread is involved.
- To perform a thread dump, it is not necessary to start the VM in the console or redirect the output.
- Thread dumps can be triggered automatically.

> **Note**
>
> While the thread information is collected, all threads are suspended to guarantee consistency of stack traces, states, and monitors.

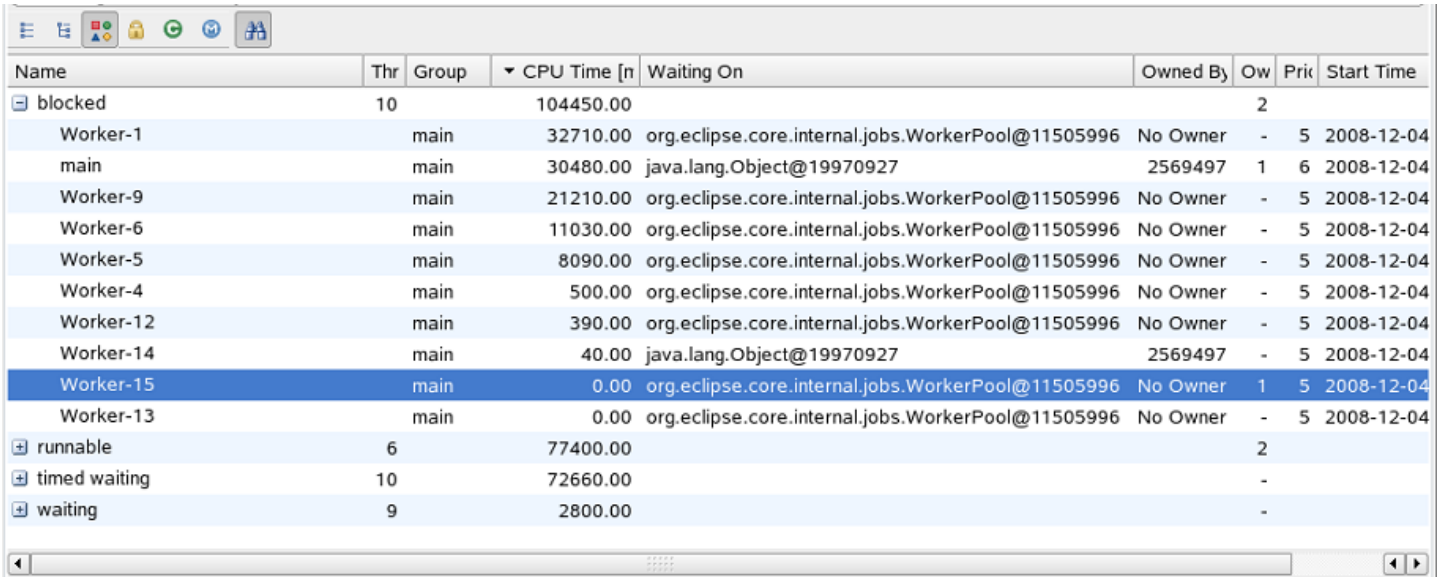
Thread dumps can be persisted, searched, and grouped. You can schedule thread dumps. For example, you might trigger thread dumps at times when the load on the system under diagnosis is low, to ensure that there are no threads leaking, or to compare the CPU usage of different threads to previously created thread dumps.

Grouping threads

It is often tedious to analyze a traditional JVM dump because the vital information is scattered in the log file or tool. For example, you have to know which thread owns which monitor or which threads are waiting for a buffer.

The ability to group threads by different criteria is very handy, especially in production scenarios with many concurrent threads. For example, threads grouped by their thread group show which thread group consumed the most processor time, because the CPU times are accumulated. Grouping by classes or methods is useful to see what is happening in an application at the time of the dump.

Thread state and *thread group* are the most common groupings. The following figure shows a thread dump taken from Eclipse under load.

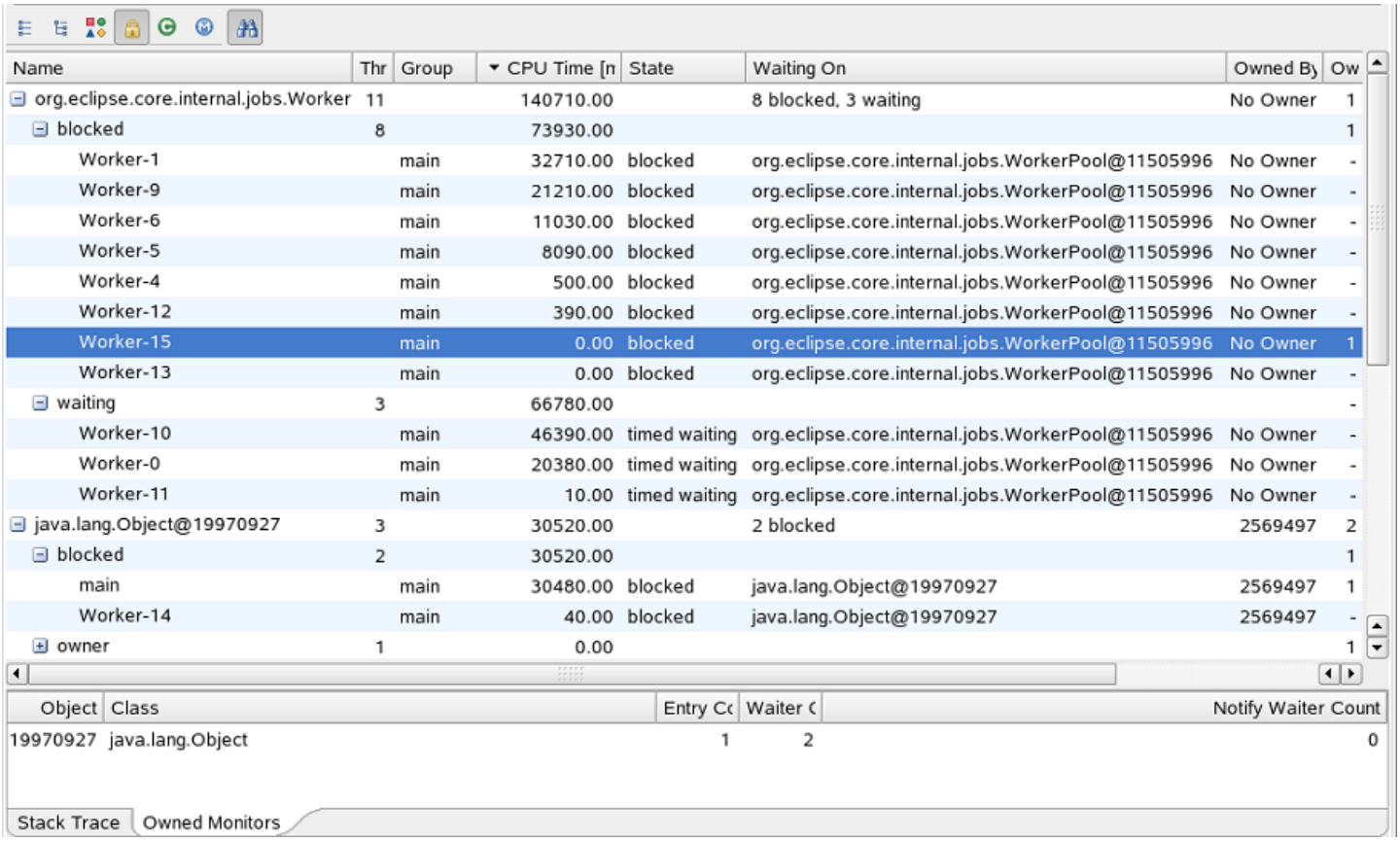


Name	Thr	Group	CPU Time [n	Waiting On	Owned By	Ow	Prio	Start Time
blocked	10		104450.00			2		
Worker-1		main	32710.00	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-	5	2008-12-04
main		main	30480.00	java.lang.Object@19970927	2569497	1	6	2008-12-04
Worker-9		main	21210.00	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-	5	2008-12-04
Worker-6		main	11030.00	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-	5	2008-12-04
Worker-5		main	8090.00	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-	5	2008-12-04
Worker-4		main	500.00	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-	5	2008-12-04
Worker-12		main	390.00	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-	5	2008-12-04
Worker-14		main	40.00	java.lang.Object@19970927	2569497	-	5	2008-12-04
Worker-15		main	0.00	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	1	5	2008-12-04
Worker-13		main	0.00	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-	5	2008-12-04
runnable	6		77400.00			2		
timed waiting	10		72660.00			-		
waiting	9		2800.00			-		

Eclipse thread dump

There are 10 blocked threads and some of them have locks on monitors. Having multiple monitors locked isn't a problem in itself, but it could indicate architectural flaws. For example, it may be that the method was not intended to be called from another method that synchronized on the owned lock.

This figure shows that two threads are waiting for this monitor. It is unclear how long they have been waiting, but it is advisable to keep an eye on them. The methods are candidates for placing additional Sensors.



Name	Thr	Group	CPU Time [n	State	Waiting On	Owned By	Ow
org.eclipse.core.internal.jobs.Worker	11		140710.00		8 blocked, 3 waiting	No Owner	1
blocked	8		73930.00				1
Worker-1		main	32710.00	blocked	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
Worker-9		main	21210.00	blocked	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
Worker-6		main	11030.00	blocked	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
Worker-5		main	8090.00	blocked	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
Worker-4		main	500.00	blocked	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
Worker-12		main	390.00	blocked	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
Worker-15		main	0.00	blocked	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	1
Worker-13		main	0.00	blocked	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
waiting	3		66780.00				-
Worker-10		main	46390.00	timed waiting	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
Worker-0		main	20380.00	timed waiting	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
Worker-11		main	10.00	timed waiting	org.eclipse.core.internal.jobs.WorkerPool@11505996	No Owner	-
java.lang.Object@19970927	3		30520.00		2 blocked	2569497	2
blocked	2		30520.00				1
main		main	30480.00	blocked	java.lang.Object@19970927	2569497	1
Worker-14		main	40.00	blocked	java.lang.Object@19970927	2569497	-
owner	1		0.00				1

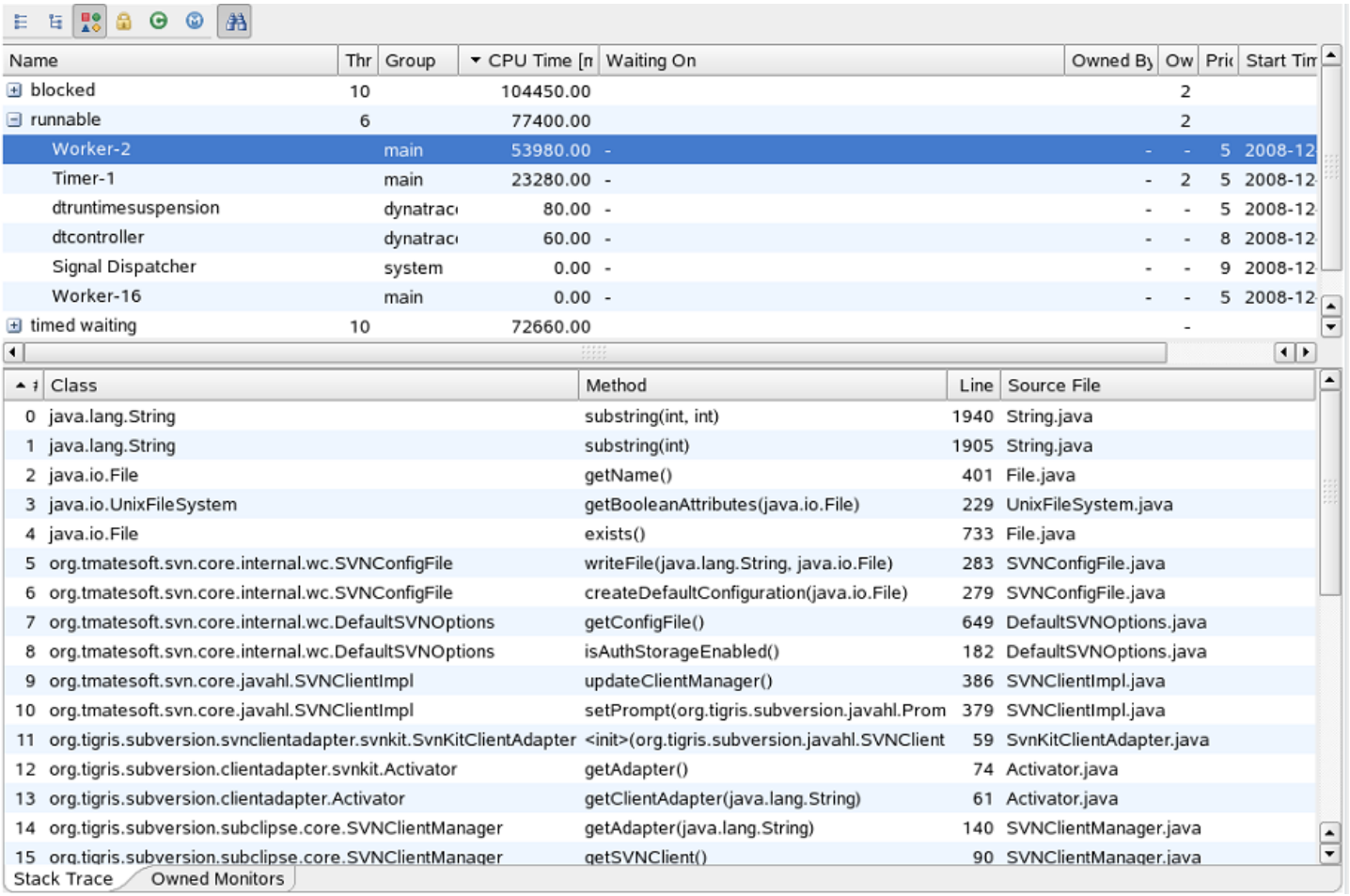
Object	Class	Entry Cc	Waiter C	Notify Waiter Count
19970927	java.lang.Object	1	2	0

Stack Trace | Owned Monitors

Blocked threads

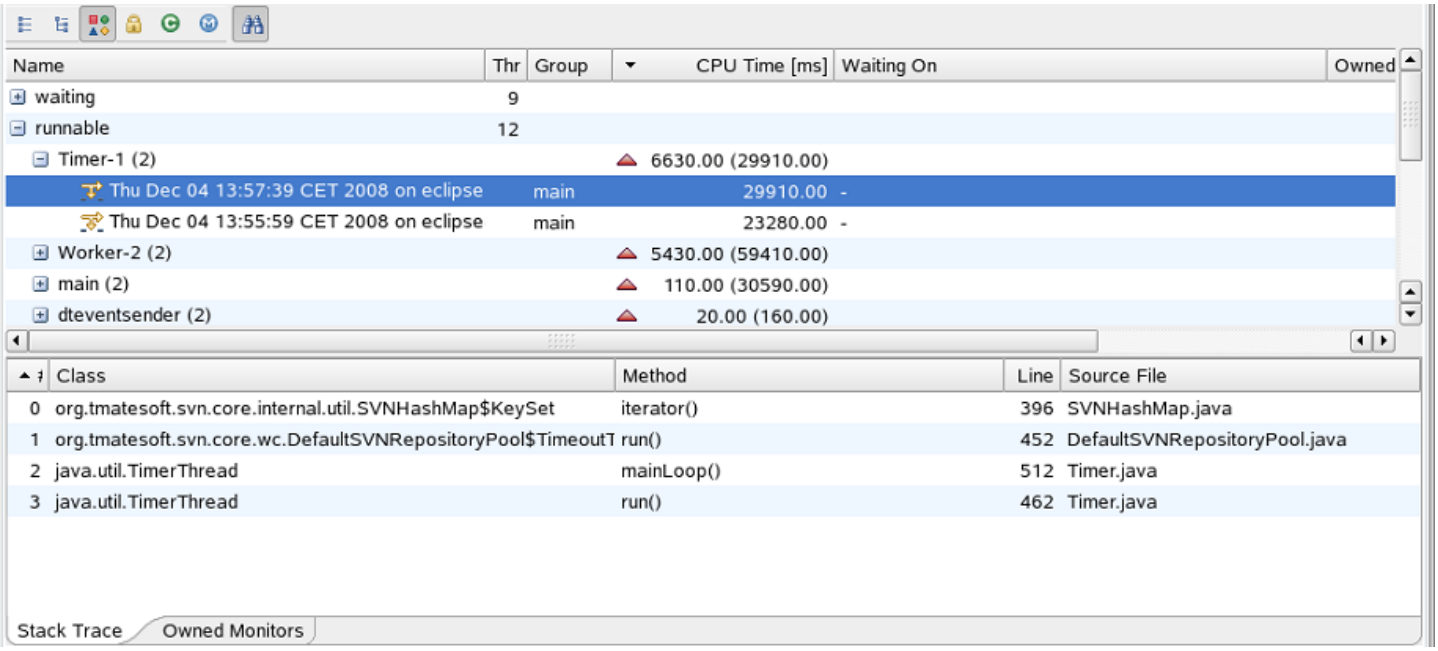
Comparing thread dumps

When the system under diagnosis appears to have random threading issues, it helps to examine all running thread.



Running threads

The most active thread is `Worker-2`. It is updating the Subversion state of all project files. To find out what's taking so long, another thread dump is requested, as shown in the following figure.



Thread dump analysis

The dump shows that `Worker-2` was not the only busy thread. `Timer-1` also consumed a lot of CPU time, probably with Subversion. To learn more, you could place additional Sensor rules for the package `org.tmatesoft.svn.core`. The filter feature shows all threads related to Subversion.

> **Note**
>
> Threads are compared by their ID. Because the ID changes after restarting the system under diagnosis, threads from different runs can only be compared by name. The thread name does not have to be unique, therefore threads with the same name cannot be compared.

Limitations

- Gathering monitor details on some Sun VMs is slow when threads are suspended. It is possible to disable thread suspension by adding the Agent option `threaddumpsuspendthreads=false`. However, with thread suspension disabled, monitor details might not match the thread states or stack traces.
- CPU time is available only with JVM 5.0 and later.
- Acquiring owned monitors under .NET is available only with .NET 4.5.

Memory diagnostics

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Memory diagnosis with AppMon

Related pages

Total Memory dashlet

Selective Memory dashlet

Total Memory Comparison dashlet

Total Memory Content dashlet

Follow References dashlet

Keep Alive Set dashlet

Shortest Root Paths dashlet

Total Memory dashlet

Memory analysis

Watch and learn

Memory diagnostics for Java & .NET

Overview

AppMon provides sophisticated means for diagnosing memory leaks and excessive memory consumption by Java and .NET applications. This page describes memory management in Java and .NET in general, shows how to use AppMon to find memory problems and optimize memory performance, and provides an example of memory diagnostics. See Memory Analysis for a walkthrough of memory analysis.

Inefficient use of application memory affects key performance indicators of service management:

- Unavailability of application due to crashes.
- Bad response time due to frequent runs of garbage collection.
- High investment costs from inefficient use of hardware resources.

At worst, memory leaks increase memory demand until out-of-memory exceptions and application malfunctions occur.

Memory problem classification

Generally, memory problems can be classified as follows:

- Memory leaks, leading to unavailability due to crashes.
- Heavy memory use, leading to high costs from hardware requirements.
- Heavy garbage collection use, impacting application performance by frequent object instantiations and high numbers of objects.

Causes of memory leaks

A *memory leak* is an allocation of memory by the application that is not freed again when the memory is not used anymore. The obvious sign of a memory leak is growing memory consumption with constant load on your application. The causes of memory leaks can be categorized as these programming errors:

- **Collections and maps**: Memory leaks are often caused by collections containing objects that are no longer used.
- **Static fields**: Objects assigned to static fields of classes are not cleaned up until the application is shut down or the fields are set to null. Static collections and maps constitute special problems.
- **Custom data structures**: Custom data structures frequently keep back-references to objects, preventing the freeing of intentionally unused objects.

- **Cleanup errors**: Code does not properly.
  - Use statements to release unmanaged resources, such as native code invoked via JNI.
  - Unregister listeners.
  - Release pooled resources, such as database connections, which can lead to database connection leaks.
- **Inappropriate use of sessions in web applications**: Objects stored in session variables live for the duration of a user's session. If sessions are not invalidated explicitly, these objects live until the session's timeout. This can prevent releasing objects for minutes to hours, depending on the configured timeout. This is not technically a memory leak, but it results in poor application performance from inefficient memory use.
- **Soft-References**: This is not a memory leak by itself, but excessive use of Soft-References makes it hard to detect memory leaks. Soft-References can keep the JVM heap full, making it impossible to use simple JVM metrics monitor the used memory assigned to hard references.
- **Inappropriate pool sizes**: Object pools enable reuse of objects without permanently creating and destroying costly objects. Using pools can restrict the number of objects created for a specific object type. Inappropriate pool sizes may result in the creation of more objects than an application would actually need. Instead of waiting for a pool object to be available, a new one is created.

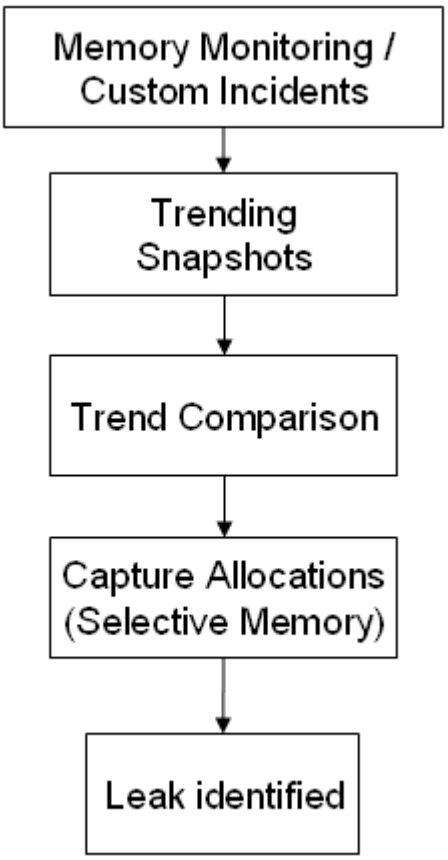
Memory diagnosis with AppMon

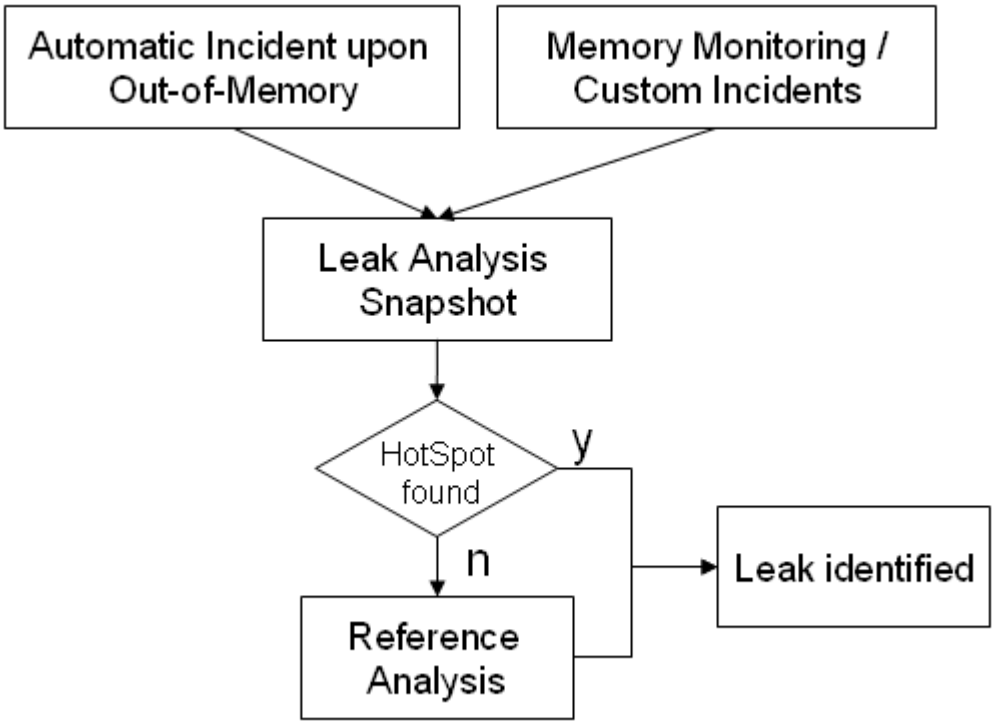
Memory diagnosis is a complex and time-consuming task. AppMon supports efficient and effective detection of memory problems and the implementation of company-specific performance tuning processes.

The diagnosis process

The process of diagnosing a memory leak depends on environmental issues. If it is possible to stall the application for a considerable amount of time (for example 15 minutes for an 8-GB JVM), you can use leak analysis snapshots, which contain all instances and their references currently in the heap. If this is not possible (e.g. the application is running in a production environment), AppMon provides the means for trending snapshots and selective memory snapshots. Trending snapshots do not contain every instance or references between instances. However, they provide an overview of which Instances are growing. You can use this information In conjunction with selective memory snapshots to find where in an application the Instances of a growing class are allocated.

The following figures illustrate the proposed process for diagnosing memory leaks for environments where an application stall is possible and for environments where it is not suitable.





Memory monitoring

The first step for memory diagnosis is usually to monitor the memory behavior using charting. For details, see Charting and Memory Analysis.

You can also use AppMon incident-based monitoring of memory behavior. With incident-based diagnosis, you analyze memory problems when they occur by defining incidents for memory usage (e.g. 80% of maximum memory) that can be combined with corresponding actions to trigger memory snapshots. For Java 6 VMs, AppMon triggers leak analysis snapshots by default if the application runs out of memory.

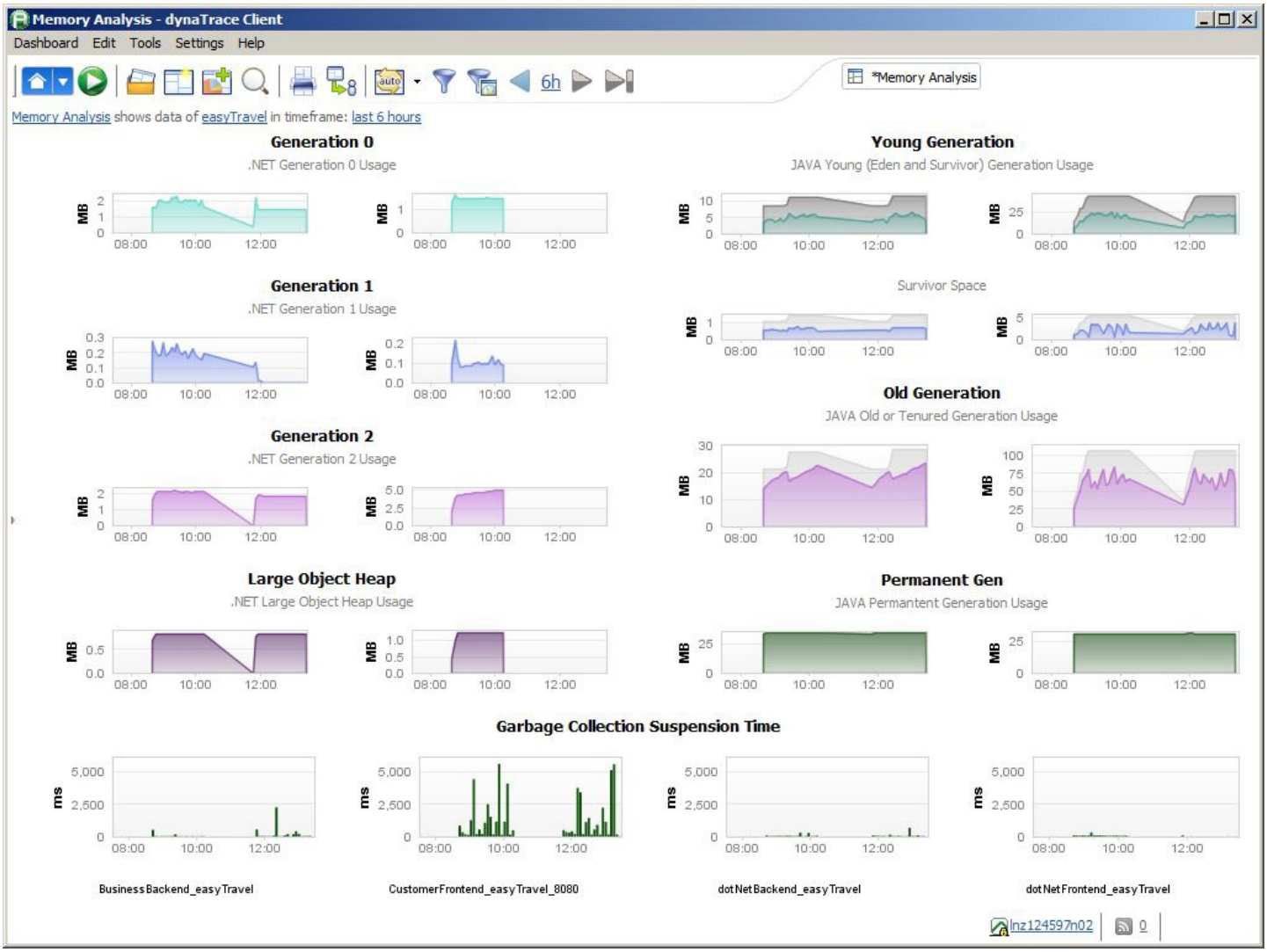
It is recommended for every running JVM or CLR, you monitor memory usage by charting measures for the system. The following measures can be used for memory monitoring:

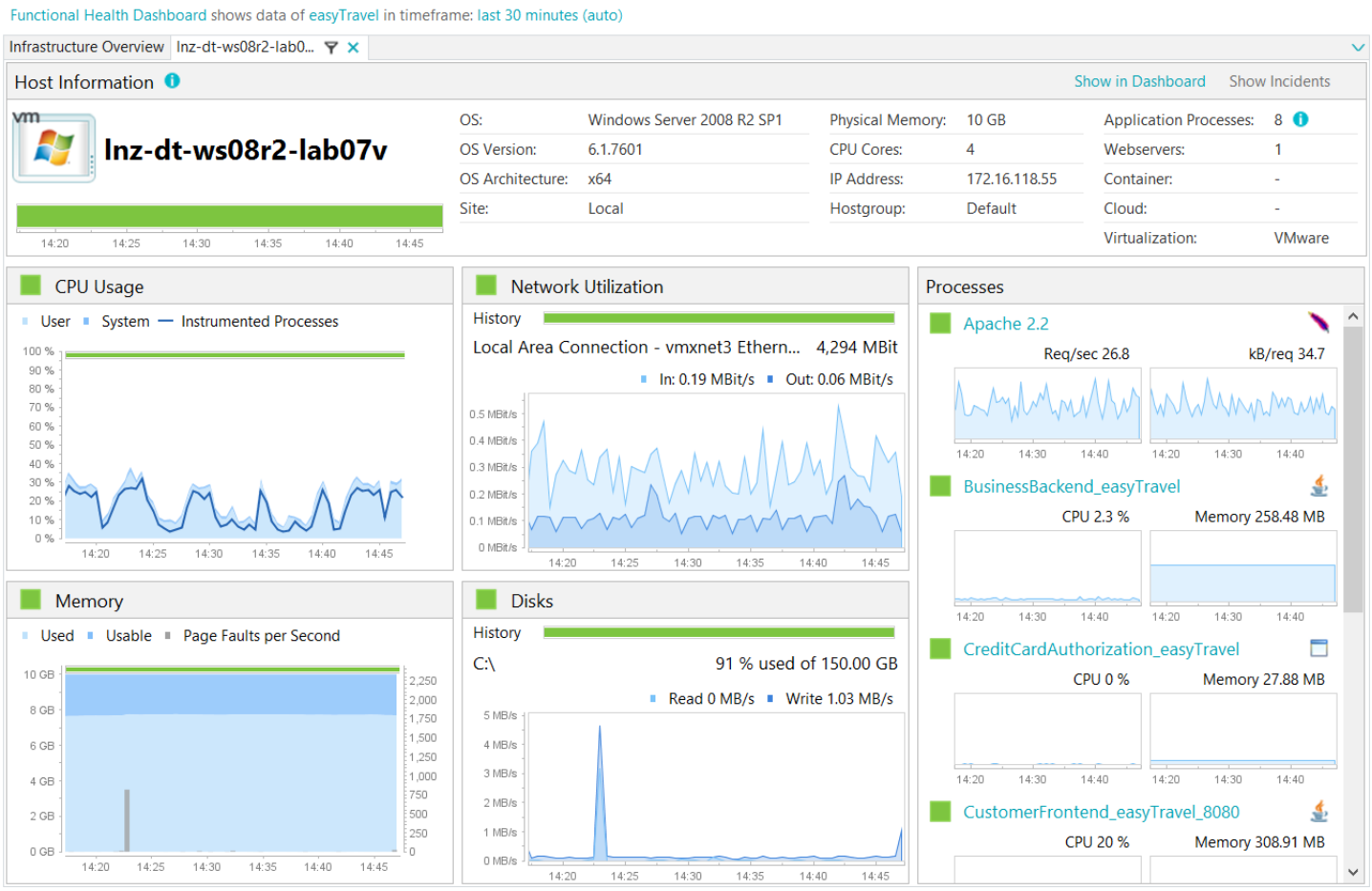
- Java
  - Used Memory
  - Committed Memory
  - Maximum Memory
  - Memory Pools (specific to the Virtual Machine, e.g. Eden, Survivor, Tenured, Perm-Gen for Sun JVMs)
  - Total GC Activations
  - Total GC Collection Time
- .NET
  - .NET Memory Consumption (Virtual)
  - .NET Memory Consumption (Heap Size Gen 0, Gen 1, Gen 2, Large Objects)
  - .NET Garbage Collection (% in GC)

AppMon provides some charts out of the box for monitoring the most important measure regarding memory.

In the Production Edition, **Start Center > Monitor Applications > Host Health Monitoring** displays the **Host Health Overview** dashboard, a comprehensive overview of host performance, including memory usage. In the Development Team Edition, you can display this dashboard via **Start Center > Monitoring > Host Health Overview**.

In the Development Team Edition, **Start Center > Memory Diagnosis > Analyze Memory Usage** opens the **Memory Analysis** dashboard in which you can quickly view the memory usage of the Agents connected to a System Profile.


Memory Analysis - dynaTrace Client
Dashboard Edit Tools Settings Help
6h
*Memory Analysis
Memory Analysis shows data of easyTravel in timeframe: last 6 hours
Generation 0
.NET Generation 0 Usage
Young Generation
JAVA Young (Eden and Survivor) Generation Usage
Generation 1
.NET Generation 1 Usage
Survivor Space
Old Generation
JAVA Old or Tenured Generation Usage
Generation 2
.NET Generation 2 Usage
Large Object Heap
.NET Large Object Heap Usage
Permanent Gen
JAVA Permantent Generation Usage
Garbage Collection Suspension Time
MB
ms
08:00
10:00
12:00
BusinessBackend_easyTravel
CustomerFrontend_easyTravel_8080
dotNetBackend_easyTravel
dotNetFrontend_easyTravel
lnz124597n02
0



You can create memory snapshots either manually or automatically through periodic tasks in the **Total Memory** dashlet of a System Profile.

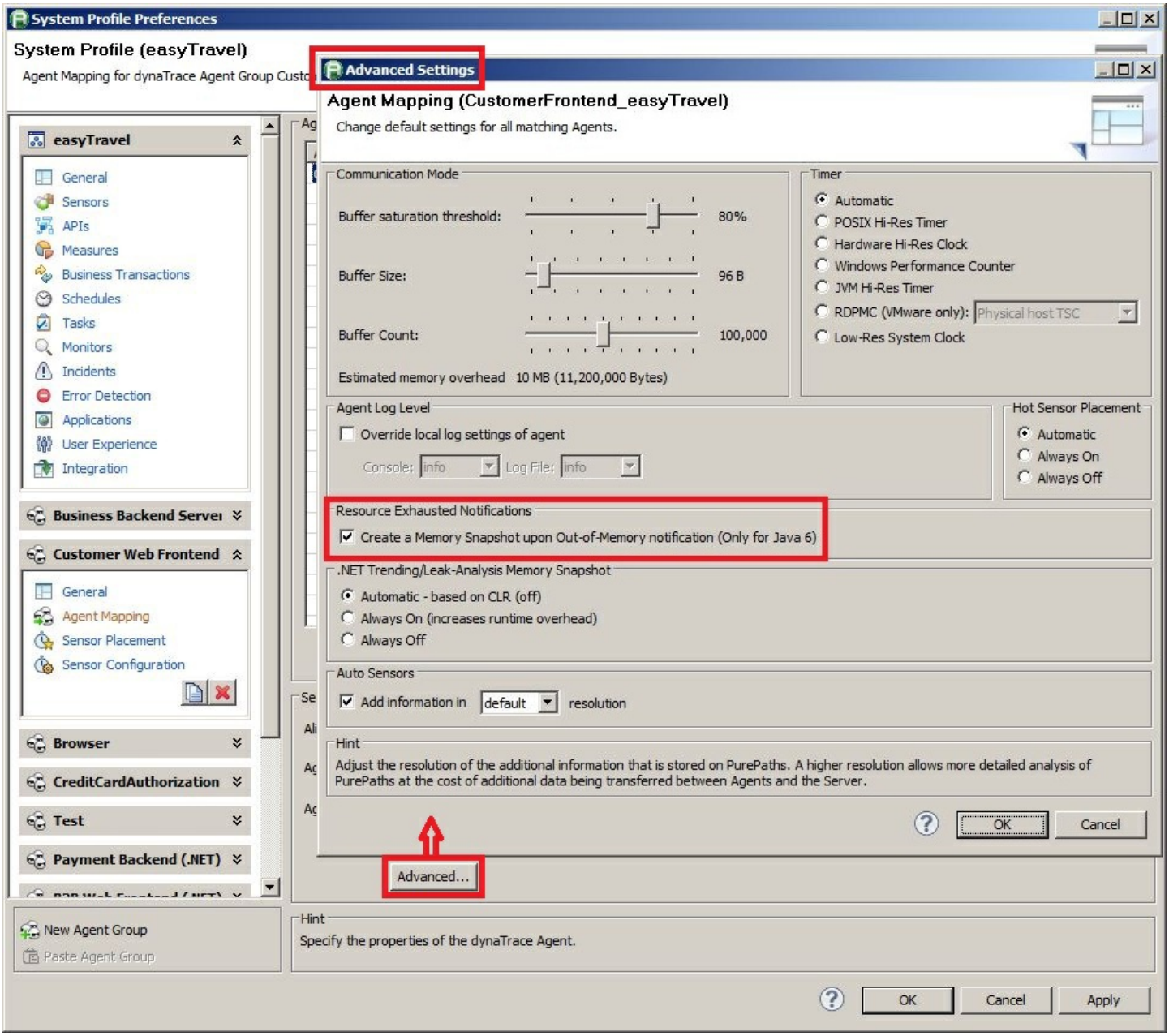
Incident-based memory diagnosis

For incident-based memory diagnosis, do either of the following:

- Automatically create a leak analysis snapshot when an application runs out of memory.
- Define Incidents (for example, if memory usage exceeds 80%) and attach actions (for example, create snapshots) to these incidents.

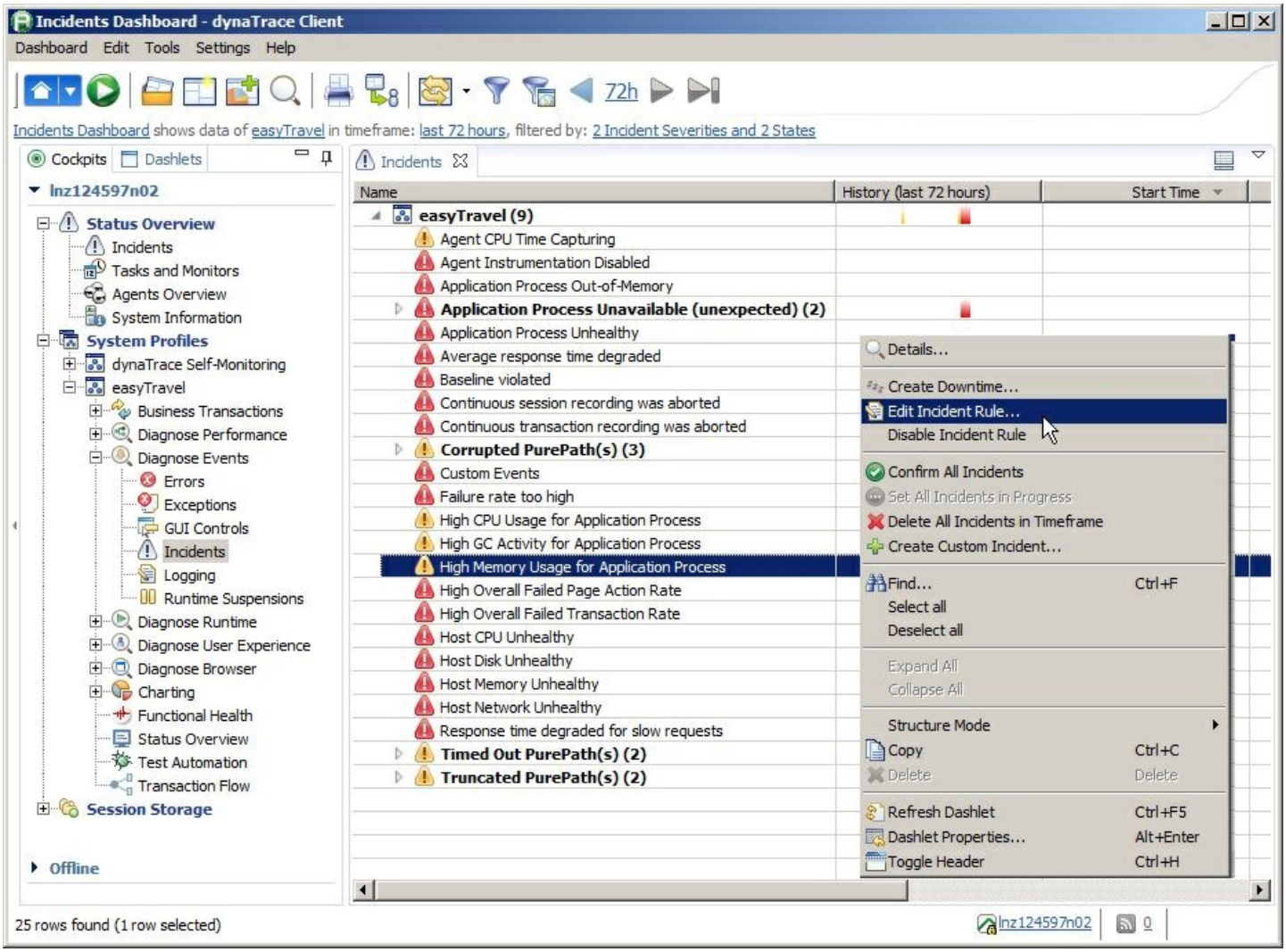
By default, AppMon automatically creates leak analysis snapshots if a Java 6 VM reports an out-of-memory error; in such a case, an incident is raised and a snapshot is triggered.

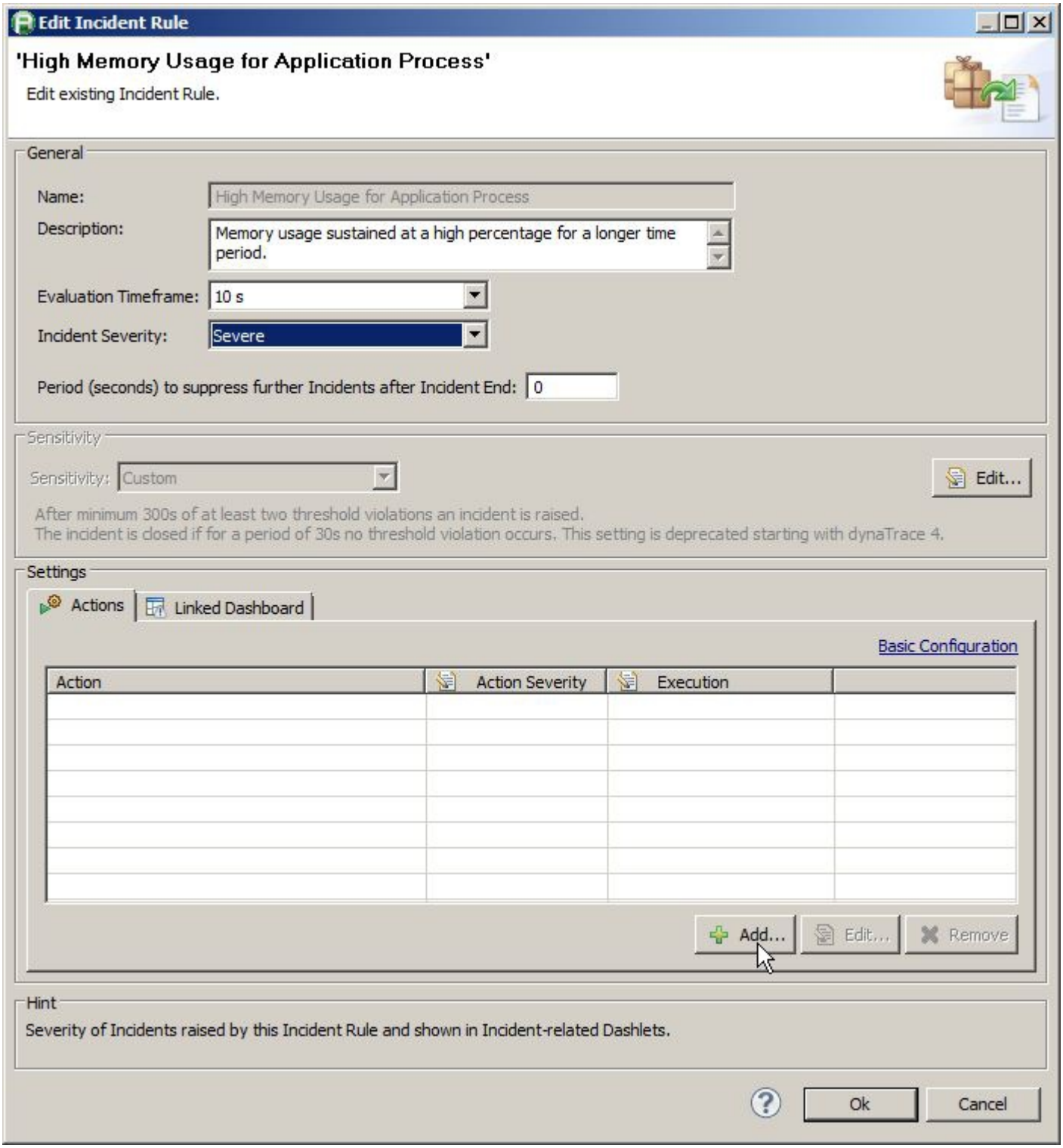
You can disable automatic creation of memory snapshots for out-of-memory errors in the **Advanced Settings** of the Agent Group, as shown in the figure below.

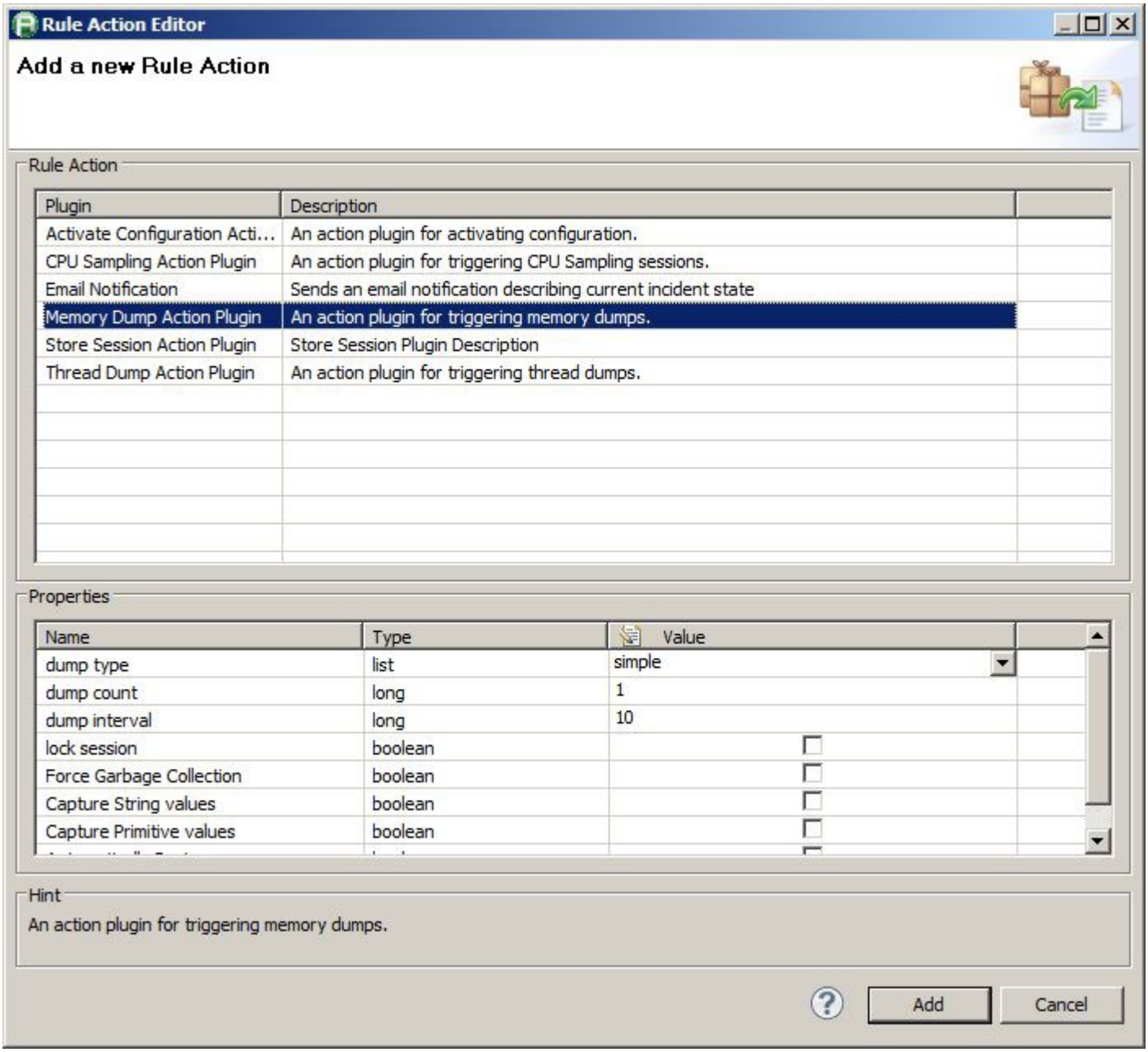


Disabling automatic memory snapshots

You can create incidents for memory-related measures like **Used memory in %**, and to define actions that trigger memory snapshots. AppMon automatically defines such incident rules. To trigger a memory snapshot, add a corresponding action to the incident rule, as shown in the following figures.


Incidents Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
72h
Incidents Dashboard shows data of easyTravel in timeframe: last 72 hours, filtered by: 2 Incident Severities and 2 States
Cockpits Dashlets
lnz124597n02
Status Overview
Incidents
Tasks and Monitors
Agents Overview
System Information
System Profiles
dynaTrace Self-Monitoring
easyTravel
Business Transactions
Diagnose Performance
Diagnose Events
Errors
Exceptions
GUI Controls
Incidents
Logging
Runtime Suspensions
Diagnose Runtime
Diagnose User Experience
Diagnose Browser
Charting
Functional Health
Status Overview
Test Automation
Transaction Flow
Session Storage
Offline
Incidents
Name
History (last 72 hours)
Start Time
easyTravel (9)
Agent CPU Time Capturing
Agent Instrumentation Disabled
Application Process Out-of-Memory
Application Process Unavailable (unexpected) (2)
Application Process Unhealthy
Average response time degraded
Baseline violated
Continuous session recording was aborted
Continuous transaction recording was aborted
Corrupted PurePath(s) (3)
Custom Events
Failure rate too high
High CPU Usage for Application Process
High GC Activity for Application Process
High Memory Usage for Application Process
High Overall Failed Page Action Rate
High Overall Failed Transaction Rate
Host CPU Unhealthy
Host Disk Unhealthy
Host Memory Unhealthy
Host Network Unhealthy
Response time degraded for slow requests
Timed Out PurePath(s) (2)
Truncated PurePath(s) (2)
Details...
Create Downtime...
Edit Incident Rule...
Disable Incident Rule
Confirm All Incidents
Set All Incidents in Progress
Delete All Incidents in Timeframe
Create Custom Incident...
Find... Ctrl+F
Select all
Deselect all
Expand All
Collapse All
Structure Mode
Copy Ctrl+C
Delete Delete
Refresh Dashlet Ctrl+F5
Dashlet Properties... Alt+Enter
Toggle Header Ctrl+H
25 rows found (1 row selected)
lnz124597n02
0


Edit Incident Rule
'High Memory Usage for Application Process'
Edit existing Incident Rule.
General
Name:
High Memory Usage for Application Process
Description:
Memory usage sustained at a high percentage for a longer time period.
Evaluation Timeframe:
10 s
Incident Severity:
Severe
Period (seconds) to suppress further Incidents after Incident End:
0
Sensitivity
Sensitivity:
Custom
Edit...
After minimum 300s of at least two threshold violations an incident is raised.
The incident is closed if for a period of 30s no threshold violation occurs. This setting is deprecated starting with dynaTrace 4.
Settings
Actions
Linked Dashboard
Basic Configuration
Action
Action Severity
Execution
Add...
Edit...
Remove
Hint
Severity of Incidents raised by this Incident Rule and shown in Incident-related Dashlets.
Ok
Cancel



Find growing objects using trending memory snapshots

When you discover a memory problem, you can use trending memory snapshots as a first step to determine which classes are top-most instantiated and compare them among multiple snapshots created over time. Trending snapshots are created:

- Using the **Create Snapshot** dialog box.
- Using incidents, as described above.
- Using automated tasks that periodically trigger a snapshot. See Snapshot Schedules for details.

Note

AppMon persists all memory dumps onto the hard disk of the AppMon Server. This can fill up the disk, especially when creating scheduled dumps of larger VMs or CLRs. Unless suppressed by clearing the options in the **Create Memory Snapshot** dialog box, garbage collection is forced before a memory dump to prevent including already unreferenced objects and to provide better accuracy. Trending snapshots are, however, quite small. One snapshot requires approximately 1–2 MB per on disk.

After identifying classes where the number of instances or their sizes are growing, use the context menu to create appropriate Memory Sensor rules. These rules are used in the next step, which involves selective memory snapshots. See Total Memory Comparison Dashlet and Memory Analysis for more details on using trending memory snapshots to find growing instances.

Capture selective memory allocations

Selective memory snapshots track the instantiation of selected classes defined by Memory Sensor rules. They identify the point in the code where instances are allocated. They do not stall the VM or CLR while they are created. However, they introduce some overhead for heap memory and CPU usage, especially when many instances are generated.

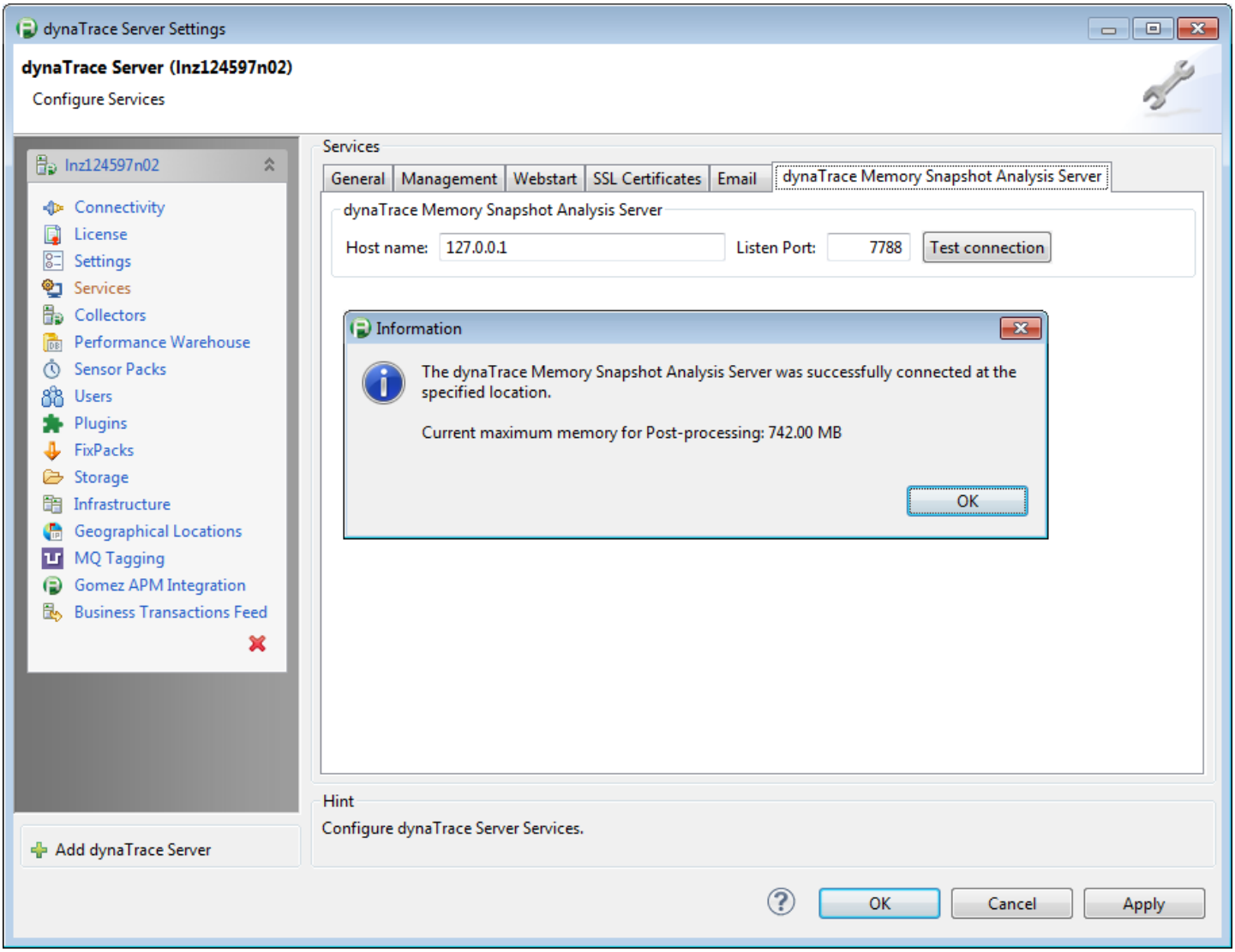
The Memory Comparison dashlet used for trend comparison allows you to define Memory Sensor rules for growing classes. See Selective Memory Dashlet for details regarding selective memory snapshots.

Identify leaks using leak analysis snapshots

Leak analysis snapshots contain all instances and classes, together with their references, that currently exist on the heap.

Creating a leak analysis snapshot stalls the JVM or CLR. For larger heaps, depending on your hardware and the bandwidth of your network connection as well as the selected snapshot options, creating the snapshot can take a long time: creating a snapshot of a 24-GB heap usually takes about 30 minutes, during which time the VM or CLR is completely stalled. Post-processing large leak analysis snapshots also requires lots of memory. To prevent the AppMon Server from purging PurePaths (as it needs memory for post-processing) use an AppMon Memory Analysis Server. For production environments, the AppMon Memory Analysis Server is mandatory.

To configure an AppMon Memory Analysis Server through the AppMon Client, select **Settings > Dynatrace Server > Services > Dynatrace Memory Analysis Server**. See Set up a Memory Analysis Server for a description of how to use the **Dynatrace Memory Analysis Server** tab, shown in the figure below.



Memory Snapshot Analysis Server tab

For a detailed overview of the functionality provided for leak analysis snapshots, see **Total Memory Dashlet**.

Note

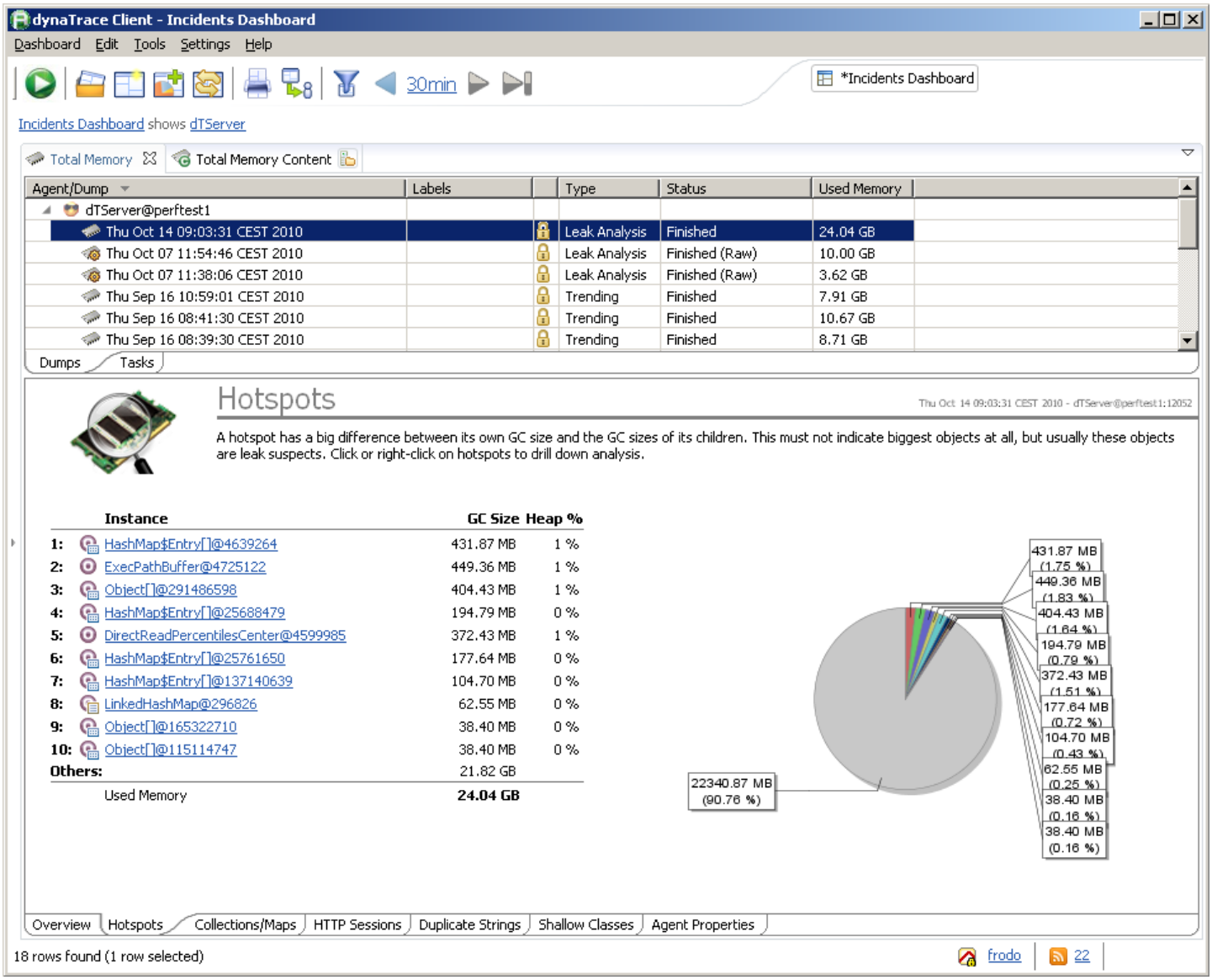
Leak analysis snapshots stall the VM while retrieving heap information. Especially for production environments, consider the impact on your system before creating a snapshot.

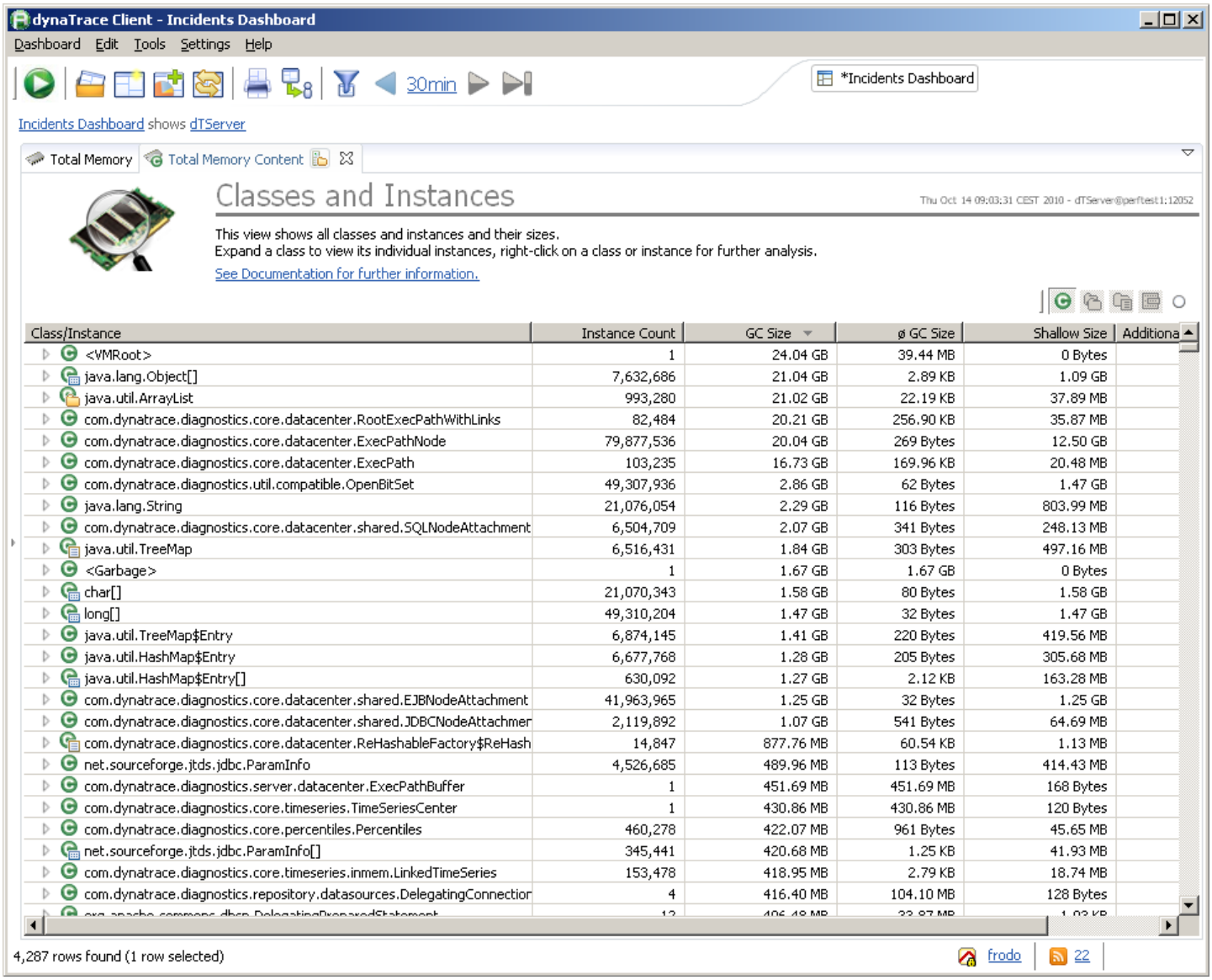
JVMs require a considerable amount of native memory for the creation of a leak analysis snapshot. For example, a JVM with 25 GB of used memory containing 340 million instances requires approximately 15 GB of native memory. CLRs, however, require very little native memory for creating a leak analysis snapshot.

The **Total Memory dashlet** provides an overview of the HotSpots found while post-processing a leak analysis snapshot. Look at the **HotSpot** pane to check whether AppMon has found a potential leak. If a leak was identified, use the analysis drilldowns to investigate leak suspects. For details, see the walkthrough in Memory Analysis. Use the **Keep Alive Set Dashlet** dashlet to see which instances are kept alive by this instance and the **Shortest Root Paths Dashlet** dashlet to investigate how this large instance is kept alive. If the keep alive set contains instances that should **not** be kept alive, drill down to the s hortest root paths of these instances to identify why they are still referenced. If the keep alive root path doesn't show you why an instance is kept alive, use direct root paths to show the direct references on the heap.

If the HotSpots do not contain the instance causing a memory leak, open the content of the snapshot to get an overview of classes and their instance counts as well as their corresponding Garbage Collection Size. This is usually a good point in your analysis to consult with an architect or developer if you do not have detailed knowledge of the implementation of your application, to help you figure out if the instance counts, the GC sizes, and in particular the references between instances are feasible.

The following figures show the **HotSpots** pane and the content of a leak analysis snapshot.


dynaTrace Client - Incidents Dashboard
Dashboard Edit Tools Settings Help
30min
*Incidents Dashboard
Incidents Dashboard shows dTServer
Total Memory
Total Memory Content
Agent/Dump
Labels
Type
Status
Used Memory
dTServer@perftest1
Thu Oct 14 09:03:31 CEST 2010 Leak Analysis Finished 24.04 GB
Thu Oct 07 11:54:46 CEST 2010 Leak Analysis Finished (Raw) 10.00 GB
Thu Oct 07 11:38:06 CEST 2010 Leak Analysis Finished (Raw) 3.62 GB
Thu Sep 16 10:59:01 CEST 2010 Trending Finished 7.91 GB
Thu Sep 16 08:41:30 CEST 2010 Trending Finished 10.67 GB
Thu Sep 16 08:39:30 CEST 2010 Trending Finished 8.71 GB
Dumps
Tasks
Hotspots
A hotspot has a big difference between its own GC size and the GC sizes of its children. This must not indicate biggest objects at all, but usually these objects are leak suspects. Click or right-click on hotspots to drill down analysis.
Instance
GC Size
Heap %
1: HashMap$Entry[]@4639264 431.87 MB 1 %
2: ExecPathBuffer@4725122 449.36 MB 1 %
3: Object[]@291486598 404.43 MB 1 %
4: HashMap$Entry[]@25688479 194.79 MB 0 %
5: DirectReadPercentilesCenter@4599985 372.43 MB 1 %
6: HashMap$Entry[]@25761650 177.64 MB 0 %
7: HashMap$Entry[]@137140639 104.70 MB 0 %
8: LinkedHashMap@296826 62.55 MB 0 %
9: Object[]@165322710 38.40 MB 0 %
10: Object[]@115114747 38.40 MB 0 %
Others: 21.82 GB
Used Memory 24.04 GB
22340.87 MB (90.76 %)
431.87 MB (1.75 %)
449.36 MB (1.83 %)
404.43 MB (1.64 %)
194.79 MB (0.79 %)
372.43 MB (1.51 %)
177.64 MB (0.72 %)
104.70 MB (0.43 %)
62.55 MB (0.25 %)
38.40 MB (0.16 %)
38.40 MB (0.16 %)
Overview
Hotspots
Collections/Maps
HTTP Sessions
Duplicate Strings
Shallow Classes
Agent Properties
18 rows found (1 row selected)
frodo
22



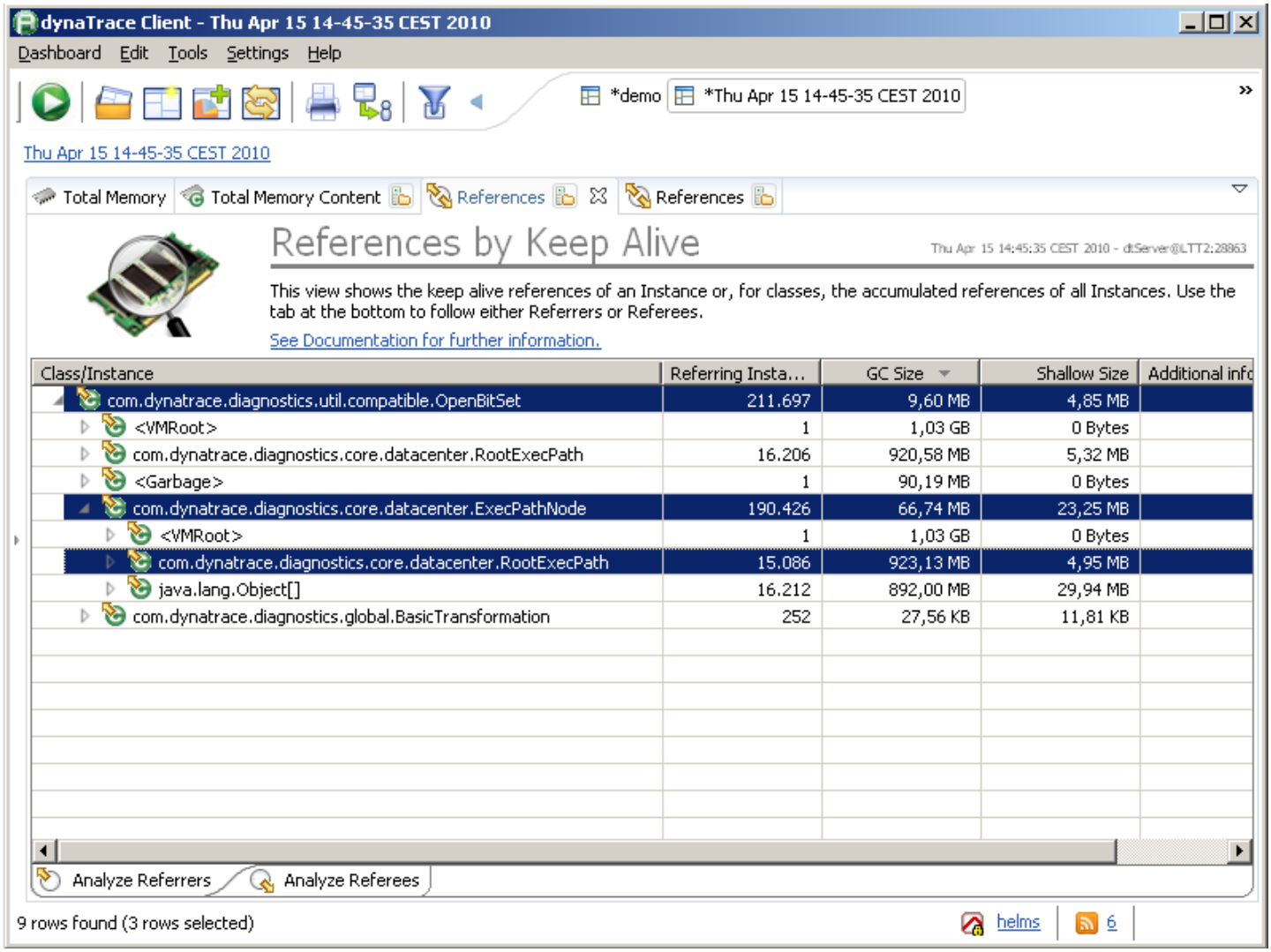
If the HotSpots view of your snapshot does not show suspicious instances and the snapshot contains a large number of instances for one or more classes, the analysis is usually more complex and time-consuming. Open the **Total Memory Content** dashlet, as shown above, and look for suspicious instance counts or Garbage Collection size of classes. You can also compare the average instance Garbage Collection size for a class to the size of individual instances to find any unreasonably large instance. If you find a single instance, use the Keep Alive Set of the instance and follow the root paths of suspicious instances of the **Keep Alive Set** contents.

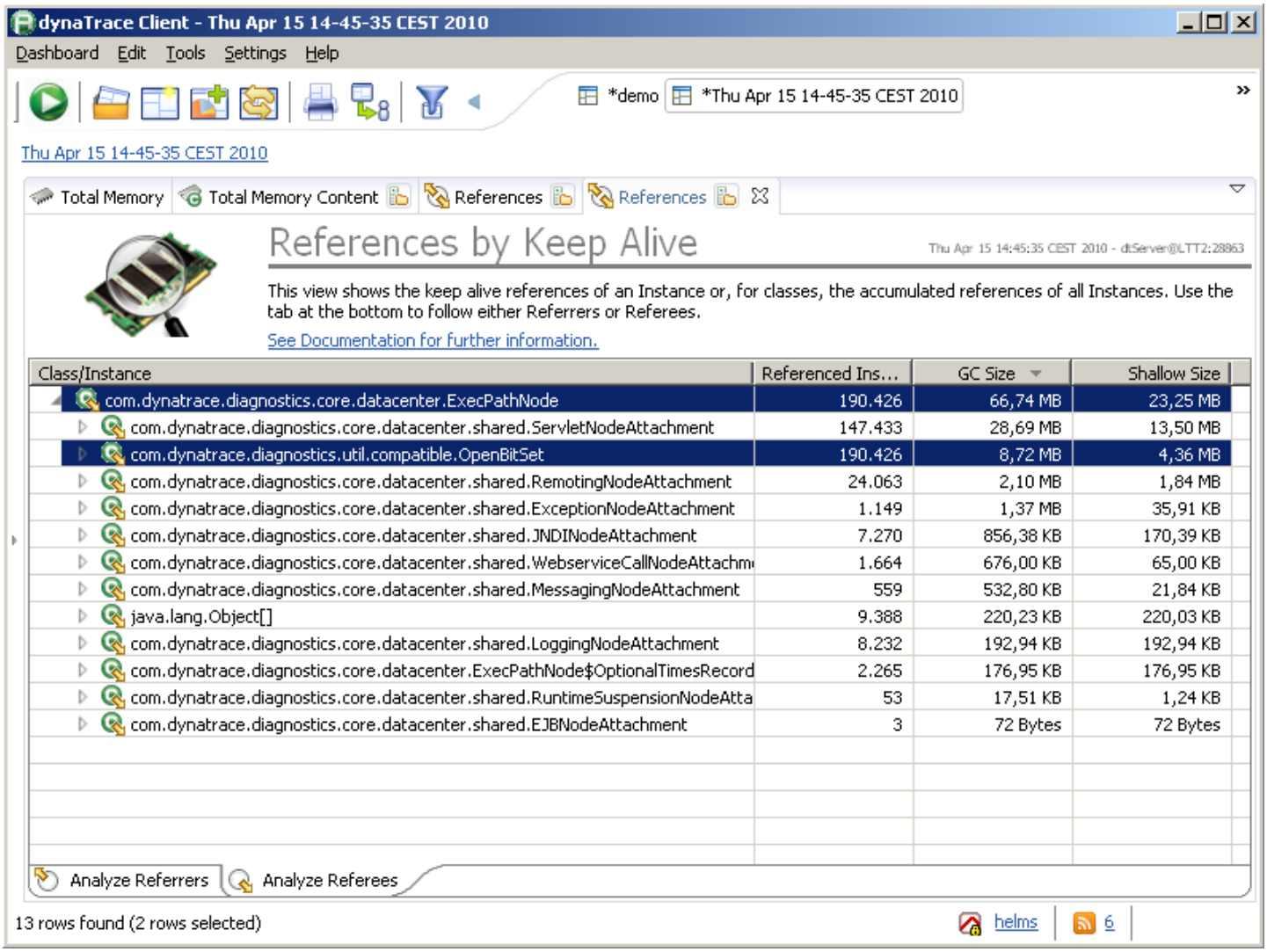
Note

For very large heaps, reproduce your memory leak with a smaller heap if possible. Less time is required to create, post-process, and investigate a snapshot with a smaller heap than with a huge heap, especially if there is no single instance that is keeping others alive. Using a fast hard disk (or SSD) for session storage improves analysis performance.

If there is no single instance to be investigated, you can analyze aggregated follow references in the **Follow References Dashlet** dashlet (drill down from a class record). *Aggregated follow references* are followed for all instances of a class. The goal is to find referrers that should not be there. Keep in mind that an aggregated reference analysis follows the references of each instance of a class and thus can require several minutes. Following by keep-alive should show how many of the instances are kept alive, and by which other instances. Following direct references should show how many of the instances are referenced directly, and by which other instances.

The figure below illustrates an aggregated **References by Keep Alive** for all `OpenBitSet` instances found in a leak analysis snapshot. The referrer analysis shows that that there are 190,426 `ExecPathNode` instances that are referring at least one `OpenBitSet`. This does not mean that all `ExecPathNode` instances are referring a total of 190,426 `OpenBitSet` instances. It might mean that one `OpenBitSet` instance is referred by 190,426 `ExecPathNode` instances. As this is suspicious, you should check whether there is really an `OpenBitSet` for each `ExecPathNode`. Drill down from the `ExecPathNodes` to keep-alive references using **References by Keep Alive**. In the new dashlet, switch to the **Analyze Referees** tab (i.e. the children) at the bottom and request the referees of those 190,426 `ExecPathNode` instances. The tab shows that 190,426 individual `OpenBitSet` instances are referenced. This identifies the problem: there should only be a single instance of `OpenBitSet` for `ExecPathNode`.


dynaTrace Client - Thu Apr 15 14-45-35 CEST 2010
Dashboard Edit Tools Settings Help
*demo
*Thu Apr 15 14-45-35 CEST 2010
Thu Apr 15 14-45-35 CEST 2010
Total Memory
Total Memory Content
References
References
References by Keep Alive
Thu Apr 15 14:45:35 CEST 2010 - dtServer@LTT2:28863
This view shows the keep alive references of an Instance or, for classes, the accumulated references of all Instances. Use the tab at the bottom to follow either Referrers or Referees.
See Documentation for further information.
Class/Instance
Referring Insta...
GC Size
Shallow Size
Additional info
com.dynatrace.diagnostics.util.compatible.OpenBitSet 211.697 9,60 MB 4,85 MB
<VMRoot> 1 1,03 GB 0 Bytes
com.dynatrace.diagnostics.core.datacenter.RootExecPath 16.206 920,58 MB 5,32 MB
<Garbage> 1 90,19 MB 0 Bytes
com.dynatrace.diagnostics.core.datacenter.ExecPathNode 190.426 66,74 MB 23,25 MB
<VMRoot> 1 1,03 GB 0 Bytes
com.dynatrace.diagnostics.core.datacenter.RootExecPath 15.086 923,13 MB 4,95 MB
java.lang.Object[] 16.212 892,00 MB 29,94 MB
com.dynatrace.diagnostics.global.BasicTransformation 252 27,56 KB 11,81 KB
Analyze Referrers
Analyze Referees
9 rows found (3 rows selected)
helms
6



While using aggregated reference analysis, you can open a result (e.g. 190,426 individual out of 5,000,000 `ExecPathNode` instances) in a new dashlet that allows to view the corresponding instances. Selecting **Show Instances** in the context menu to open a dashlet that contains only the corresponding class. Expand this class, to see only the corresponding Instances.

See Memory Analysis for a walkthrough of leak analysis using the easyTravel application.

Sampling based diagnostics

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

When to use sampling

Sampling FYIs

Limitations

Wording

Sampling sessions

Sampling session interpretation

Errors

Related pages

Runtime Specific dashlets

Selective Memory dashlet

Threads dashlet

Total Memory dashlet

When to use sampling

AppMon automatically captures performance relevant method invocations (shown in the **CPU Sampling** dashlet) and does not define explicit sensors. Therefore, you would use CPU sampling to:

- Identify an entry point: Find your custom entry points to define Custom Entry Point Sensors.
- Analyze background threads: If AppMon does not capture threads as PurePaths, analyze them with CPU sampling.

Sampling FYIs

- Sampling investigates the current stack of all running Java or .NET application threads to determine if a method is long running or called very often.
- Sampling doesn't see calls across thread or application boundaries as do PurePaths Dashlet.
- CPU times are measured on a per thread basis, not per method. To visualize it per method, CPU time is assigned to methods based on the number of times a method is seen as the executing one.

Limitations

CPU sampling for Java is available for Java versions 1.5+ and .NET versions 1.1+.

Wording

leaf method	A method that has been seen at least once at the top of a stack trace. When the method sees the stack trace, it is executing.
active thread	A method is counted active if the thread is currently in a runnable state.
inactive thread	A method is counted inactive if the thread it is executed in is blocked or waiting.

Sampling sessions

Sampling sessions receive an *incomplete* state when sampling stops unexpectedly. This occurs when the collector at the agent connection stops, or when the agent itself stops. You can open incomplete sampling sessions like you open correctly finished sessions.

Sampling session interpretation

Stacktraces grouping

Hotspot view

In the Hotspot tree, all methods that are seen executing are root nodes. The callers of the methods are the tree children.

Group by root methods

In this view, all grouped stack traces ignore the thread in which they were executed. This is useful if you don't care about the thread in which your method is executed. This is true for code that is executed by thread pools (like worker threads of web-applications).

Group by thread

All stack traces are grouped by the thread in which they were executed.

Filtering capabilities

Click `Ctrl-F` to filter the tree. The following predefined filtering capabilities are at the top-right:

- The Package filter hides methods that are called from the same package. This shows only methods where the parent node in the tree is a method of another package.
- The Hide Delegates filter hides methods that are always seen together with the calling method.
- The Only Active filter ignores threads that are only seen inactive. This is enabled by default. If you are looking for synchronization or similar problems, disable this filter.

The nodes that match your filter text always display, even if the Hide Same Package filter or Hide Delegates filter is enabled. When you switch to search mode, the behavior is different. Nodes that match your search do not display if they are hidden by these filters.

Accuracy

Methods may be shown as hotspots, however these methods are cheap. The JVM doesn't hold the threads exactly when the stack traces are requested, but only on so-called *safe points*. Safe points have their origins in garbage collection. The Hotspot JVM sets a safe point when it exits a method and will see a stack trace when a method just returns.

Errors

No methods consuming CPU time have been found

Reasons:

- No methods have been seen active while the thread in which they were running consumed CPU time.
- The sampling session was created with an AppMon version lower than 3.5, where this data wasn't written.

No waiting methods have been found

Reasons:

- No methods have been seen in inactive state. This applies to .NET because sampling only regards managed code.
- The sampling session was created with an AppMon version lower than 3.5, where this data wasn't written.

CPU usage in overview differs from an operating system process monitor output

This can happen because AppMon sampling does not apply to native threads.

Reporting

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Types and formats

Report content

Related pages

Client reporting

Reporting limits

Scheduled reporting

Dashboards and reporting (REST)

XML reporting (REST)

Overview

AppMon provides the possibility to extract information from dashboards and dashlets and turn it into customized reports.

Reports can either be created by the client or by the server. Scheduled reports are one kind of server side reports.

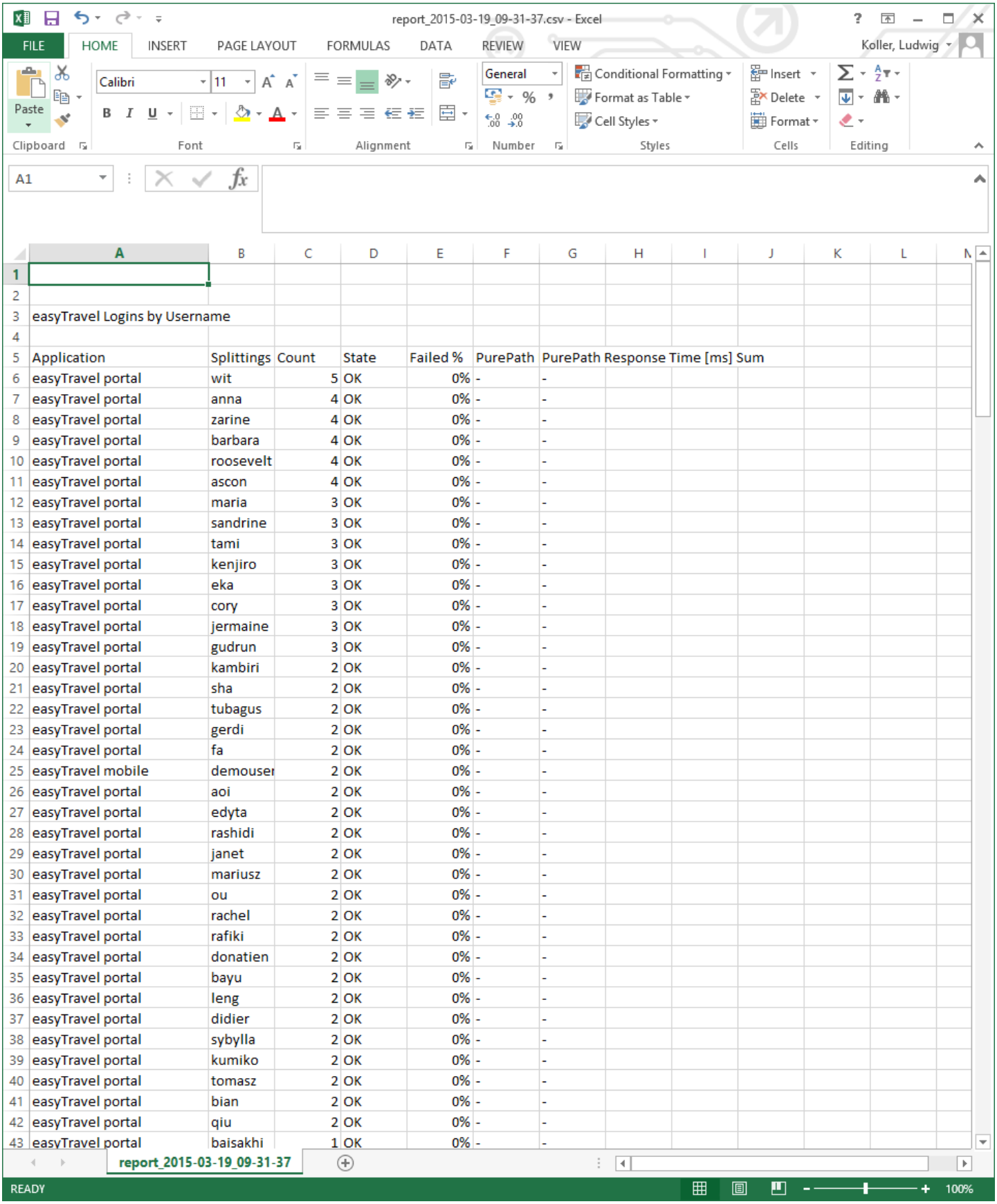
- Find more info on client-side reporting.
- Find more info on scheduled reporting.
- Find more info on server side reporting and Server REST Interfaces.

Reporting has certain limitations which are listed on this page.

Types and formats

AppMon includes the following customizable report formats:

- PDF
- HTML and Tabbed HTML
- XML
- Microsoft Excel and Excel 2007
- CSV



easyTravel Logins by Username

Application	Splittings	Count	State	Failed %	PurePath	PurePath Response Time [ms] Sum
easyTravel portal	wit	5	OK	0%	-	-
easyTravel portal	anna	4	OK	0%	-	-
easyTravel portal	zarine	4	OK	0%	-	-
easyTravel portal	barbara	4	OK	0%	-	-
easyTravel portal	roosevelt	4	OK	0%	-	-
easyTravel portal	ascon	4	OK	0%	-	-
easyTravel portal	maria	3	OK	0%	-	-
easyTravel portal	sandrine	3	OK	0%	-	-
easyTravel portal	tami	3	OK	0%	-	-
easyTravel portal	kenjiro	3	OK	0%	-	-
easyTravel portal	eka	3	OK	0%	-	-
easyTravel portal	cory	3	OK	0%	-	-
easyTravel portal	jermaine	3	OK	0%	-	-
easyTravel portal	gudrun	3	OK	0%	-	-
easyTravel portal	kambiri	2	OK	0%	-	-
easyTravel portal	sha	2	OK	0%	-	-
easyTravel portal	tubagus	2	OK	0%	-	-
easyTravel portal	gerdi	2	OK	0%	-	-
easyTravel portal	fa	2	OK	0%	-	-
easyTravel mobile	demouser	2	OK	0%	-	-
easyTravel portal	aoi	2	OK	0%	-	-
easyTravel portal	edyta	2	OK	0%	-	-
easyTravel portal	rashidi	2	OK	0%	-	-
easyTravel portal	janet	2	OK	0%	-	-
easyTravel portal	mariusz	2	OK	0%	-	-
easyTravel portal	ou	2	OK	0%	-	-
easyTravel portal	rachel	2	OK	0%	-	-
easyTravel portal	rafiki	2	OK	0%	-	-
easyTravel portal	donatien	2	OK	0%	-	-
easyTravel portal	bayu	2	OK	0%	-	-
easyTravel portal	leng	2	OK	0%	-	-
easyTravel portal	didier	2	OK	0%	-	-
easyTravel portal	sybylla	2	OK	0%	-	-
easyTravel portal	kumiko	2	OK	0%	-	-
easyTravel portal	tomasz	2	OK	0%	-	-
easyTravel portal	bian	2	OK	0%	-	-
easyTravel portal	qiu	2	OK	0%	-	-
easyTravel portal	baisakhi	1	OK	0%	-	-

Report: easyTravel Business Da...

file:///C:/workspaces/trunk-classic/jloadtrace/client/reports/report_2015-03-19_09-14-04.l

easyTravel Business Dashboard

This dashboard shows an overview of bookings and revenue per tenant and destination.

Contents

Conversion Rate: Logins vs. Bookings
Bookings by Tenant
Bookings by Destination
Revenue per 15 minutes
Revenue by Tenant
Revenue by Destination
Bookings by Location - Comparison
easyTravel Trip Search by Destination

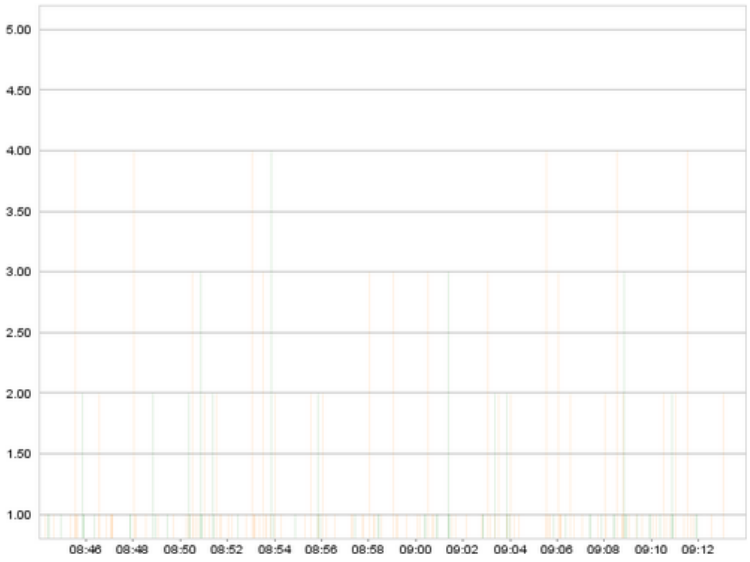
Details

Created on Thu Mar 19 09:14:07 CET 2015
Created by admin
Source: dynaday - easyTravel
Filter: Tests, last 30 minutes , change: Select
Go

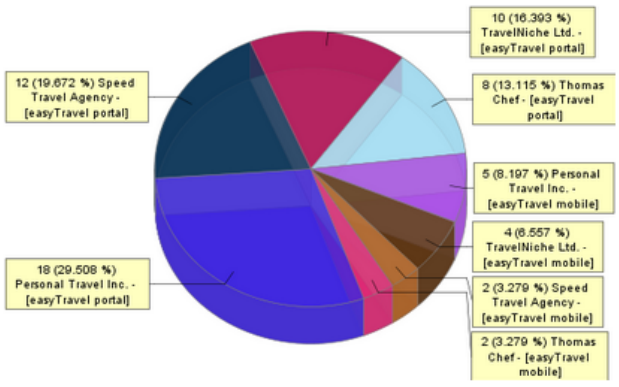
View in dynaTrace

Conversion Rate: Logins vs. Bookings

Compares logins (orange) to actual bookings (green)



Bookings by Tenant



report_2015-03-19_09-30-14.pdf - Adobe Reader

File Edit View Window Help

1 (1 of 4) 96,4% Fill & Sign Comment

easyTravel Memory

Contents

Text and Image
Memory Utilization
Memory Breakdown
Major vs Minor GC

Details

Created on Thu Mar 19 09:30:16 CET 2015
Created by admin
Source: dynaday - easyTravel
Filter: Tests, last 30 minutes

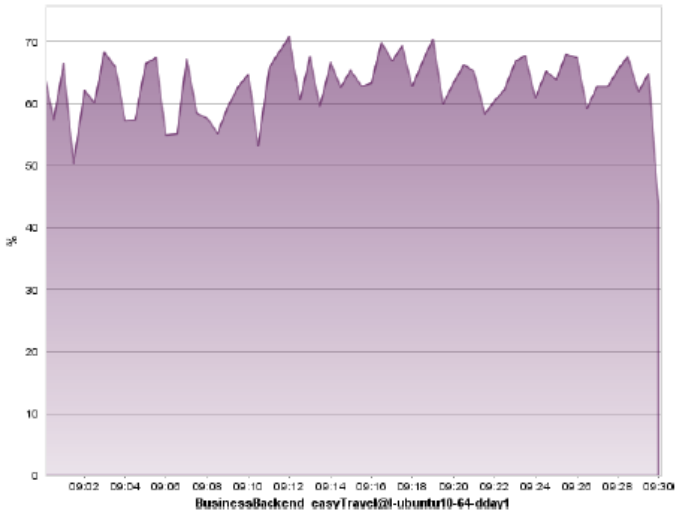
View in dynaTrace
View in Browser

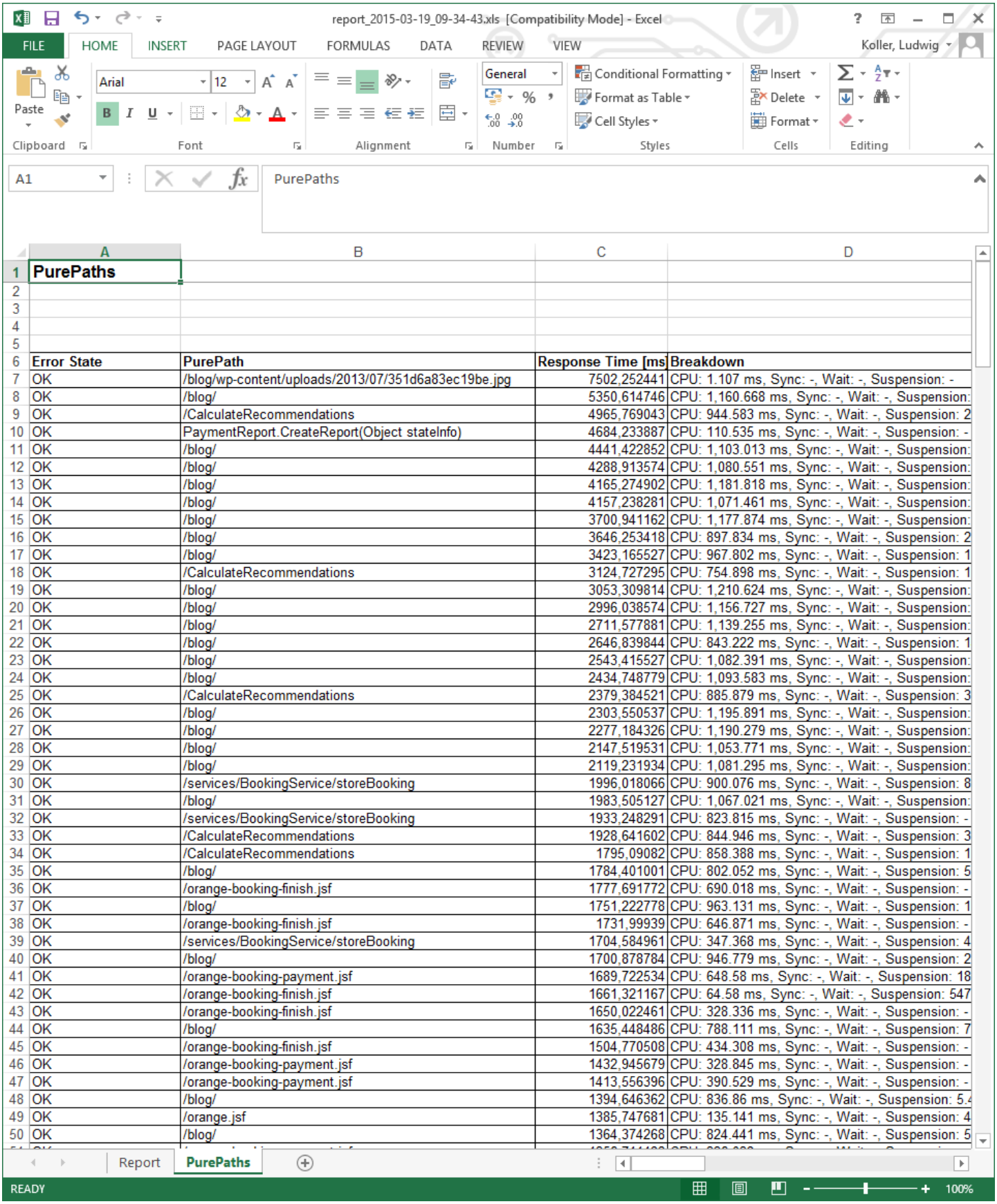
Text and Image

easyTravel - Memory

Memory Utilization

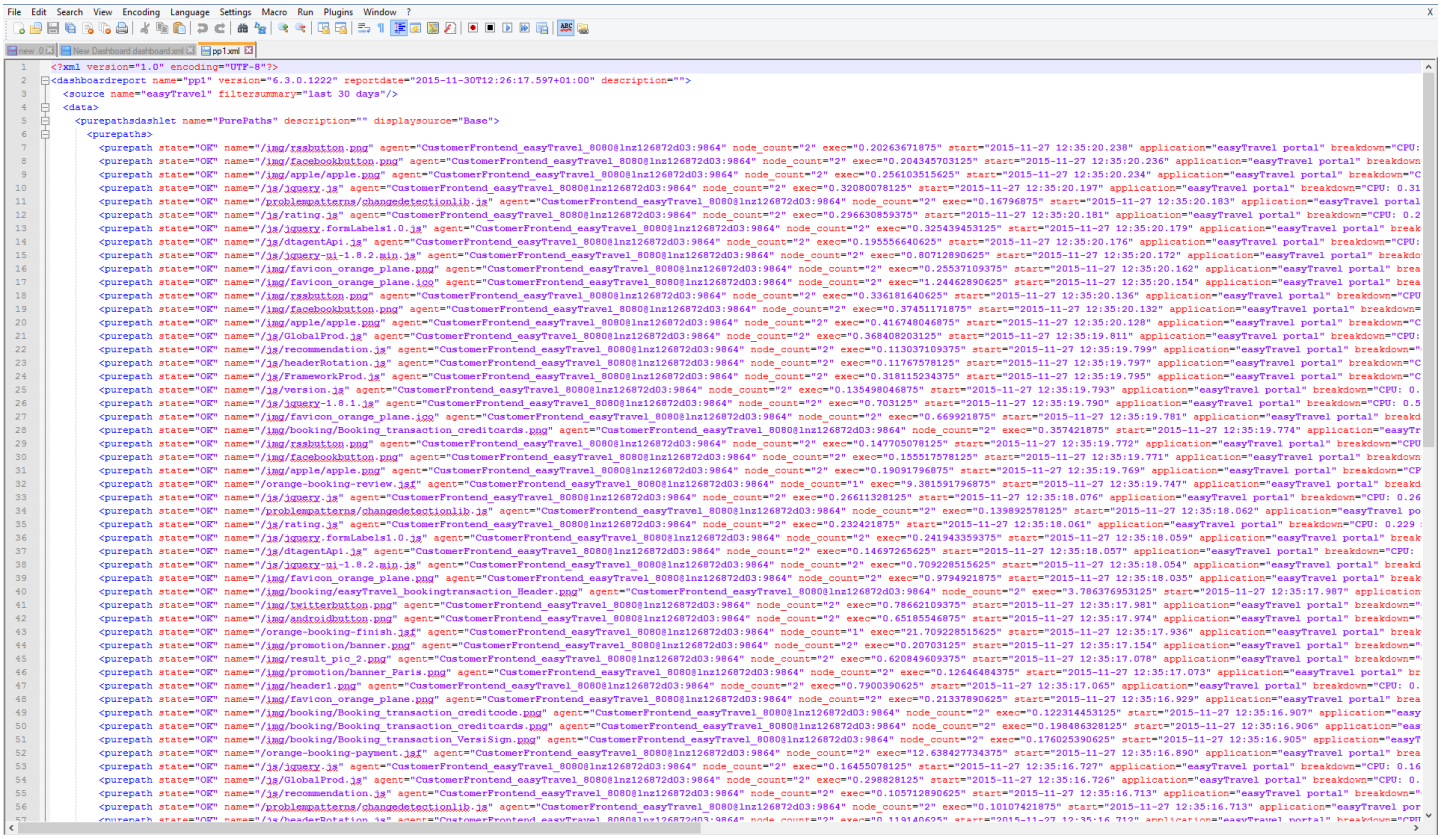
Chart visualizes Memory Utilization (0-100%) for each application tier.





PurePaths

Error State	PurePath	Response Time [ms]	Breakdown
OK	/blog/wp-content/uploads/2013/07/351d6a83ec19be.jpg	7502,252441	CPU: 1.107 ms, Sync: -, Wait: -, Suspension: -
OK	/blog/	5350,614746	CPU: 1,160.668 ms, Sync: -, Wait: -, Suspension:
OK	/CalculateRecommendations	4965,769043	CPU: 944.583 ms, Sync: -, Wait: -, Suspension: 2
OK	PaymentReport.CreateReport(Object stateInfo)	4684,233887	CPU: 110.535 ms, Sync: -, Wait: -, Suspension: -
OK	/blog/	4441,422852	CPU: 1,103.013 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	4288,913574	CPU: 1,080.551 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	4165,274902	CPU: 1,181.818 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	4157,238281	CPU: 1,071.461 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	3700,941162	CPU: 1,177.874 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	3646,253418	CPU: 897.834 ms, Sync: -, Wait: -, Suspension: 2
OK	/blog/	3423,165527	CPU: 967.802 ms, Sync: -, Wait: -, Suspension: 1
OK	/CalculateRecommendations	3124,727295	CPU: 754.898 ms, Sync: -, Wait: -, Suspension: 1
OK	/blog/	3053,309814	CPU: 1,210.624 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	2996,038574	CPU: 1,156.727 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	2711,577881	CPU: 1,139.255 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	2646,839844	CPU: 843.222 ms, Sync: -, Wait: -, Suspension: 1
OK	/blog/	2543,415527	CPU: 1,082.391 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	2434,748779	CPU: 1,093.583 ms, Sync: -, Wait: -, Suspension:
OK	/CalculateRecommendations	2379,384521	CPU: 885.879 ms, Sync: -, Wait: -, Suspension: 3
OK	/blog/	2303,550537	CPU: 1,195.891 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	2277,184326	CPU: 1,190.279 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	2147,519531	CPU: 1,053.771 ms, Sync: -, Wait: -, Suspension:
OK	/blog/	2119,231934	CPU: 1,081.295 ms, Sync: -, Wait: -, Suspension:
OK	/services/BookingService/storeBooking	1996,018066	CPU: 900.076 ms, Sync: -, Wait: -, Suspension: 8
OK	/blog/	1983,505127	CPU: 1,067.021 ms, Sync: -, Wait: -, Suspension:
OK	/services/BookingService/storeBooking	1933,248291	CPU: 823.815 ms, Sync: -, Wait: -, Suspension: -
OK	/CalculateRecommendations	1928,641602	CPU: 844.946 ms, Sync: -, Wait: -, Suspension: 3
OK	/CalculateRecommendations	1795,09082	CPU: 858.388 ms, Sync: -, Wait: -, Suspension: 1
OK	/blog/	1784,401001	CPU: 802.052 ms, Sync: -, Wait: -, Suspension: 5
OK	/orange-booking-finish.jsf	1777,691772	CPU: 690.018 ms, Sync: -, Wait: -, Suspension: -
OK	/blog/	1751,222778	CPU: 963.131 ms, Sync: -, Wait: -, Suspension: 1
OK	/orange-booking-finish.jsf	1731,99939	CPU: 646.871 ms, Sync: -, Wait: -, Suspension: -
OK	/services/BookingService/storeBooking	1704,584961	CPU: 347.368 ms, Sync: -, Wait: -, Suspension: 4
OK	/blog/	1700,878784	CPU: 946.779 ms, Sync: -, Wait: -, Suspension: 2
OK	/orange-booking-payment.jsf	1689,722534	CPU: 648.58 ms, Sync: -, Wait: -, Suspension: 18
OK	/orange-booking-finish.jsf	1661,321167	CPU: 64.58 ms, Sync: -, Wait: -, Suspension: 547
OK	/orange-booking-finish.jsf	1650,022461	CPU: 328.336 ms, Sync: -, Wait: -, Suspension: -
OK	/blog/	1635,448486	CPU: 788.111 ms, Sync: -, Wait: -, Suspension: 7
OK	/orange-booking-finish.jsf	1504,770508	CPU: 434.308 ms, Sync: -, Wait: -, Suspension: -
OK	/orange-booking-payment.jsf	1432,945679	CPU: 328.845 ms, Sync: -, Wait: -, Suspension: -
OK	/orange-booking-payment.jsf	1413,556396	CPU: 390.529 ms, Sync: -, Wait: -, Suspension: -
OK	/blog/	1394,646362	CPU: 836.86 ms, Sync: -, Wait: -, Suspension: 5.4
OK	/orange.jsf	1385,747681	CPU: 135.141 ms, Sync: -, Wait: -, Suspension: 4
OK	/blog/	1364,374268	CPU: 824.441 ms, Sync: -, Wait: -, Suspension: 5



Report content

Reports contain the dashlet's main section and show the data that would be shown when a dashlet is loaded. This means that selections are normally not considered in reports, only a few dashlets do consider selections in client reports or contain more than just the main part.

Client reporting

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Default report

Customize reports for dashboards

Related pages

Reporting

Scheduled reporting

Dashboards and reporting (REST)

XML reporting (REST)

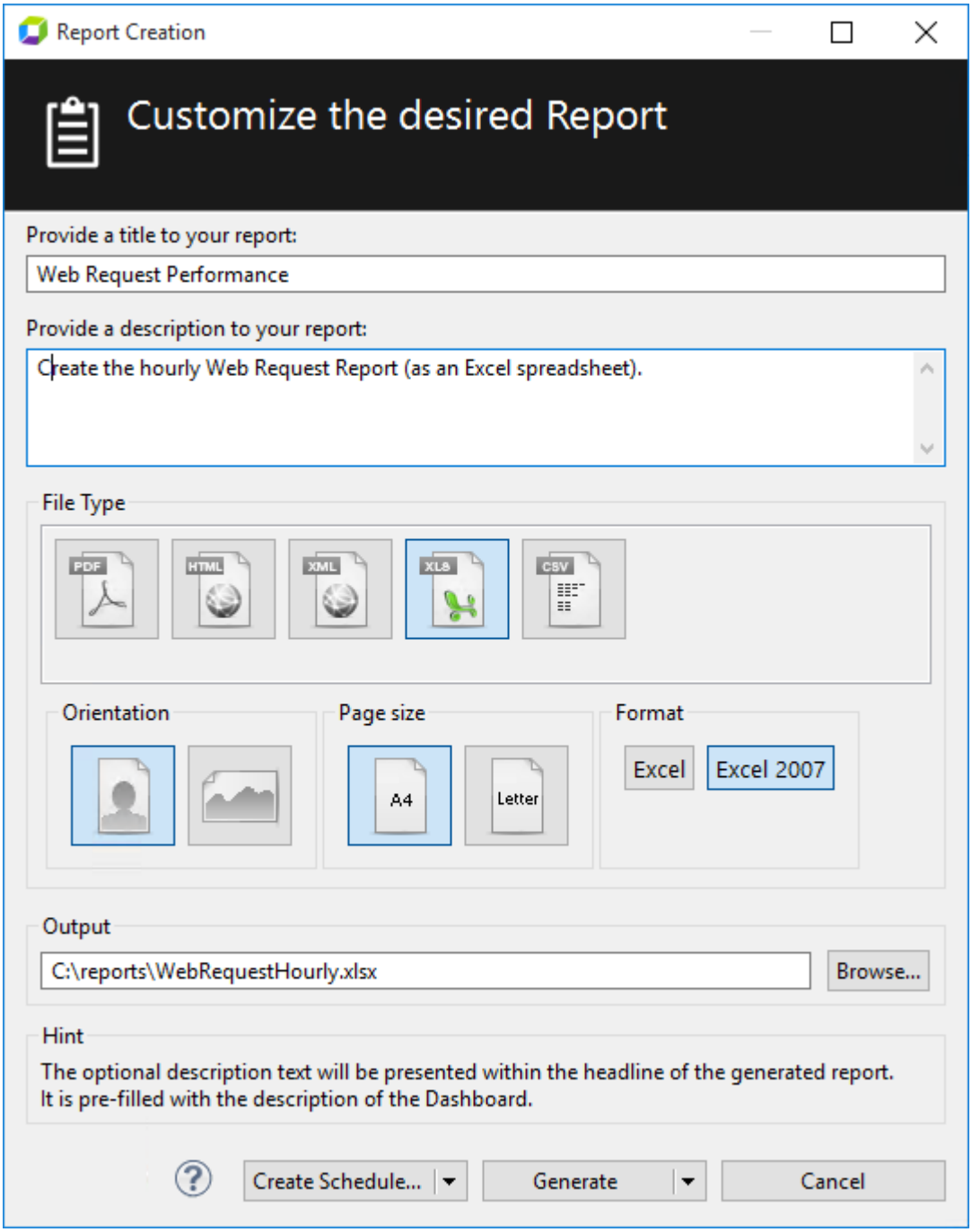
Reporting limits

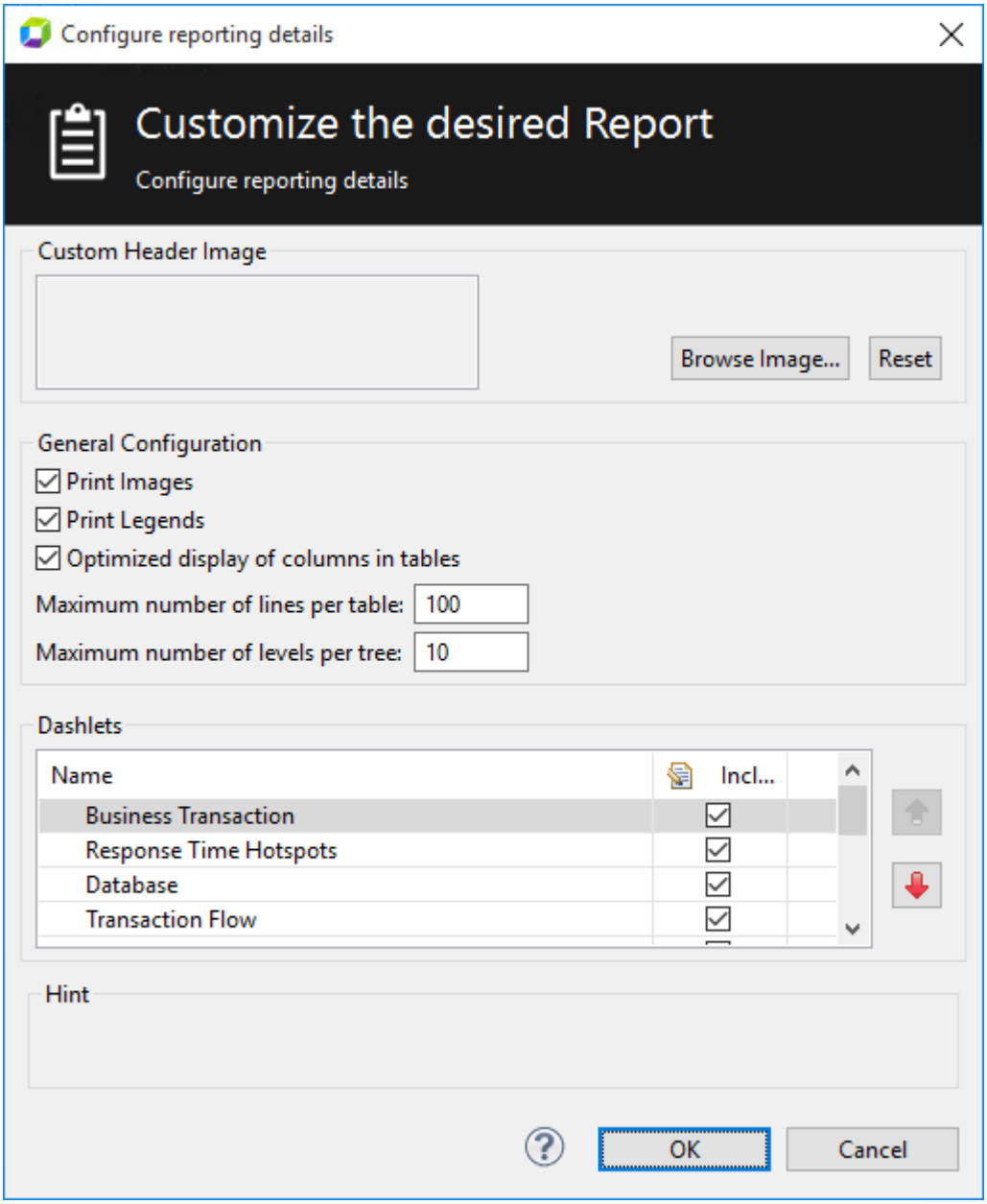
Default report

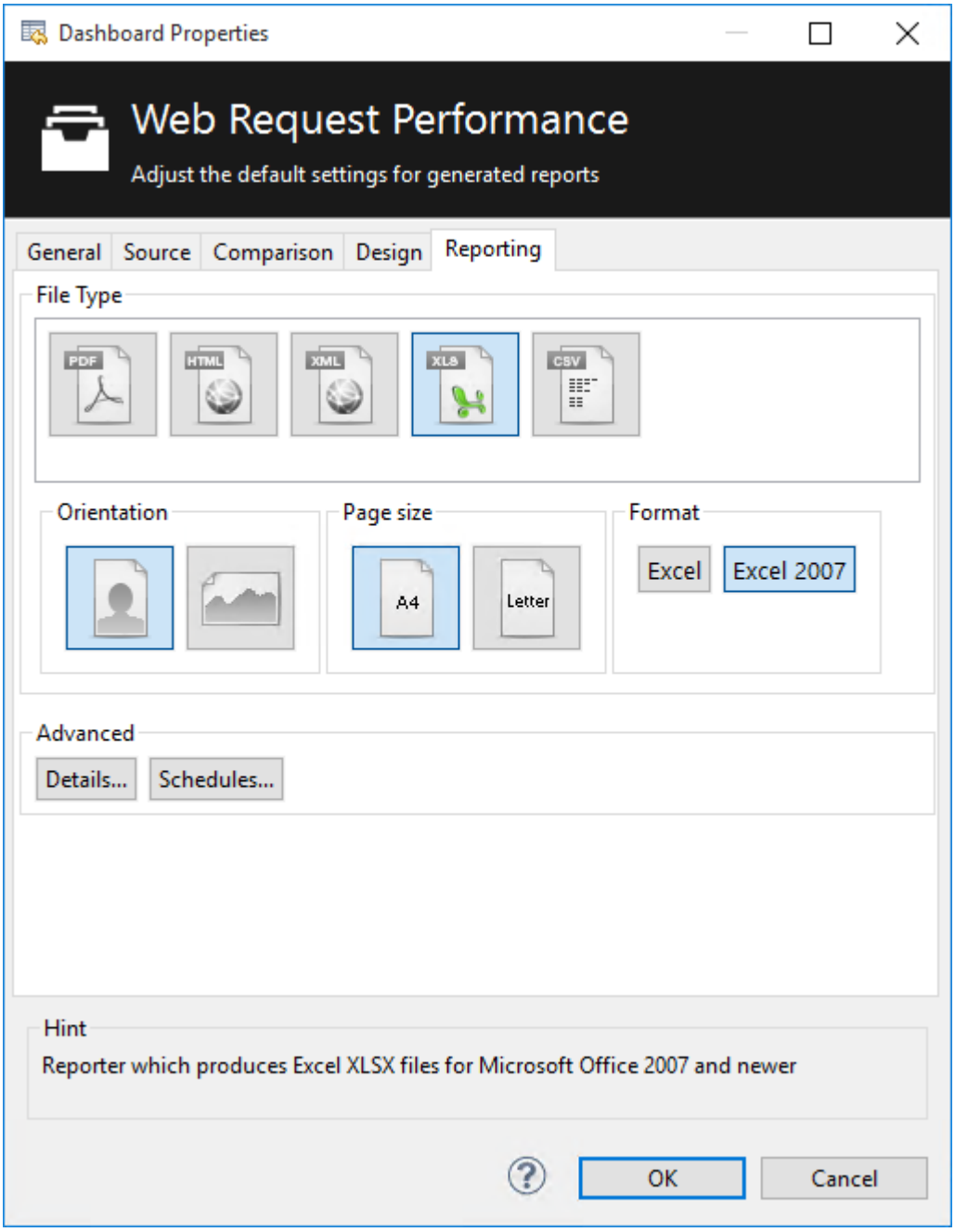
Create a report by doing either of the following:

- To report a dashboard, select **Report** from the **Dashboard** menu or right-click the dashboard icon in the top right corner and choose **Report**.
- To report a single dashlet right click the dashlet's tab and select **Report**.

The **Generate Report** dialog box appears where you can choose additional settings.


Report Creation
Customize the desired Report
Provide a title to your report:
Web Request Performance
Provide a description to your report:
Create the hourly Web Request Report (as an Excel spreadsheet).
File Type
PDF
HTML
XML
XLS
CSV
Orientation
Page size
A4
Letter
Format
Excel
Excel 2007
Output
C:\reports\WebRequestHourly.xlsx
Browse...
Hint
The optional description text will be presented within the headline of the generated report.
It is pre-filled with the description of the Dashboard.
Create Schedule...
Generate
Cancel


Configure reporting details
Customize the desired Report
Configure reporting details
Custom Header Image
Browse Image...
Reset
General Configuration
Print Images
Print Legends
Optimized display of columns in tables
Maximum number of lines per table: 100
Maximum number of levels per tree: 10
Dashlets
Name
Incl...
Business Transaction
Response Time Hotspots
Database
Transaction Flow
Hint
OK
Cancel



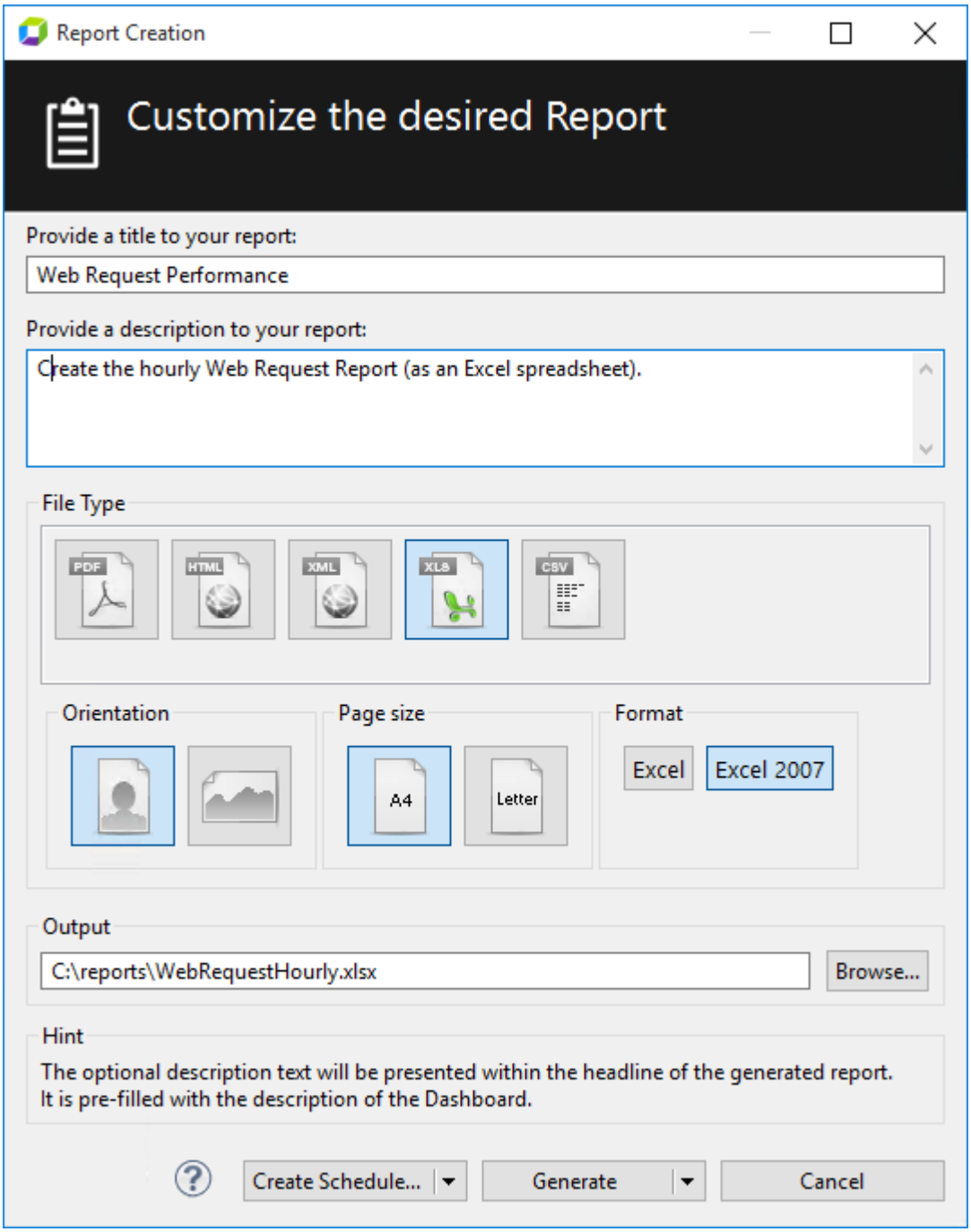
1. The report title and description are automatically created from the dashboard. You can modify these.
2. Choose a report format.
3. Change orientation and page size.
4. Define the output file.
5. Click **Generate** to create the report. When complete, it will open automatically.

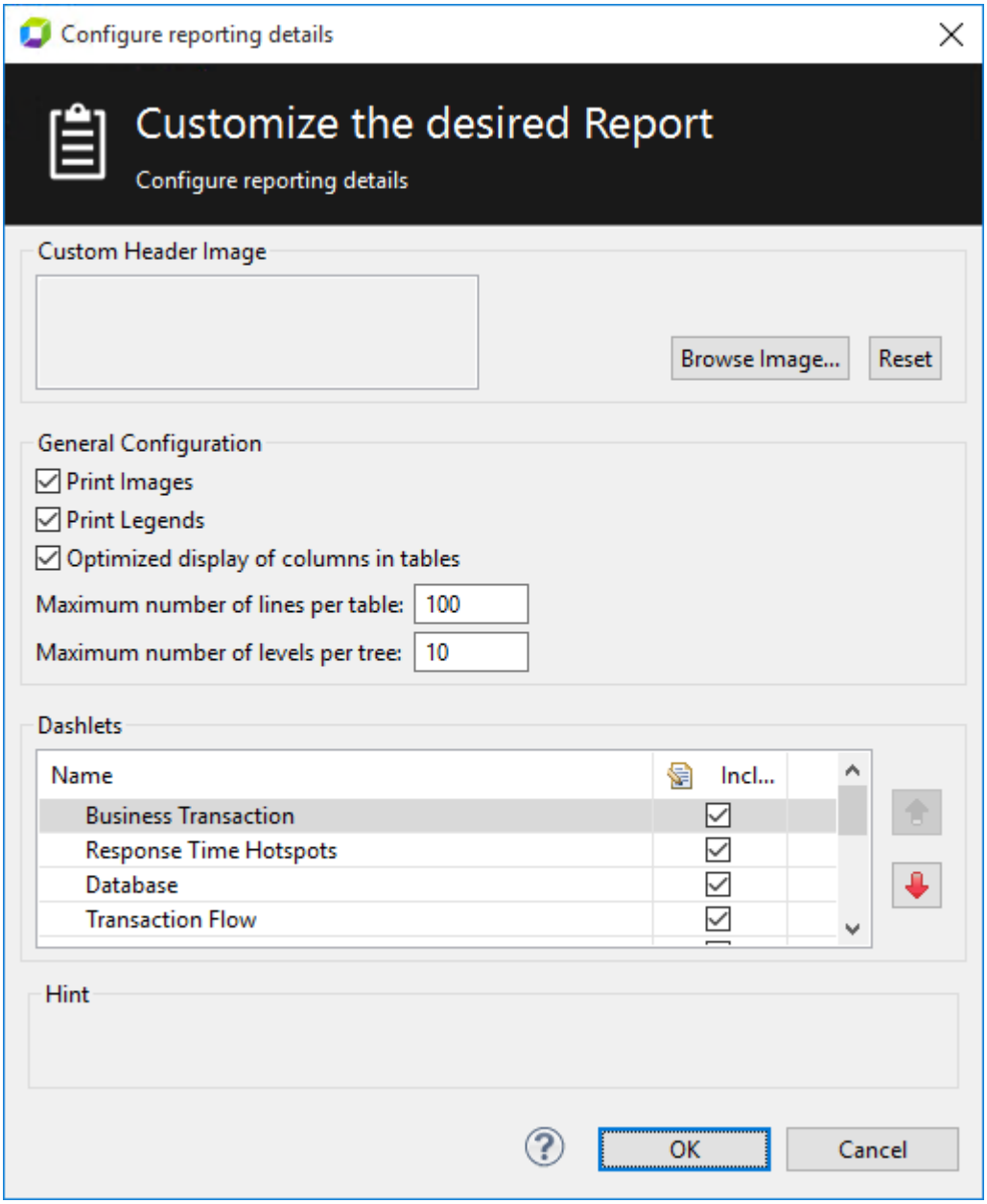
Changes only apply to one report.

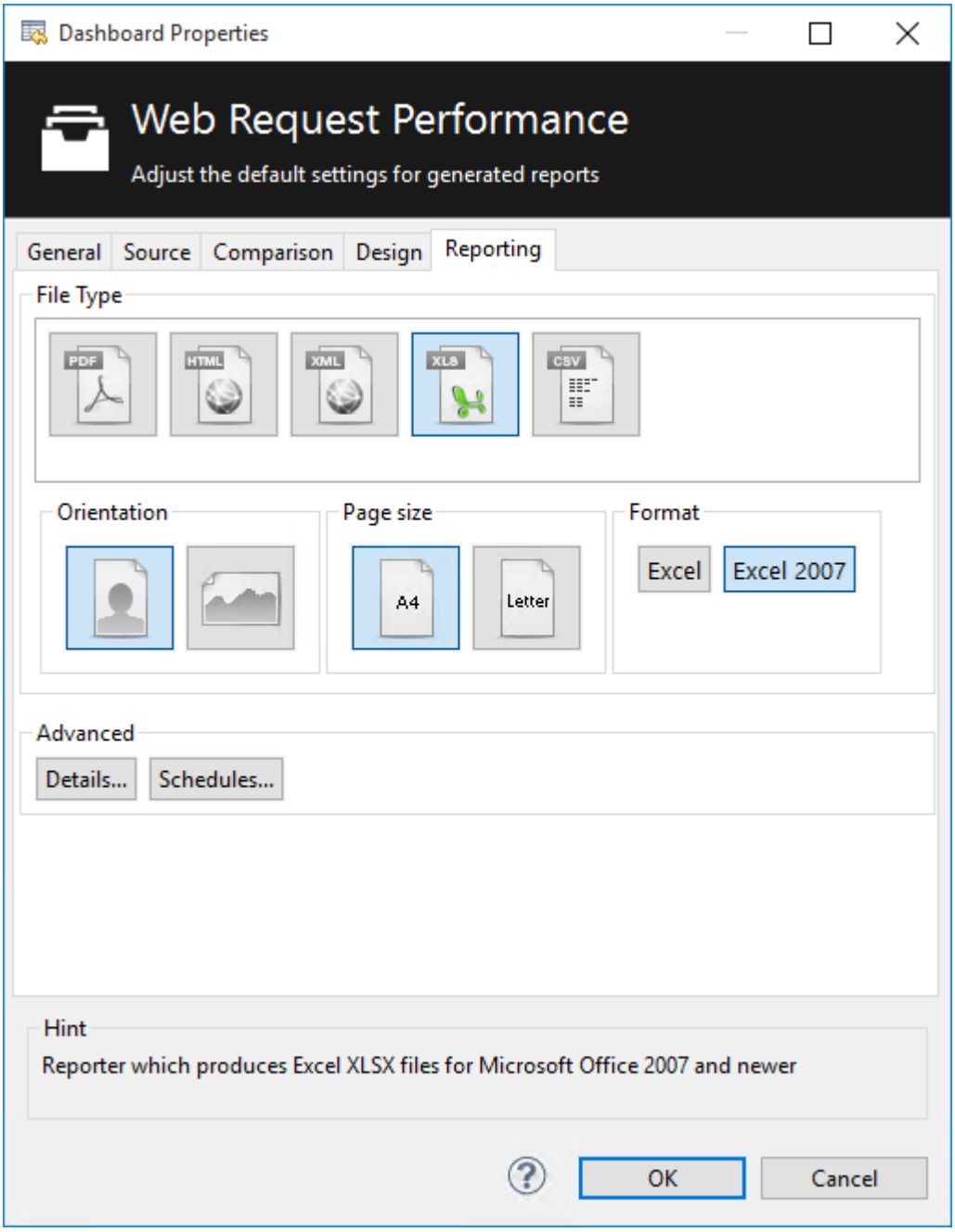
Customize reports for dashboards

In the Dashboard Properties choose the tab **Reporting** to change the following settings:

- Default format.
- Orientation and page size.
- Custom header image.
- Table row and tree level limits.
- Dashlet order and inclusion.
- Scheduling of report creation.




Configure reporting details
Customize the desired Report
Configure reporting details
Custom Header Image
Browse Image...
Reset
General Configuration
Print Images
Print Legends
Optimized display of columns in tables
Maximum number of lines per table: 100
Maximum number of levels per tree: 10
Dashlets
Name
Incl...
Business Transaction
Response Time Hotspots
Database
Transaction Flow
Hint
OK
Cancel



Changes apply to every report generated from this dashboard.

Scheduled reporting

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Scheduled reporting

Related pages

Automated reporting

Reporting

Client reporting

Dashboards and reporting (REST)

XML reporting (REST)

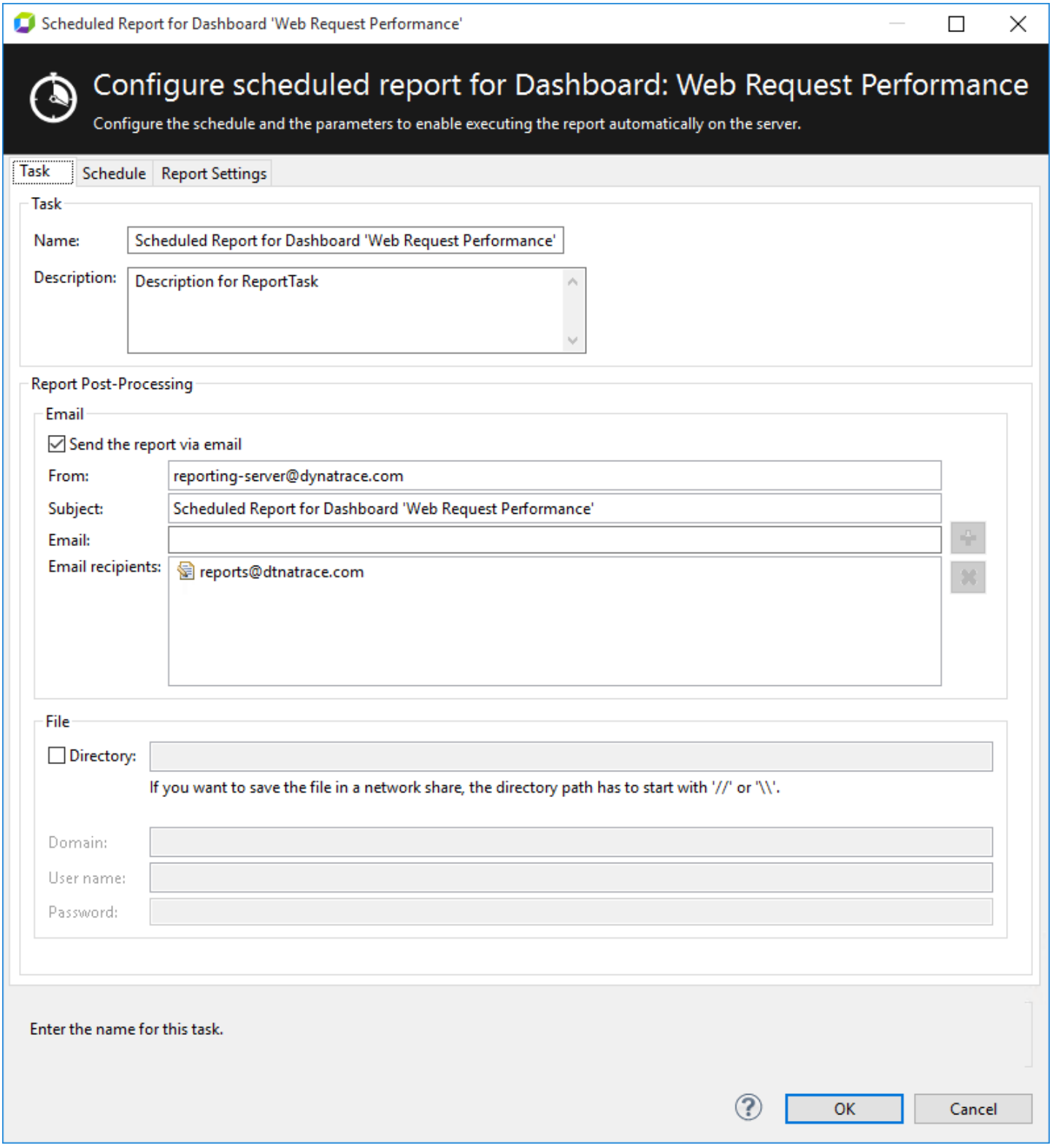
Scheduled reporting

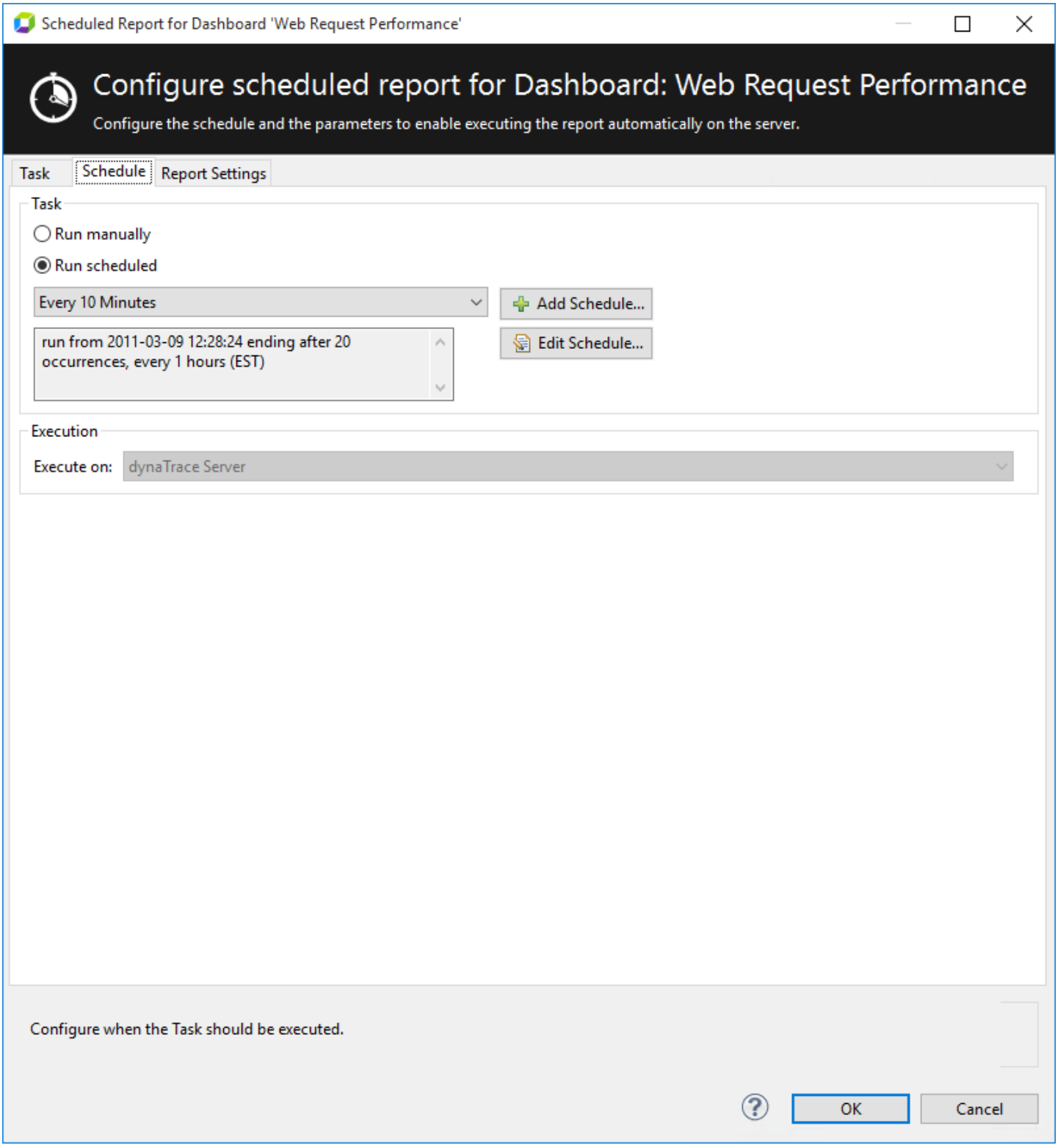
Reports can be created automatically at given intervals and can either be stored in the AppMon Server's filesystem or be sent via e-mail. Scheduled reports are created by the AppMon Server.

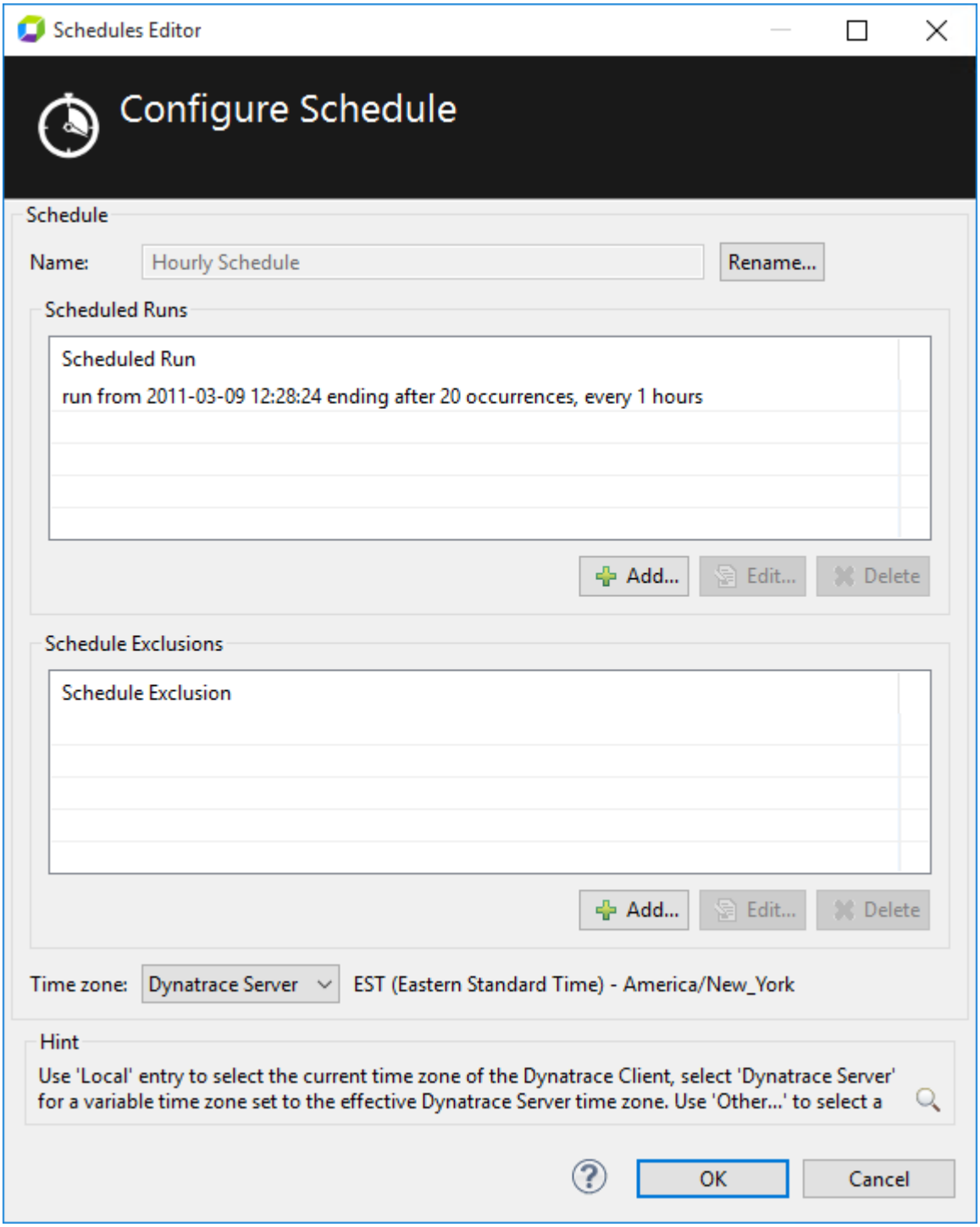
To create a reporting task, execute the following steps, or see the detailed walkthrough of scheduled reporting.

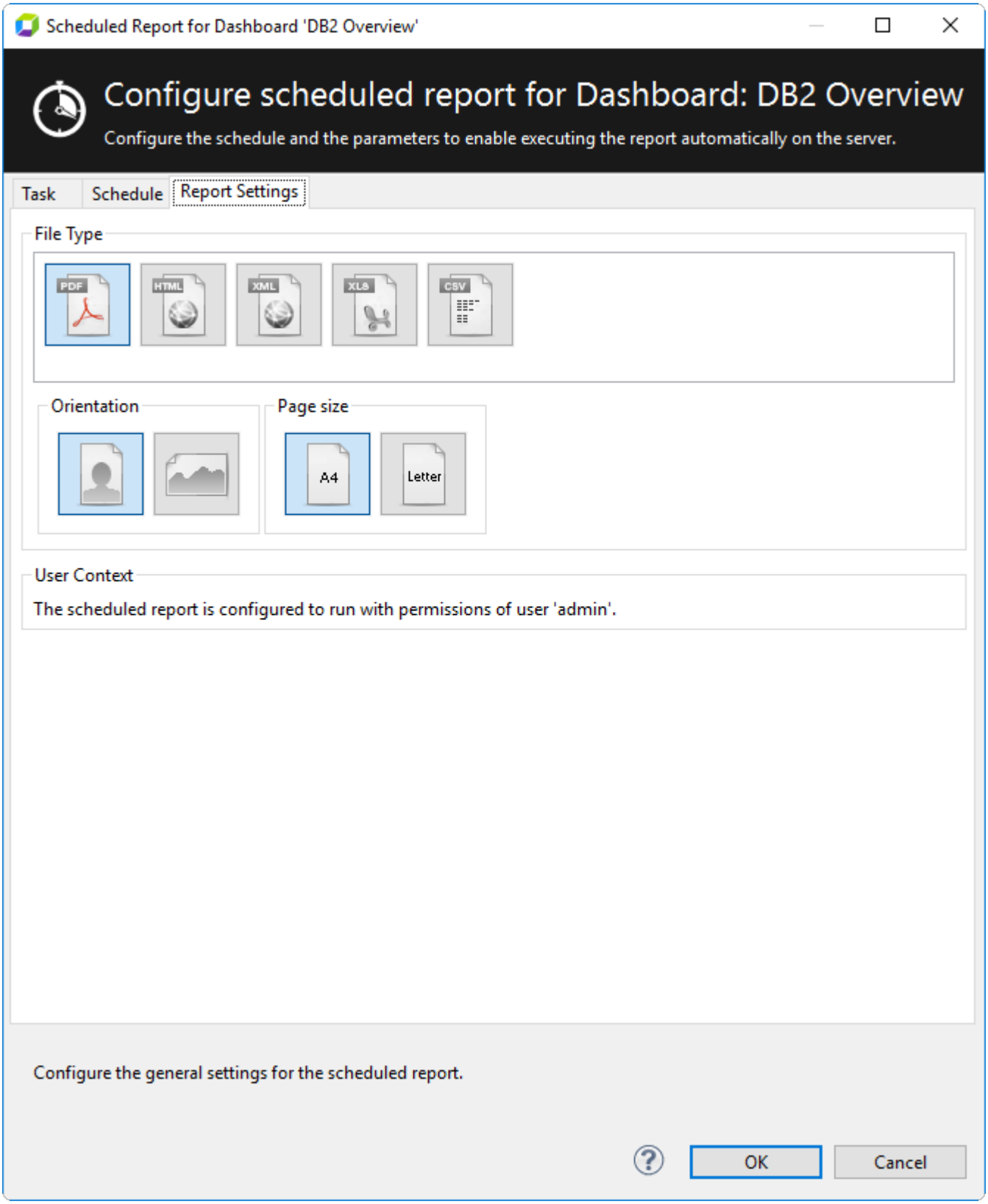
1. Open the dashboard you want to report.
2. Right-click the dashboard title or open the dashboard menu and select **Report**. The Report Creation dialog box appears.
3. Customize the report by changing the title, description, file type, page orientation, page size and filename.
4. Click **Create Schedule**. The Scheduled Report dialog box appears.
5. Customize the post-processing to send the report as e-mail and/or store it in the AppMon Server's file system.

6. Click the **Schedule** tab.
7. Add or edit a schedule with in the Schedules Editor.
8. Click the **Report Settings** tab to define the File Type, Orientation and Page Size.
   You can also define the user which is used to run the report here.




Scheduled Report for Dashboard 'Web Request Performance'
Configure scheduled report for Dashboard: Web Request Performance
Configure the schedule and the parameters to enable executing the report automatically on the server.
Task
Schedule
Report Settings
Task
Run manually
Run scheduled
Every 10 Minutes
Add Schedule...
run from 2011-03-09 12:28:24 ending after 20 occurrences, every 1 hours (EST)
Edit Schedule...
Execution
Execute on:
dynaTrace Server
Configure when the Task should be executed.
OK
Cancel


Schedules Editor
Configure Schedule
Schedule
Name:
Hourly Schedule
Rename...
Scheduled Runs
Scheduled Run
run from 2011-03-09 12:28:24 ending after 20 occurrences, every 1 hours
Add...
Edit...
Delete
Schedule Exclusions
Schedule Exclusion
Add...
Edit...
Delete
Time zone:
Dynatrace Server
EST (Eastern Standard Time) - America/New_York
Hint
Use 'Local' entry to select the current time zone of the Dynatrace Client, select 'Dynatrace Server' for a variable time zone set to the effective Dynatrace Server time zone. Use 'Other...' to select a
OK
Cancel


Scheduled Report for Dashboard 'DB2 Overview'
Configure scheduled report for Dashboard: DB2 Overview
Configure the schedule and the parameters to enable executing the report automatically on the server.
Task
Schedule
Report Settings
File Type
PDF
HTML
XML
XLS
CSV
Orientation
Page size
A4
Letter
User Context
The scheduled report is configured to run with permissions of user 'admin'.
Configure the general settings for the scheduled report.
OK
Cancel

Automated reporting

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Report automation options

Sample use cases

Walkthrough

Related pages

Reporting Plugin

Server REST interfaces

XML reporting (REST)

Overview

Use the reporting functionality in the AppMon Client to produce a variety of reports from dashboards that you can easily share. Generate manual reports on the fly or schedule reports to run at regular intervals, like weekly performance reports.

To automate monitoring tasks, AppMon schedules reports or uses external actions to trigger them via REST interfaces. See Sample Use Cases below for more information.

The AppMon automated reporting functionality also lets you schedule sending email reports using SMTP protocol. Embedded images and styling information in the HTML email so the content appears in email clients without security warnings.

Report automation options

AppMon includes additions to the core reporting functionality so you can automate the process in various ways.

Scheduled Report Execution: Schedule a report to run at various intervals and send it using email. You can store the report in the file system and process it further. For example, you can publish it to a company web portal.

Retrieve Reports via REST Web Services Interface: You can use your own scheduler and receive reports via the REST interfaces. For more information, see Server REST Interfaces.

Trigger Reports via REST Web Services Interface: To trigger and send a report at certain times such as when a test run completes, and send it via email, you can create a reporting task. See the walkthrough on this page to learn how.

You can use the REST interfaces to trigger reports. See Server REST Interfaces for more information.

Retrieve Detail Data for Further Processing: You can process the information in XML reports more than other report types. The report data includes charts, and you can also retrieve full details about PurePaths. See Server REST Interfaces and Server- and Client-Side XML Reporting for more information.

Sample use cases

This section lists common use cases and how you can implement them.

Use Case	Implementation
The operator wants to send a weekly status report via email to the management team.	• Define the dashboard to send. • Configure report details in the dashboard properties: Select report format, exclude unnecessary dashlets, and set correct reporting order for dashlets. • Create a scheduled report in the reporting dialog box and define the email details, including the recipient and schedule.
The build engineer wants to email a report after the nightly build and test run is complete.	• Set up a dashboard and a report schedule similar to the previous use case. • Set the schedule to **manual execution.** • In your build scripts at the point when building and testing is finished, trigger the reporting task via the REST interfaces, see Server REST Interfaces for details.

A performance engineer wants to retrieve details from a chart for more detailed analysis.	• Set up a dashboard with charts that display the information. • Retrieve the XML report that contains the necessary details for each chart via Server-side XML Reporting. • For a different time frame, adjust the dashboard filter or specify additional filter parameters for the REST calls.
A developer wants to retrieve details from PurePaths for statistical analysis of called code locations.	• Set up a dashboard with a PurePath dashlet that includes the correct source and filter. • Retrieve the XML report, which contains the necessary details for the selected or all PurePaths, via the interfaces for Server-side XML Reporting and provide the parameter `purePathDetails`. • For a different time frame, adjust the dashboard filter or specify additional filter parameters for the REST calls.

Walkthrough

With automated reporting, you can present accumulated data to stakeholders or deliver the daily KPI status update to your inbox. Reporting is mandatory, and you don't want to spend too much time to gather the data, format the report, and process it. If you run a report twice, you should automate it with AppMon's reporting features.

This walkthrough demonstrates the AppMon reporting features and how they make reporting easier. Learn how to schedule your reports.

Create your own dashboards or use those that come with AppMon. If you don't have a System Profile, and have no data from Agents, use the built-in dashboards for self-monitoring.

Follow these steps to automate reporting:

1. Open the reporting dialog box.
2. Customize the necessary information.
3. Create a reporting schedule.
4. Automate report generation based on these schedules.

Create a report schedule

Create a schedule for your report if you want it to be sent out via email daily, weekly or both to various recipients.

To do this, click **Report** from the **Dashboard** menu, then click **Create Schedule** and create a new schedule. Create a schedule so that the PDF report is sent to your mailbox daily at 8am, and weekly, to your manager. You must create two schedules.

For the daily schedule, use the following definitions:

Start on: Today

Time to Send: 08:00AM

End: Forever

Recurrence: Every Day in Every Month

Define the exclusion for Saturday and Sunday separately:

Schedule Exclusion: Exclude Always

On: Saturday and Sunday

In: Every Month

The second schedule executes every month on the first of the month:

Start on: First day of the month

End: Forever

Recurrence: Every Month

Monthly Schedule: Send the Report on the First Day of the Month After you configure your dashboards with relevant data, automate reporting to save time. Also this automation capability can serve as a data export engine to your favorite tools. Get the data delivered at certain times or pull it into your automation tools for further processing via scheduled tasks.

Note

If the AppMon Server is not available on the network with the internal host name, but with some other host name or only when the link-address contains the domain name, you can still generated working links in the reports. Use the **External hostname** field to specify the fully qualified host name or IP address that AppMon uses to create links in reports, emails, and the server web interface. To access this, click **Settings > Dynatrace Servers > Services > Management**.

Reporting limits

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Limitations to client side reporting

Size limits

Non-latin scripts

Trees in reports

Related pages

Reporting

Client reporting

Scheduled reporting

Dashboards and reporting (REST)

XML reporting (REST)

Limitations to client side reporting

Reporting in the AppMon Client has certain limitations:

- Only data already shown in the client are reported. No refresh is triggered before reporting.
- Dashlets that were not yet activated in the client are skipped in the reports. To make sure all of the dashboard's dashlets show data in the report, manually refresh them before creating the report. This does not apply to server side reporting via REST or scheduled reporting.

Size limits

AppMon limits the report size so the AppMon Server or AppMon Client is not affected by very large reports. To generate a large report, use the following configuration options:

Soft limit: 100 rows per dashlet

Dashlets and dashboards are set to report 100 rows of data per table. Adjust this in the dashboard and dashlet settings. If you set it higher than the current hard limit of 5000 rows per table, the hard limit is still imposed.

Hard limit: 5000 rows per report

A report cannot have more than 5000 rows for all dashlets in the report. If you require more, you can set the following system property:

`com.dynatrace.diagnostics.reporting.maxTableRows`	int (default = 5000)	Limits the number of overall table rows in one report. After the limit is reached, Dashlets are excluded from the report. They are listed at the end of the report.

You may have to adjust this setting in multiple places. The following describes report types and their settings:

Client side reports: client installed locally

The locally installed client uses the file `dtclient.ini,` which is located in `C:/Users/<username>/.dynaTrace/dynaTrace <version>/dtclient.ini` (on Linux/Mac OS X this is `~/.dynaTrace/...`).

```
-vmargs
-Dcom.dynatrace.diagnostics.reporting.maxTableRows=10000`
-Xmx512M
...
```

Add the setting to this file directly after -`vmargs`, not at the end.

Maximum of 50 dashlets per report

To prevent extremely large reports and performance issues on the AppMon Client or the AppMon Server every Report is limited to 50 dashlets. Of course you can create multiple reports with 50 dashlets each.

Client side reports: Webstart Client launched via AppMon Server website

Because there is no `dtclient.ini` for Webstart Clients, the system property has to be set differently, for example by setting the JAVA_TOOL_OPTIONS environment variable:

```
JAVA_TOOL_OPTIONS=-Dcom.dynatrace.diagnostics.reporting.maxTableRows=10000
```

Server Side Reports: Scheduled Reports and REST Reports

When you create a report via scheduled tasks or the REST interface, the AppMon Server creates the report. You must configure it to have higher limits, if necessary. To do this adjust the file `dtserver.ini` which is usually located at `<dynaTrace Installation>\dtserver.ini`, for example `C:\Program Files\dynaTrace\dynaTrace 5.5.0\dtserver.ini`

```
-restartonfailure

-vmargs

-Dcom.dynatrace.diagnostics.reporting.maxTableRows=10000

-Xmx1920M
```

Add the setting to this file directly after `-vmargs`, not at the end.

Troubleshooting

- After the setting is applied and the server or client is restarted, you should see the following line in the logfile, after the first report is triggered:

```
2015-14-03 09:26:53 INFO [ServerReportManager] Using non-default limit of table rows per
```

If the line is not there, verify that the config-files are adjusted correctly and that the server and client are restarted to activate the new settings.

- You must also adjust the soft limit in the dashboard or dashlet because the default limit of 100 rows still applies, even if you raise the hard limit correctly.
- `dtclient.ini` files must only be located under `C:\Users`. If they are located in places like `C:\Program Files (x86)\dynaTrace\dynaTrace <version>`, remove them.

Server side reports: queue

> **Note**
>
> Changing the default settings for the reporting queue will increase the resources needed by your AppMon Server

Only a certain number of reports gets executed simultaneously (by default 3). This setting can be adapted with the system property

```
-Dcom.dynatrace.diagnostics.reporting.maxConcurrentReports=5
```

As soon as the maximum number is reached, new reports get queued and wait until at least one previously started report is done. The timeout for reports waiting to be processed (by default 300 seconds) can be changed by the system property

```
-Dcom.dynatrace.diagnostics.reporting.reportWaitTimeout=400
```

To change the size of the queue of reports (by default 8) use the system property

```
-Dcom.dynatrace.diagnostics.reporting.maxWaitingReports=10
```

Non-latin scripts

To enable non-latin scripts for PDF reports it may be necessary to set the system property `com.dynatrace.diagnostics.reporting.pdf.english` to `false`. This could lead to errors with hyphenation but enables right-to-left writing.

Depending on whether the AppMon Client or the AppMon Server create the report, the system property has to be set in the `dtclient.ini` or `dtserver.ini`.

Trees in reports

Dashlets that contain trees report those as flat tables - at least for non-XML reports. Since the relation between the table rows can have a completely different meaning if the tree depth information is lost, a column “level” is always added to such reports.

Application Monitoring walkthroughs

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Analyzing a slow transaction

Identify problems in your application

Memory analysis

Analyzing out of memory errors

Infrastructure monitoring walkthroughs

How to retrieve performance metrics with Windows Performance Monitor

Setting up application availability monitoring

Setting up infrastructure monitoring

Using host and hostgroups for large system monitoring

Using host and process health monitoring

Walkthroughs

The application monitoring walkthroughs provide detailed instructions and examples for using AppMon to monitor and troubleshoot common application performance problems.

Identify problems in your application

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Goal of this walkthrough

Problem isolation overview

Detailed steps

Conclusion

Related pages

How to use Business Transactions

Using host and process health monitoring

Goal of this walkthrough

This walkthrough explains how to use AppMon to find out which applications and transactions have problems, and if those problems are related to the application or the infrastructure.

Problem isolation overview

The first task in any diagnostic process is to isolate the problem. You can do this in high volume and production environments. Your goal is to quickly narrow down the root cause area.

For example, if a transaction has a slow response time, follow these steps:

1. Isolate the tier that produces the problem. The quickest way to do this is to use the Transaction Flow and look at the biggest contributors.
2. Dig deeper and look at the biggest contributors inside the tiers.

3. Dig again and look at the transaction load trend and resource utilization.

This approach quickly traces the response time problem to a particular component or API inside a particular Agent group or specific transaction type Agent. You may also discover that the network latency is at fault.

Use the easyTravel demo application to learn about various AppMon features and functions, and to simulate load, and problems that typically occur.

Detailed steps

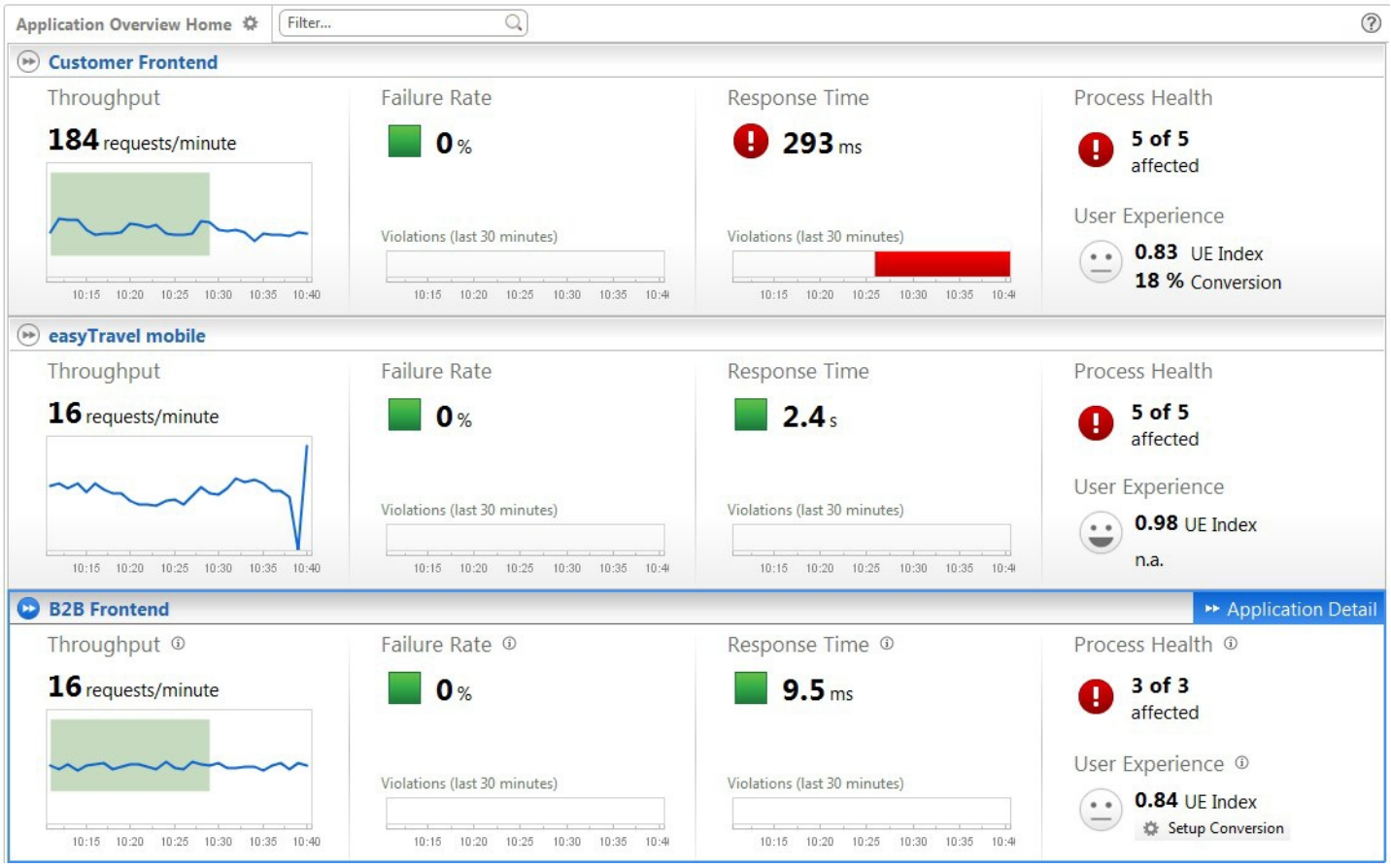
1 Identify the Application Problem

AppMon automatically detects applications based on the host or application ID. It also lets you manually configure an application, if automatic configuration doesn't work for you. For more information, see System Profile - Applications.

Open the **Application Overview Dashboard** to see an overview of all the selected System Profile applications. AppMon looks at throughput, response time, failure rate, infrastructure health and user experience.
Applications with problems, like a slow response time that violates the automatic calculated baseline, appear at the top of the list.

The following shows that the **Customer Frontend** application violates the **Response Time** baseline. That means that certain business transactions have a response time that violates the baseline. Notice that the underlying infrastructure has health problems that impact the application also. Five out of five associated application processes are unhealthy.

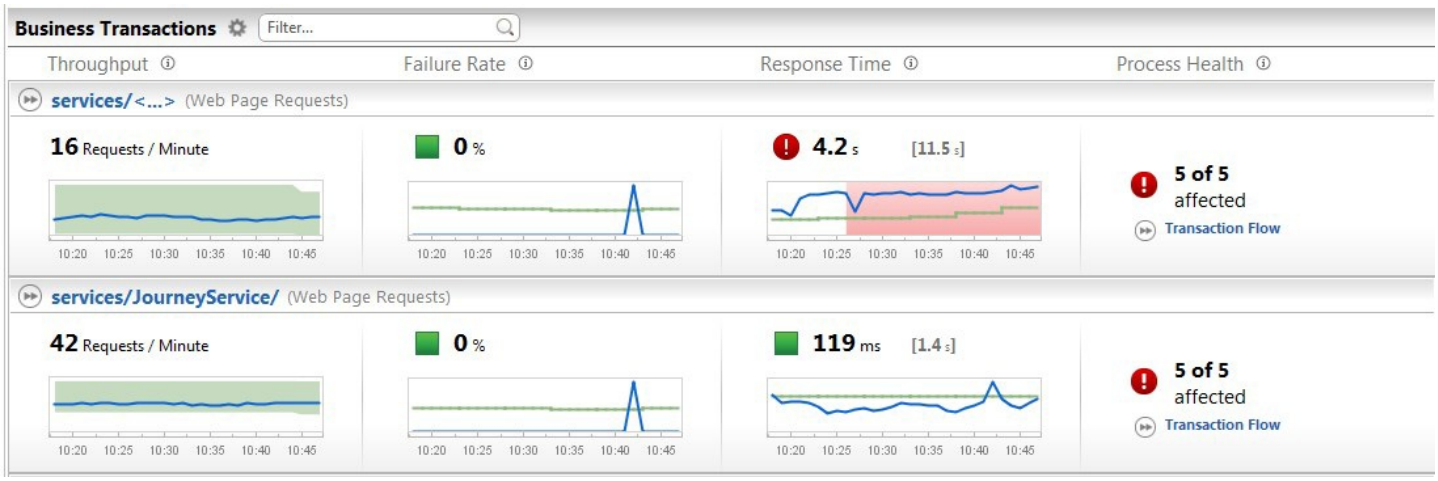
Customer Frontend with Slow Response Time



2 Identify the Application Problem

AppMon learns which URLs you access most often via webpage requests. It uses this information to identify your critical business transactions.
To change the **Business Transaction** view, simply define the business transactions that you add to the **Application Overview**. See
How to Use Business Transactions for more information.

Click on the problematic application. The following window appears and displays all the business transactions with the problematic ones at the top:

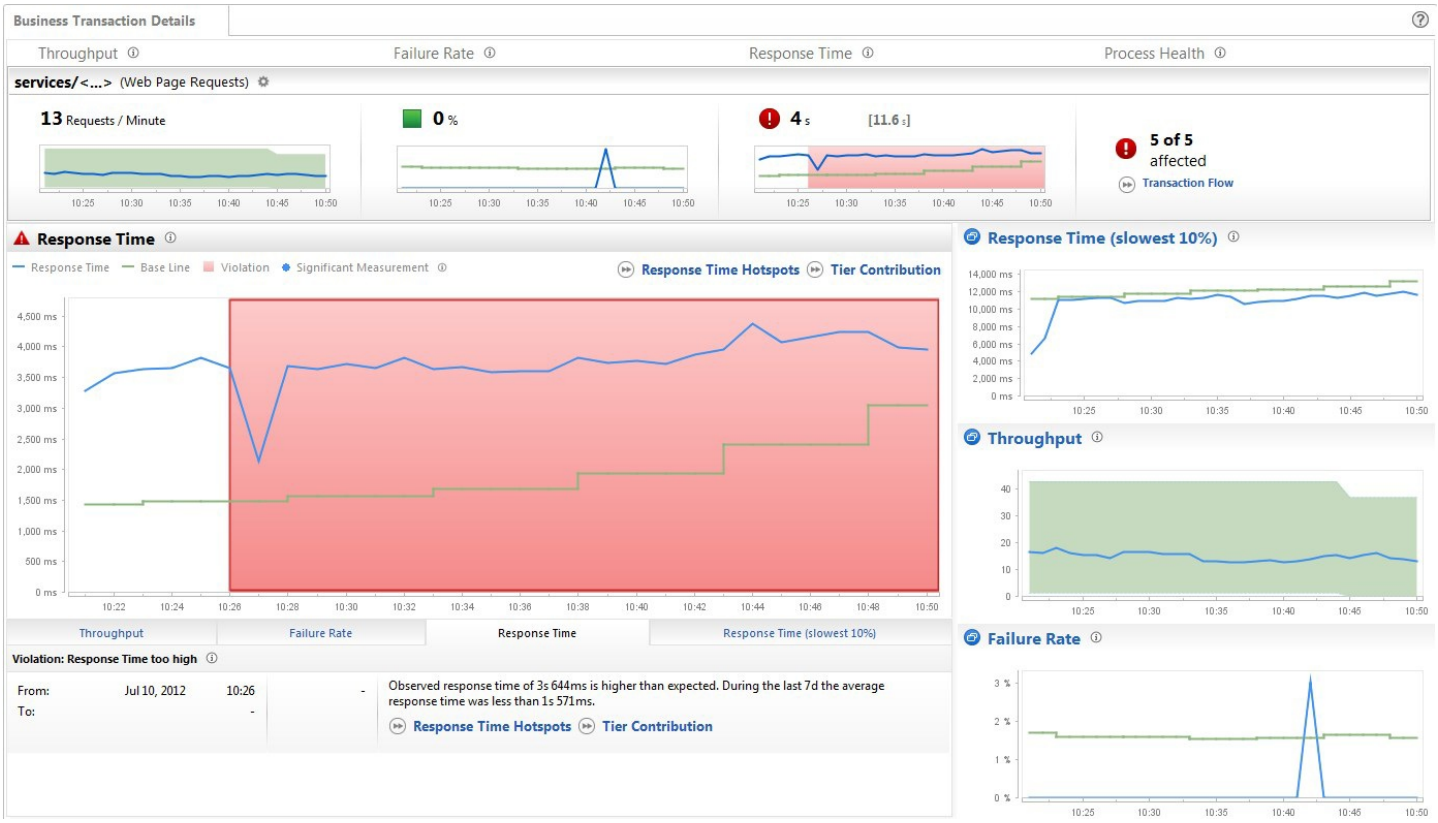


List of All Business Transactions

See Application Details for more information.

3 Analyze the Transaction Problem Details

The **Business Transaction Details** tell you if the problem is related to performance, failure or process health. They also tell you if you currently have more throughput than usual, which can also cause unusual behavior. The following shows that the response time surpasses the automatically calculated baseline. This is why AppMon is alerted.



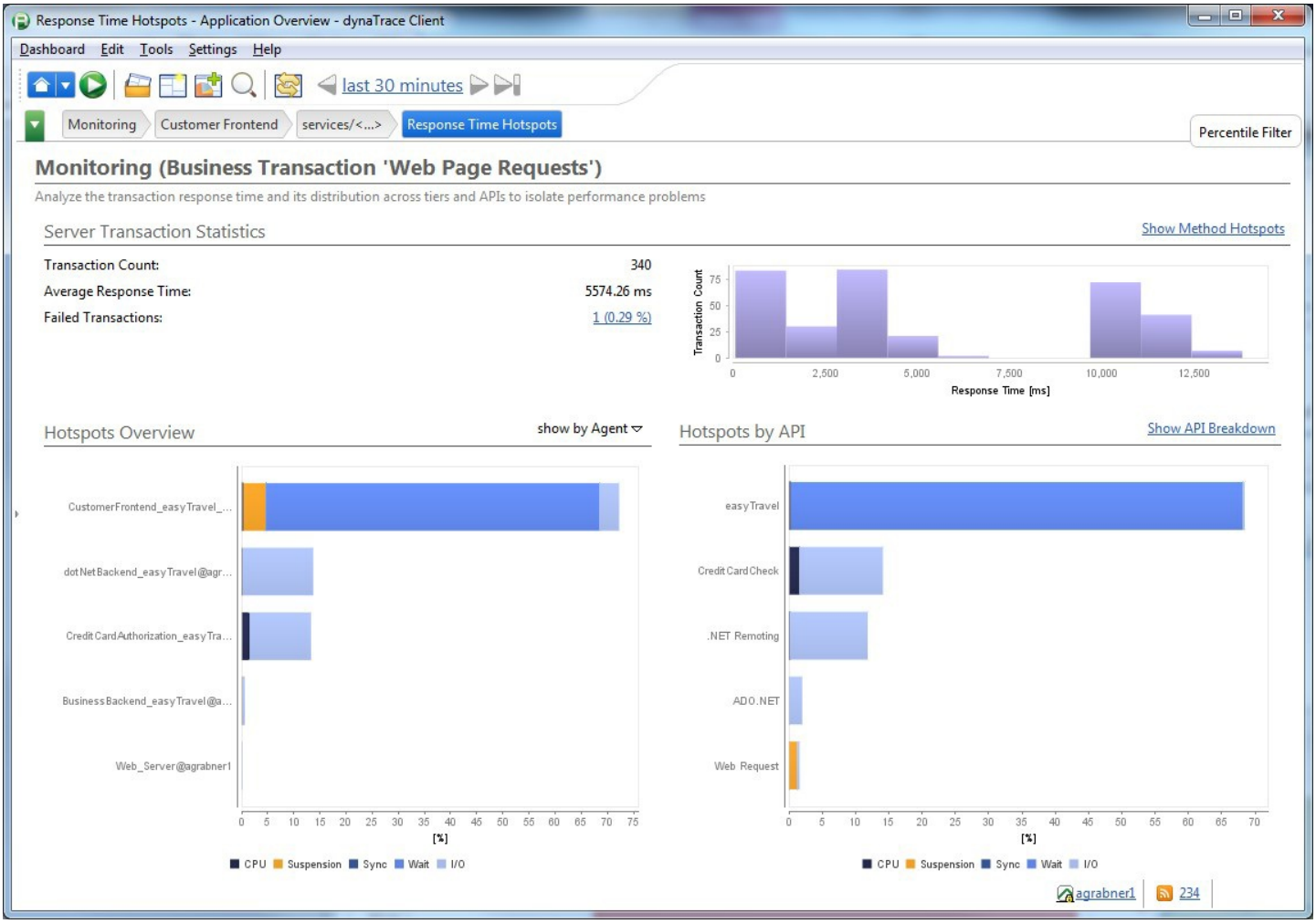
Business Transaction Details with Slow Response Time

AppMon automatically calculates baselines for median response time (50%) and the slowest 10% of response time (90%).
It also calculates a baseline for the failure rate and calculates the expected throughput per transaction, based on historical throughput data.
If any of these values cause a baseline violation, drill into more details such as the **Response Time Hotspots Dashlet**, **Transaction Flow Dashlet** or **Errors Dashlet**.

See Business Transaction Details for more information.

4 Analyze Response Time Hotspots

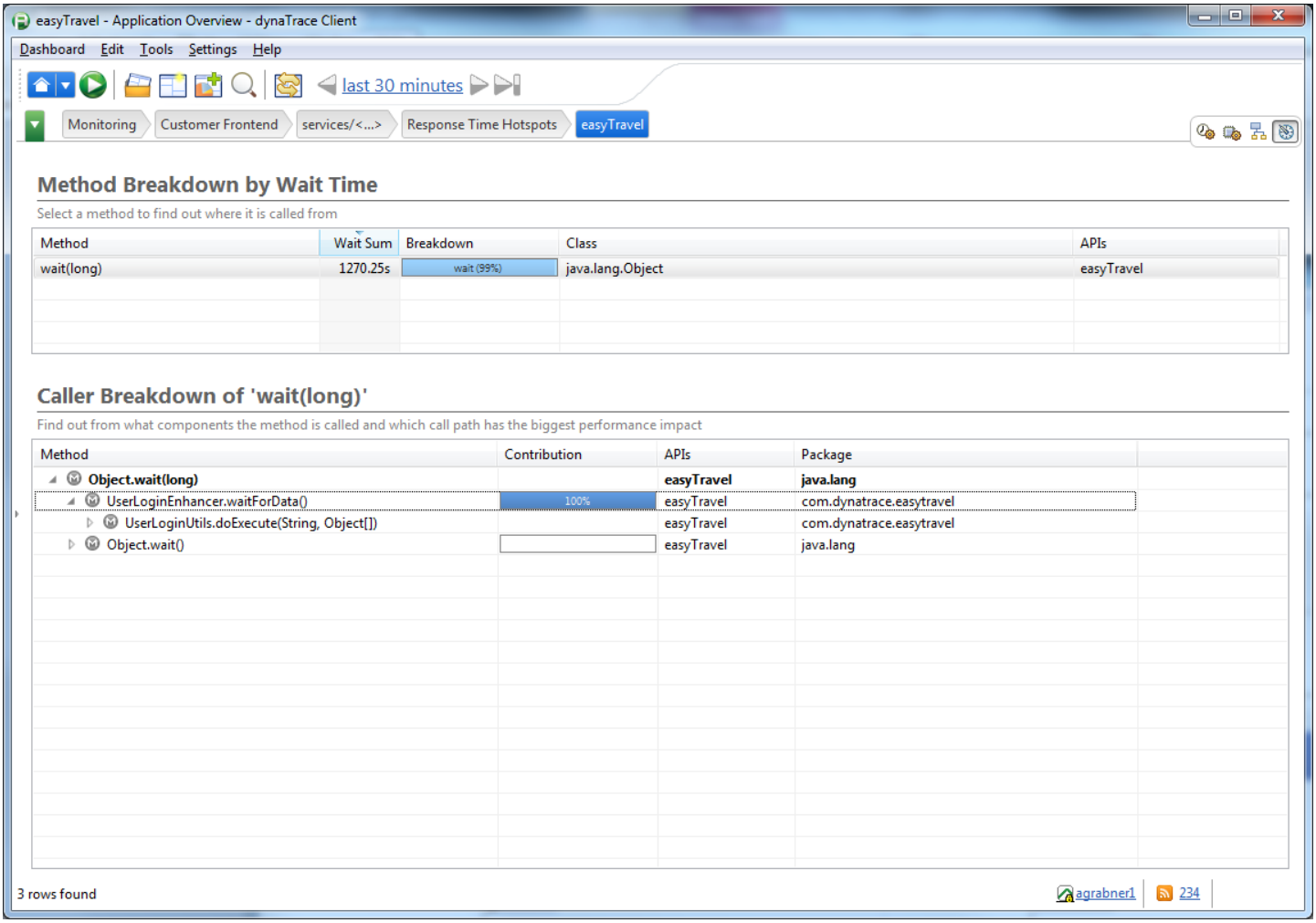
If the problem is related to degrading response time, the first step is to drill to the **Response Time Hotspots**.
The following image shows that one particular API is responsible for almost 70% of the response time that executes on one particular agent:



Response Time Hotspots

Click on the problematic API to see the **Method Hotspots Dashlet**. This displays which methods are slow to execute, and who is calling these methods.

The following image shows that the `UserLoginEnhancer` class is waiting for data with a call to `Object.wait`. This contributes almost 70% to the business transactions' response time.



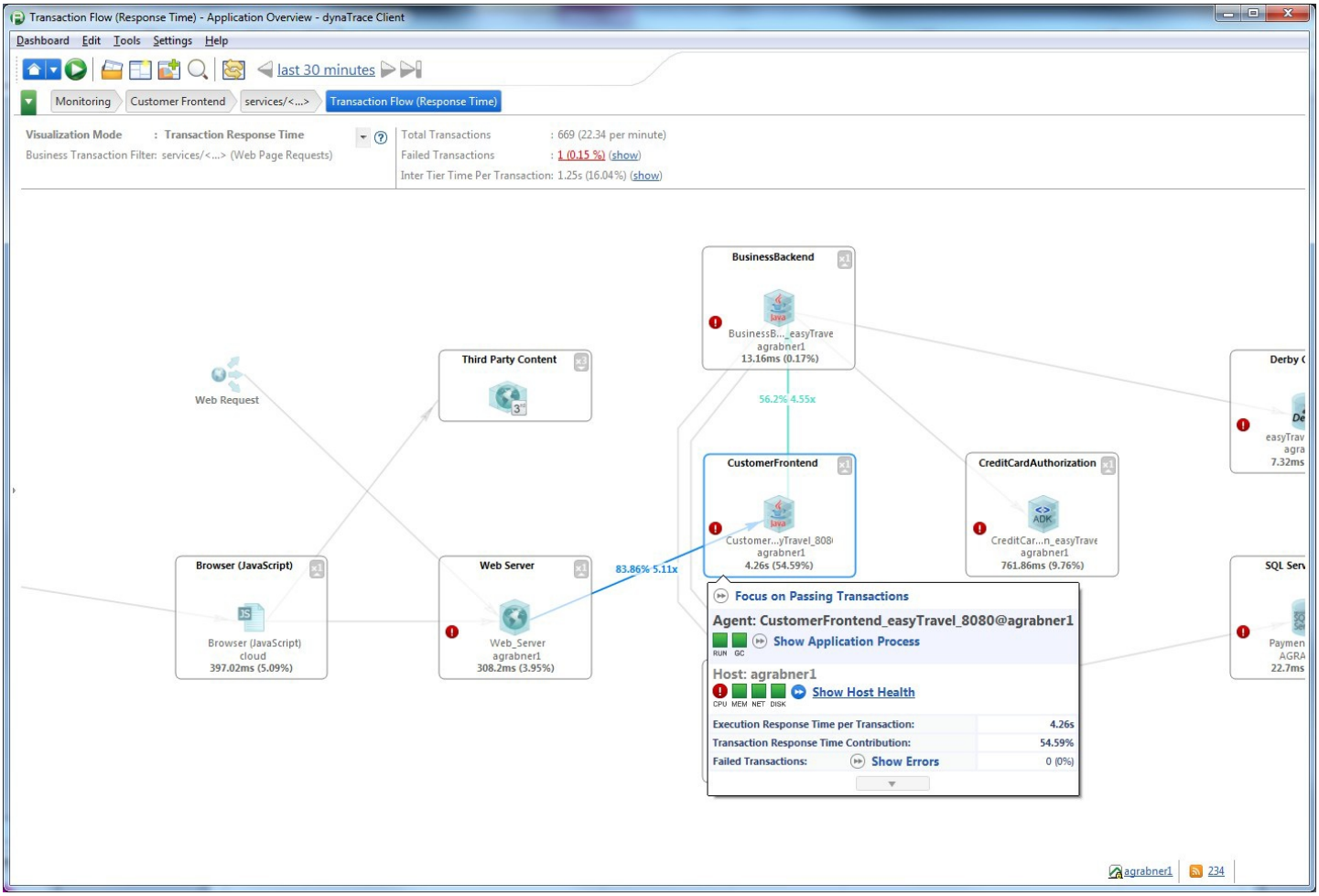
Method Breakdown by Wait Time

5 Analyze Transaction Flow and Host Health

Drill into the **Transaction Flow Dashlet** for another way to analyze application problems. The **Transaction Flow** provides an overview of the application tiers associated with this business transaction. Use the **Transaction Flow** to determine the following:

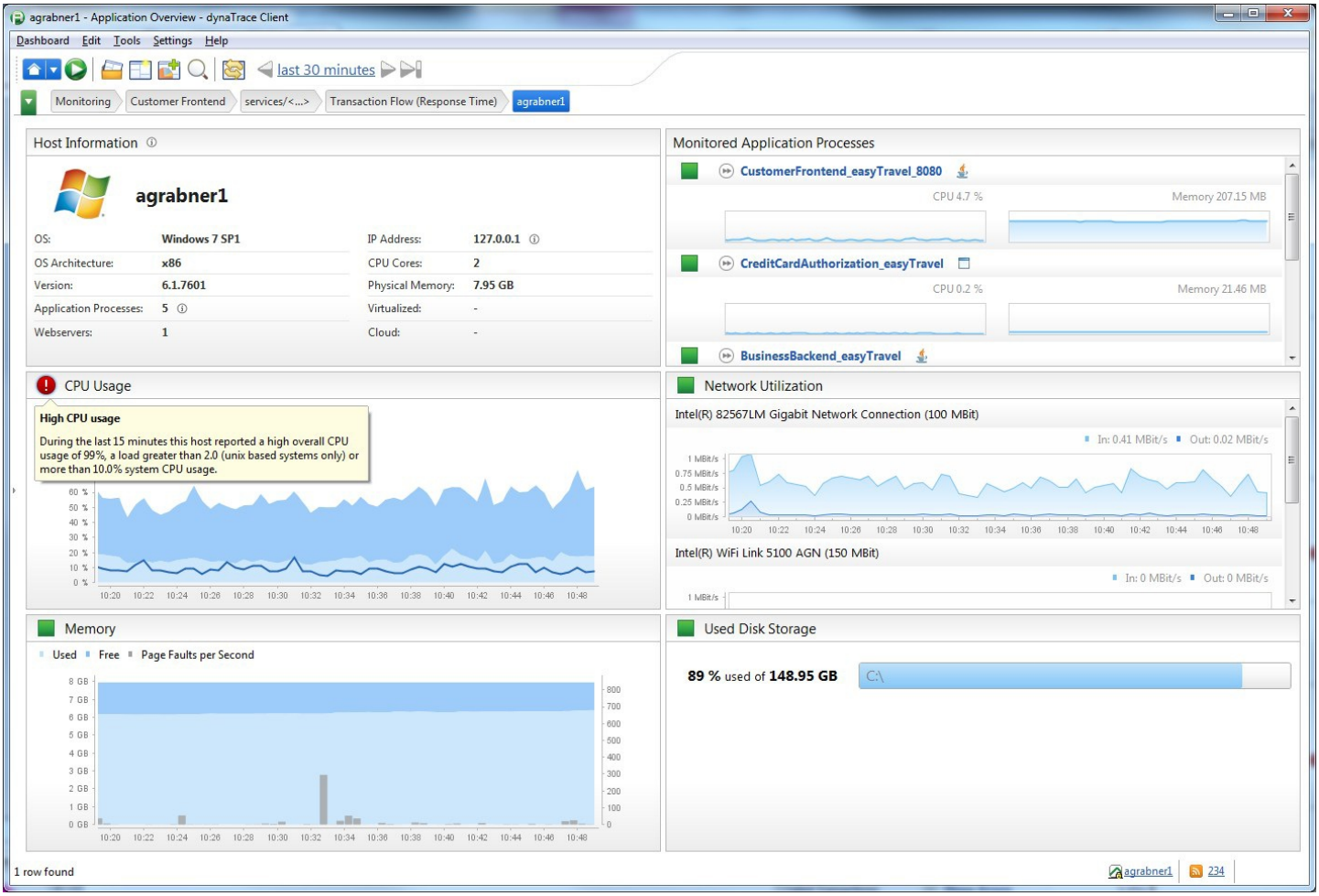
- Do certain tiers have a problem?
- Is there too much time spent between tiers?
- Is there too much traffic on the database?
- Are there problems outside the data center, such as third party content?

In addition to analyzing the **Transaction Flow**, you can get information about the supporting infrastructure. Every node shows host and process health. This gets calculated on different dimensions. The following image shows that the host that runs the **Customer Frontend** application has a CPU Problem.



Transaction Flow

Click **Show Host Health**. The **Host Health Overview** shows that the host has a very high CPU utilization. You can also see which application components run on that host (top right). This lets you determine if the CPU overhead is the result of these components, or another process running on the machine.



The Host Health Overview

See Follow Your Transactions with Transaction Flow for more information on how to analyze the Transaction Flow.
See Host Health Monitoring, Process Health Monitoring for more information on host and process health.

Conclusion

The **Application Overview** makes it very easy to locate problems in individual applications, a specific business transaction or in your infrastructure. The following tasks help to identify problems in your application:

- Configure your System Profile - Applications.
- Learn How to Use Business Transactions.
- Leverage the built-in Host and Process Health Monitoring.
- Use automatic Baselining.

Regression analysis

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is regression analysis?

Goal of this walkthrough

Detailed steps

Conclusion

Related topics

What is regression analysis?

Regression analysis estimates the differences between application executions. If you fix bugs, introduce new features or refactor an application, you affect other application areas. Regression analysis estimates how your application performs if you:

- Have two different application versions or releases.
- Execute the application on two different environments or platforms.
- Change third-party libraries.

Goal of this walkthrough

This walkthrough describes how to do a simple regression analysis based on a problem pattern in easyTravel.

Create a dashboard to pinpoint the differences in two recorded sessions. Use one problem pattern and show the difference.

Scenario

Assume that you have recently refactored your application and modified some APIs. Also, you fixed some bugs and created new features. The main features remain the same. Determine if the changes affect the main features negatively or positively. Use the SlowAuthentication problem pattern in easyTravel to demonstrate the two different application variants. Do not deploy two different versions of easyTravel.

Requirements

Install, configure and run the following components:

- easyTravel.
- AppMon with a default easyTravel System Profile.
- Mozilla Firefox browser with Selenium IDE.

1. Click **easyTravel Configuration UI > Install System Profile**.
2. Click **UEM - Lightweight** to start a scenario.
3. Select **Manual Visits**.
4. Click **Cockpits > Status Overview > Agent Overview** to make sure that all Agents are connected to the AppMon Server.
5. Click **Cockpits > System Profiles > easyTravel > Diagnose Performance > PurePaths** to make sure you get events and PurePaths in your live session.

Detailed steps

The following is a high level overview of the testing process:

1. Create a repeatable automated easyTravel test suite (with Selenium) or use the basic example Selenium test suite for this and the next step.
2. Automate a session recording through the AppMon REST interface.
3. Create a **Regression Analysis** dashboard to show the performance differences.

If you already have two comparable stored sessions, you can skip to Step three.

Use an easyTravel problem pattern to simulate a regression.

Step One: Create Repeatable Automated Tests

For a meaningful regression test, make sure you compare the same scenarios or identical test cases.

For web applications, use Selenium to create an automated test suite that you can replay. Install Selenium as a Mozilla Firefox plugin and click on various functions in your application to record test cases. You can use any tool that is capable of test-driving a web app.

For this scenario, use the easyTravel customer frontend login and the SlowAuthentication problem pattern to simulate two different easyTravel versions:

1. Start the easyTravel customer frontend.
2. Click the **Lock** icon to get a list of predefined users.
3. Click **monica.**. Record this in Selenium or look at the edited result in its IDE:
4. Open **Test Suite > Test Suite Part 1**.
5. Double-click **Login monica Test Case**.

Step Two: Automate Session Recording with AppMon's REST Interface

You must compare two identical sessions. Use AppMon's REST interface to automate the session recording.

The following are the necessary functions:

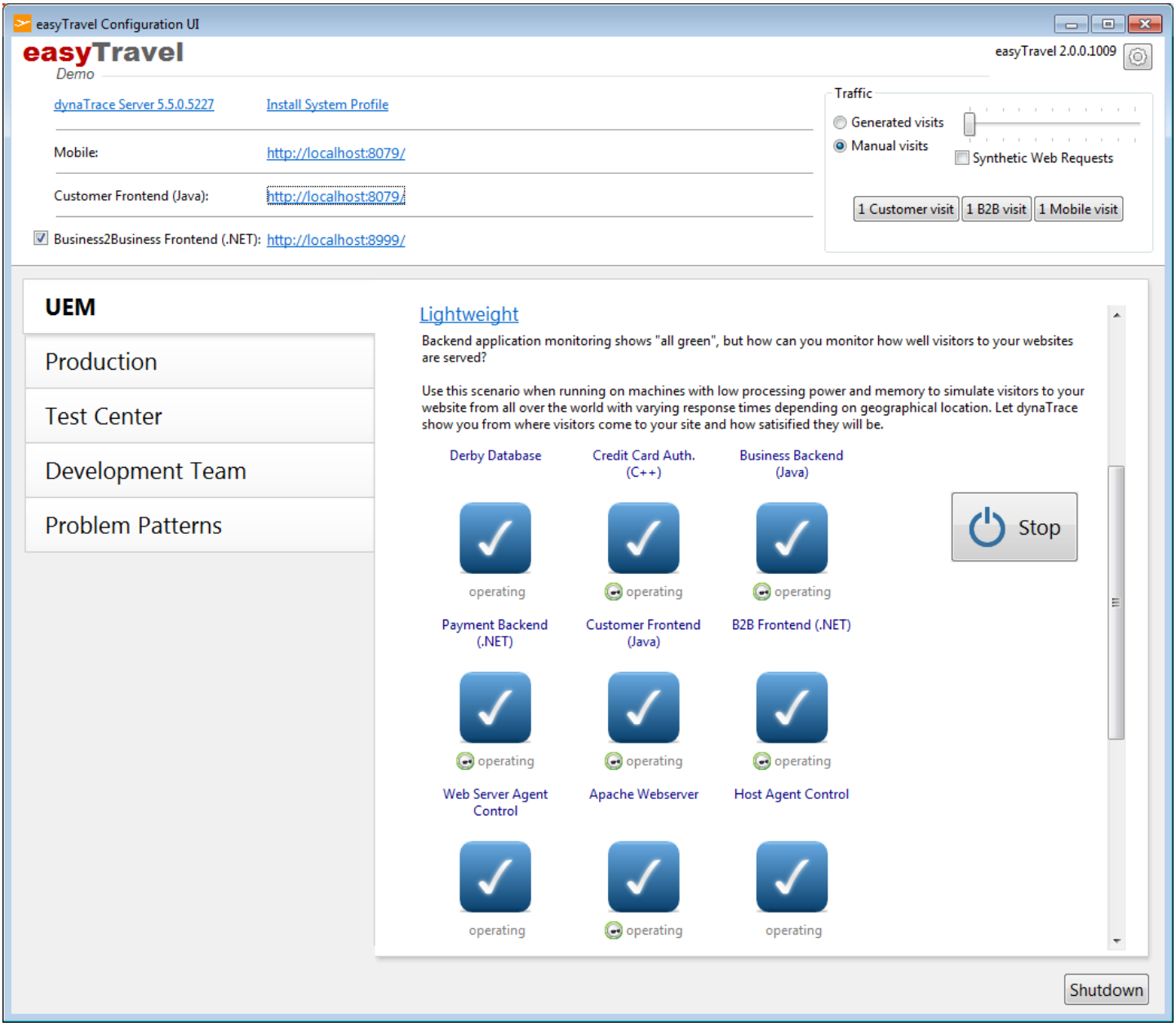
- http://localhost:8020/rest/management/profiles/easyTravel/clear: Clears the live session. Replace localhost with your AppMon Server's name.
- http://localhost:8020/rest/management/profiles/easyTravel/startrecording: Provides a post. http://localhost:8020/rest/html/management/profiles/easyTravel: Provide the session name and click **Start** to record.
- http://localhost:8020/rest/management/profiles/easyTravel/stoprecording: To stop.

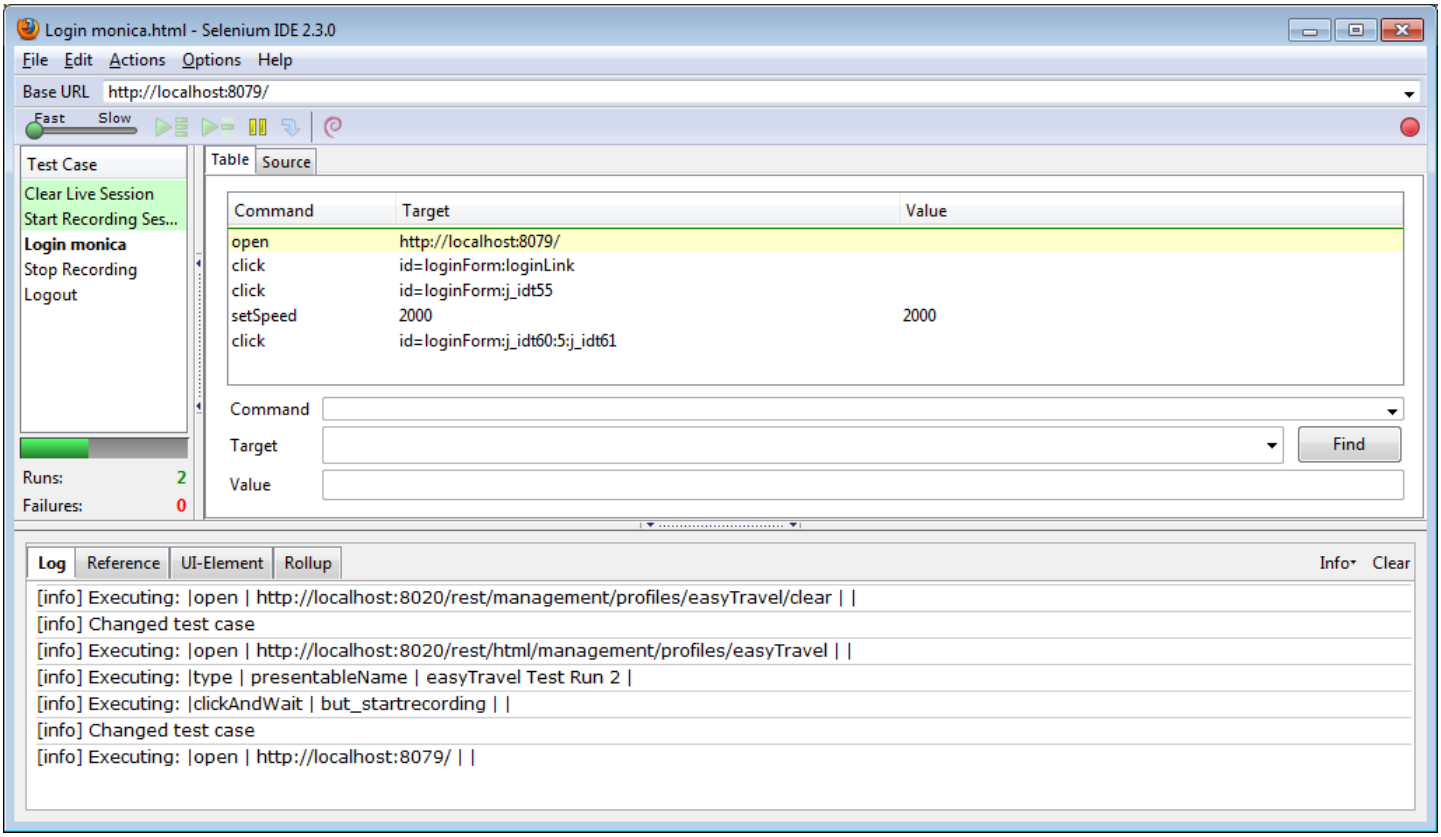
Perform these on the easyTravel System Profile. You can embed the necessary commands to operate AppMon in the Selenium test suite. Follow these steps:

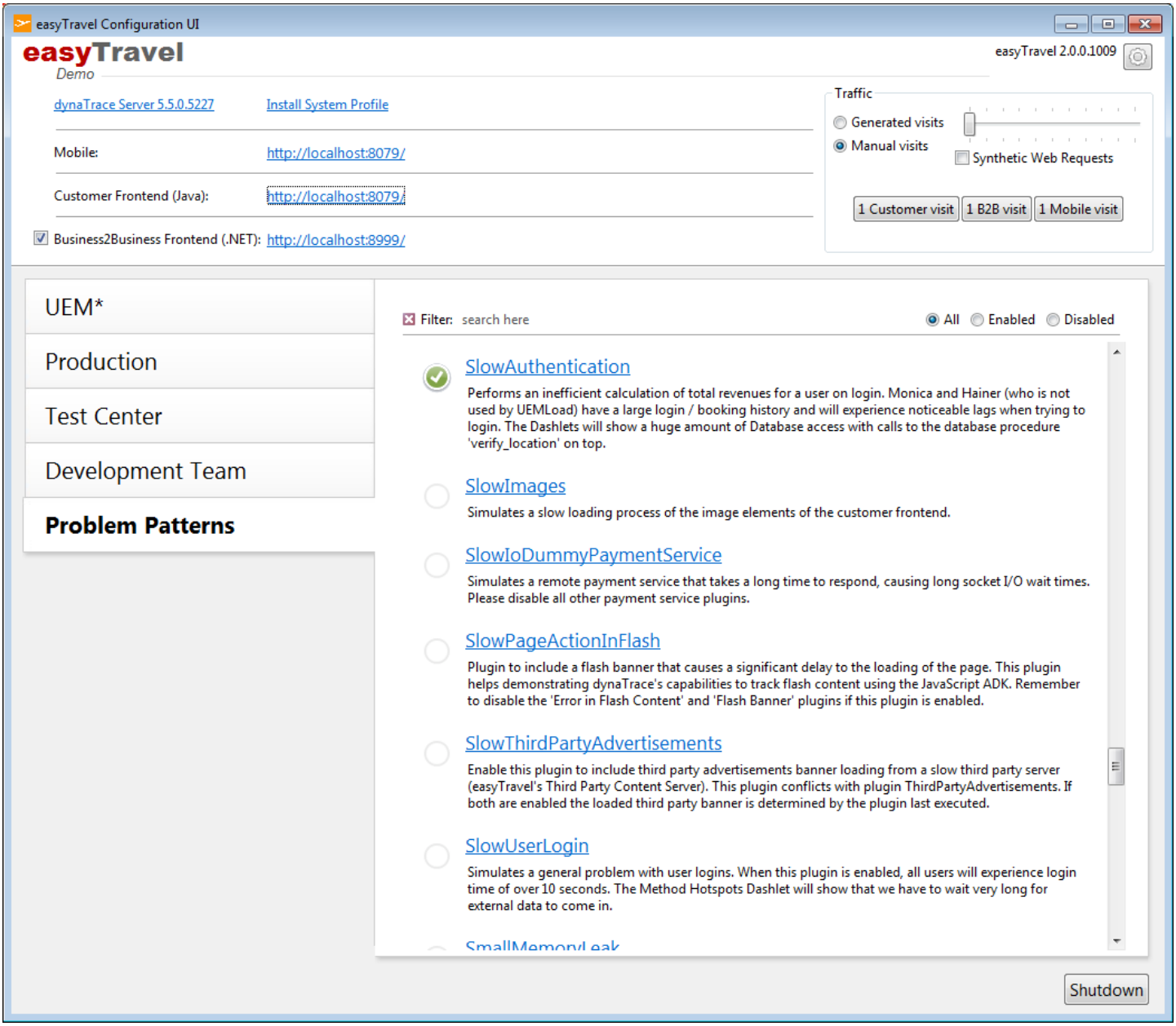
1. Clear the AppMon live session.
2. Start the recording session.
3. Execute test suite steps.
4. Stop recording.

Follow these steps to complete this step:

1. Click **easyTravel Configuration UI > Install System Profile**.
2. Click **UEM - Lightweight** to start a scenario.
3. Select **Manual Visits**.
4. Execute the (Selenium) test suite.
5. The problem patterns are hidden by default. To show them, click the **Settings** icon in the easyTravel Configuration UI.




Login monica.html - Selenium IDE 2.3.0
File Edit Actions Options Help
Base URL http://localhost:8079/
Fast Slow
Test Case
Clear Live Session
Start Recording Ses...
Login monica
Stop Recording
Logout
Runs: 2
Failures: 0
Table Source
Command Target Value
open http://localhost:8079/
click id=loginForm:loginLink
click id=loginForm:j_idt55
setSpeed 2000 2000
click id=loginForm:j_idt60:5:j_idt61
Command
Target
Find
Value
Log Reference UI-Element Rollup
Info Clear
[info] Executing: |open | http://localhost:8020/rest/management/profiles/easyTravel/clear | |
[info] Changed test case
[info] Executing: |open | http://localhost:8020/rest/html/management/profiles/easyTravel | |
[info] Executing: |type | presentableName | easyTravel Test Run 2 |
[info] Executing: |clickAndWait | but_startrecording | |
[info] Changed test case
[info] Executing: |open | http://localhost:8079/ | |



Step Three: Create a Regression Analysis Dashboard

Check the recorded session

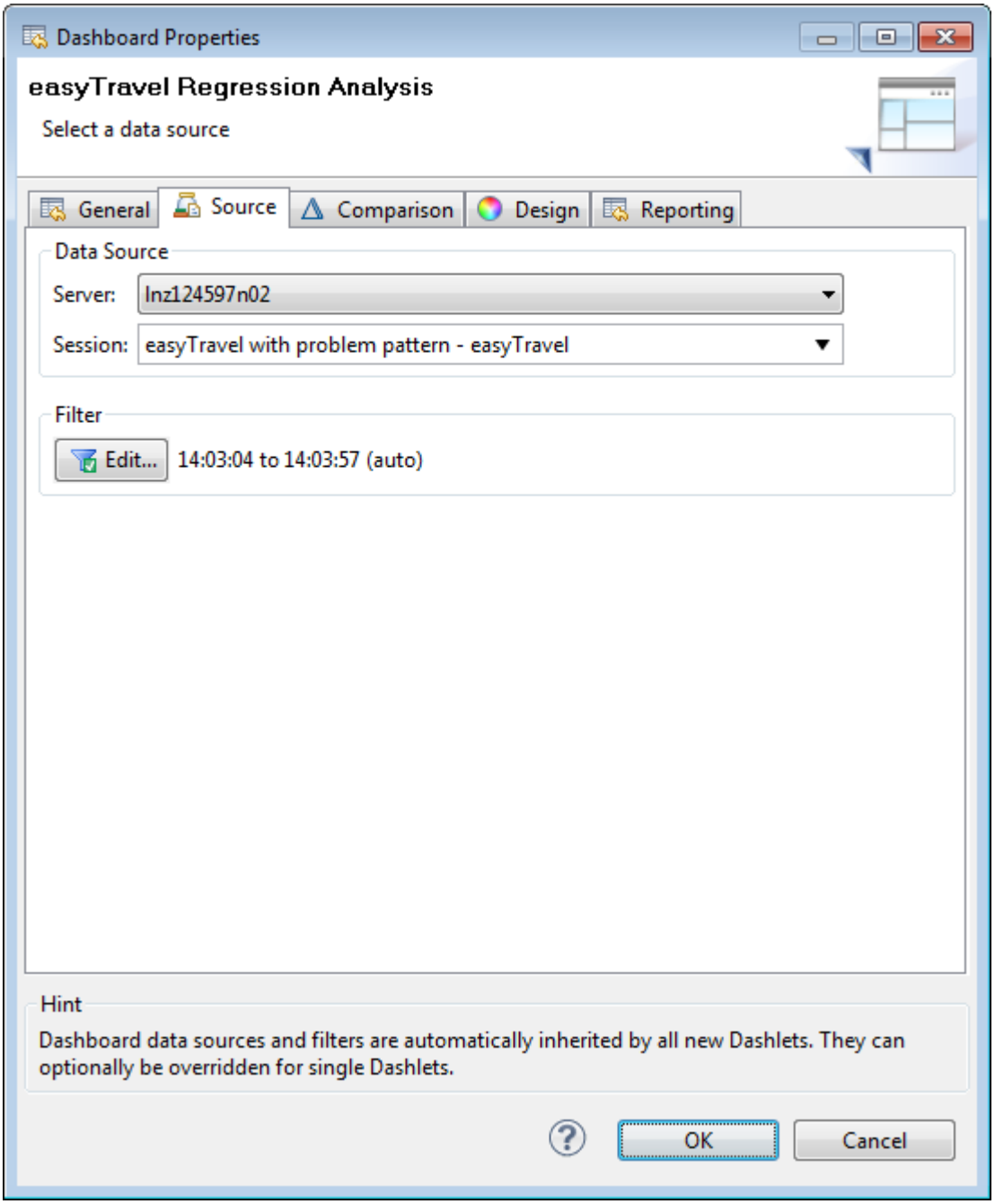
If the automated test suite executes successfully, you will see two stored sessions on the AppMon Server. The sessions should be identical because there are no other requests running while you record. Take a look at the sessions and verify that:

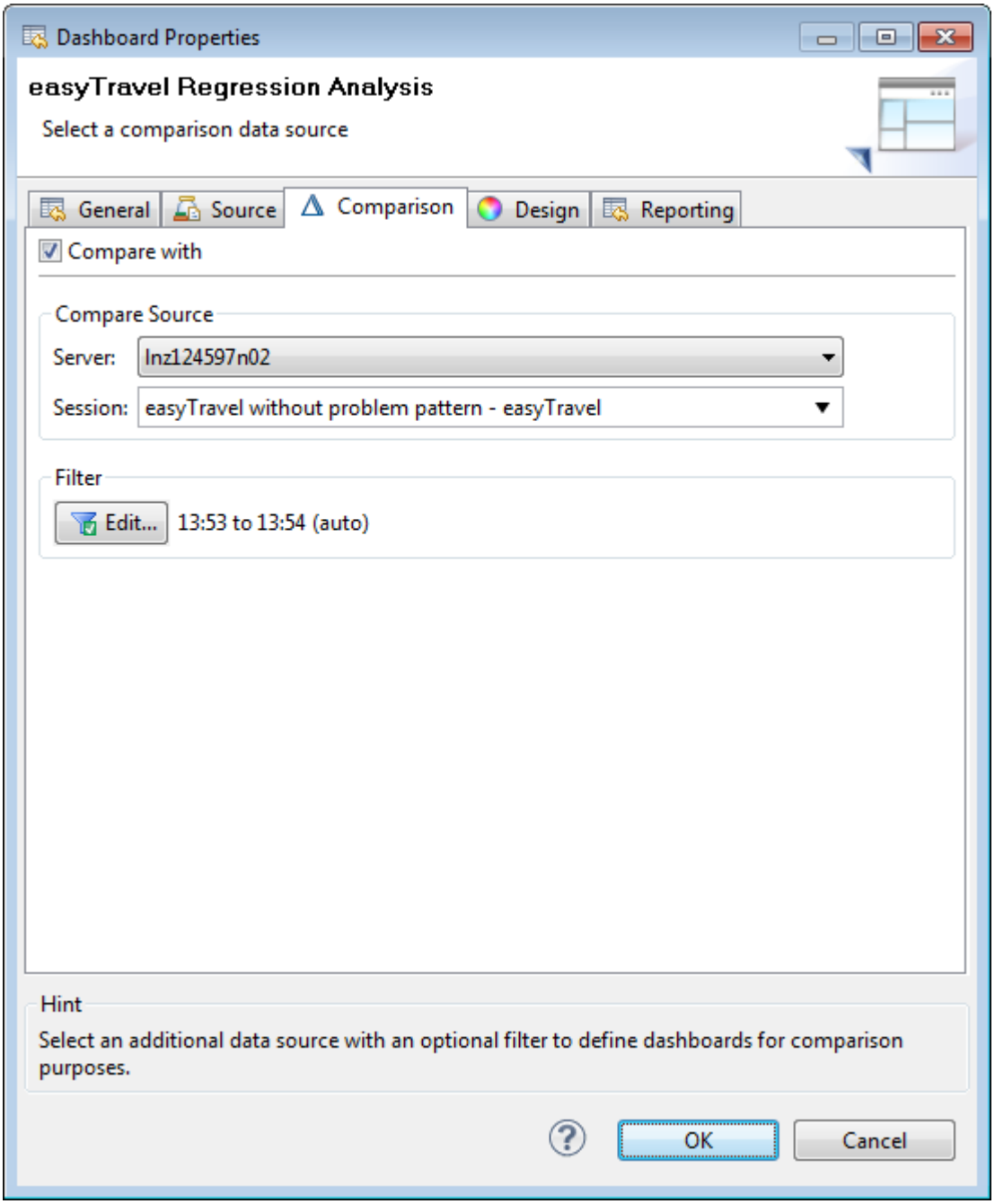
- All PurePaths are complete.
- You have the same amount of PurePaths in the sessions.
- The PurePaths have about the same length (same instrumentation depth).

In a real-world scenario, the sessions will be different and you may have a different number of PurePaths.

Comparison dashboard

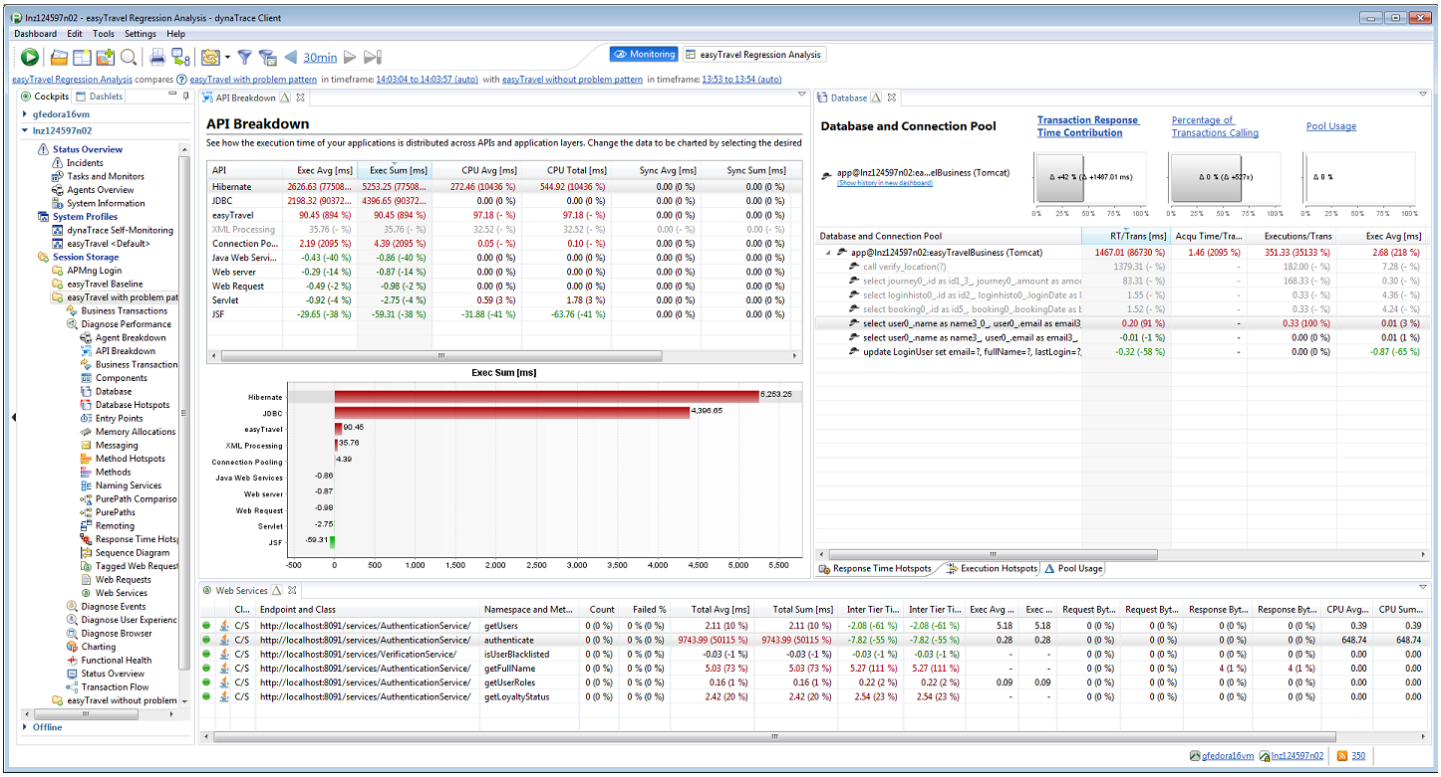
1. Create a new empty dashboard (^N).
2. Right-click the **Comparison** tab.
3. Click the **Properties** dialog box.
4. Set the data source to the stored session 1, which is the baseline for the regression.
5. Right-click the **Source** tab.
6. Click the **Properties** dialog box.
7. Set the data source to the stored session #2. Do not apply filters.


Dashboard Properties
easyTravel Regression Analysis
Select a data source
General
Source
Comparison
Design
Reporting
Data Source
Server:
lnz124597n02
Session:
easyTravel with problem pattern - easyTravel
Filter
Edit...
14:03:04 to 14:03:57 (auto)
Hint
Dashboard data sources and filters are automatically inherited by all new Dashlets. They can optionally be overridden for single Dashlets.
OK
Cancel



Drag the following dashlets to the dashboard so they appear on the same page:

- API Breakdown Dashlet
- Database
- Web services
- Remoting
- Method



Example Regression Dashboard

Due to the SlowAuthentication artificial regression, some timings differ drastically. You may want to explore this deeper.

Conclusion

AppMon provides regression information and quickly puts you on the path to discovery, even if you are not familiar with the application. For best results, use the AppMon automation interface to perform regression analysis in an automated environment.

Memory analysis

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is memory analysis?

Goal of this walkthrough

Detailed steps

Conclusion

Related pages

Analyzing out of memory errors

Analyze exceptions and log messages

Analyzing a slow transaction

What is memory analysis?

The Java memory model - or garbage collector (GC) - solves many memory problems. However, new strains of memory leaks pop up constantly. Many applications simply consume too much memory or thrash the GC.

Memory analysis is the art of identifying all these problems, and their root cause.

Goal of this walkthrough

This walkthrough reviews the of memory analysis process. Analyze your memory whether you have problems or not. You can save a lot CPU (less GC) and memory.

Scenario

Use easyTravel to simulate memory problems and analyze them.

Requirements

You need the following components installed and configured:

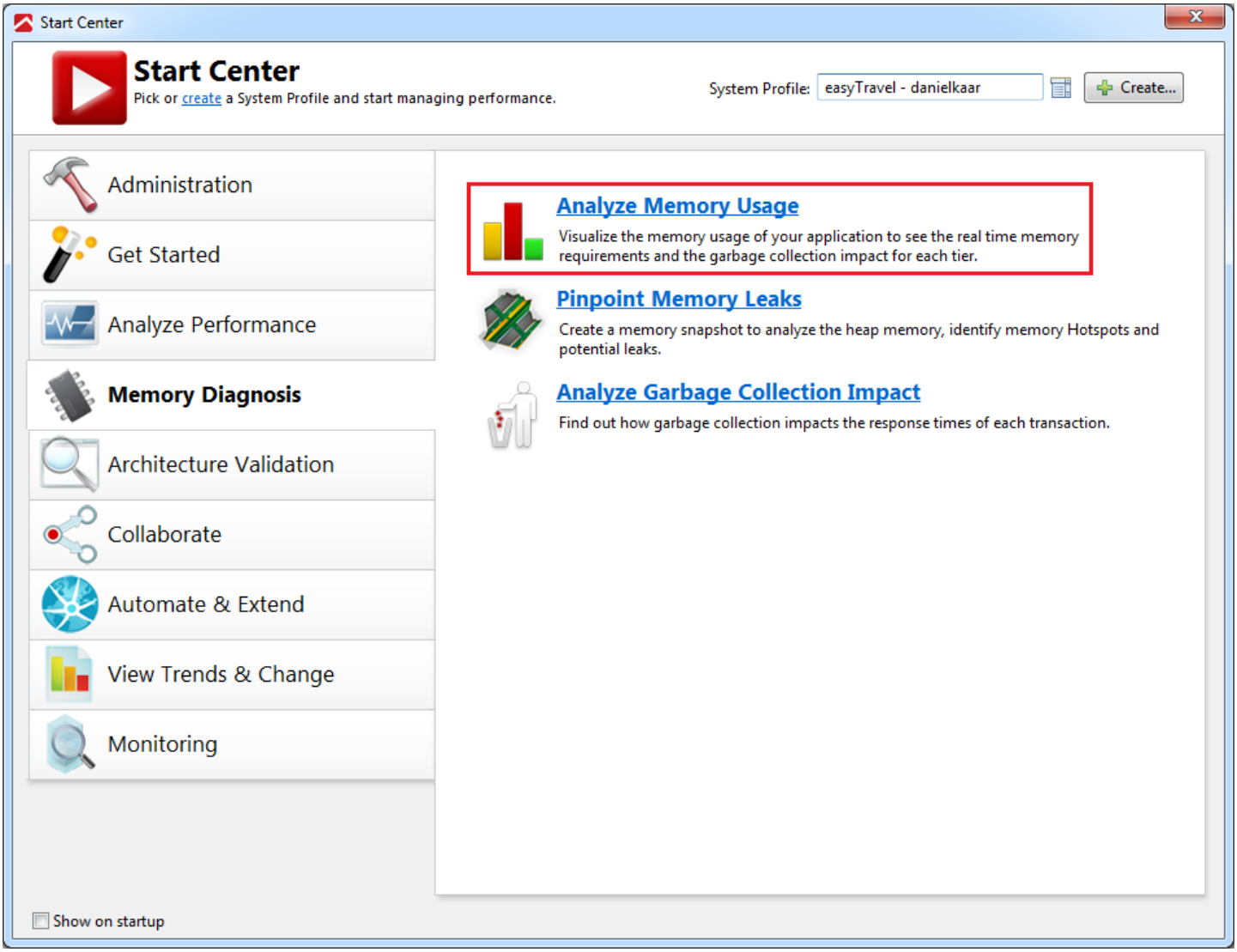
- easyTravel demo application.
- AppMon installation with a Test Center Edition license.
- Default easyTravel System Profile.

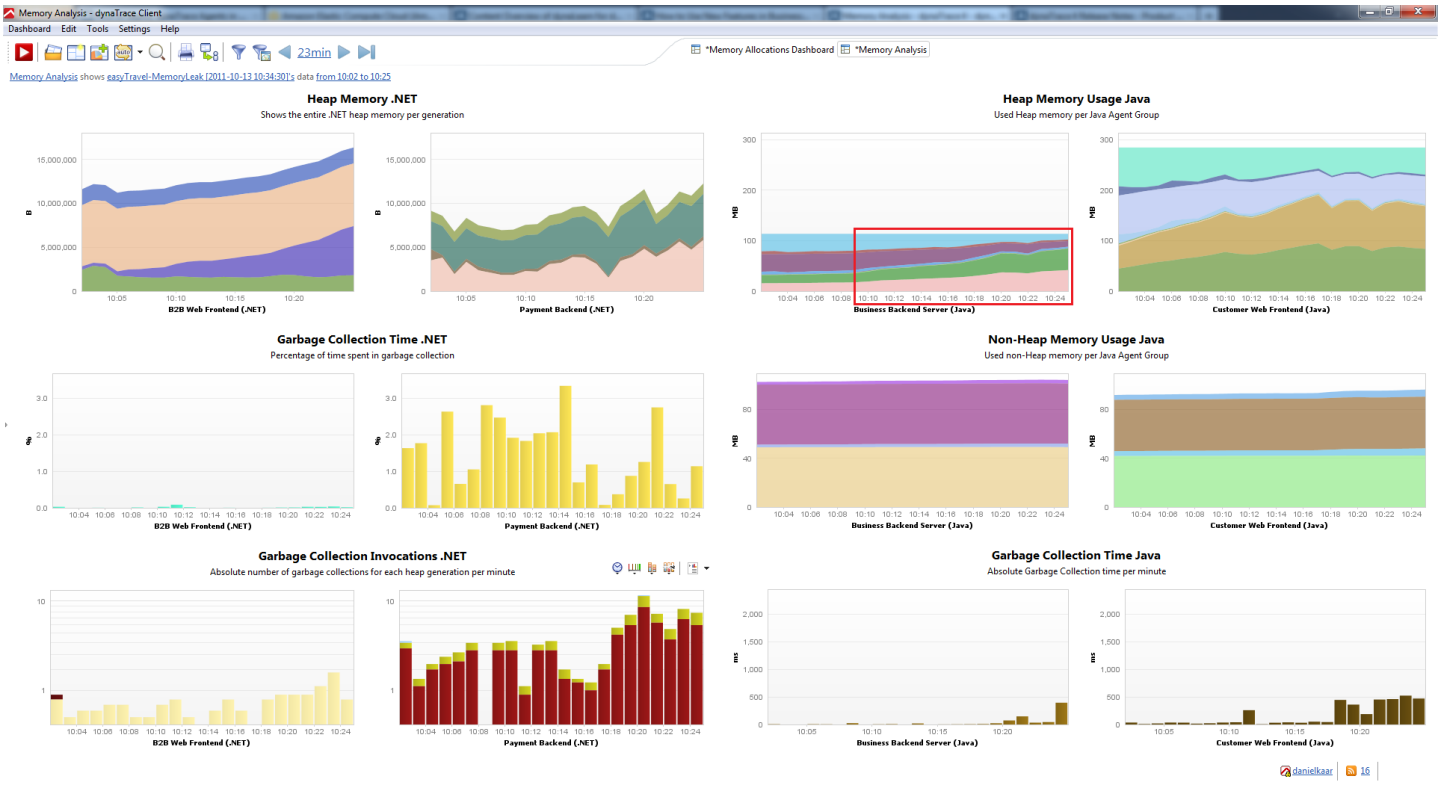
Detailed steps

Monitor memory trend

Monitor the memory trends when you run your application with or without load. If the memory increases continuously, you should analyze this. To see a trend, activate the **Medium Memory Leak Problem Pattern**. See the attached session of this walkthrough for more information.

1. Open the **Start Center**.
2. Select **Memory Diagnosis**.
3. Select the correct System Profile.
4. Select **Analyze Memory Usage**.


Start Center
Start Center
Pick or create a System Profile and start managing performance.
System Profile: easyTravel - danielkaar
Create...
Administration
Get Started
Analyze Performance
Memory Diagnosis
Architecture Validation
Collaborate
Automate & Extend
View Trends & Change
Monitoring
Analyze Memory Usage
Visualize the memory usage of your application to see the real time memory requirements and the garbage collection impact for each tier.
Pinpoint Memory Leaks
Create a memory snapshot to analyze the heap memory, identify memory Hotspots and potential leaks.
Analyze Garbage Collection Impact
Find out how garbage collection impacts the response times of each transaction.
Show on startup


Memory Analysis - dynaTrace Client
Heap Memory .NET
Heap Memory Usage Java
Garbage Collection Time .NET
Non-Heap Memory Usage Java
Garbage Collection Invocations .NET
Garbage Collection Time Java

The **Memory Analysis** dashboard opens and displays easyTravel's memory usage trends. The Memory Leak Problem pattern displays a growing memory demand. When you monitor the memory, keep an eye on the GC behavior. High GC can lead to high CPU utilization and slow response times. This is due to suspensions. See Memory Problems on the Performance Wiki for more information.

Analyze memory trend

Memory increases for two reasons: A bad GC memory setting or an application problem. The application problem is either a memory leak or one of the several other memory problems. For more information, see Memory Problems in the Performance wiki.

If the memory is consistently high, with GCs, but no `OutOfMemory` errors, analyze the constant high memory usage. Identify how GC impacts your response time. If memory increases and you get an `Out Of Memory Error`, try to pinpoint memory leaks.

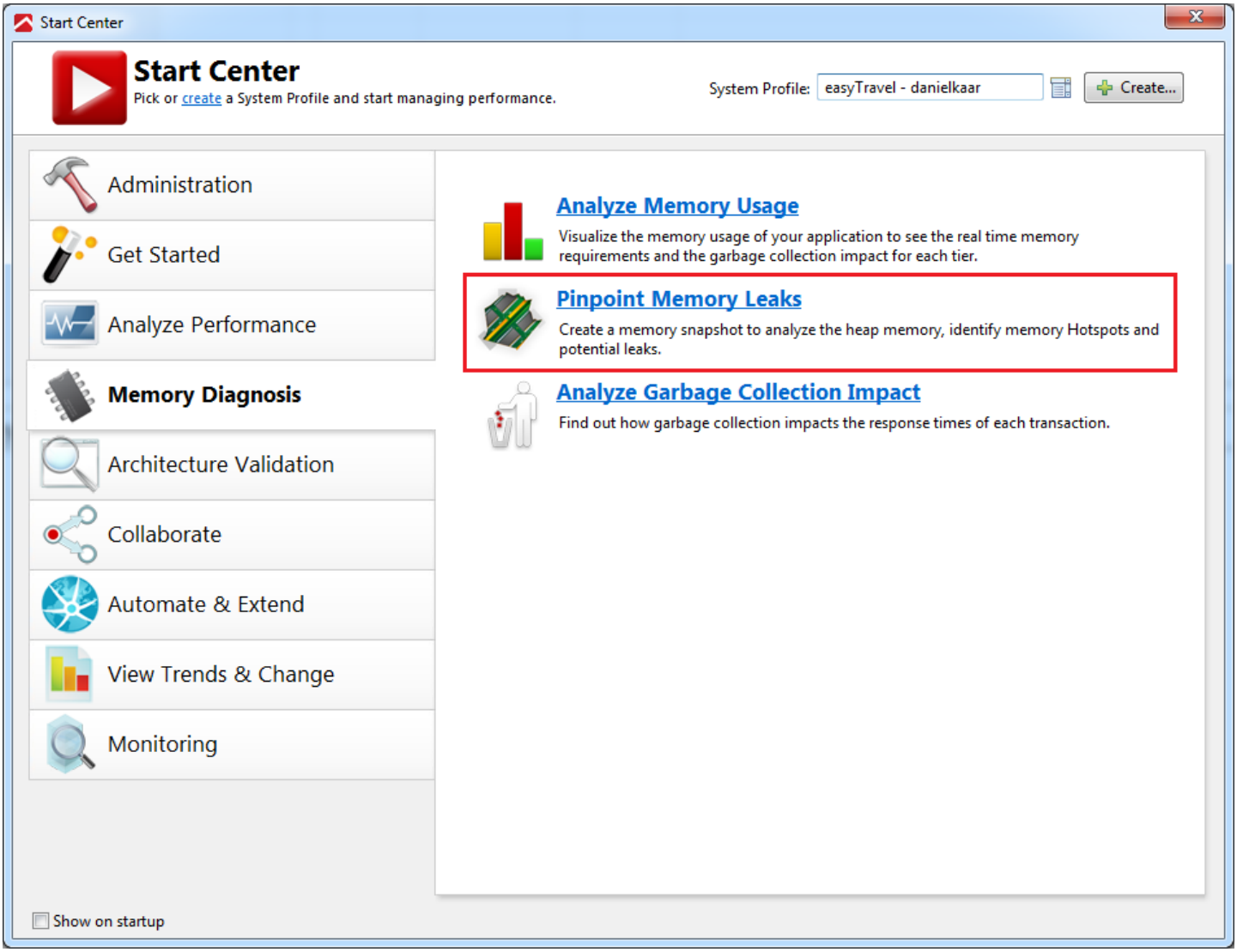
If memory increases slowly, and a GC decreases memory, but it increases again, check your GC/memory settings. They may be incorrect. See Understanding and Optimizing your Garbage Collector for more information.

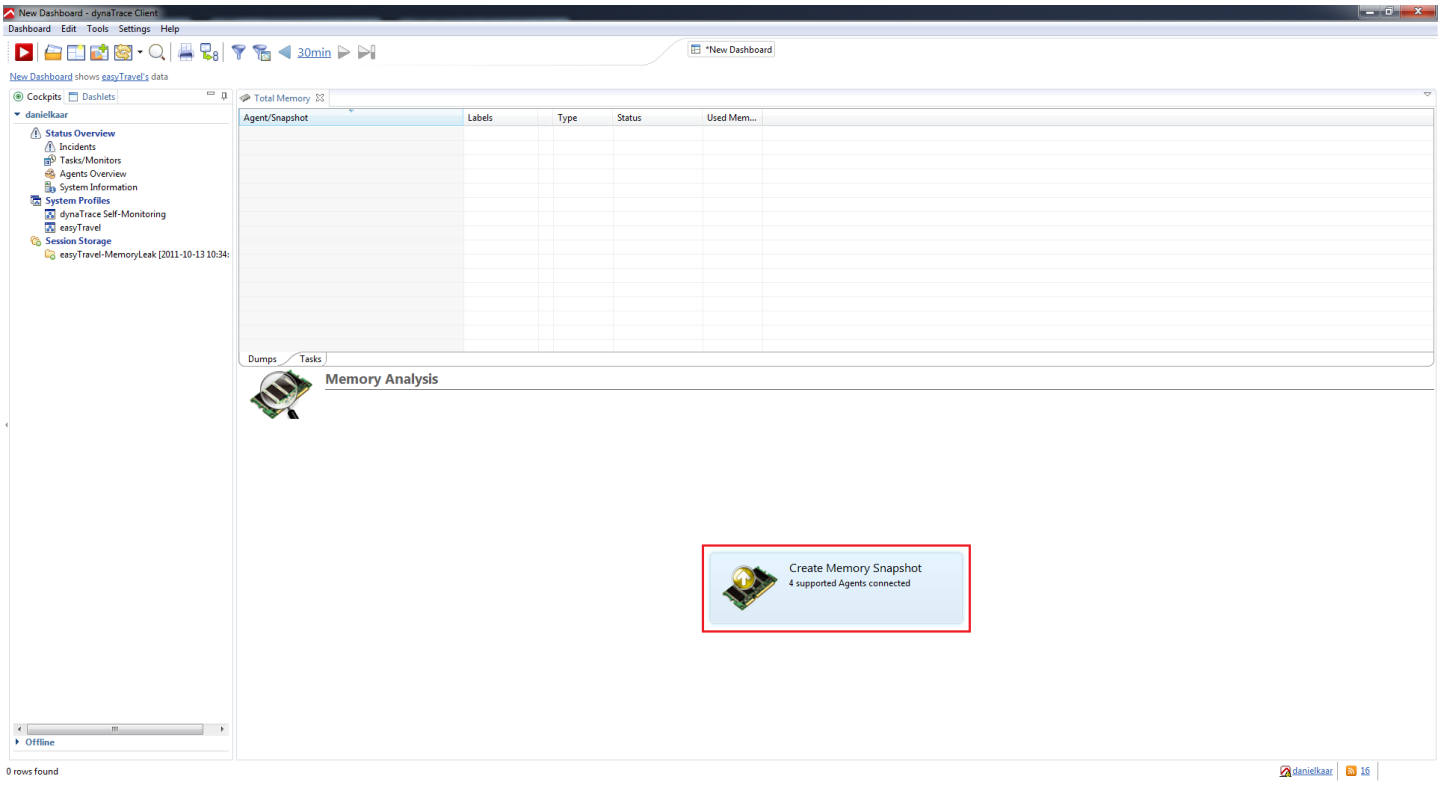
If you have many GCs and volatile memory, but your GC settings are correct, your transactions may consume too much memory. Analyze the transaction memory usage.

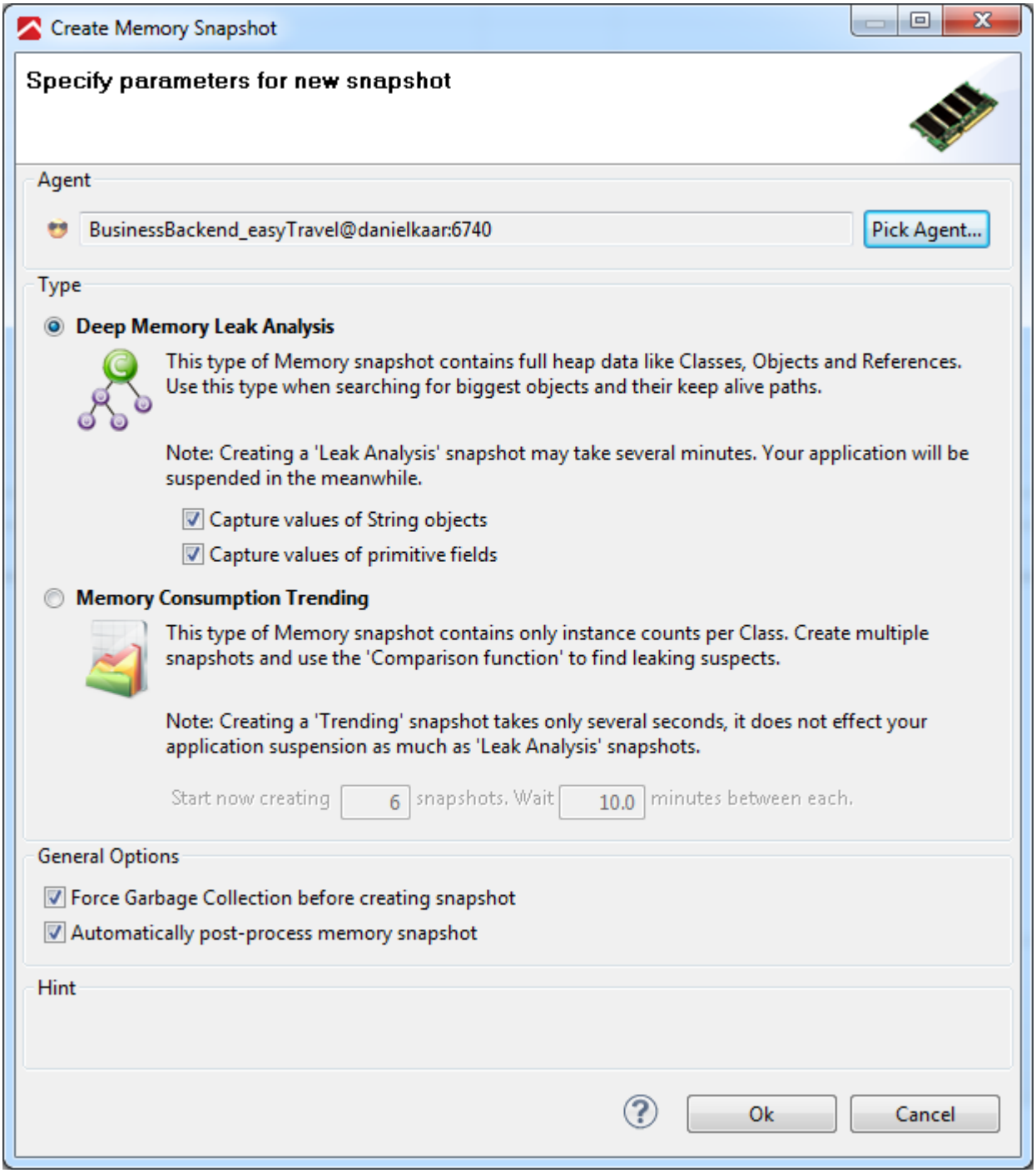
Analyze constant high memory usage

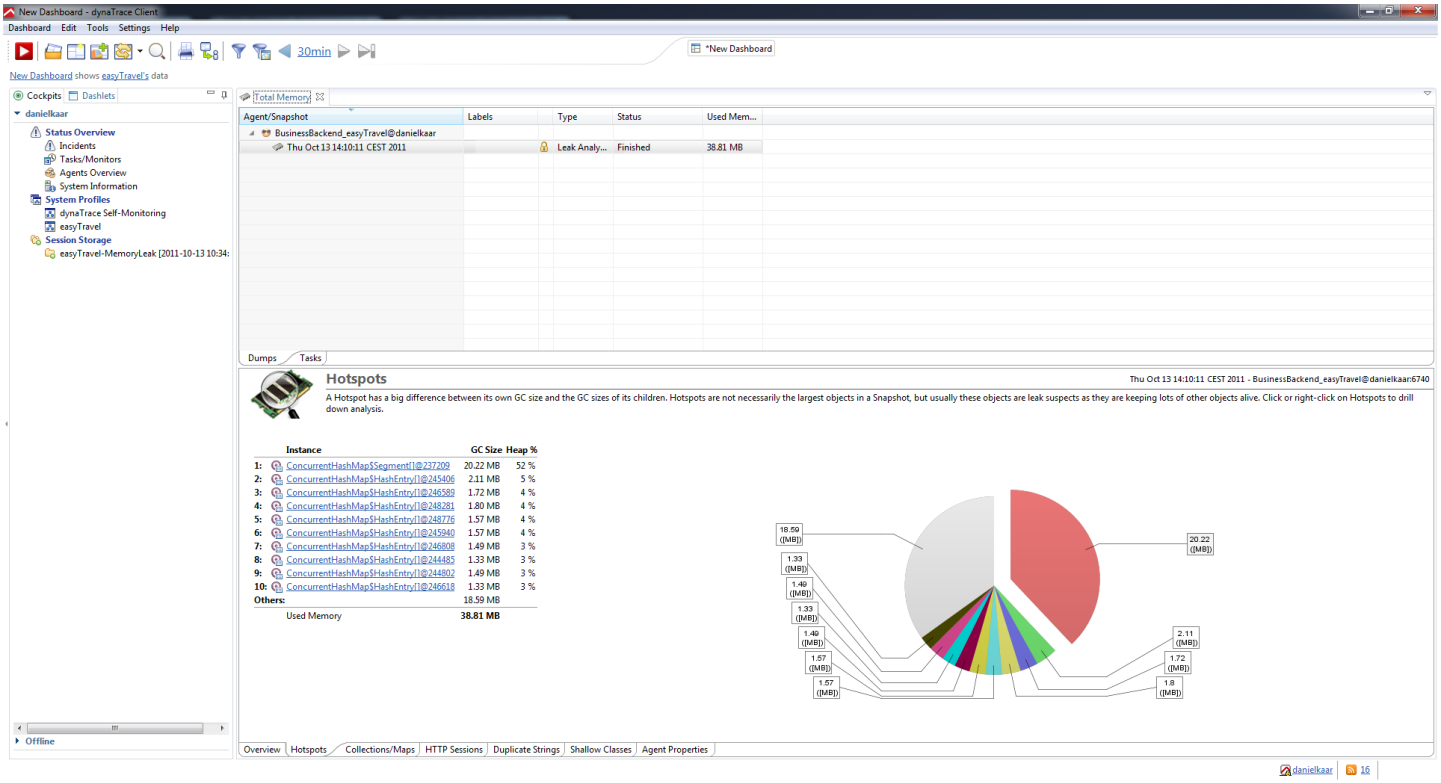
Take a leak snapshot to discover why your JVM/CLR consumes too much memory. Click **Start Center > Memory Diagnosis > Pinpoint Memory > Deep Leak Analysis**.

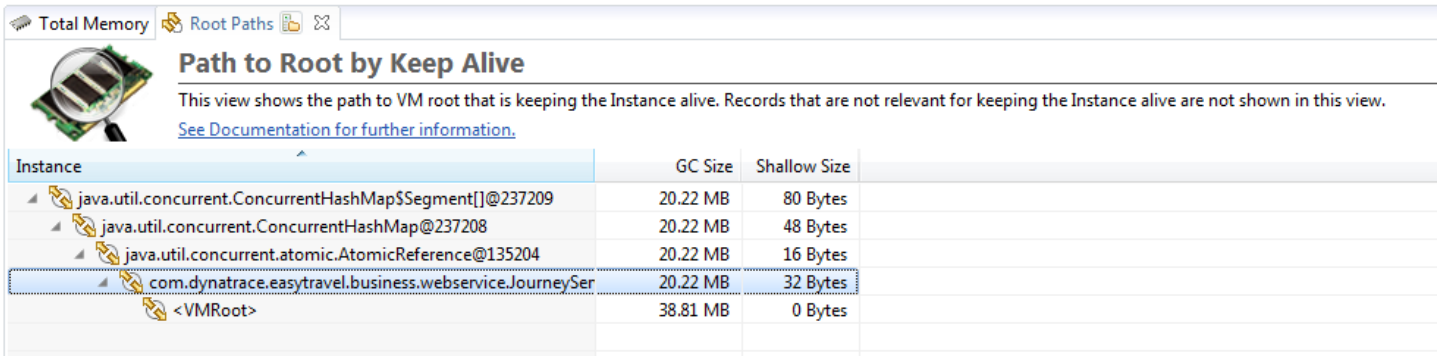
To find a memory leak, or something else that consumes memory, but is not load dependent, force a GC. If an application does not perform optimally and does not GC properly, do not choose this option.


Start Center
Start Center
Pick or create a System Profile and start managing performance.
System Profile: easyTravel - danielkaar
Create...
Administration
Get Started
Analyze Performance
Memory Diagnosis
Architecture Validation
Collaborate
Automate & Extend
View Trends & Change
Monitoring
Analyze Memory Usage
Visualize the memory usage of your application to see the real time memory requirements and the garbage collection impact for each tier.
Pinpoint Memory Leaks
Create a memory snapshot to analyze the heap memory, identify memory Hotspots and potential leaks.
Analyze Garbage Collection Impact
Find out how garbage collection impacts the response times of each transaction.
Show on startup


New Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
*New Dashboard
Cockpits Dashlets
Total Memory
Agent/Snapshot
Labels
Type
Status
Used Mem...
Status Overview
Incidents
Tasks/Monitors
Agents Overview
System Information
System Profiles
dynaTrace Self-Monitoring
easyTravel
Session Storage
Dumps
Tasks
Memory Analysis
Create Memory Snapshot
4 supported Agents connected
Offline
0 rows found


Create Memory Snapshot
Specify parameters for new snapshot
Agent
BusinessBackend_easyTravel@danielkaar:6740
Pick Agent...
Type
Deep Memory Leak Analysis
This type of Memory snapshot contains full heap data like Classes, Objects and References. Use this type when searching for biggest objects and their keep alive paths.
Note: Creating a 'Leak Analysis' snapshot may take several minutes. Your application will be suspended in the meanwhile.
Capture values of String objects
Capture values of primitive fields
Memory Consumption Trending
This type of Memory snapshot contains only instance counts per Class. Create multiple snapshots and use the 'Comparison function' to find leaking suspects.
Note: Creating a 'Trending' snapshot takes only several seconds, it does not effect your application suspension as much as 'Leak Analysis' snapshots.
Start now creating 6 snapshots. Wait 10.0 minutes between each.
General Options
Force Garbage Collection before creating snapshot
Automatically post-process memory snapshot
Hint
Ok
Cancel





This example forces the GC. Look at the hotspot to see what holds most of the memory in the JVM. Use this view to look for memory leaks or to understand which objects keep most of the objects alive. Notice that a HashMap Segment holds 52%.

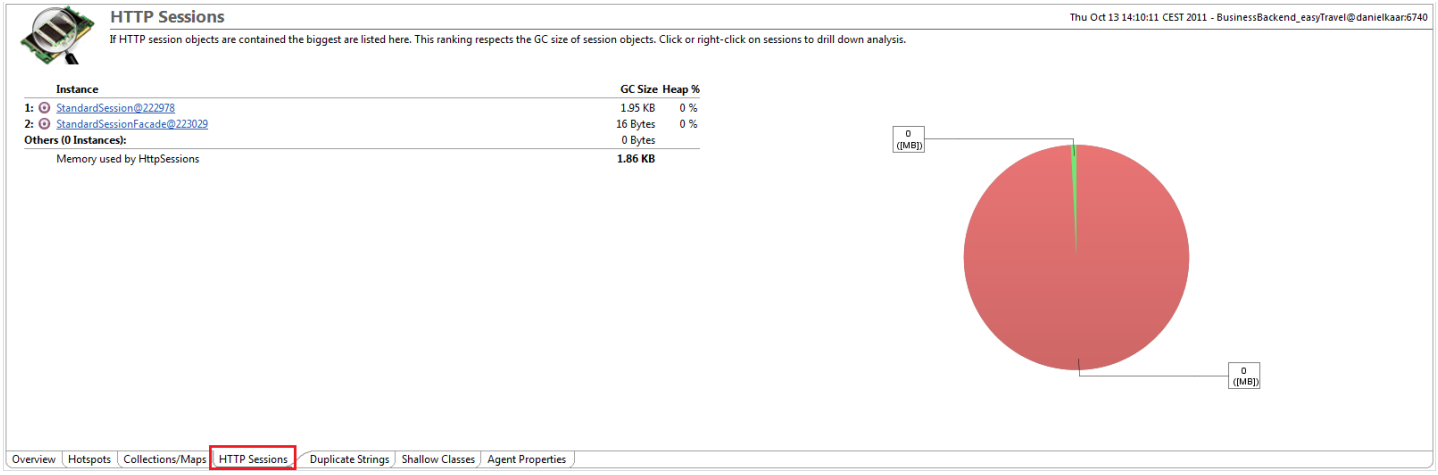
1. Right-click the **Hot Spot**.
2. Choose the **Keep Alive** set. Check to see who holds this object.
3. Look at Follow References to see which references this object holds. This displays instances that the selected object keeps alive, and an aggregated set with instance count.

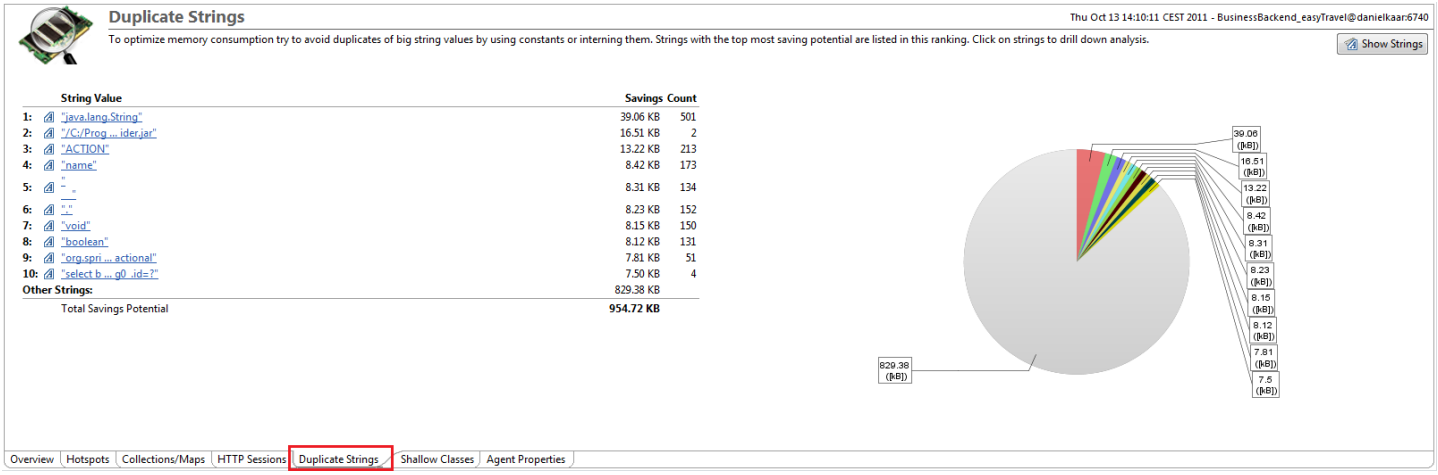
Use this to get an overview:

- The GC size tells how much free memory you would have if the objected were garbage collected.
- Follow References does not aggregate, but shows the real reference tree.

- **Keep Alive** only shows you what is kept alive or what keeps alive. It also aggregates views of things like instance count.
- Direct follows the object structure and displays which memory holds the reference.

Look at the HTTP sessions or duplicated strings. These can cause high memory because they keep too many things. For more information, see the Excessive Memory Usage section in Memory Problems on the Performance Wiki.





Conclusion

Memory analysis is very important if you want your application to perform and scale. Use AppMon to accomplish this.

Analyzing out of memory errors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is an out of memory error?

Goal of this walkthrough

Detailed steps

Conclusion

Related pages

Memory analysis

Memory leak

Out of memory error

Analyze exceptions and log messages

Analyzing a slow transaction

What is an out of memory error?

Out of Memory errors in Java and .NET occur when the VM/CLR cannot create a requested object because it cannot allocate any more memory. This can happen for several reasons:

- Memory Leak.
- Insufficient memory configuration for the JVM.
- Single transactions need too much memory so parallel running transactions do not have enough.

See Out Of Memory Error for more information.

Goal of this walkthrough

This walkthrough explains how AppMon communicates out of memory errors, and how to respond.

Scenario

The easyTravel application is running and has run out of memory. To induce a memory leak in easyTravel, locate the Running Out of Memory scenario in the Test Center.

Requirements

- Standard Demo Applications - easyTravel installation.
- AppMon installation with default easyTravel System Profile.
- The easyTravel Memory Leak activated, for example using the Running out of Memory Scenario.

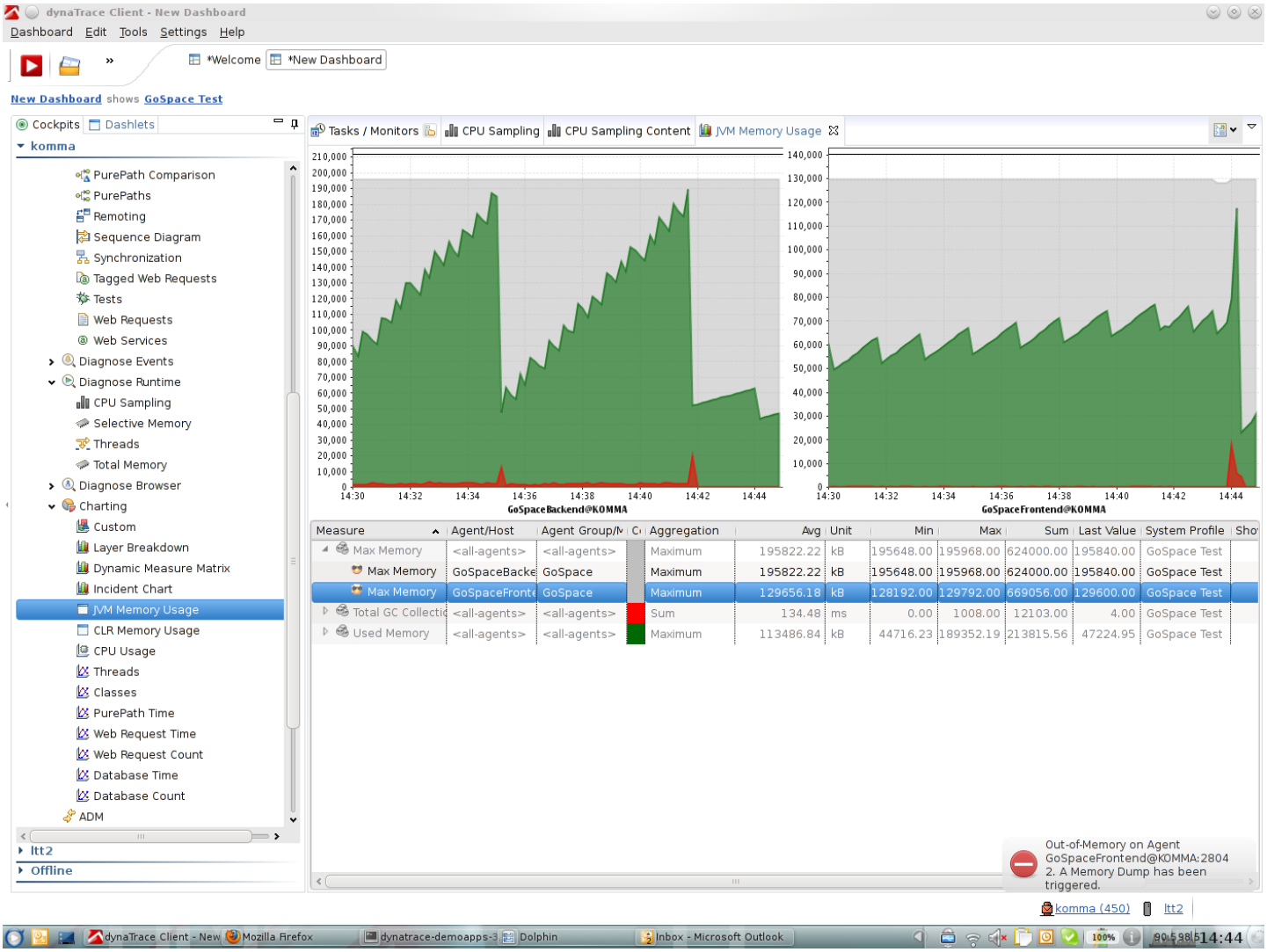
Detailed steps

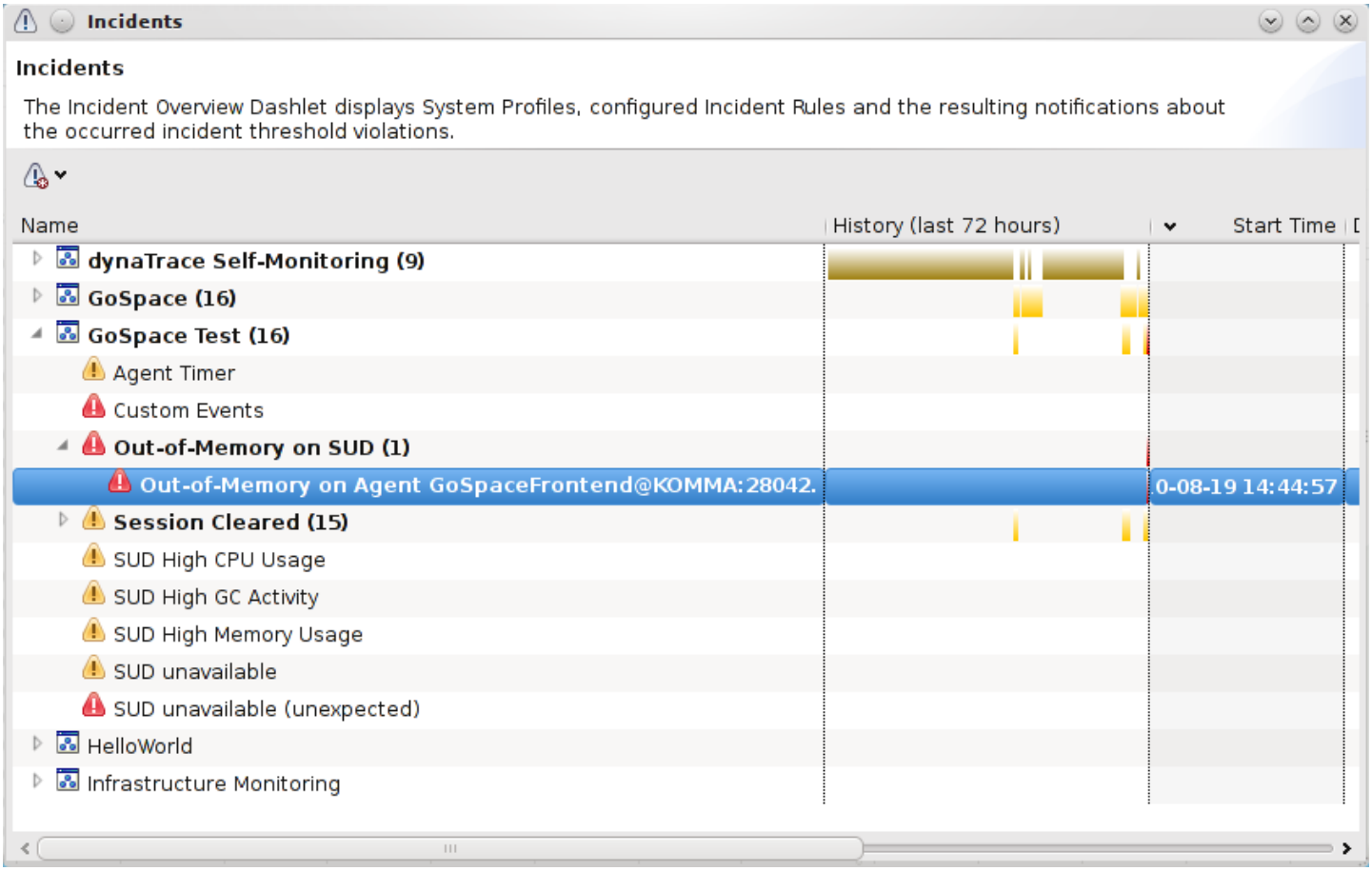
Detect the out of memory error

After the memory leak in easyTravel has exhausted the memory, AppMon raises a new incident. This displays in the lower right corner of the application.

For Java 6 and later, AppMon automatically captures a leak analysis memory snapshot. Use this to analyze the problem. To create a leak analysis memory snapshot in earlier Java releases, set up an incident that triggers when the memory usage reaches very high levels such as 95%.

To open the **Incident** dashlet, click on the AppMon **Server** name in the lower right corner of the AppMon Client to get a list of incidents.


dynaTrace Client - New Dashboard
Dashboard Edit Tools Settings Help
*Welcome *New Dashboard
New Dashboard shows GoSpace Test
Cockpits Dashlets
komma
PurePath Comparison
PurePaths
Remoting
Sequence Diagram
Synchronization
Tagged Web Requests
Tests
Web Requests
Web Services
Diagnose Events
Diagnose Runtime
CPU Sampling
Selective Memory
Threads
Total Memory
Diagnose Browser
Charting
Custom
Layer Breakdown
Dynamic Measure Matrix
Incident Chart
JVM Memory Usage
CLR Memory Usage
CPU Usage
Threads
Classes
PurePath Time
Web Request Time
Web Request Count
Database Time
Database Count
ADM
ltt2
Offline
Tasks / Monitors
CPU Sampling
CPU Sampling Content
JVM Memory Usage
GoSpaceBackend@KOMMA
GoSpaceFrontend@KOMMA
Measure
Agent/Host
Agent Group/M
Aggregation
Avg
Unit
Min
Max
Sum
Last Value
System Profile
Max Memory <all-agents> <all-agents> Maximum 195822.22 kB 195648.00 195968.00 624000.00 195840.00 GoSpace Test
Max Memory GoSpaceBacke GoSpace Maximum 195822.22 kB 195648.00 195968.00 624000.00 195840.00 GoSpace Test
Max Memory GoSpaceFronte GoSpace Maximum 129656.18 kB 128192.00 129792.00 669056.00 129600.00 GoSpace Test
Total GC Collecti <all-agents> <all-agents> Sum 134.48 ms 0.00 1008.00 12103.00 4.00 GoSpace Test
Used Memory <all-agents> <all-agents> Maximum 113486.84 kB 44716.23 189352.19 213815.56 47224.95 GoSpace Test
Out-of-Memory on Agent GoSpaceFrontend@KOMMA:2804
2. A Memory Dump has been triggered.
komma (450) ltt2
14:44



Right-click the **Out Of Memory** incident and open the memory snapshot.

Conclusion

AppMon detects out-of-memory errors and captures a leak analysis snapshot for further diagnosis.

Analyze exceptions and log messages

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Goal of this walkthrough

Detailed steps

Conclusion

Related pages

Memory analysis

Analyzing out of memory errors

Analyzing a slow transaction

Goal of this walkthrough

This walkthrough explains how to use AppMon to diagnose the root cause of exceptions and error log messages.

Scenario

Use the attached easyTravel session or your own application with an injected AppMon agent.

Requirements

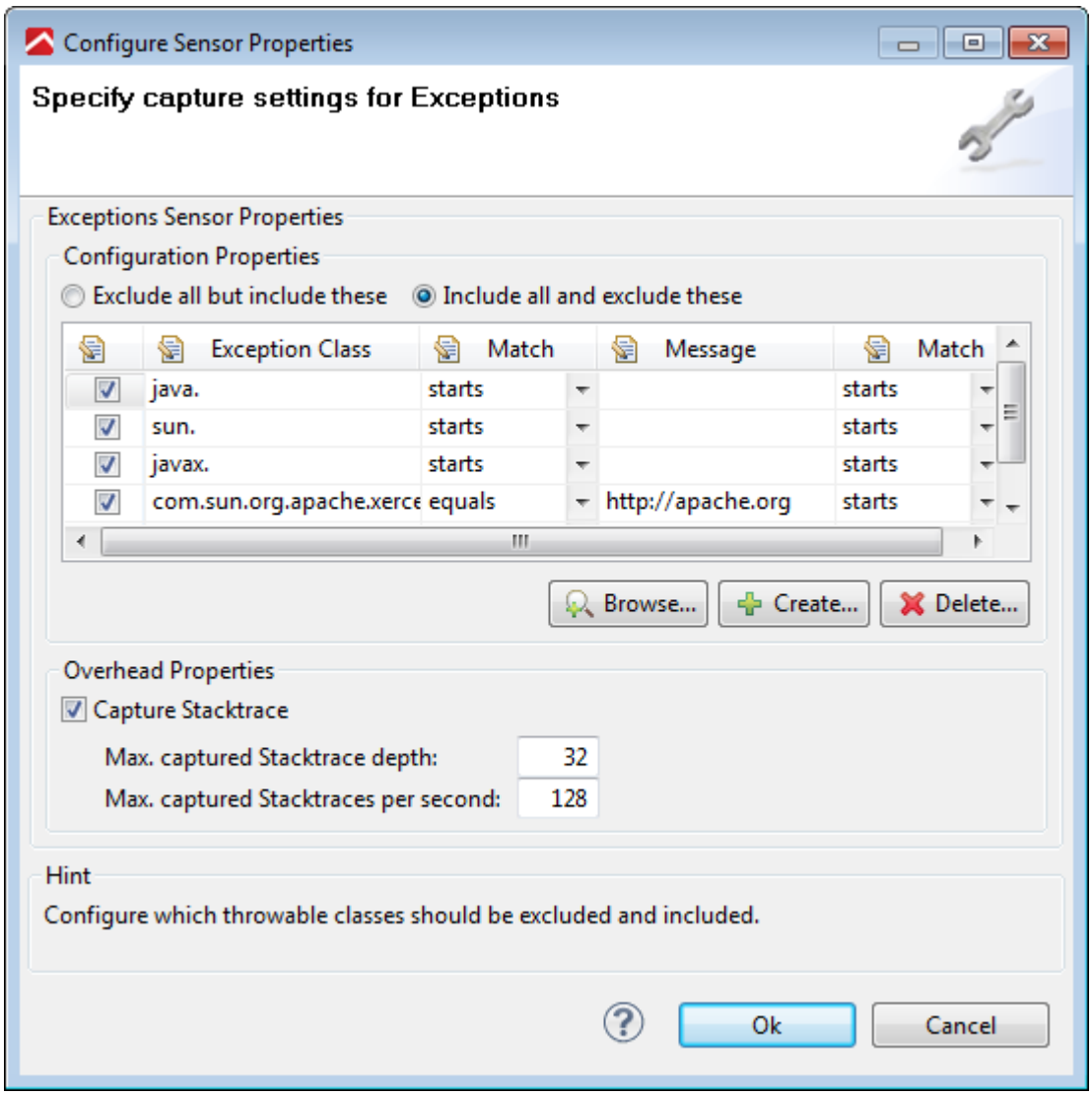
- The AppMon Client if you use the attached easyTravel session.
  or
- An application with an injected AppMon agent.

Detailed steps

Prepare the application

If you use your own application, follow these steps to set the Exception and Logging Sensors to run by default:

1. Go to the **Sensor Configuration** in the appropriate Agent.
2. Click **Properties** for the Exception Sensor (or .NET Exceptions for .NET). The Java Exception Sensor does not capture JDK exceptions by default.
3. Change this in your System Profile.

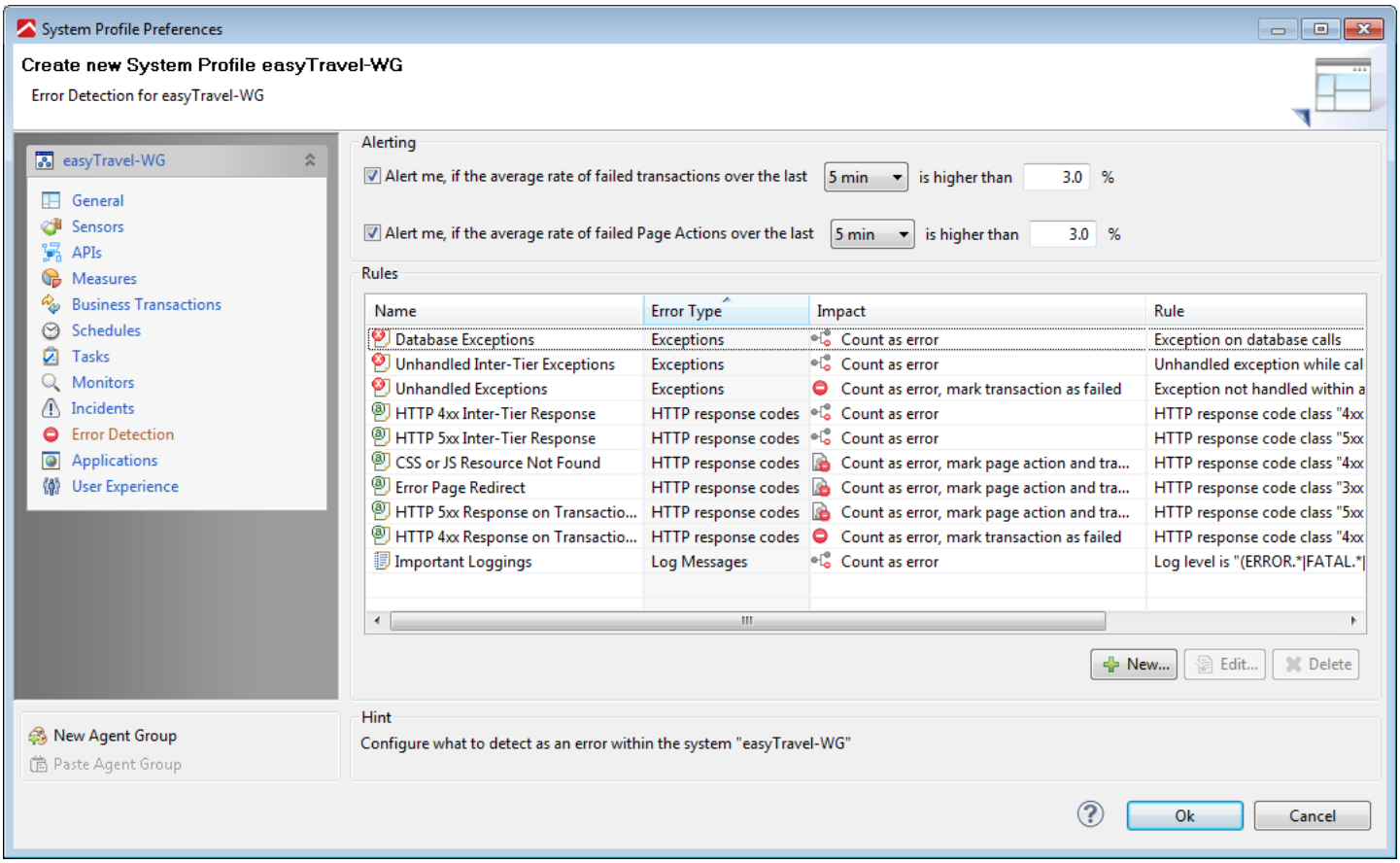


Specify Capture Settings For Exceptions

AppMon logging captures errors and warnings. To capture DEBUG messages, go to **Sensor Configuration > Properties > Logging and .NET Logging Sensor**.

> **Warning**
>
> DEBUG logging captures can cause considerable overhead in your application.

AppMon detects errors in your application via a configurable set of error detection rules. These rules are based on exceptions, HTTP response codes, and log messages. AppMon counts errors by transaction. The node that triggers the error is marked in the PurePath. Optionally, the transactions are marked as failed. The System Profile dictates the configuration for error detection rules. These rules are for common problems in applications, like unhandled exceptions that are outside of the transaction scope, or HTTP 500 responses. See System Profile - Error Detection for information about configuring error detection rules.

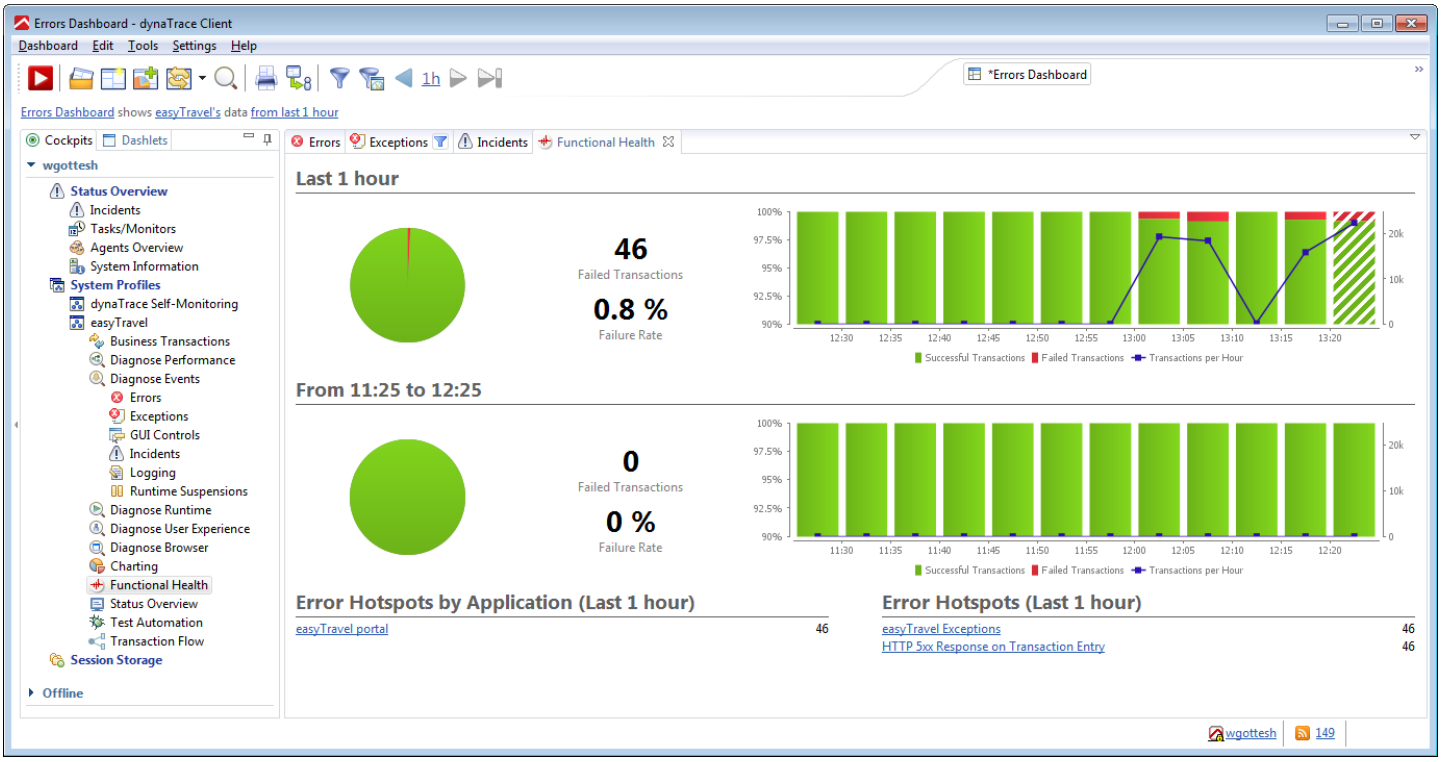


Create a New System Profile

Monitor functional health

For an overview of the functional health of your application, use the **Functional Health Dashlet**. You can see how the transaction's load and failure rate compare to another time frame.

For an overview of errors within a certain time frame, click **Diagnose Events/Errors > Errors dashlet**.

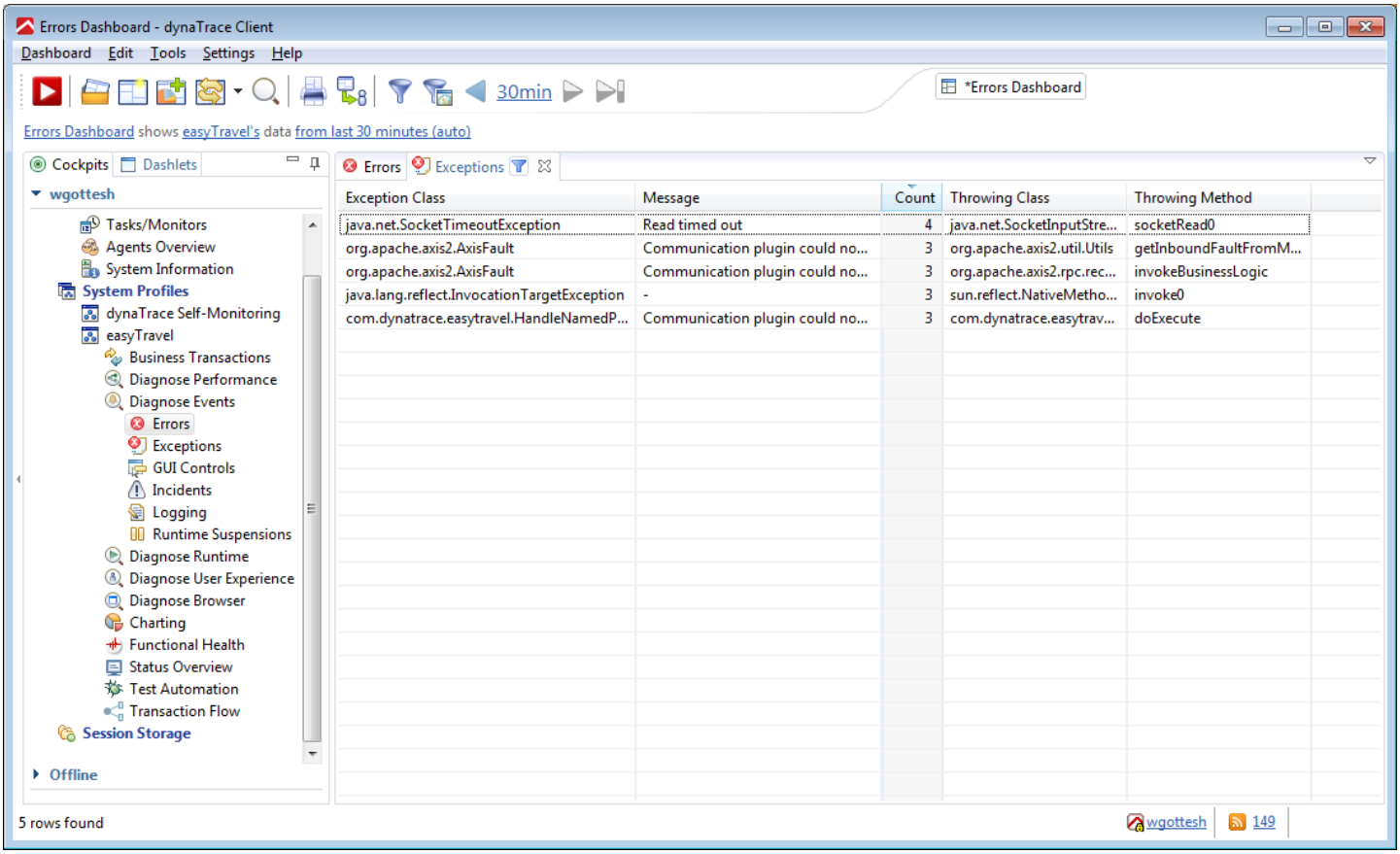


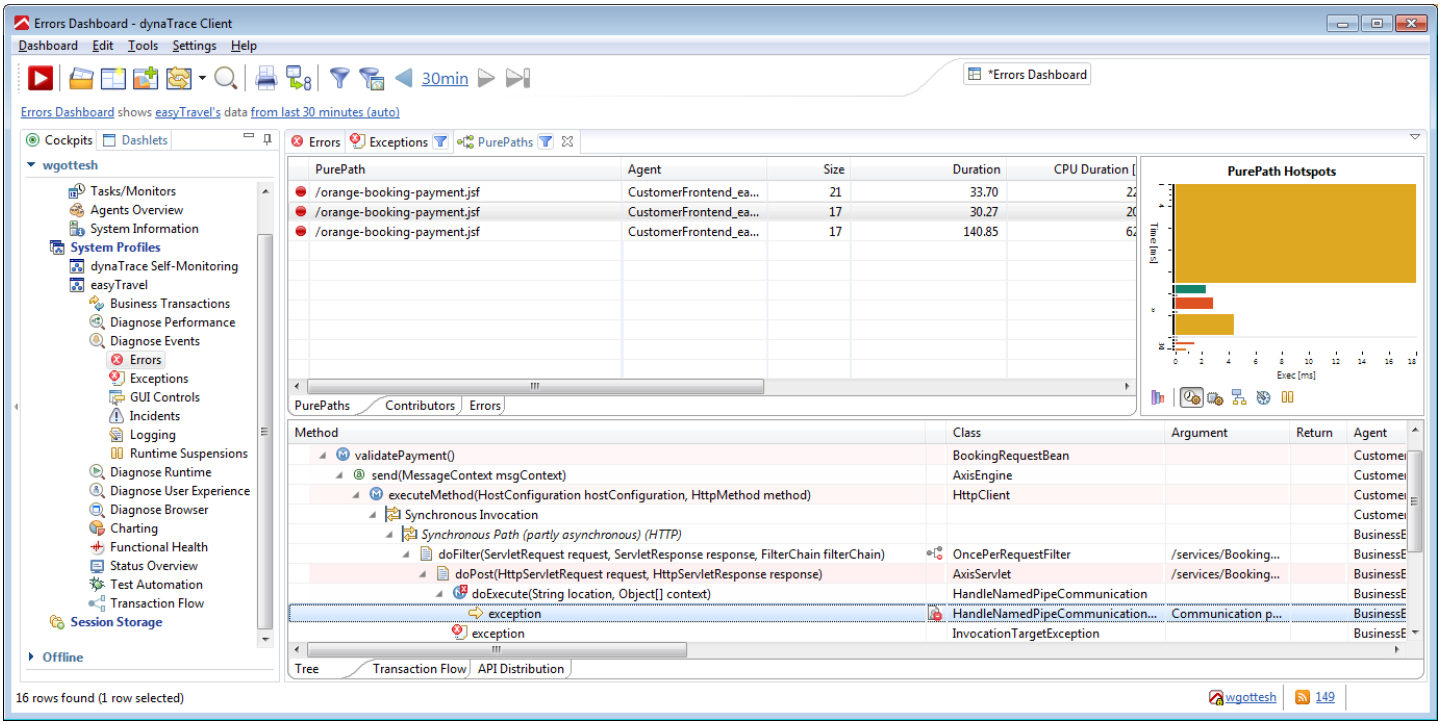
The Errors Dashboard

Identify errors

The **Errors** dashlet includes the following:

- The Error Overview pane - Details errors and their occurrences.
- The Error Details pane - Details the exception class and message for the selected group of error rules or a single rule.



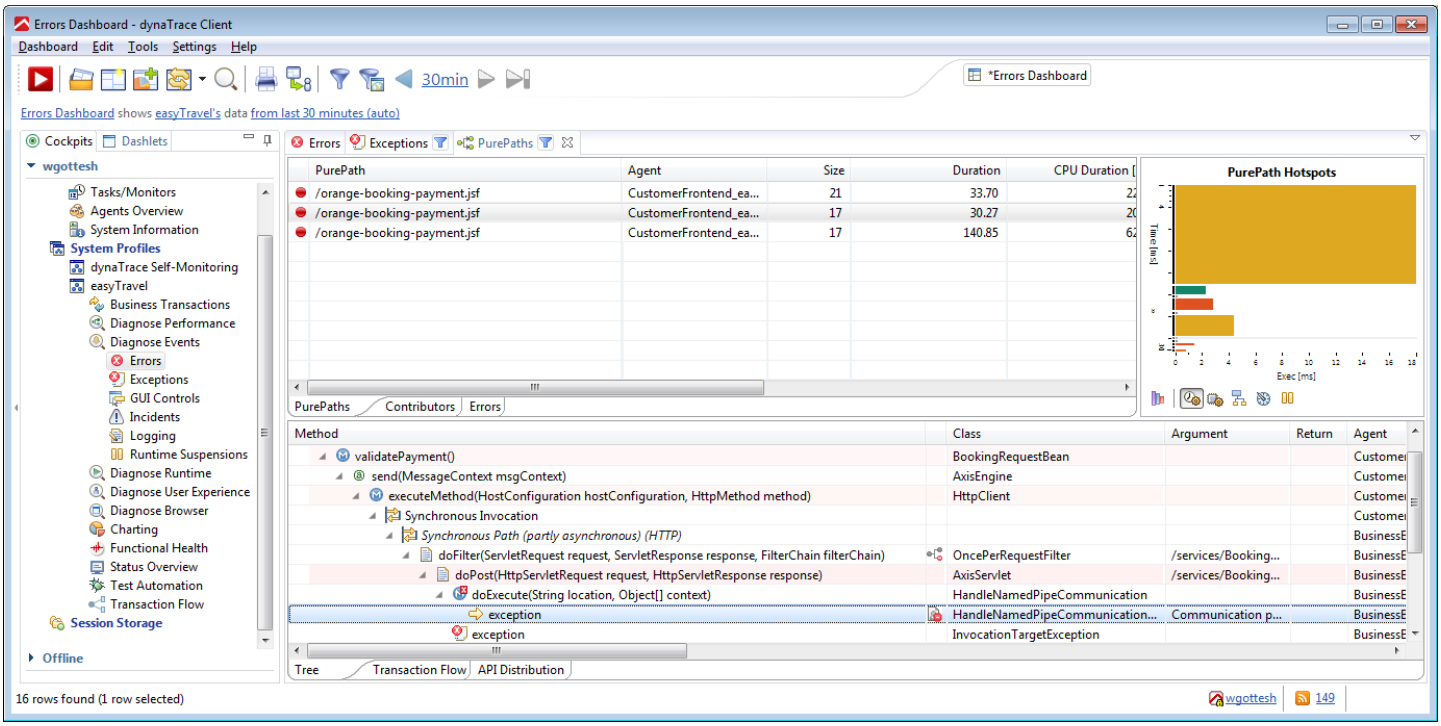


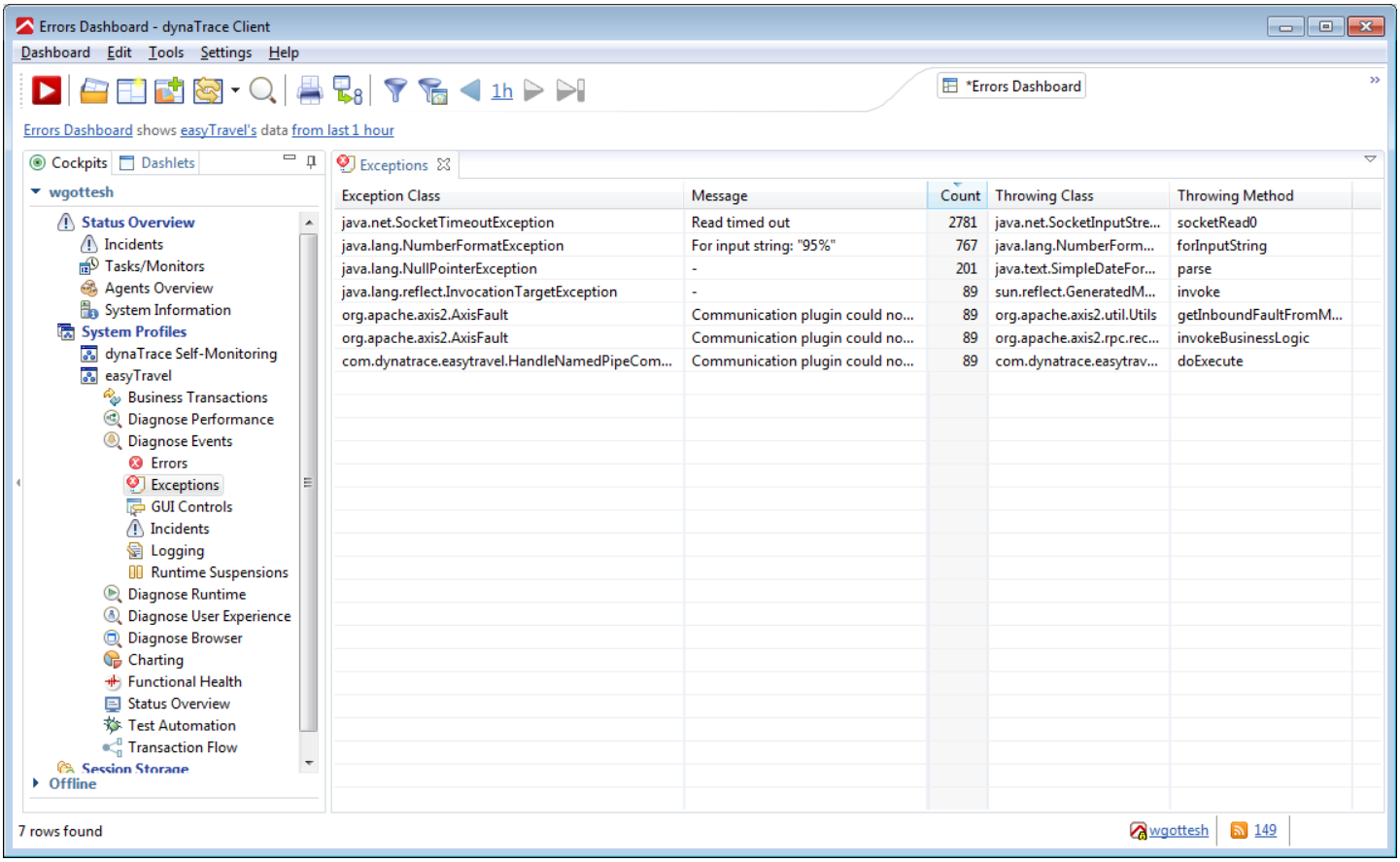
Identify exceptions

To find where an exception occurs, open your System Profile or use the attached session and follow these steps:

1. Click **Diagnose Events**.
2. Open the **Exceptions** dashlet. This displays all of the exceptions in the stored session, or in the PurePaths of your live session. You can see the exception class, message, which class threw the exceptions, and how often it occurs.
3. Right-click and select **Drill Down to PurePath** to identify where the exception comes from and why it occurs.

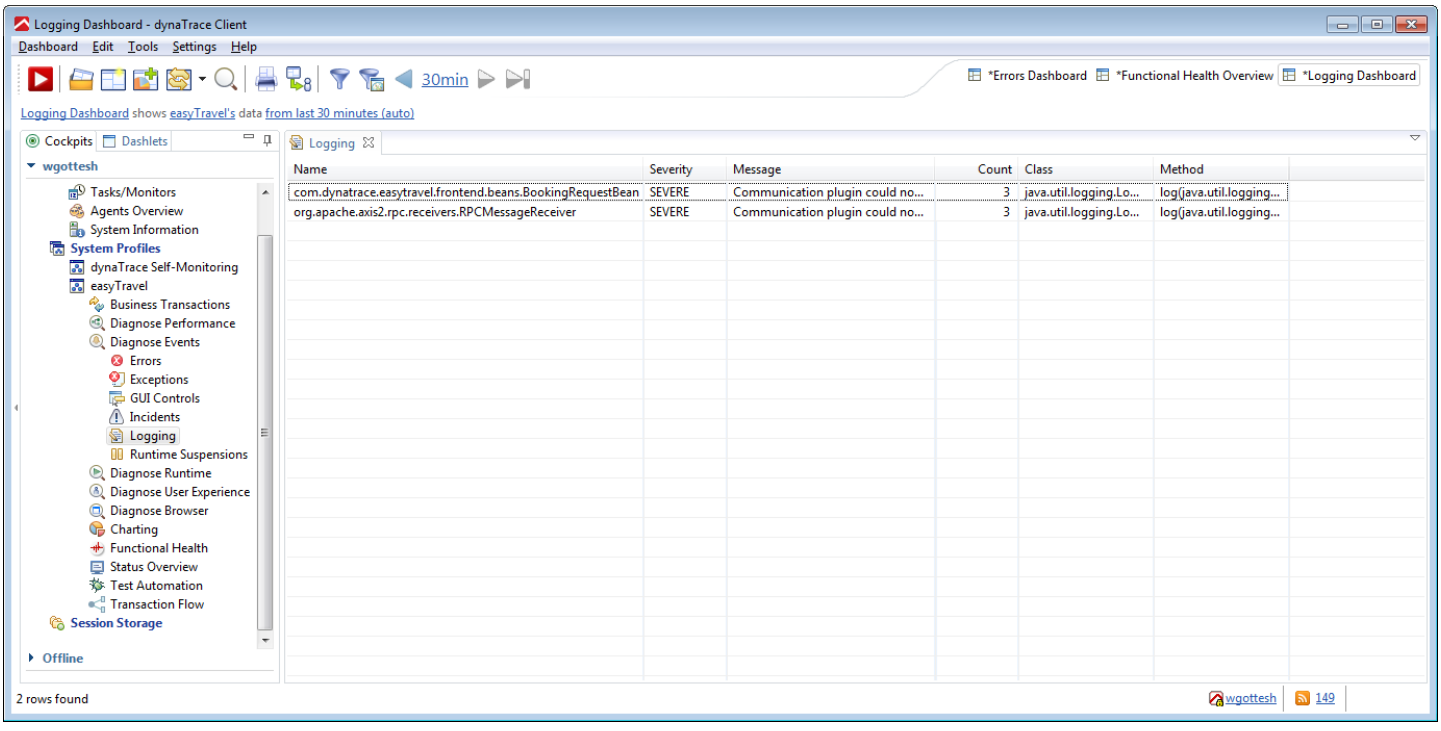
Drill down to business transactions (if available) or web requests to identify the transaction that triggers the exception.

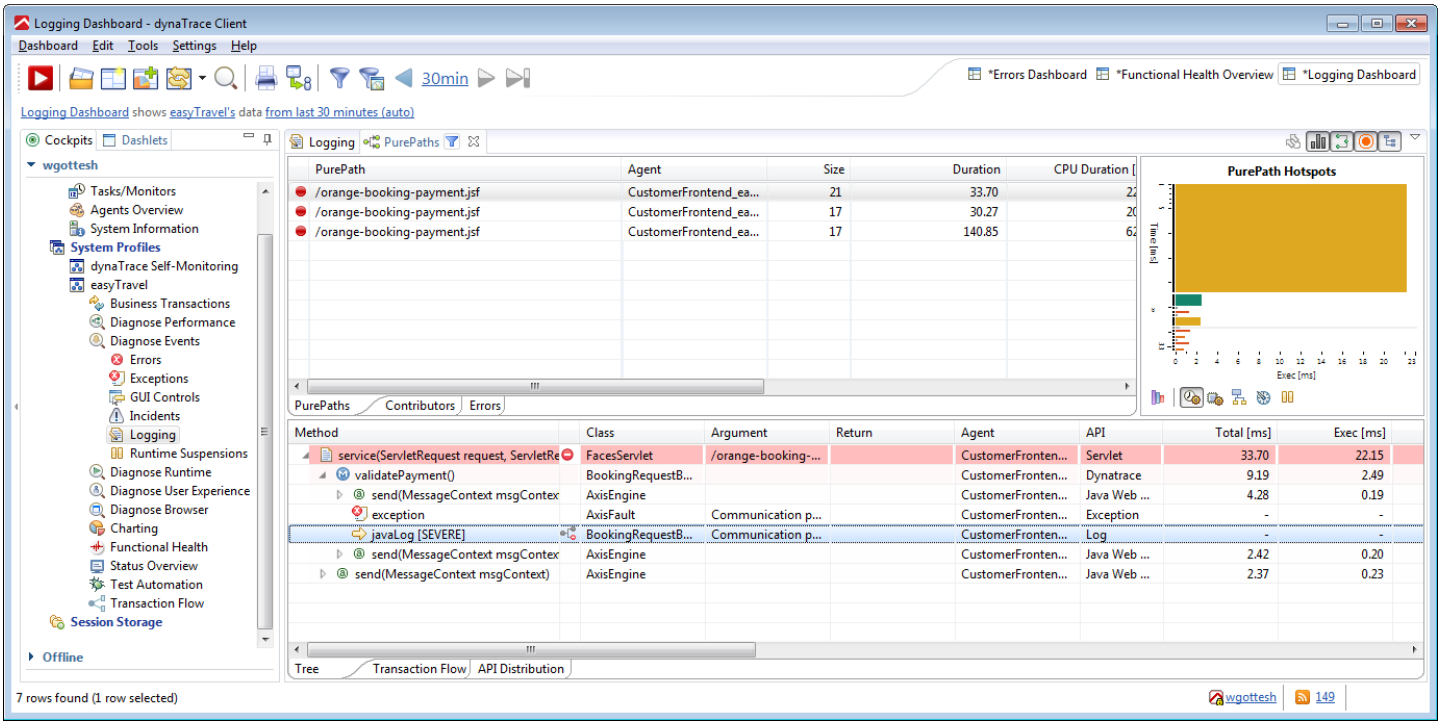




Identify the root cause of error logs

Identify the root cause the same way you identify exceptions. Choose the **Logging** dashlet instead of the **Exceptions** dashlet. Drill down to the PurePath and determine the reason for the exception.

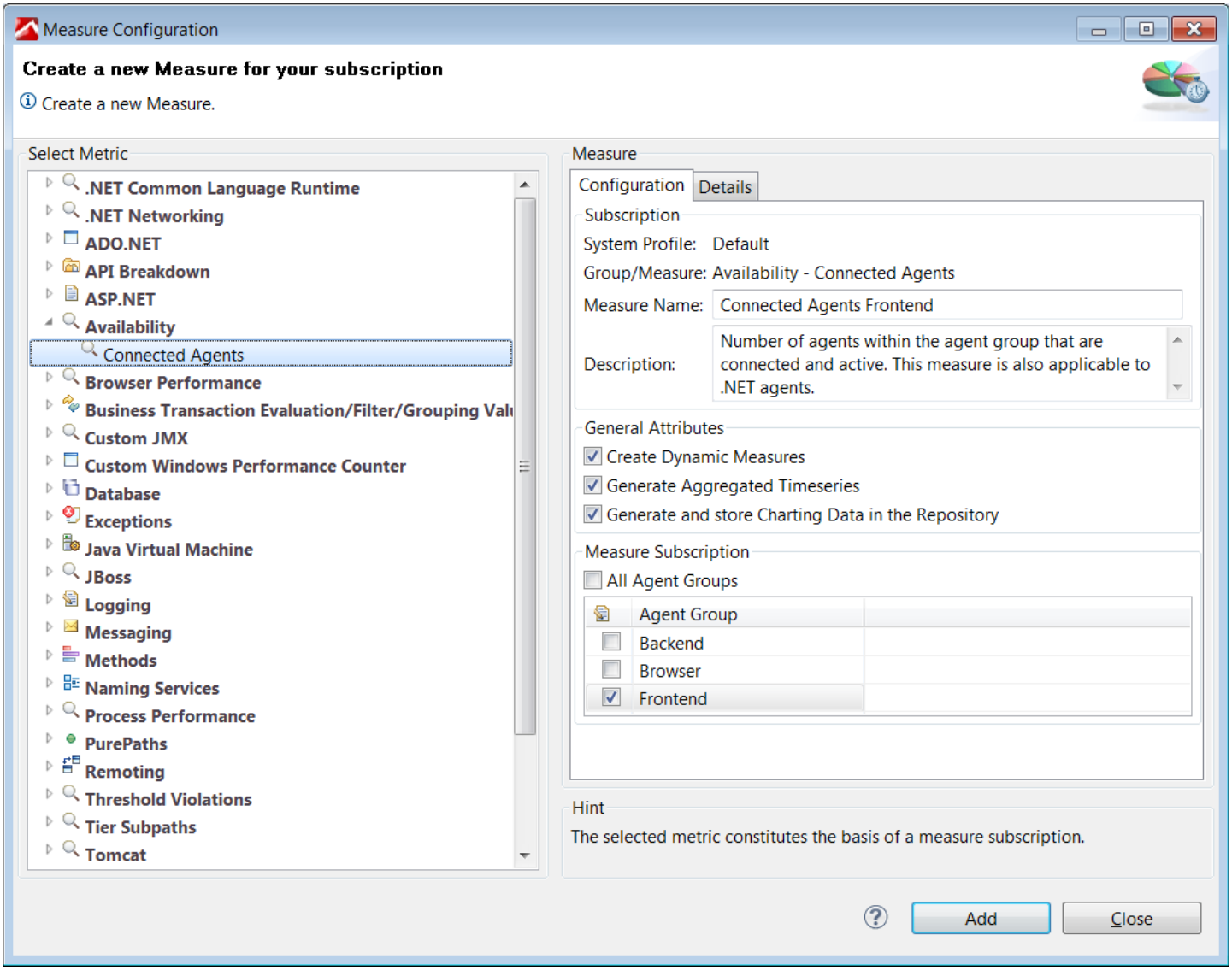




Automatically capture PurePath on exception

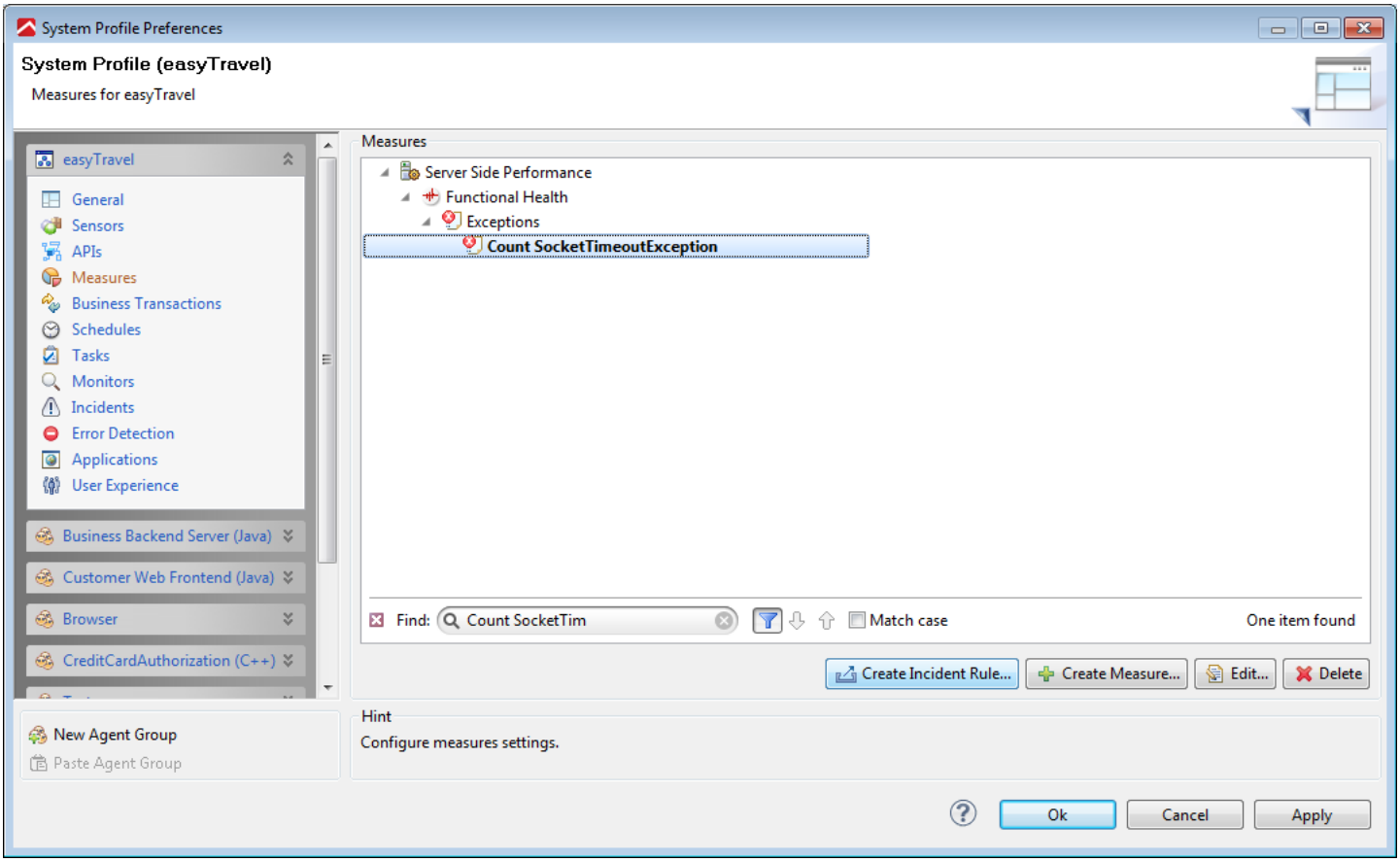
It's not always possible to check exceptions immediately in production or load test environments. It's more efficient to automatically store a transaction. To do this follow these steps:

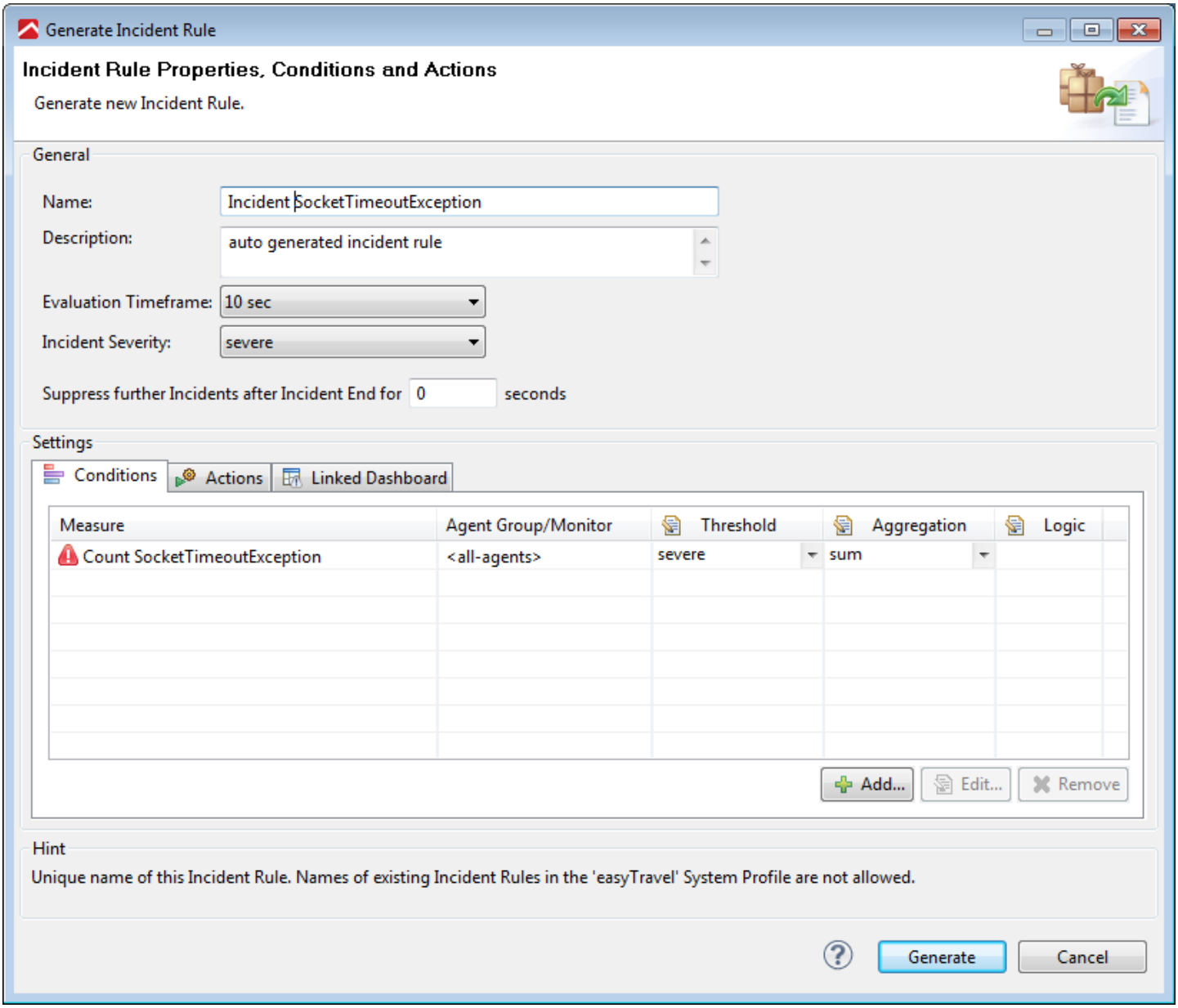
1. Subscribe a Measure and define an incident from the **Exceptions** dashlet.
2. Right-click on an exception and subscribe a Measure for it (give it a meaningful name).
3. Set the **Severe** threshold to one. You must use a running application to do this - either your own or easyTravel.



To create an Incident Rule, follow these steps:

1. Click **System Profile > Measures** and look for your new Measure.
2. If you set the threshold, click the **Create Incident Rule** button.
3. To enable the Incident Rule, edit the Measure and set a threshold.


System Profile Preferences
System Profile (easyTravel)
Measures for easyTravel
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Business Backend Server (Java)
Customer Web Frontend (Java)
Browser
CreditCardAuthorization (C++)
New Agent Group
Paste Agent Group
Measures
Server Side Performance
Functional Health
Exceptions
Count SocketTimeoutException
Find: Count SocketTim
Match case
One item found
Create Incident Rule...
Create Measure...
Edit...
Delete
Hint
Configure measures settings.
Ok
Cancel
Apply



Click the **Actions** tab to configure additional actions for an incident, like an email notification.

When an exception occurs, you receive an incident and you can drill down to the PurePaths. If the exception occurs multiple times, you receive a single incident with multiple PurePaths attached to it.

Conclusion

Use AppMon to easily identify where and why exceptions happen.

Analyzing a slow transaction

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is a slow transaction?

Goal of this walkthrough

Requirements

Detailed steps

Related pages

Memory analysis

Analyzing out of memory errors

Analyze exceptions and log messages

What is a slow transaction?

Slow transactions are user requests that do not respond in an appropriate amount of time. A transaction is slow if:

- A single transaction is more than 20% slower than others of the same type.
- A web requests takes longer than one second to complete.

Goal of this walkthrough

The goal of this walkthrough is to learn how to diagnose the root cause of a slow transaction.

Scenario

Install the easyTravel demo application and follow these steps:

1. Run the Production / Standard scenario.
2. Switch on the SlowAuthentication Problem Pattern.
3. Switch off generic load. Move the slider to the left.
4. Open the customer frontend, and log in as a user, such as maria/maria and then as monica/monica.

Import the attached session to perform this walkthrough with the recorded session.

Requirements

- AppMon Client
- Imported attached session

Detailed steps

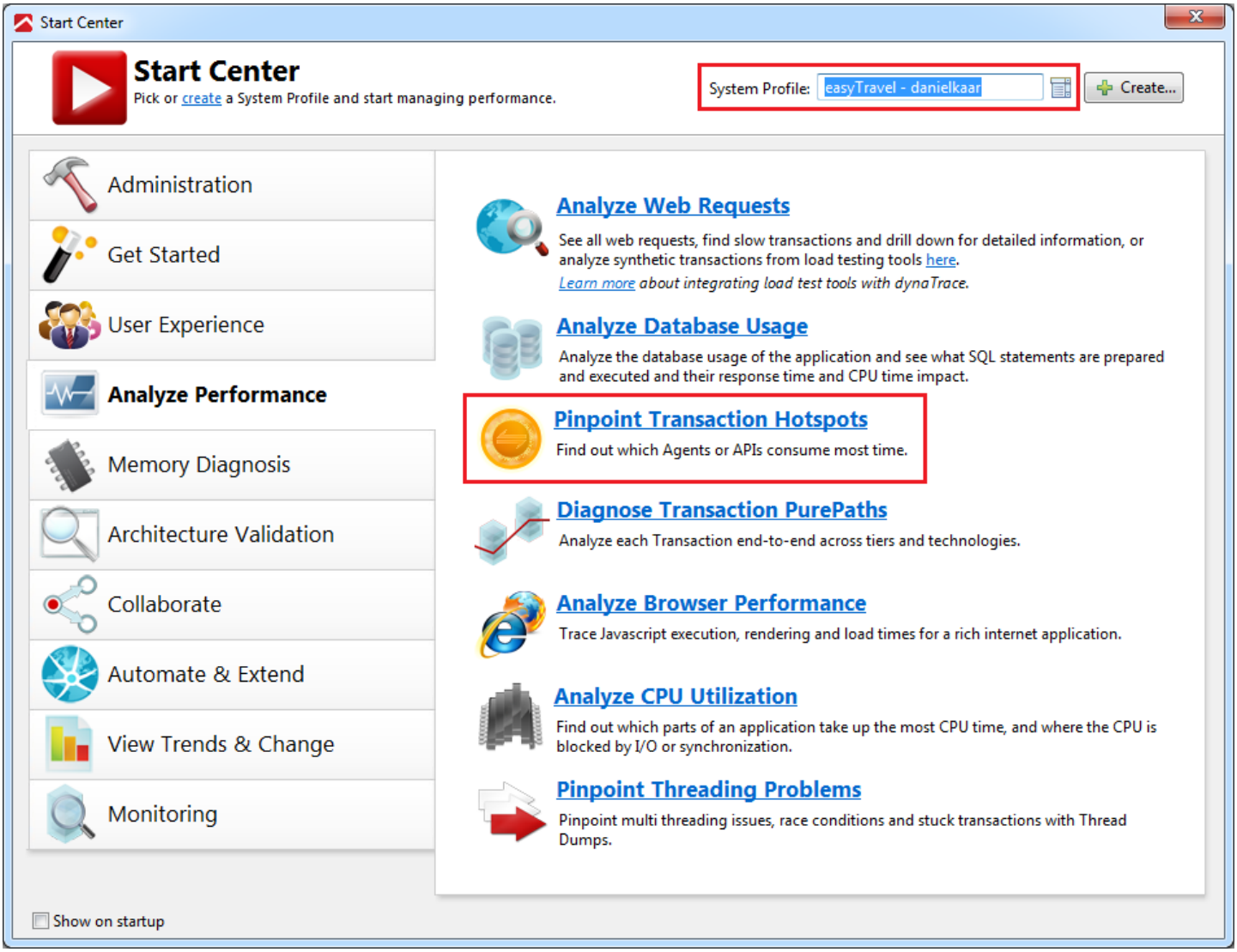
Response time HotSpots

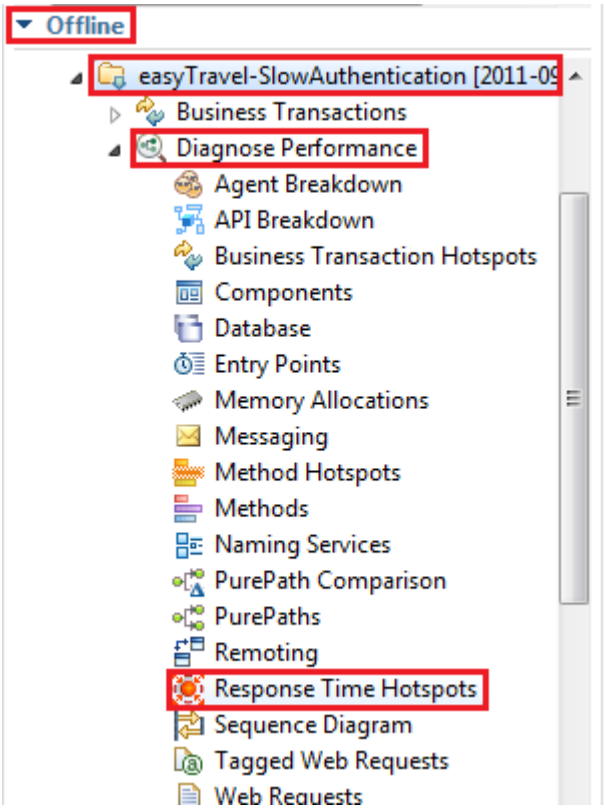
If you are running the scenario on your own, follow these steps:

1. Open the **Start Center**.
2. Go to **Analyze Performance**.
3. Select the **Pinpoint Transaction Hotspot**. The **Response Time Hotspots** dashlet opens.

If you are using the recorded session, follow these steps:

1. Go to the Cockpit.
2. Click the **Diagnose Performance** node in the recorded session. The **Response Time Hotspots** dashlet opens.

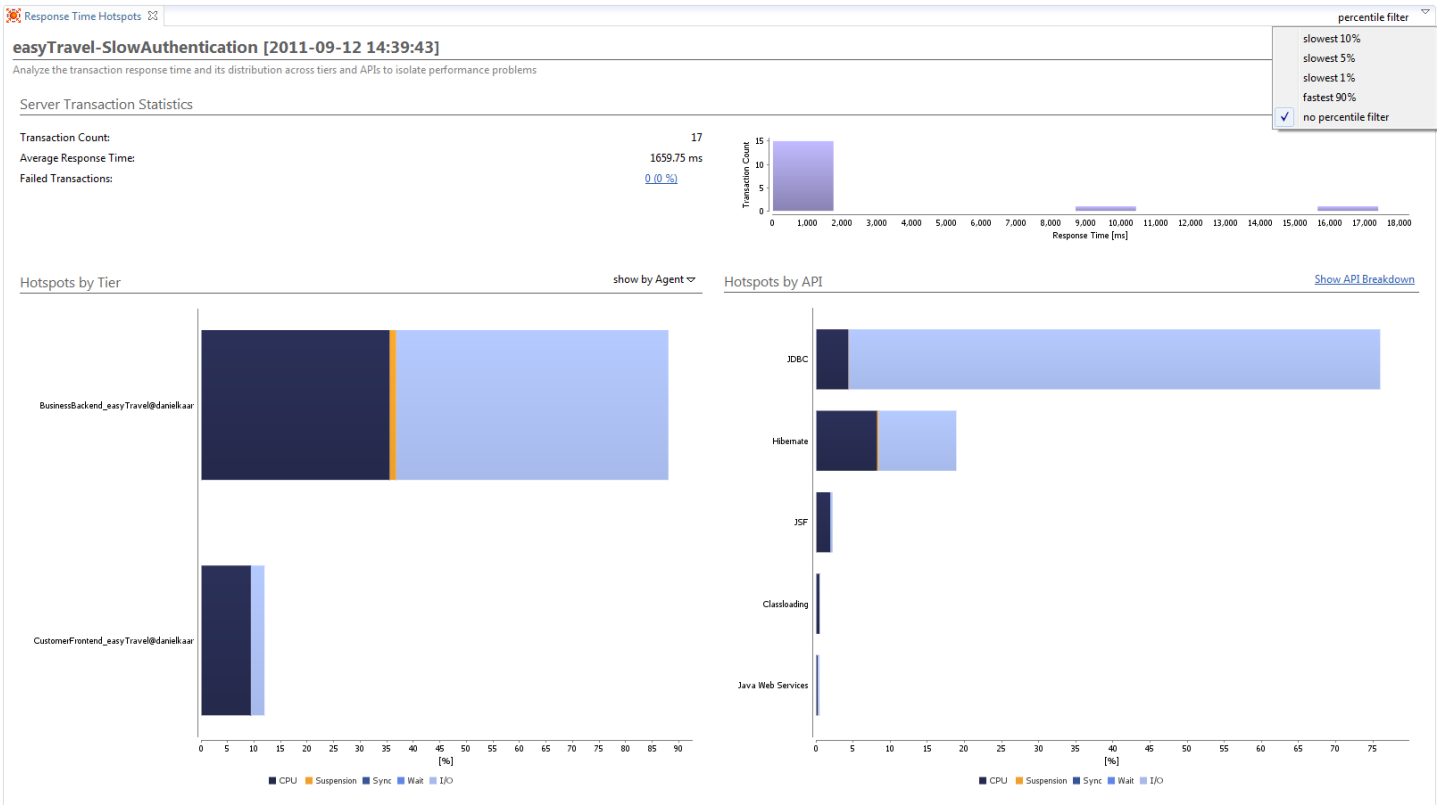

Start Center
Start Center
Pick or create a System Profile and start managing performance.
System Profile: easyTravel - danielkaar
Create...
Administration
Get Started
User Experience
Analyze Performance
Memory Diagnosis
Architecture Validation
Collaborate
Automate & Extend
View Trends & Change
Monitoring
Analyze Web Requests
See all web requests, find slow transactions and drill down for detailed information, or analyze synthetic transactions from load testing tools here.
Learn more about integrating load test tools with dynaTrace.
Analyze Database Usage
Analyze the database usage of the application and see what SQL statements are prepared and executed and their response time and CPU time impact.
Pinpoint Transaction Hotspots
Find out which Agents or APIs consume most time.
Diagnose Transaction PurePaths
Analyze each Transaction end-to-end across tiers and technologies.
Analyze Browser Performance
Trace Javascript execution, rendering and load times for a rich internet application.
Analyze CPU Utilization
Find out which parts of an application take up the most CPU time, and where the CPU is blocked by I/O or synchronization.
Pinpoint Threading Problems
Pinpoint multi threading issues, race conditions and stuck transactions with Thread Dumps.
Show on startup



With just a click, you can identify which tiers and APIs are slow and contribute to the response time:

- The backend uses Approx. 90% of the time.
- The JDBC uses 75% of the API.
- Hibernate uses approximately 20%.

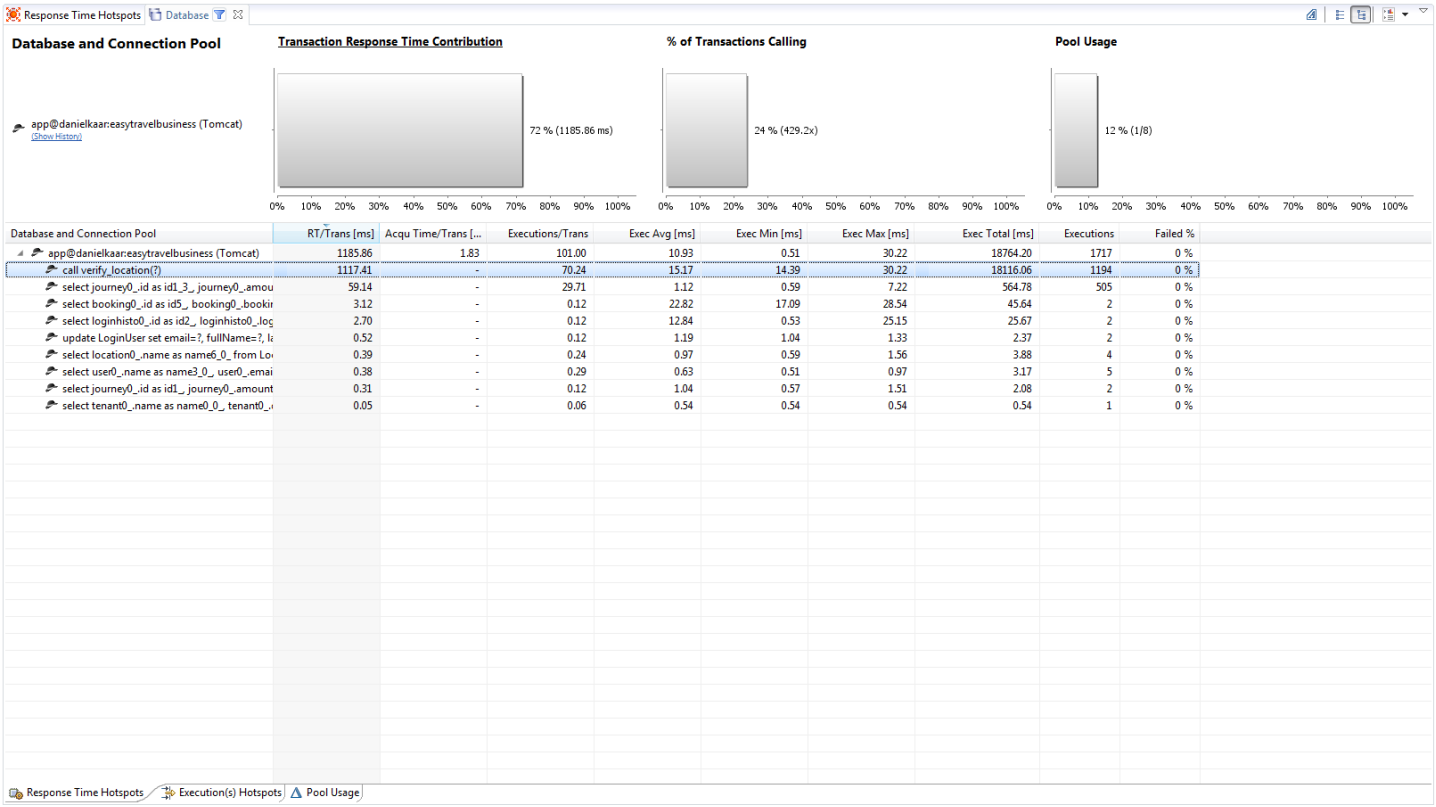
The chart provides even more detail when you set the percentile filter on the top right corner of the dashlet to the slowest 5%.



Response Time Hotspots

One click to root cause

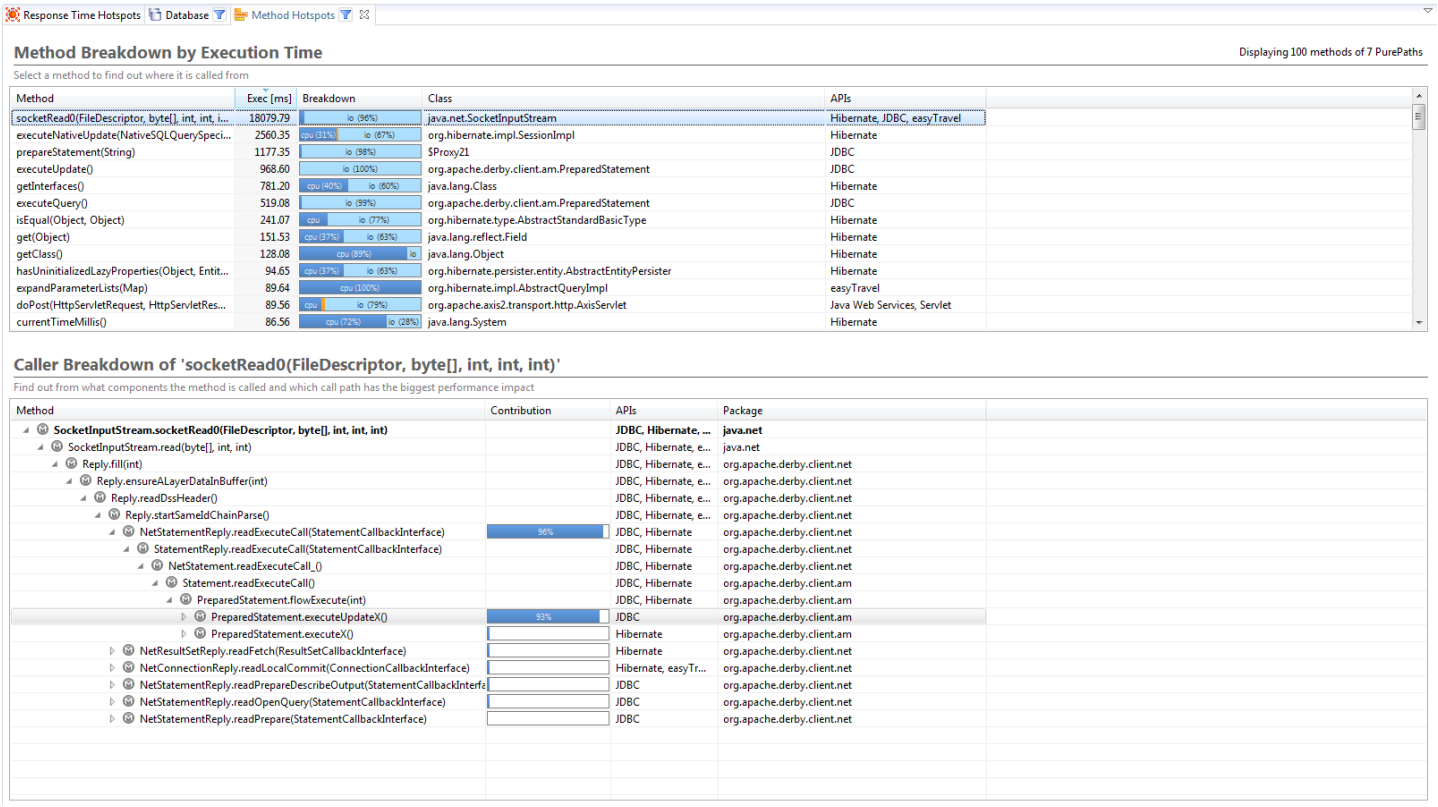
The JDBC I/O time is the most significant contributor. For further analysis, click on this area in the chart and the Database dashlet opens. You can see a detailed list of all the executed SQL statements. Notice that 70 calls of verify_location are responsible for approximately 70% of the response time.

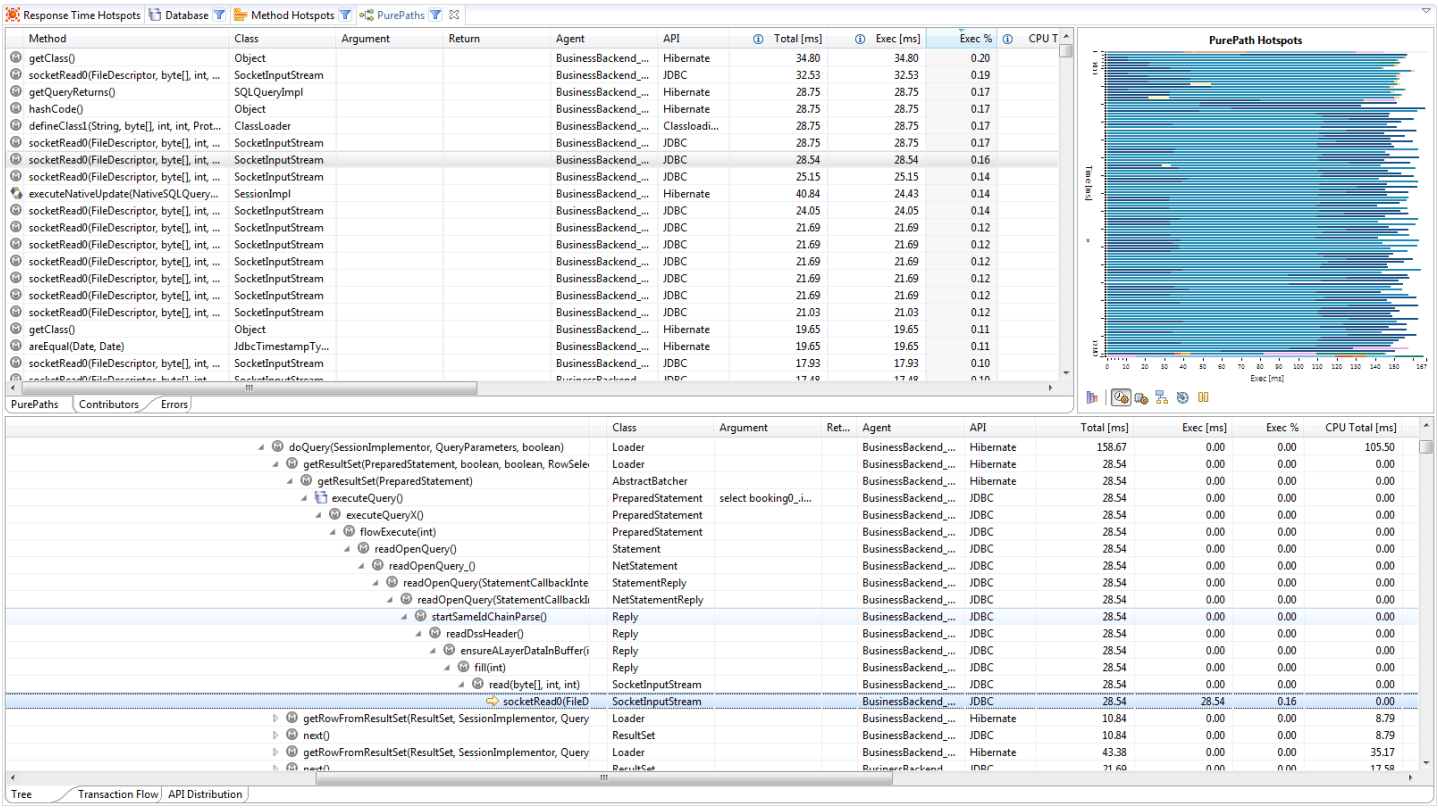


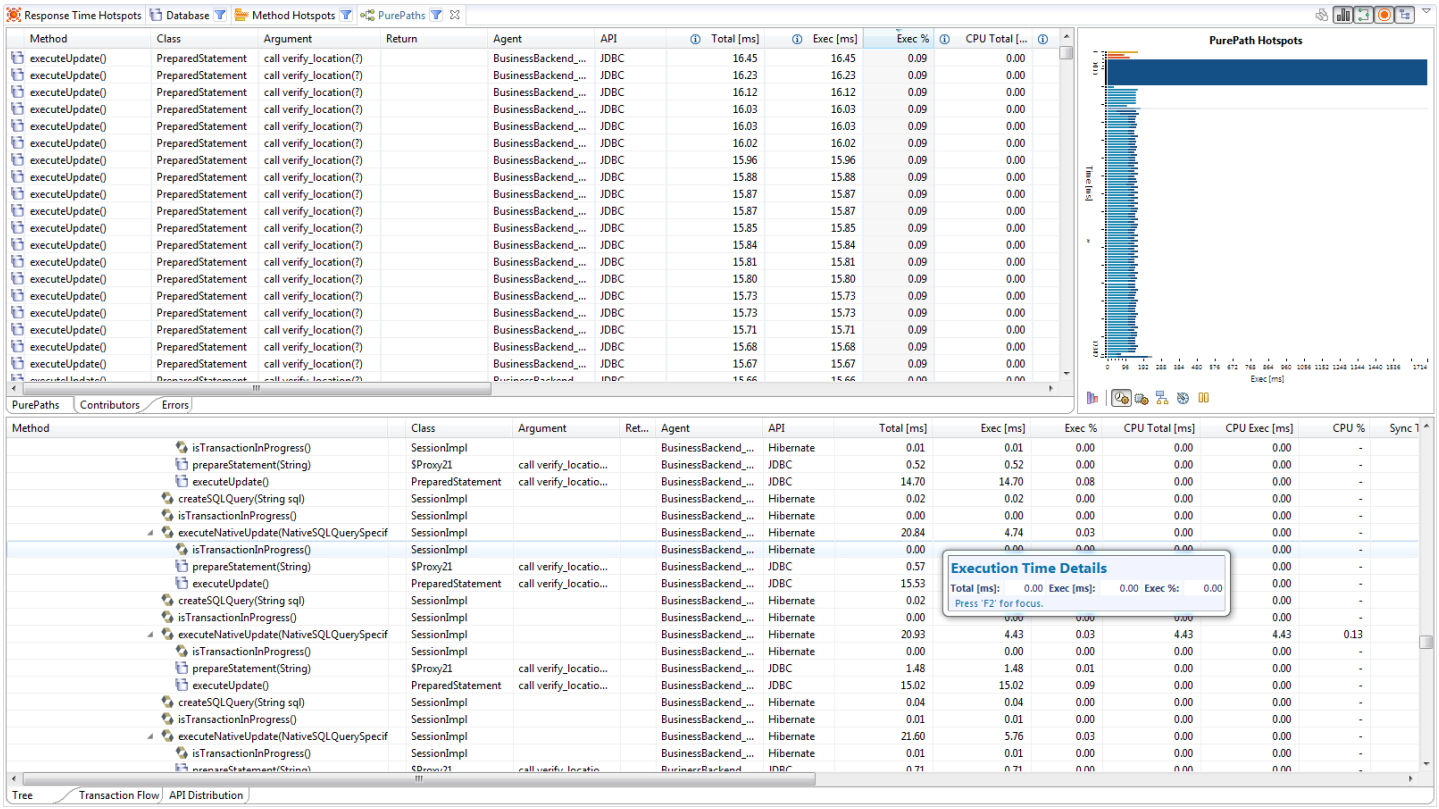
Database Dashlet

Go back to the **Response Time Hotspots** dashlet and follow these steps:

1. Click on the **Response Time Hotspots** dashlet to drill down from the business backend. The **Method Hotspots** dashlet opens.
2. Select the method `socketRead0(...),` which consumes a vast amount of time. The detailed caller breakdown appears below. It shows, that 93% originate from `PreparedStatement.ExecuteUpdateX()`.
3. Drill down to the PurePaths.
4. Look at the **Contributors** tab and the PurePath Hotspots. Notice that you have many single method calls, as opposed to one big time-consuming hotspot, of `socketRead0(...)`. Each last just about 20ms.
5. Look at the tree below and note that `socketRead0` is called by `executeQuery` of `PreparedStatement`.
6. Switch off the Auto Sensors. You can identify the structure of the problem pattern: Many single SQL statements.


Response Time Hotspots
Database
Method Hotspots
Method Breakdown by Execution Time
Displaying 100 methods of 7 PurePaths
Select a method to find out where it is called from
Method
Exec [ms]
Breakdown
Class
APIs
socketRead0(FileDescriptor, byte[], int, int, i... 18079.79 io (96%) java.net.SocketInputStream Hibernate, JDBC, easyTravel
executeNativeUpdate(NativeSQLQuerySpeci... 2560.35 cpu (31%) io (67%) org.hibernate.impl.SessionImpl Hibernate
prepareStatement(String) 1177.35 io (98%) $Proxy21 JDBC
executeUpdate() 968.60 io (100%) org.apache.derby.client.am.PreparedStatement JDBC
getInterfaces() 781.20 cpu (40%) io (60%) java.lang.Class Hibernate
executeQuery() 519.08 io (99%) org.apache.derby.client.am.PreparedStatement JDBC
isEqual(Object, Object) 241.07 cpu io (77%) org.hibernate.type.AbstractStandardBasicType Hibernate
get(Object) 151.53 cpu (37%) io (63%) java.lang.reflect.Field Hibernate
getClass() 128.08 cpu (89%) io java.lang.Object Hibernate
hasUninitializedLazyProperties(Object, Entit... 94.65 cpu (37%) io (63%) org.hibernate.persister.entity.AbstractEntityPersister Hibernate
expandParameterLists(Map) 89.64 cpu (100%) org.hibernate.impl.AbstractQueryImpl easyTravel
doPost(HttpServletRequest, HttpServletRes... 89.56 cpu io (79%) org.apache.axis2.transport.http.AxisServlet Java Web Services, Servlet
currentTimeMillis() 86.56 cpu (72%) io (28%) java.lang.System Hibernate
Caller Breakdown of 'socketRead0(FileDescriptor, byte[], int, int, int)'
Find out from what components the method is called and which call path has the biggest performance impact
Method
Contribution
APIs
Package
SocketInputStream.socketRead0(FileDescriptor, byte[], int, int, int) JDBC, Hibernate, ... java.net
SocketInputStream.read(byte[], int, int) JDBC, Hibernate, e... java.net
Reply.fill(int) JDBC, Hibernate, e... org.apache.derby.client.net
Reply.ensureALayerDataInBuffer(int) JDBC, Hibernate, e... org.apache.derby.client.net
Reply.readDssHeader() JDBC, Hibernate, e... org.apache.derby.client.net
Reply.startSameIdChainParse() JDBC, Hibernate, e... org.apache.derby.client.net
NetStatementReply.readExecuteCall(StatementCallbackInterface) 96% JDBC, Hibernate org.apache.derby.client.net
StatementReply.readExecuteCall(StatementCallbackInterface) JDBC, Hibernate org.apache.derby.client.net
NetStatement.readExecuteCall_() JDBC, Hibernate org.apache.derby.client.net
Statement.readExecuteCall() JDBC, Hibernate org.apache.derby.client.am
PreparedStatement.flowExecute(int) JDBC, Hibernate org.apache.derby.client.am
PreparedStatement.executeUpdateX() 93% JDBC org.apache.derby.client.am
PreparedStatement.executeX() Hibernate org.apache.derby.client.am
NetResultSetReply.readFetch(ResultSetCallbackInterface) Hibernate org.apache.derby.client.net
NetConnectionReply.readLocalCommit(ConnectionCallbackInterface) Hibernate, easyTr... org.apache.derby.client.net
NetStatementReply.readPrepareDescribeOutput(StatementCallbackInterfa JDBC org.apache.derby.client.net
NetStatementReply.readOpenQuery(StatementCallbackInterface) JDBC org.apache.derby.client.net
NetStatementReply.readPrepare(StatementCallbackInterface) JDBC org.apache.derby.client.net


Response Time Hotspots
Database
Method Hotspots
PurePaths
Method
Class
Argument
Return
Agent
API
Total [ms]
Exec [ms]
Exec %
CPU T
getClass() Object BusinessBackend_... Hibernate 34.80 34.80 0.20
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 32.53 32.53 0.19
getQueryReturns() SQLQueryImpl BusinessBackend_... Hibernate 28.75 28.75 0.17
hashCode() Object BusinessBackend_... Hibernate 28.75 28.75 0.17
defineClass1(String, byte[], int, int, Prot... ClassLoader BusinessBackend_... Classloadi... 28.75 28.75 0.17
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 28.75 28.75 0.17
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 28.54 28.54 0.16
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 25.15 25.15 0.14
executeNativeUpdate(NativeSQLQuery... SessionImpl BusinessBackend_... Hibernate 40.84 24.43 0.14
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 24.05 24.05 0.14
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 21.69 21.69 0.12
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 21.69 21.69 0.12
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 21.69 21.69 0.12
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 21.69 21.69 0.12
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 21.69 21.69 0.12
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 21.03 21.03 0.12
getClass() Object BusinessBackend_... Hibernate 19.65 19.65 0.11
areEqual(Date, Date) JdbcTimestampTy... BusinessBackend_... Hibernate 19.65 19.65 0.11
socketRead0(FileDescriptor, byte[], int, ... SocketInputStream BusinessBackend_... JDBC 17.93 17.93 0.10
PurePaths
Contributors
Errors
PurePath Hotspots
Time [ms]
Exec [ms]
Class
Argument
Ret...
Agent
API
Total [ms]
Exec [ms]
Exec %
CPU Total [ms]
doQuery(SessionImplementor, QueryParameters, boolean) Loader BusinessBackend_... Hibernate 158.67 0.00 0.00 105.50
getResultSet(PreparedStatement, boolean, boolean, RowSele Loader BusinessBackend_... Hibernate 28.54 0.00 0.00 0.00
getResultSet(PreparedStatement) AbstractBatcher BusinessBackend_... Hibernate 28.54 0.00 0.00 0.00
executeQuery() PreparedStatement select booking0_.i... BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
executeQueryX() PreparedStatement BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
flowExecute(int) PreparedStatement BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
readOpenQuery() Statement BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
readOpenQuery_() NetStatement BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
readOpenQuery(StatementCallbackInte StatementReply BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
readOpenQuery(StatementCallbackI NetStatementReply BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
startSameIdChainParse() Reply BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
readDssHeader() Reply BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
ensureALayerDataInBuffer(i Reply BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
fill(int) Reply BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
read(byte[], int, int) SocketInputStream BusinessBackend_... JDBC 28.54 0.00 0.00 0.00
socketRead0(FileD SocketInputStream BusinessBackend_... JDBC 28.54 28.54 0.16 0.00
getRowFromResultSet(ResultSet, SessionImplementor, Query Loader BusinessBackend_... Hibernate 10.84 0.00 0.00 8.79
next() ResultSet BusinessBackend_... JDBC 10.84 0.00 0.00 8.79
getRowFromResultSet(ResultSet, SessionImplementor, Query Loader BusinessBackend_... Hibernate 43.38 0.00 0.00 35.17
Tree
Transaction Flow
API Distribution


Response Time Hotspots
Database
Method Hotspots
PurePaths
Method
Class
Argument
Return
Agent
API
Total [ms]
Exec [ms]
Exec %
CPU Total [...
executeUpdate()
PreparedStatement
call verify_location(?)
BusinessBackend_...
JDBC
PurePath Hotspots
Time [ms]
Exec [ms]
PurePaths
Contributors
Errors
Method
Class
Argument
Ret...
Agent
API
Total [ms]
Exec [ms]
Exec %
CPU Total [ms]
CPU Exec [ms]
CPU %
Sync T
isTransactionInProgress()
prepareStatement(String)
executeUpdate()
createSQLQuery(String sql)
executeNativeUpdate(NativeSQLQuerySpecif
SessionImpl
Hibernate
$Proxy21
call verify_locatio...
PreparedStatement
Execution Time Details
Total [ms]: 0.00 Exec [ms]: 0.00 Exec %: 0.00
Press 'F2' for focus.
Tree
Transaction Flow
API Distribution

Tips and tricks

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Client commandline arguments

FAQ - measures and charting

Glossary - measures and charting

RegEx - using regular expressions

This section provides helpful product usage tips.

- FAQ - Measures and Charting
- Glossary - Measures and Charting
- Client Commandline Arguments
- Handling of Confidential Strings
- RegEx - Using Regular Expressions

FAQ - measures and charting

You can only re-compute measurements or Measures for PurePath data, not for time series.

Can measures be re-computed?

If you store a session from continuous session recording, then there is (only) a one-minute resolution for data older than last one hour, as AppMon fetches it from the Performance Warehouse.

If you store your session via start or stop recording, then you will have a 10-second resolution in the stored session.

Will the measure data get aggregated based on the age of the session data?

Common time frame	Resolution	Time frame boundaries for resolution
5 Minutes	10 seconds	[0 … 5:10]
15 Minutes	30 seconds	[5:10 … 15:30]
30 Minutes	1 minute	[15:30 … 31:00]
1 hour	5 minutes	[31:00 … 2:35:00]
6 hours	15 minutes	[2:35:00 … 7:45:00]
24 hours	1 hour	[7:45:00 … 31:00:00]
72 hours, 7 days	6 hours	[31:00:00 … 7 days 18:00:00]
30 days	1 day	[7 days 18:00:00 … 31 days]
-	1 week	[31 days … 7 months 7 days]

365 days	1 month	[7 months 7 days … 2 years 6 months 20 days]
-	6 months	[2 years 6 months 20 days … 15 years 3 months 15 days]
-	1 year	>= 15 years 3 months 15 days

The following table shows the mapping from common time frames to the resolution of the chart data. The time frame boundaries show the exact interval in which a certain resolution is used. Time format: hh:mm:ss or mm:ss (h: hours, m: minutes, s: seconds).

Resolution mapping for stacked charts

Common time frame	Resolution	Time frame boundaries for resolution
5 and 15 minutes	10 seconds	[0 … 16:20]
30 minutes	30 seconds	[16:10 … 48:30]
1 hour	1 minute	[48:30 … 1:37:00]
6 hours	5 minutes	[1:37:00 … 8:05:00]
24 hours (1 day)	15 minutes	[8:05:00 … 24:15:00]
72 hours (3 days)	1 hour	[24:15:00 … 4 days 1:00:00]
7 days	6 hours	[4 days 1:00:00 … 24 days 6:00:00]
30 days	1 day	[24 days 6:00:00 … 3 months 7 days]
365 days	1 week	[3 months 7 days … 1 year 10 months 14 days]
-	1 month	[1 year 10 months 14 days … 7 years 11 months 25 days]

	1 month	[1 year 10 months 11 days … 1 year 11 months 25 days]
-	6 months	[7 years 11 months 25 days … 47 years 10 months 5 days]
-	1 year	>= 47 years 10 months 5 days

The following table shows the mapping from common time frames to chart data resolution. The time frame boundaries show the exact interval in which a certain resolution is used. Timeformat: hh:mm:ss or mm:ss (h: hours, m: minutes, s: seconds). The fixed size for a month (= 30 days) and a year (= 365 days).

Resolution mapping for non-stacked charts

The resolution is set automatically, based on the time frame and chart type you select. The following tables show the mappings:

How is the resolution related to the selected time frame?

You can track threshold changes only if you connect a Performance Warehouse. Typically, after an AppMon Server restart, all thresholds, except the current ones, are lost.

Why are the historical thresholds lost after the server is restarted?

This occurs if you delete a Measure and create a new Measure with the same name, metric, and metric group (all within one System Profile).

Why is there old data for a newly created measure?

We recommend that you use time frames of >= 1h with percentile aggregations, because AppMon calculates percentiles per hour (resolution of 1 hour).

Why is data missing after I switch to percentile aggregation?

The Percentile chart is the only type to use a special aggregation. When you switch back to another chart type, the series aggregation gets reset to average.

Why does the configured series aggregation get lost when I switch to a percentile chart?

Why does the configured series aggregation get lost when I switch to a percentile chart?

Data that you retrieve from the Performance Warehouse is available at the highest resolution of one minute.

Why doesn't changing the resolution to 10 seconds have an effect on the data before server restart?


You remove the Measure from System Profile and reconstruct it from Performance Warehouse.


The Measure no longer exists in System Profile.


You configure the Measure from a stored or offline Session

Overlays:


Measure Configuration


Dynamic Measure


Business Transaction


Monitor

Icons:

What is the meaning of the different icons in the chart measure table?

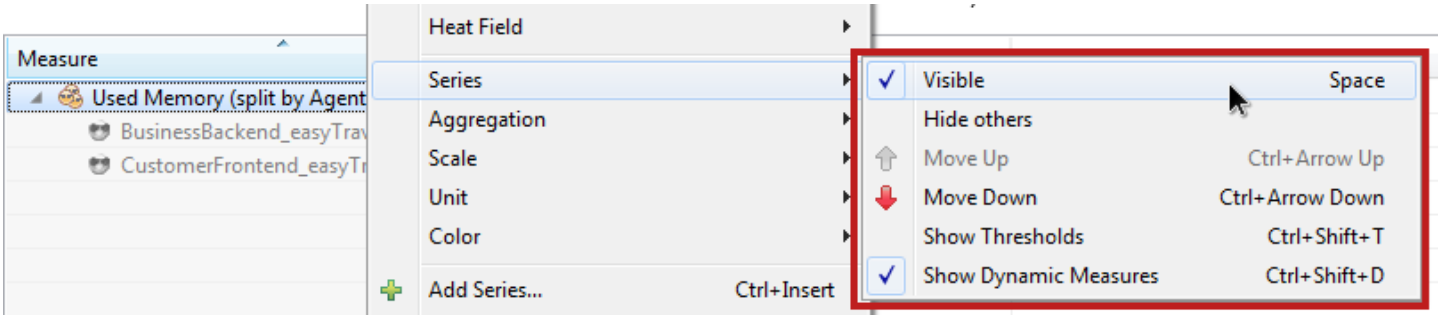
> **Note**
>
> All Measures and splittings in stored and offline sessions appear in the Measure picker.

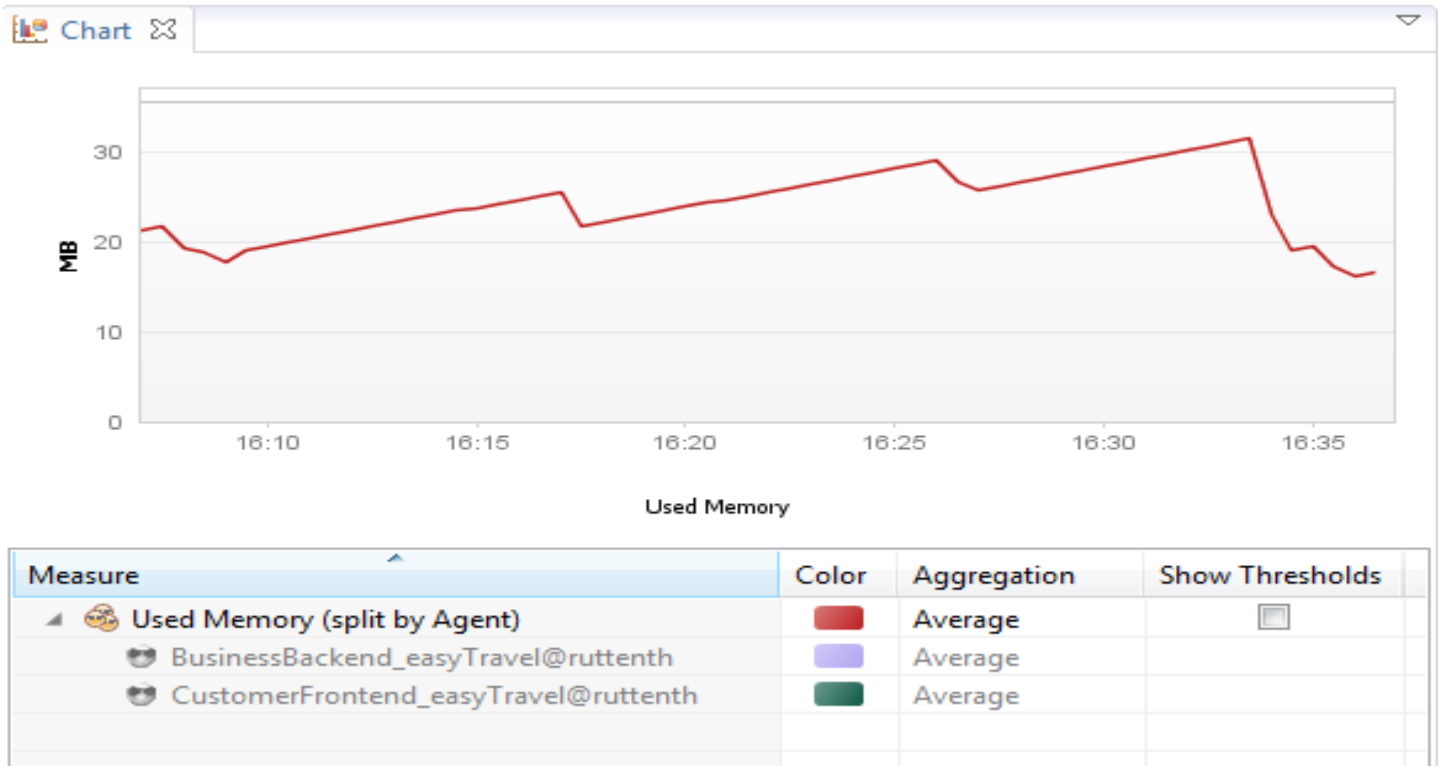
- AppMon reconstructs splittings from the Performance Warehouse.

- AppMon Server restarts and the live session is empty.
- The scope of splitting is outside the live session time frame of one hour.

The Measure picker shows splittings from the live session only. Due to performance reasons, AppMon does not show splittings in the Performance Warehouse in the Measure picker. In the following scenarios,

Why does the chart show more splittings than the measure picker?





> **Toggle visibility**
>
> To toggle the visibility, click the **Space bar** for selected Measures or splittings in the table.

This happens for chart series that have a visible flag set to false. By default, AppMon hides Measures that you add to charts, and only shows the Measure splittings.

Why doesn't the displayed chart reflect the amount of measures shown in the table?

Dynamic measures are measures that can either appear or not appear on a chart based on whether the split for the measure creates data given the chart's selected time frame. A Measure is split by Agent, application or a business transaction criterion. If a splitting value occurs for the Measure within the selected time frame, AppMon creates an entry (a line in a chart table or a series in a chart) for this Measure in the chart.

Why do measures appear or disappear dynamically in charts?

- Via a chart table:
  1. Open the Measure context menu.
  2. Select **Color**.
  3. Select a color or select **Random Color**, **Related Color**, or **Choose Color**.
- Via a chart:
  1. Click on the line, area, bar, etc. that represents the Measure, which is also selected in the chart table.
  2. Open the context menu.
  3. Select **Color**.
  4. Select a color or select **Random Color**, **Related Color**, or **Choose Color**.

This may be done in one of the following ways:

How can I change the color of a series?

If you disconnect the Performance Warehouse, the only measurements available in the server live session are available to charts. By default, the time series center is configured to cache measurements up to one hour.

Why do older measures and incidents disappear if I disconnect the Performance Warehouse?

By default, bar charts use a lower resolution than line charts and, therefore, values may differ.

Why do values seem to differ between line and bar charts?

- **Auto-Scaling**: AppMon enables automatic scaling by default. The Measure types that you add, apply different scaling strategies. If you add a Measure with a valid threshold setting and activate

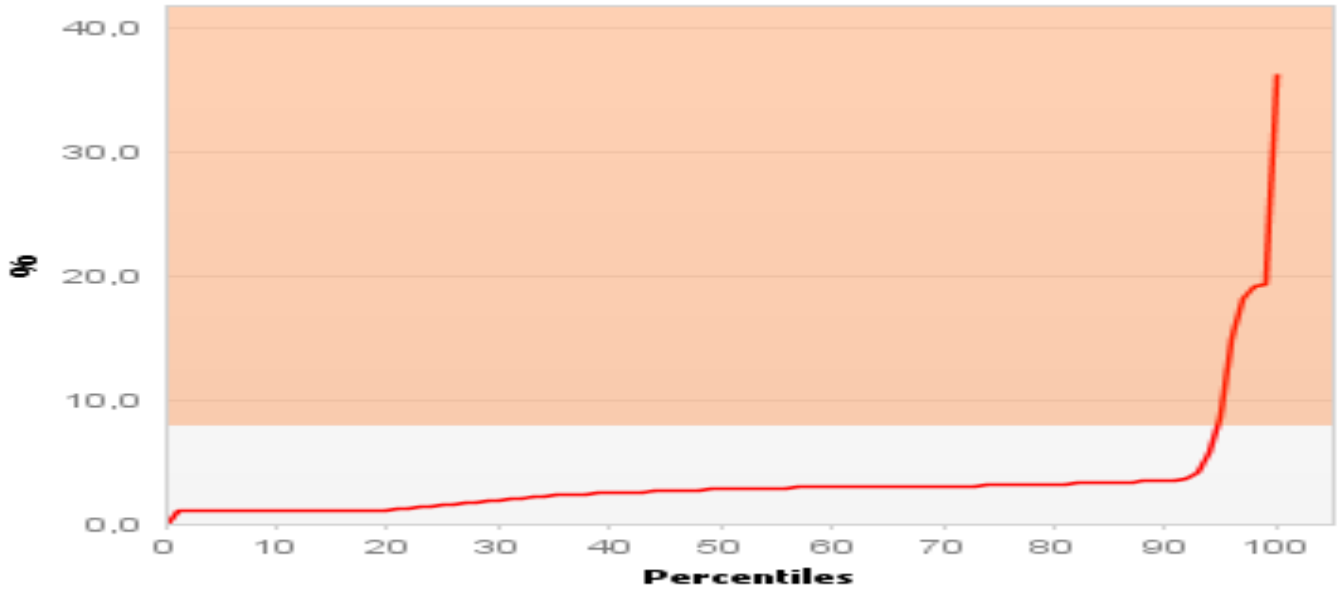
Show Thresholds for a series, the scaling strategy attempts to scale the chart to include a portion (~20%) of threshold.

- **Y-Axis scaling in Meter Charts**: Meter charts scale their y-axis based on the currently shown value, e.g. value 512ms > y-axis: 0-1000. If you want to apply the same axis to all meter charts in a **Chart** dashlet, click **Dashlet Properties > Axis > Pin Axis** and specify lower and upper bounds.

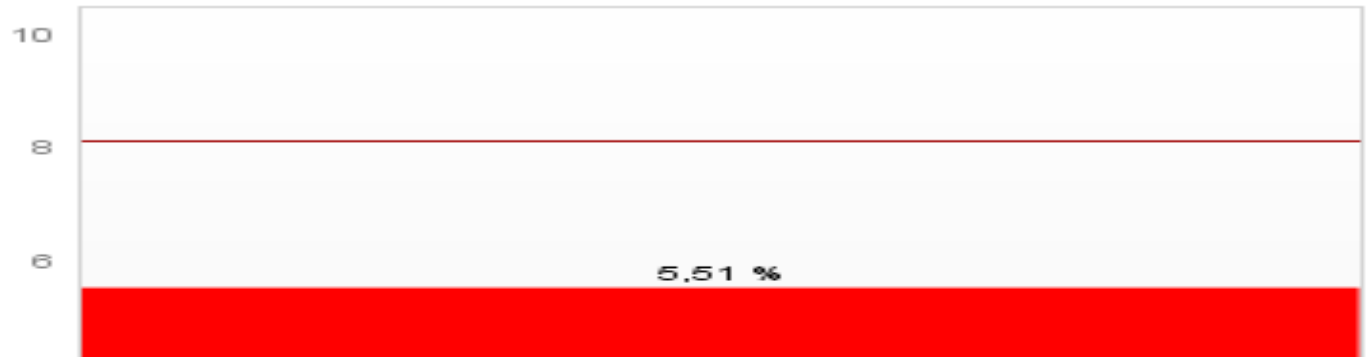
Scaling occurs in the Line and Area chart types. To adapt the scale factor per series, select the Scale submenu in the chart context menu, or click **Dashlet Properties > Series > Override default settings**.

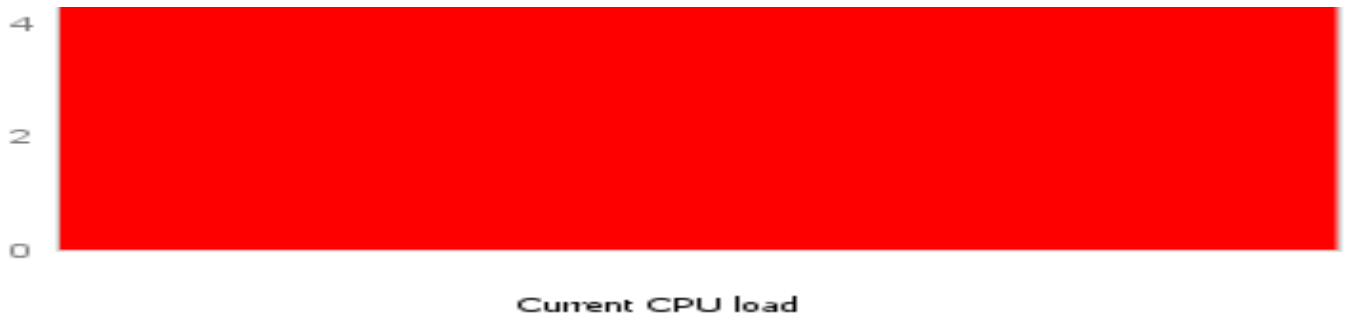
How does scaling work within charts?

- Charted values are evaluated against the threshold that is valid at measurement time.
- Meter charts and Percentile charts visualize the currently active threshold.
- The charted value may be the average or other aggregated value over the specified time frame.
- A Traffic Light reflects the highest severity state that occurs within the specified time frame.
- XY-Charts show the threshold, and its alterations over time, as a painted area or dashed line.

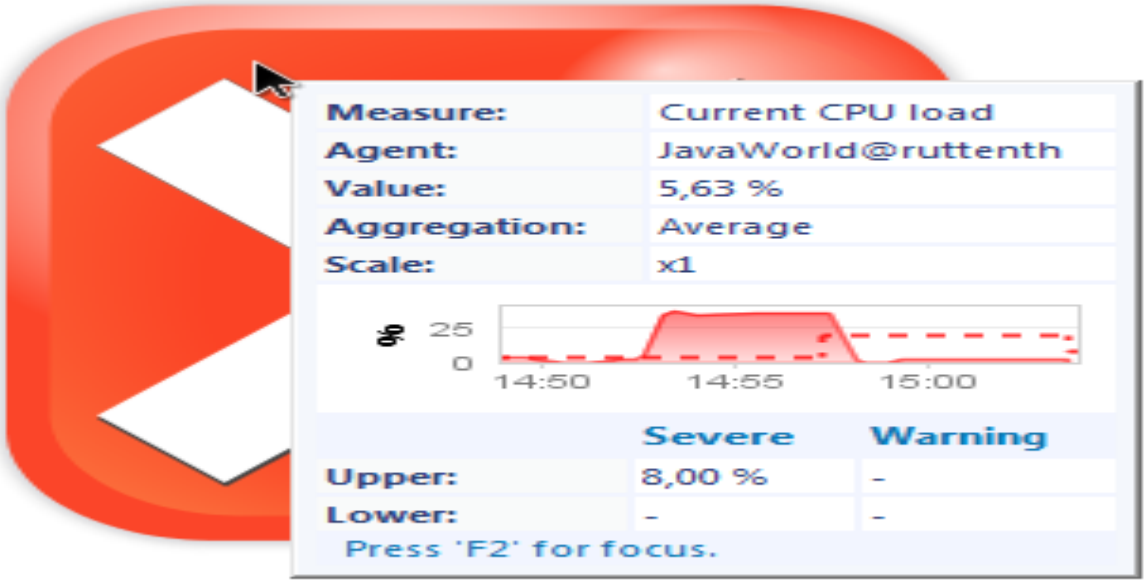


Current CPU load

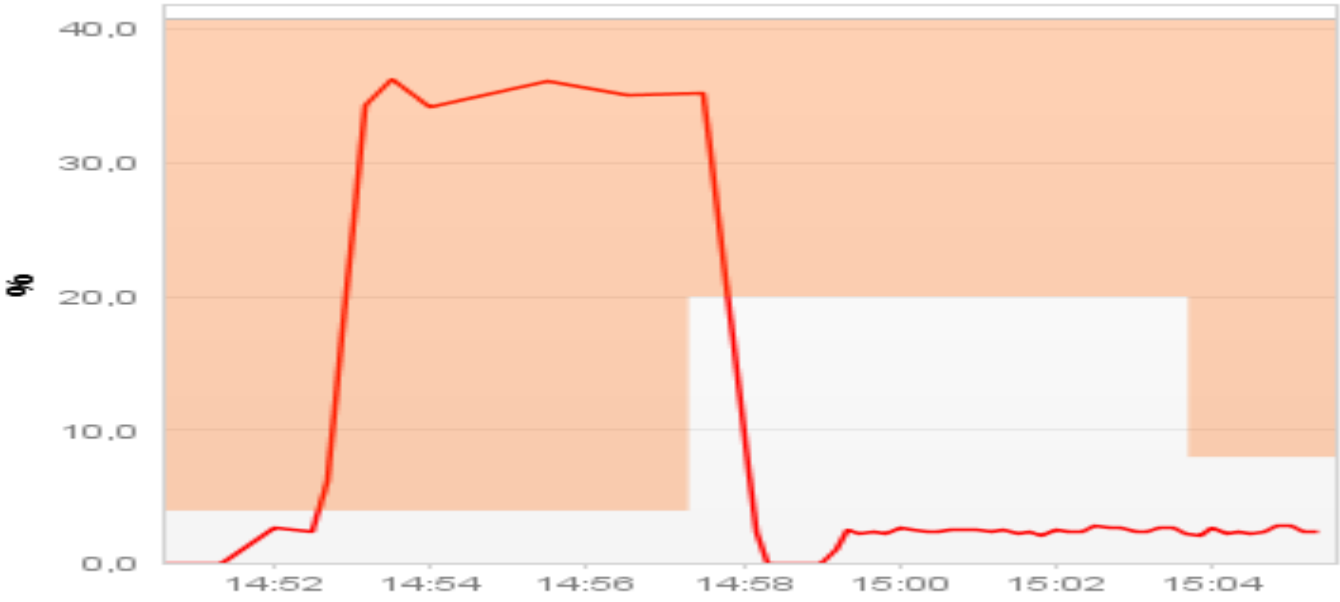




Current CPU load



Current CPU load



Current CPU load

How do you evaluate thresholds?

- System Profile preferences:
  1. In the Cockpit, select the **System Profile** where the Measure to edit is defined.
  2. Double-click the **System Profile**, or click **Edit System Profile** on the context menu.
  3. Click **Measures** on the left side.
  4. Select a Measure.
  5. Double-click to open **Measure Properties**, or click **Edit** on the context menu.
  6. Edit thresholds in the Thresholds section.
- If the Measure is already in the chart, follow these steps:
  1. Select the Measure in the chart table and click **Edit Measure** on the context menu.
  2. Edit thresholds in the Thresholds section.

Thresholds are defined for Measures that exist in System Profiles in the following ways:

How do I configure a threshold?

Glossary - measures and charting

On this page

Unit

Aggregation

Time frames

Resolution

Measure picker

Unit

- Automatically set to **[1]** for chart series with configured count aggregation
- **B, kB, MB, GB** for byte Measures
- **ns, us, ms, s, min, h** for time Measures

Aggregation

Events captured for a Measure on the AppMon Server are aggregated into a measurement per 10 sec. interval.

- **Average**: Average of all incoming event values per interval.
- **Minimum**: Minimum encountered value per interval.
- **Maximum**: Maximum encountered value per interval.
- **Sum**: Sum of event values per interval.
- **Count**: Event count per interval.

The interpretation of an aggregation is based on the chosen chart type. For non-XY-Charts like the Meter chart or Pie chart, you can choose the following aggregations:

- **Last**: Last Measure value
- **Percentile** Choose a percentage value from {25, 50, 75, 90, 95, 99}

Tip

The aggregation interval for plotted values is based on the selected chart resolution. For more information, see How is the resolution related to the selected time frame?.

Time frames

- **Auto**: Measures from a live System Profile resolve to the last 15min time frame. For Measures from a stored or offline session, the time frame should automatically adapt to incorporate captured data (may require refresh).
- **Last Value**: Retrieves the last captured measurement for a Measure and includes data in Performance Warehouse, if connected.
- **Last [X]**: Last x minutes/hours/days/etc.
- **Last Year, Last Week, Last Month, Yesterday**: Time span of the last calendar year/week/month/day.
- **This Year, This Week, This Month, Today**: Time span from beginning of year/week/month/day till now.

Time stamps source

AppMon Server

Resolution

The resolution dictates how a chart represents detailed data. The maximum resolution for live data is 10 seconds. Performance Warehouse data is available at a maximum resolution of 1 minute.

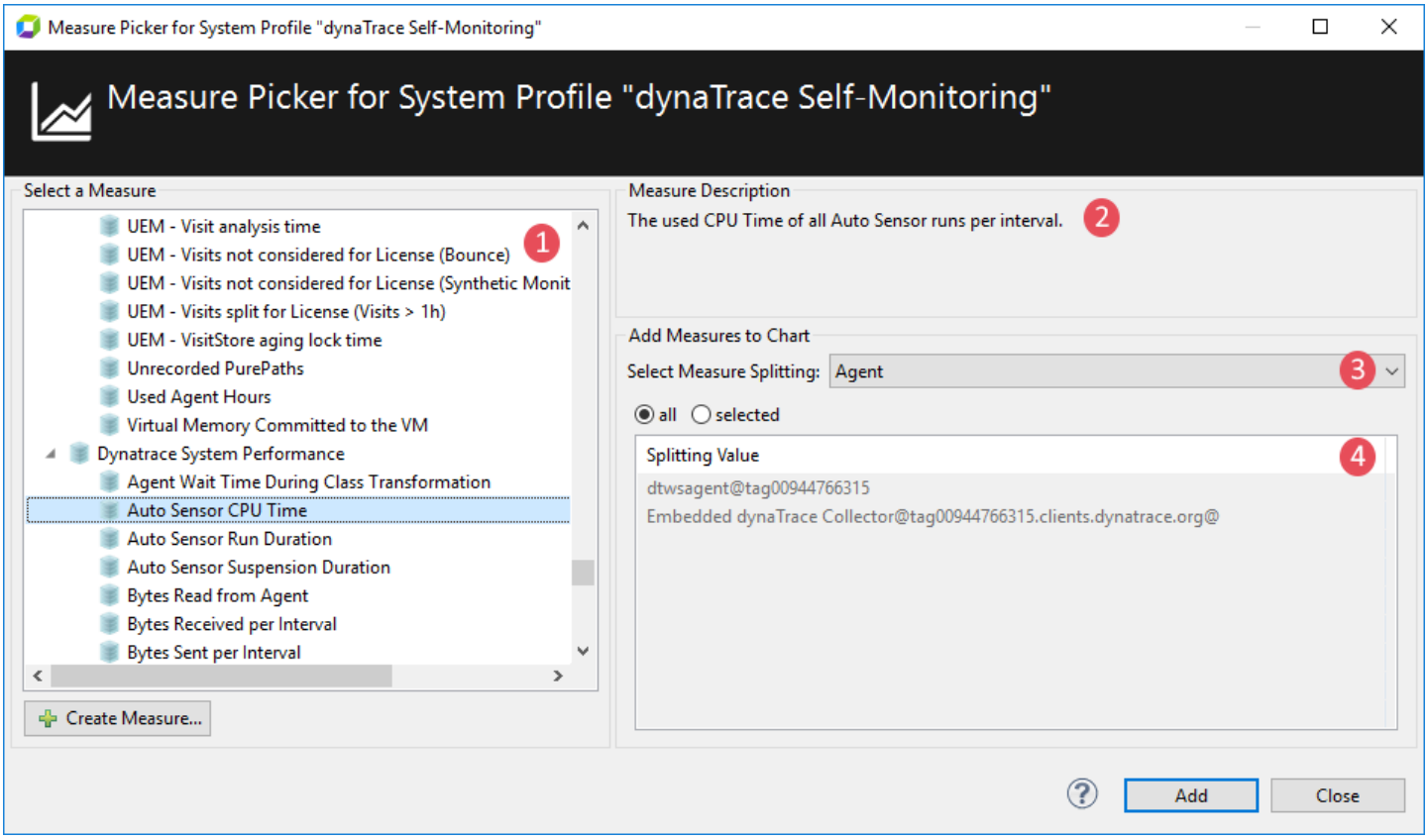
Automatic chart resolution

Even though the resolution may be set manually in chart context menu, the chart can override it to avoid flooding the plot with too many records.

Measure picker

Use the Measure Picker to add Measures to a chart. The System Profile Measures appear in a tree structure on the left side (1). When you select a Measure, its description appears in the Measure Description area (2) and activates the splitting selection list (3). If splittings are available, these appear in the table on the right side (4).

To add a distinct set of splittings to the chart, select a set of splittings in the table to activate the **Select** option.



(1) measures, (2) measure description, (3) splitting selection, (4) list of splittings

Client commandline arguments

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Arguments

Troubleshooting

Arguments

Apply to `<DT_HOME>\dtclient`:

```
-kiosk [interval]
```

Optional value: Positive integer value for the interval (seconds, default 30, minimum 1)

Starts the client in kiosk mode. It automatically hides the mouse cursor, expands to full screen, and cycles through all open dashboards at the default or specified interval. For example `-kiosk 10`.

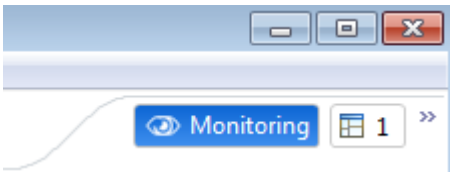
Use the Webstart Client in the kiosk mode by applying the arguments in the Web-start URL: `http://<dynaTraceServerName>:8021/webstart/Client/client.jnlp?argument=-kiosk&argument=10`

See Webstart Client for other functionality.

Kiosk mode control: Open or close dashboards and they are included in or excluded from the process.
Click **ESC** to pauses or resume automatic switching. Use **Page Up** and **Page Down** to go to the previous or next dashboard. Click **F11** to toggle the full screen mode.

If a dashboard tab disappears, drag it from the right.





`-dashboard [dashboardlocation]`

`[dashboardlocation]`: Optionally includes overriding source, compare source, and filter settings.

Opens the specified dashboard when the client starts. If you specify a source, compare source, or filters, an unsaved copy of the dashboard is created so that the original does not change.

Argument format:
`dashboardlocation = location [ ? query ] location = online://<servername>/<dashboard-name> | local://<file-system-path> | internal://<dashboard-name>`

While `online` and `local` are self-explanatory, `internal` refers to dashboards that are not listed explicitly, but are available in the Start Center or are integrated with other products.
`query = querypart [ & querypart ... ]`
` querypart = [source=sourcestring] | [compare=sourcestring] | [filter=filterstring [&filter=filterstring …]]`

See XML Reporting (REST) and REST Filters for information on `sourcestring` and `filterstring`.

Note

`<servername>` is case-sensitive. Type it as it appears in the client (not in upper case like in the Windows System control panel). Use the System Profile names that (i.e. the Alias), not the internal ID. For `local://` do not supply the `.dashboard.xml` extension, AppMon supplies it automatically.

Examples: `<DT_HOME>/dtclient -dashboard "online://<dynaTraceServerName>/Incident Dashboard?filter=tf:Last7d"` `<DT_HOME>/dtclient -dashboard "online://<dynaTraceServerName>/Incident Dashboard?filter=tf:Last7d"` `<DT_HOME>/dtclient -dashboard "local://C:/tmp/System Information"`

There may be path length limits for dashboards that are stored in the `Users` directory. The following code block does not work: `<DT_HOME>/dtclient -dashboard "local://C:/Users/test-user/.dynaTrace/dynaTrace <version>/client/conf/dashboards/System Information"`

`-installplugin [pluginfile]`

`pluginfile`: A plugin installation file

Installs the plugin when the client starts.

`-forceFullAnalysis`

Prevents the client from executing a quick analysis.

`-hidestartcenter`

Prevents the client from opening the Start Center automatically.

See Webstart Client > Webstart Client URL Equivalent to Command Line Arguments for the same functionality in the Set up the Webstart Client.

Troubleshooting

If the dashboard does not open in the client, check the log file. To do this, follow these steps:

1. Double-click the **System Information** in the Cockpit.
2. Expand DynatraceClient@host_name:nnnn and log files.
3. Open the most recent log file.

4. Scroll down and look for errors. For example, `<dateTime> WARNING Client Invalid syntax for Dashboard argument: 'online://easyTravel Database?filter=tf:Last3d'`.

For this example, the filter `Last3d` doesn´t exist. Its name is `Last72h`. For information on valid filter expressions, see REST Filters.

RegEx - using regular expressions

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Test regular expressions

Test regular group expressions

String comparison options

Related pages

Useful regular expressions

Useful regular expressions

RegEx technical reference

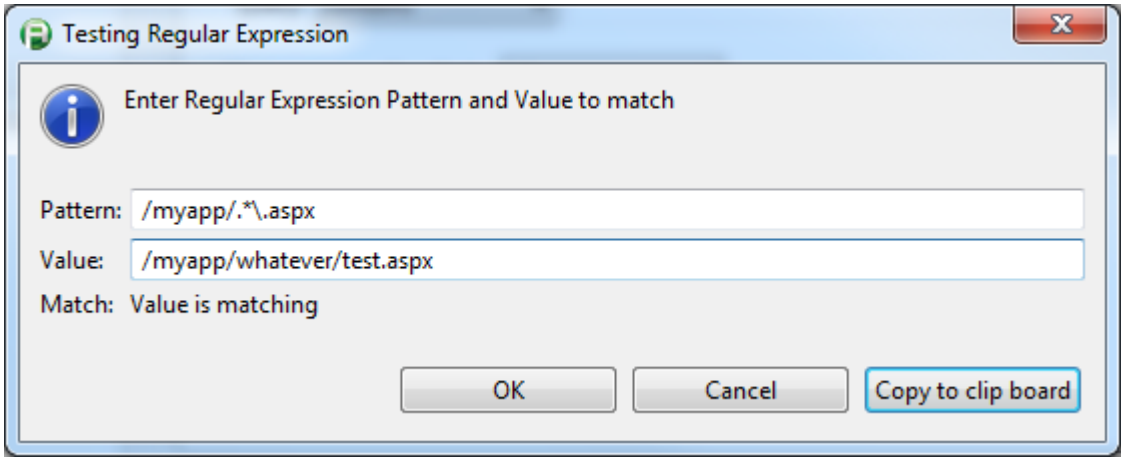
The following are examples of some of the regular expressions you can use throughout AppMon:

- In Filters: Only show PurePaths that match a certain URI.
- In Measures: Web request time for URIs matching a certain URI pattern.
- In Sensor Rules: Instrument methods that match a certain method nam.e
- As transformation of a Measure resul.t

Regular expressions are very powerful and more expensive to analyze than regular string comparisons such as *Starts With*, *Contains*, and *Ends With*. The AppMon Server needs extra CPU to analyze complex regular expressions. This particularly affects PurePath Measures that use RegEx, which can cause extra CPU overhead. A RegEx in a *web request count* Measure is evaluated for each web request.

Test regular expressions

To test your regular expressions, use the **Test Regular Expression** dialog box in the same dialog box where you define your regular expression.
Use this dialog box to define the regular expression and a test value, and to find out if the test value matches your regular expression.

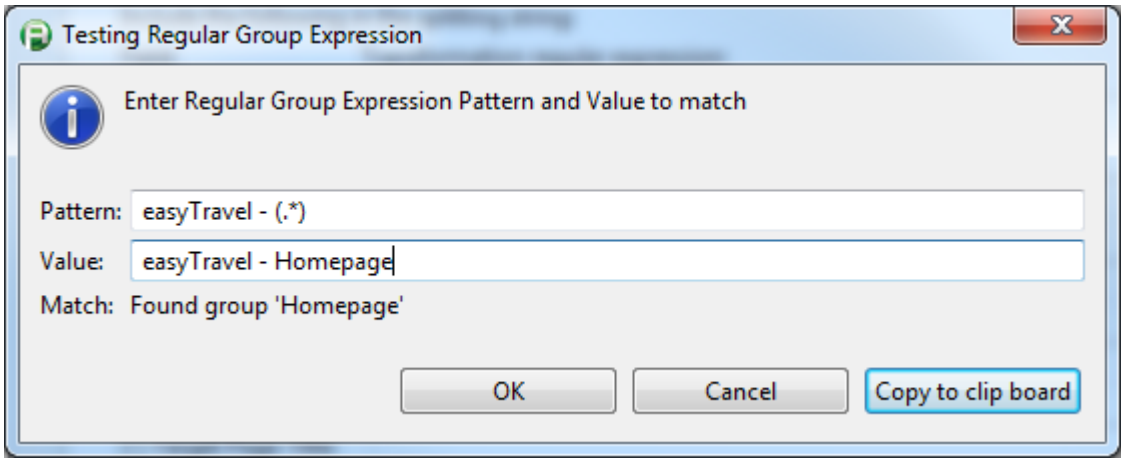


Test Regular Expression dialog box

Test regular group expressions

These are Measures that return a string value that is used as a split value in a business transaction. These provide an optional Transformational Regular expression. For example, if you want to split your user actions by page title, and all your application page titles start with easyTravel, you may want a transformation regular expression that removes the first part of the page title. AppMon requires that you specify a regular expression with a group matching, and will return the individual groups that match.

Use the following dialog box to ensure your transformation regex works as expected.



Test Regular Group Expression dialog box

String comparison options

AppMon offers RegEx and other string comparison options. The following is an overview:

Starts With	Matches strings that start with the given value (case sensitive)
Ends With	Matches strings that end with the given value (case sensitive)
Match	Allows you to use '*' as a placeholder similar to SQL For example, *TEST will match everything that ends with TEST (case sensitive)
Contains	Matches strings that contain the given value (case sensitive)
Equals	Matches strings that exactly match (are equal) to the given value (case sensitive)

AppMon uses the Java RegEx implementation internally. See the technical reference for more information.

Useful regular expressions

Related pages

RegEx - using regular expressions

RegEx technical reference

Regular expressions can be used in different Measure definitions, such as URI Pattern or Cookie Values. Regular expressions are very powerful but sometimes a little hard to write if not working with them regularly. This page shows useful Regular expressions frequently used in their AppMon Installation. You can use them to leverage regular expressions for your measures.

Besides the built-in Regular Expression Tester in most AppMon dialog boxes, you can also check out this online regex testing service: http://www.regextester.com/

See http://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html for a documentation of regular expression syntax.

Pulling a value from a method argument (for example an XML field)

Full Method Argument:	<XML File> Value1=”1234” Value2=”5678” WantedValue=”AndiG” Lots of stuff </XML File>
RegEx:	\bWantedValue=”([^”]*)
Returns:	AndiG

Parsing cookie value (for example JSESSIONID)

Full Cookie Text:	VALUE_A=Stuff; JSESSIONID=f423cc8100ab1990e70e4c34c4d8; JROUTE=HZrB;MORECOOKIES

RegEx: `JSESSIONID=([\w]\*)`

Returns: f423cc8100ab1990e70e4c34c4d8

Parsing header/session/cookie/attrib

Just plug in (.*)

Full Method Argument: AUTH_USER:domain/user@emailaddr.com

RegEx: domain/(.*) or domain/(.*)@

Returns: user@emailaddr.com or user

Parsing EMail address from a cookie

The following allows you to extract an email value from a key/value pair where the keyname is userid. This can be used for example when an email is passed in a Cookie.

Full Cookie Text: somekey=somevalue;userid=community@dynatrace.com

RegEx: `\buserid=([a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4})\b`

Returns: community@dynatrace.com

Extracting part of a URI

(for example for a Business Transaction Split Measure)

URI: http://dtstoragetest.blob.core.windows.net/samplecontainer

RegEx: `https?://([^.]*)\.blob\.core\.windows\.net/.*`

Returns: dtstoragetest

Identify classes ending with a certain string (limited to `[cls]` or `[bas]`)

Class MyClass[cls] or MyOtherClass[bas]

RegEx `(\[cls\]|\[bas\])`

Extract string of a constant length

The following allows you to extract a string that is of 20 character length. The string of interest is the value after `corellationid`.

Full String Text: >>subtractResult= >>>result=2, corellationid=vamsitest987654321, _env= >>>locale=, tag=166, activation=wm6ed6e9c70-bcc3-11e2-81a2-e32be605e442,

RegEx: corellationid=(.{20})

How to instrument an application

2. **Instrument application**

   LAMPP Apache and PHP instrumentation
   XAMPP Apache and PHP instrumentation
   XAMPP Apache instrumentation
   IIS instrumentation
   Application server / Tomcat / Alfresco instrumentation
   Application server / ASP.NET / DotNetNuke instrumentation

1. **Set up agents**

   General information
   .NET Agent
   Java Agent
   JavaScript Agent
   Database Agent
   Node.js Agent
   PHP Agent
   Web Server Agent
   Host Monitoring Agent
   zOS Agent

To see the data from your application, perform these general steps:

Initial configuration

Monitor specific applications

Related pages

Analyze and monitor big data solutions

Monitor Cloud Foundry

Monitor Microsoft Azure

Diagnose AWT and swing applications

Gatling/AppMon 6.2 integration

Gatling/AppMon 7.0 integration

Monitor Amazon applications

Monitor cloud applications

Monitor Docker applications

Monitor a WAS liberty profile server

Set up Performance Warehouse on Oracle Database (separate tablespace for indexes)

Use AspectJ to connect PurePaths in Tibco BusinessWorks

This section of the AppMon walkthroughs provide detailed instructions and examples instrumenting and monitoring specific applications.

Click an entry below or in the Related pages to learn more about monitoring a specific application or group of applications.

Cloud applications
Microsoft Azure applications
Amazon applications
Dockerized applications
Big Data solutions
AWT and Swing applications
Gatling AppMon 6.2 integration
Gatling AppMon 7.0 integration
WAS Liberty Profile Server
Performance Warehouse on Oracle Database
Use AspectJ to comment PurePaths in Tibco Businessworks

Monitor Amazon applications

Related pages

Deploy AppMon agents in Amazon EC2

Monitor Amazon cloud environments

AppMon includes special support for Amazon's AWS and cloud environments:

- Amazon EC2
- Amazon cloud environments

Deploy AppMon agents in Amazon EC2

On this page

Overview

Adding AppMon to an EC2 image

AppMon Server in EC2

AppMon Collector in EC2

The firewall

Tying it together

Next steps

Related pages

Monitor Amazon cloud environments

.NET Agent configuration

Deployment Guide

Overview

Amazon EC2 is Amazon's IaaS cloud offering that lets you to deploy anything from native to java applications. You can learn more on the Amazon Elastic Compute Cloud Website

Adding AppMon to an EC2 image

EC2 uses the concept of images similar to VMWare. The best way to start is to use one of the predefined Windows or Linux Images. Start from an image if you already have one.

Simply copy the Agent MSI or installer `.jar` to the image and install it. Because the agent boot straps itself, it does not require manual update.

If working with Java, you can configure the agent like you would in any normal installation. However, instead of supplying a hard coded AppMon Server IP, it is automatically a parameter of the image.

Every EC2 instance has meta-data assigned to it. This meta or user data can be used to parametrize an image. Inside the image you would add a script prior to your JVM start:

```
`#!/bin/sh
# Linux shell script to retrieve the AppMon Server IP from EC2 meta-data
USER_DATA=curl http://169.254.169.254/latest/user-data
if [ -z "$USER_DATA" ]
then
  echo "User Data was not supplied. Do Nothing"
  exit 1
fi

export DT_SERVER=echo $USER_DATA|grep "dynaTraceServer="|sed "s#.*dynaTraceServer=\([^ ]*
```

The resulting User Data would be similar to the following:

```
dynaTraceServer=10.1.1.90
```

Use the defined environment variable instead of a hardcoded AppMon Server or Collector IP. You can also do this for Windows and .NET, but it is actually easier to have a couple of lines of java code to do this. You can also use Powershell if you are familiar with it. For .NET, make sure that the environment variable is named **DT_SERVER**. Set it in the same context as the starting CLR. You might also execute the following to automatically change the AppMon .NET Agent Configuration:

```
REG ADD "HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\dynaTrace\Agent\Whitelist\1"  /f /v serv
```

Check the entry with `regedit.exe` to ensure that you pick the correct Whitelist entry. Then create a new image based on the modification and use it in the future.

AppMon Server in EC2

The AppMon Server setup in EC2 works pretty much like on premise. Use a 64-bit Amazon Linux Image with at least 8 GB memory and sufficient EC Units of CPU (check the deployment guide. You can use Amazon's Oracle RDS, install your own Oracle or use PostGres as a Performance Warehouse.

Once installed use the AppMon Server `init.d` script to start the AppMon server on Image startup. The startup script ensures the AppMon Server is running correctly in EC2. It reads the user data to extract the host for the AppMon Performance Warehouse. In your Performance Warehouse configuration dialog box, use **dynatracePerformanceWarehouse.localdomain** as the hostname.

To use a local PostGres instead, just remove that piece from the startup script and ensure that PostGres is started prior the AppMon Server.

Finally you need to create an Image based on the instance.

AppMon Collector in EC2

Install an AppMon collector if running the AppMon Server on premise or to run a real production environment. See the deployment guide for AppMon collector sizing information. Start the AppMon collector with an `init.d` script. If you run the AppMon Server in EC2, parameterize it similar to the Agent.

```
`#!/bin/sh
# Linux shell script to retrieve the AppMon Server IP from EC2 meta-data
USER_DATA=curl http://169.254.169.254/latest/user-data
if [ -z "$USER_DATA" ]
then
  echo "User Data was not supplied. Do Nothing"
  exit 1
fi

export DT_SERVER=echo $USER_DATA|grep "dynaTraceServer="|sed "s#.*dynaTraceServer=\([^ ]*

#start collector
dtcollector --server $DT_SERVER`
```

The resulting user data is similar to the following:

```
dynaTraceServer=10.1.1.90
```

Create an image for the AppMon Collector. You can start multiple Collector instances based on the number of instances that you want to monitor. When starting the instances simply configure the IP of the AppMon Collector in the meta-data instead of the AppMon Server.

The firewall

Amazon EC2 Instances are secured by a firewall that is maintained outside the actual instance. Learn how to use Security Groups in the Amazon Documentation. What is important for a successful AppMon setup is that the incoming ports on the AppMon Server and Collector must be opened in those firewall rules. Otherwise, the Agent cannot connect and you cannot connect your Client to the AppMon Server.

Tying it together

The easiest way to start an environment and tie it all together is using Amazon Cloud Formation. It lets you configure and start multiple instances and wire them together. A sample formation would is similar to the following:

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "easyTravel Demo-in-the-Cloud setup",

  "Parameters" : {
    "KeyName" : {
      "Description" : "Name of and existing EC2 KeyPair to enable SSH access to the insta
      "Type" : "String"
    }
  },


  "Resources" : {

    "dynaTraceServer" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "InstanceType" : "c1.medium" ,
        "SecurityGroups" : [ { "Ref" : "SSH" }, { "Ref" : "HTTP" }, { "Ref" : "dynaTrace"
```

```
        "SecurityGroups" : [ { "Ref" : "SSH" }, { "Ref" : "HTTP" }, { "Ref" : "dynaTrace
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : "ami-dynaTrace",
        "UserData" : {
                "Fn::Base64" : {
                    "Fn::Join" : [ "\n", [
                            { "Fn::Join" : [ "=", [ "dynaTracePerformanceWarehouse", "loc
                                ] ]
                 } },
        "Tags" : [ {
                    "Key" : "Name",
                    "Value" : "dynaTrace Server"
                 }
                ]
      }
    },


    "MyJavaInstance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "InstanceType" : "t1.micro" ,
        "SecurityGroups" : [ { "Ref" : "SSH" }, { "Ref" : "HTTP" }, { "Ref" : "MyOwnPorts
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : "MyAMI",
        "UserData" : {
                "Fn::Base64" : {
                    "Fn::Join" : [ "\n", [
                            { "Fn::Join" : [ "=", [ "dynaTraceServer", { "Fn::GetAtt" : [
                                ] ]
                 } },
        "Tags" : [ {
                    "Key" : "Name",
                    "Value" : "My EC2 Instance"
                 }
          ]
      }
    },
    "SSH" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
```

```
        "GroupDescription" : "Enable SSH access via port 22",
        "SecurityGroupIngress" : [ {
          "IpProtocol" : "tcp",
          "FromPort" : "22",
          "ToPort" : "22",
          "CidrIp" : "0.0.0.0/0"
        } ]
      }
    },
    "HTTP" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable http and https",
        "SecurityGroupIngress" : [ {
          "IpProtocol" : "tcp",
          "FromPort" : "80",
          "ToPort" : "80",
          "CidrIp" : "0.0.0.0/0"
        }, {
          "IpProtocol" : "tcp",
          "FromPort" : "443",
          "ToPort" : "443",
          "CidrIp" : "0.0.0.0/0"
        } ]
      }
    },
    "dynaTrace" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "Enable agent, collector and web client port",
        "SecurityGroupIngress" : [ {
          "IpProtocol" : "tcp",
          "FromPort" : "9998",
          "ToPort" : "9998",
          "CidrIp" : "0.0.0.0/0"
        },
        {
          "IpProtocol" : "tcp",
          "FromPort" : "6998",
          "ToPort" : "6998",
          "CidrIp" : "0.0.0.0/0"
```

```
          "CidrIp" : "0.0.0.0/0"
        },
        {
          "IpProtocol" : "tcp",
          "FromPort" : "8021",
          "ToPort" : "8021",
          "CidrIp" : "0.0.0.0/0"
        } ]
      }
    },
    "MyOwnPorts" : {
      "Type" : "AWS::EC2::SecurityGroup",
      "Properties" : {
        "GroupDescription" : "My Own Ports",
        "SecurityGroupIngress" : [ {
          "IpProtocol" : "tcp",
          "FromPort" : "1697",
          "ToPort" : "1698",
          "CidrIp" : "0.0.0.0/0"
        },
        {
          "IpProtocol" : "tcp",
          "FromPort" : "1527",
          "ToPort" : "1527",
          "CidrIp" : "0.0.0.0/0"
        } ]
      }
    }
  },

  "Outputs" : {
    "MyInstanceIP" : {
      "Description" : "Public IP of the Web Launcher instance",
      "Value" : { "Fn::GetAtt" : [ "MyJavaInstance", "PublicIp" ] }
    },
    "dynaTraceServer" : {
      "Description" : "Public IP of the dynaTrace Server instance",
      "Value" : { "Fn::GetAtt" : [ "dynaTraceServer", "PublicIp" ] }
    },
    "dynaTraceServerURL" : {
      "Description" : "Web-URL of the dynaTrace Server",
```

```
      "Value" : { "Fn::Join" : [ "", ["https://",{ "Fn::GetAtt" : [ "dynaTraceServer", "P
    },
    "dynaTraceWebstart" : {
      "Description" : "URL of Webstart Client",
      "Value" : { "Fn::Join" : [ "", ["https://",{ "Fn::GetAtt" : [ "dynaTraceServer", "P
    }
  }
}
```

Our own instance gets some meta-data assigned. This meta data assigns the internal IP of the AppMon Server. The important thing to notice is that AppMon uses the internal IPs for the agent to collector/server configuration as well as for AppMon Server to Performance Warehouse. Internal communication is free. Using public IPs would incur additional cost and higher latency.

You can also configure the firewall rules here.

Once you start the stack you are good to go.

Next steps

The next logical step is to setup EC2 Cloud Monitoring using the Amazon EC2 Fastpack.

Monitor Amazon cloud environments

On this page

How to deploy your application with AppMon monitoring in EC2

End user monitoring for Amazon EC2 based applications

Running the AppMon Server in EC2

Related pages

Deploy AppMon agents in Amazon EC2

Monitor cloud applications

How does UEM work?



Amazon AWS offers a range of cloud applications, from the IaaS offering EC2, to Amazon RDS for Database, and much more. Regardless of which services you choose to use, your application at the core. Monitoring an Amazon Cloud Environment must be focused on your Application and how it performs in context of the Amazon Cloud.

AppMon fully supports Amazon EC2 as a platform for your applications. It lets you achieve End-to-End Monitoring, starting at the End User, across your Web Server and Application Servers running on EC2, all the way down to your own database or Amazon's RDS. In addition to AppMon's unique capabilities to monitoring and analyze application performance, it also integrates with Amazon AWS CloudWatch to closely monitor EC2 instances, Elastic Load Balancers, or RDS Services. It identifies any performance impact on your end user.

You can also run the complete AppMon infrastructure inside EC2.

How to deploy your application with AppMon monitoring in EC2

It is very easy to deploy AppMon in EC2. Learn more about how to do this and how to automatically inject the agent in the EC2 Deployment Page.

Once installed, AppMon monitors newly started instances of your applications, either manually or due to auto scaling.

End user monitoring for Amazon EC2 based applications

In a public Cloud environment, your provider controls the infrastructure and network. You also most likely use Load Balancers and other services that are located between your end users and your JVM. As such, it is important to get an end-to-end understanding of your application's performance. AppMon UEM achieves just that and is fully cloud-ready.

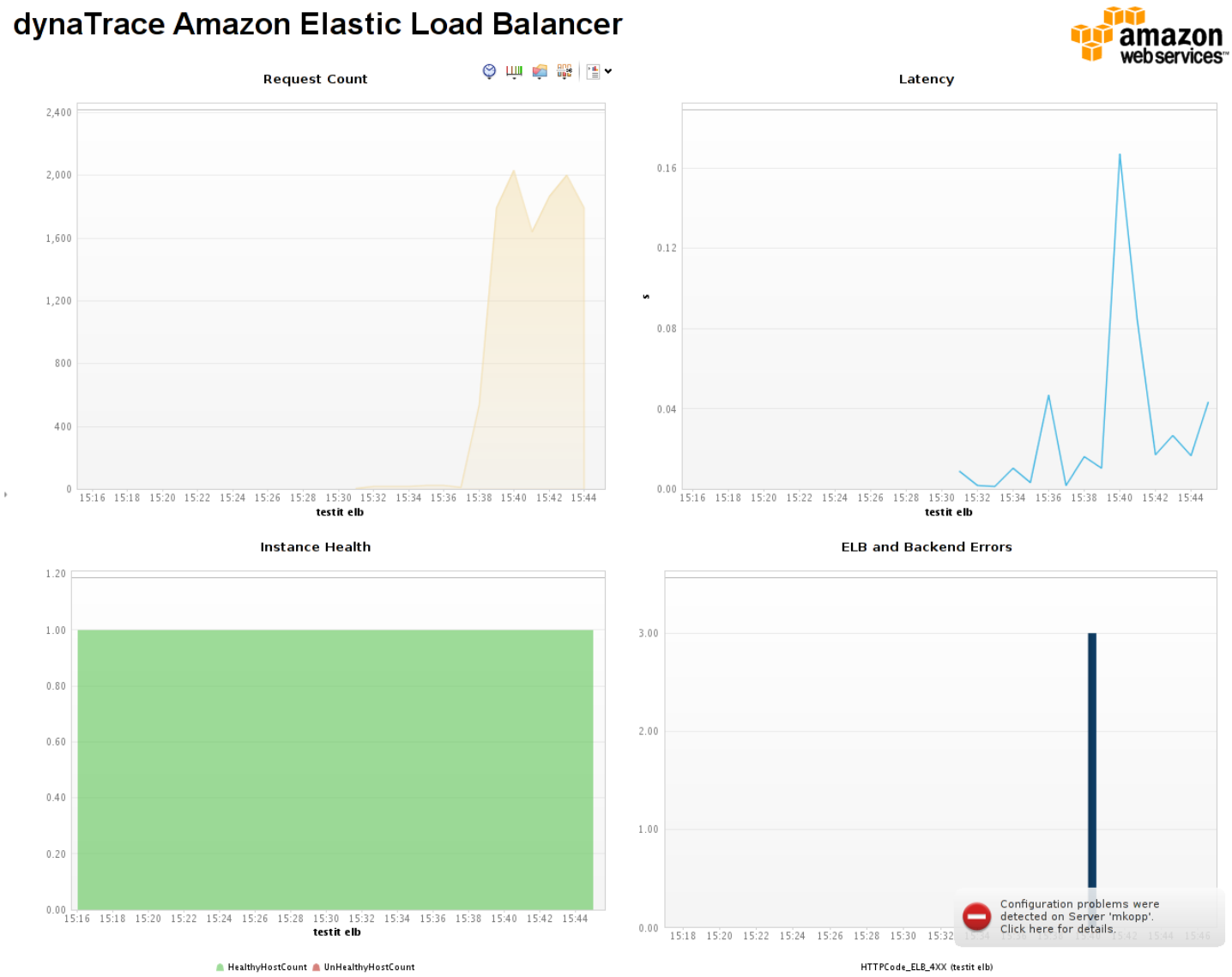
UEM lets you understand the real end user performance of your users and how CDN, or the lack of it, impacts the end user satisfaction at any given location. It lets you decide whether you should add more CDN locations or even add another AWS region to your application deployment. It tells where end user response time is acceptable and where it has negative impact on the conversion rate. Finally, because of AppMon's unique end-to-end 100% transaction type approach, you can identify the root case of end user response time problems. This is regardless of whether it is your Application, Amazons RDS, the Elastic Load Balancer, or simply a degrading EC2 instance.

Learn more about UEM

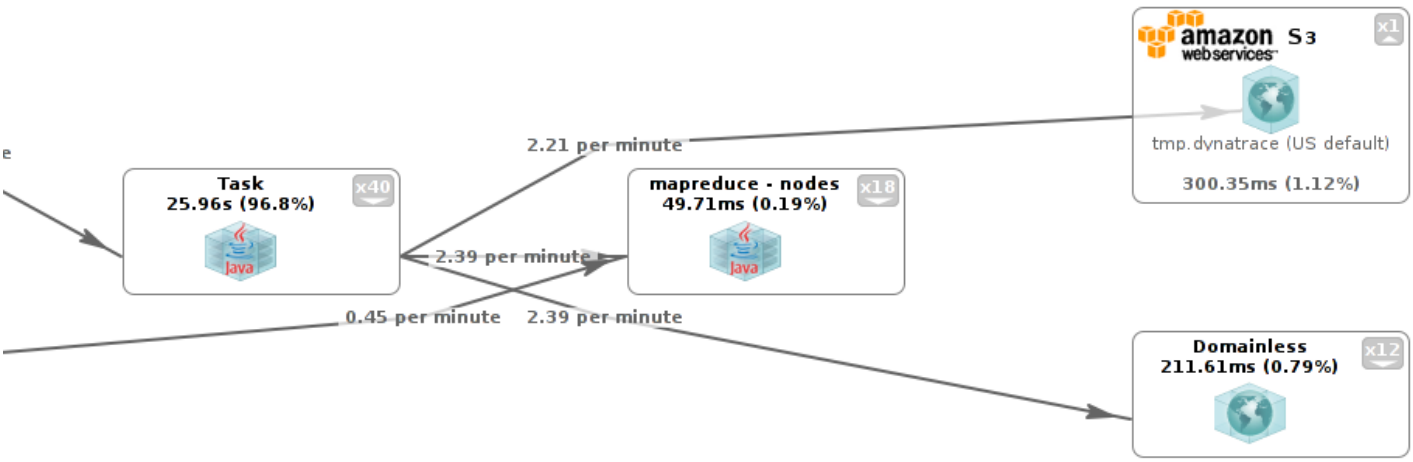
Understand the impact of elastic load balancer latency

The Amazon Elastic Load Balancer provides insight into how transaction load affects application latency. You can use this to decide whether to add more EC2 instances or do a performance hotspot analysis on your application.

Integrate the Elastic Load Balancer metrics into AppMon.



Understand application performance and the impact of AWS infrastructure

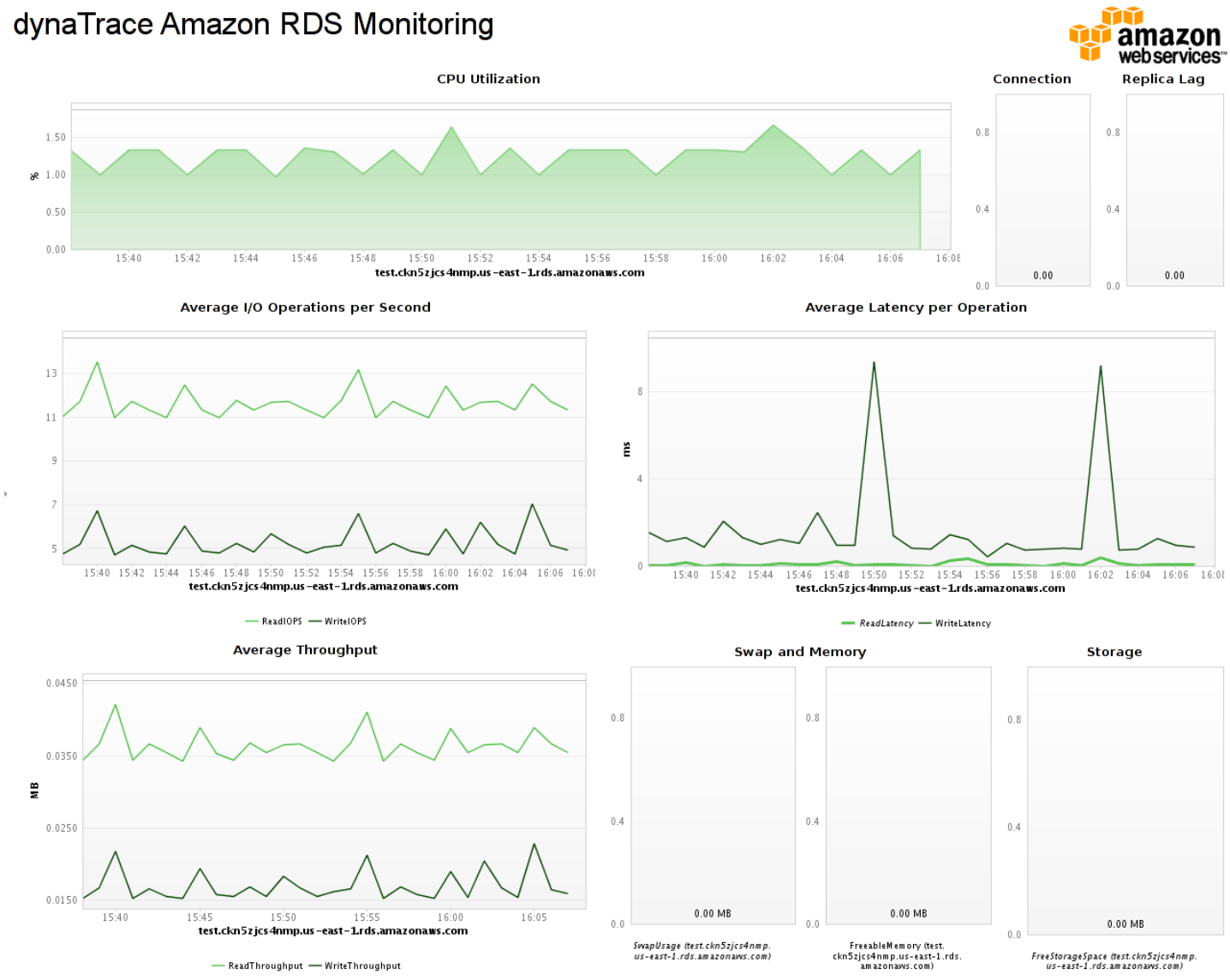


AppMon provides a full range of performance and scalability analysis options for EC2-based Applications. You can analyze end-to-end transaction flow and identify hotspots down to the method level. AppMon can also monitor the AWS infrastructure using Amazon Cloud Watch and lets you to infrastructure based performance issues. AppMon even detects and visualizes any outgoing calls that you make to the Amazon Webservice Infrastructure.

Monitor Amazon AWS

Understand the performance impact of Amazon RDS

AppMon fully supports Amazons RDS services. Not only can you analyze and monitor every database statement that your application executes, you can monitor the RDS instance itself to help understand why a particular instance or statement is slower than expected. Learn more on how to monitor Amazon RDS



Running the AppMon Server in EC2

You can run the AppMon Server infrastrcture in your datacenter, but can optionally leverage Amazon AWS for that as well.

Learn how to deploy AppMon in EC2.

Analyze and monitor big data solutions

On this page

Overview

Get full end to end database visibility with Dynatrace Cassandra support

Optimize your MapReduce jobs to achieve more with less easily

Full end-to-end visibility into your SOLR searches

Understand how Amazon SimpleDB impacts your response times

Related pages

Monitor specific applications

MongoDB

Cassandra

Set up Performance Warehouse on Oracle Database (separate tablespace for indexes)

Overview

Big Data Solutions are on the rise. They range from NoSQL databases like Cassandra, to Full Text Search solutions like Apache SOLR to distributed batch processing solutions such as Apache Hadoop MapReduce. AppMon enables you to ensure your Big Data Applications performance.

Get full end to end database visibility with Dynatrace Cassandra support



AppMon gives you end to end visibility into your application's usage of Cassandra.

Learn more about setting up AppMon to monitor Cassandra.

Identify the following Cassandra access patterns

- Transaction Contribution Time
- Number and type of statements per BT
- Access Distribution
- Consistency Level

End-to-end visibility

- JMX/Cache Monitoring
- GC and I/O impact
- Cassandra Health

Optimize your MapReduce jobs to achieve more with less easily



The AppMon Apache Hadoop fastpack provides easy out-of-the-box monitoring of Hadoop HDFS and Map/Reduce Jobs. It also allows the performance analysis of specific Map/Reduce Jobs and Tasks in order to improve them. Learn how to easily spot why your MapReduce job can finish in 90% of the time or with 20% less resources.

Full end-to-end visibility into your SOLR searches



The AppMon SOLR FastPack provides easy out-of-the-box monitoring for applications using the SOLR Search Engine. AppMon automatically provides an end-to-end view of each Business Transaction down into SOLR itself. This lets you not only understand how much each SOLR query contributes to your Business Transaction, but also why a specific SOLR query was slow.

Understand how Amazon SimpleDB impacts your response times



AppMon automatically detects and visualizes requests to Amazon's SimpleDB. See how much of your Business Transactions time is spent in EC2 and why. The AppMon Amazon SimpleDB FastPack provides easy out-of-the-box monitoring of read and write performance of Amazon SimpleDB use of your application.

Monitor Microsoft Azure

On this page

Supported Azure Computer Services

Deployment Guides

Additional Resources

Supported Azure Computer Services

	AppMon Agents	AppMon Collector	AppMon Server
Virtual Machines			
Windows			
Linux			
Cloud Service			
Web Roles			
Worker Roles			
App Services			
Web Apps (Websites)	1)		
Mobile Apps	1)		

Supported

Unsupported / Not Available

1)
Compatible agents: .NET, Java, NodeJS
Early acess program for IIS agent available
-> Send your request to participate to patrick.thurner@dynatrace.com

Deployment Guides

- How to deploy AppMon Agents in a Cloud-Service
- How to deploy AppMon .NET Agent in a Web App

Additional Resources

- AppMon Deployment Automation for Windows with Powershell

- AppMon Deployment Automation using Chef
- AppMon Deployment Automation using Puppet
- AppMon Deployment Automation using Ansible
- UEM manual injection

Deploy AppMon agents in MS Azure cloud service

On this page

Overview

Prior to Deployment

Add necessary files to your solution

Configure cloud-service's ServiceDefinition file

Related pages

Deploy AppMon .NET Agent in MS Azure web apps

Monitor Microsoft Azure

Monitor specific applications

Monitor cloud applications

Overview

This guide describes the steps necessary to setup AppMon Agents in Microsoft Azure Cloud-Services.

The installation uses Powershell scripts available from https://github.com/dynatrace/dynatrace-powershell to install the necessary agents, executed by a startup-task.

In this guide, **WebRole** installs agents to monitor IIS and ASP.NET. **WorkerRole** installs .NET agent to monitor `WaWorkerHost.exe`.

Prior to Deployment

1. Download "All Agents for Windows" Installer from one of the following:
   a. The community portal.
   b. The public download location at http://downloads.dynatracesaas.com
      .

2. Download scripts and modules from https://github.com/dynatrace/dynatrace-powershell.

Add necessary files to your solution

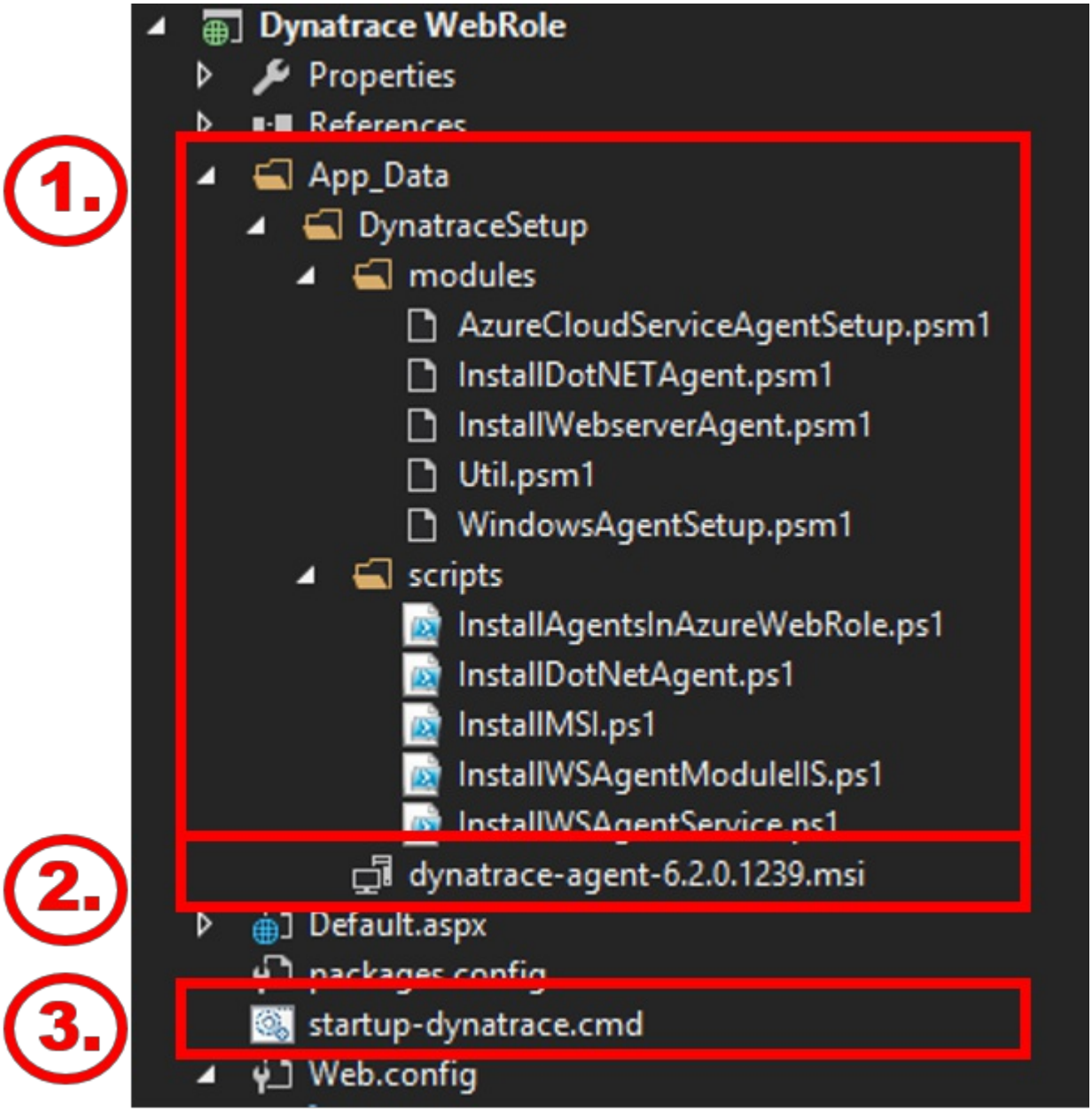
WebRole

1. Within the App_Data folder, create a new folder named with a descriptive name. For example, ‘DynatraceSetup’.

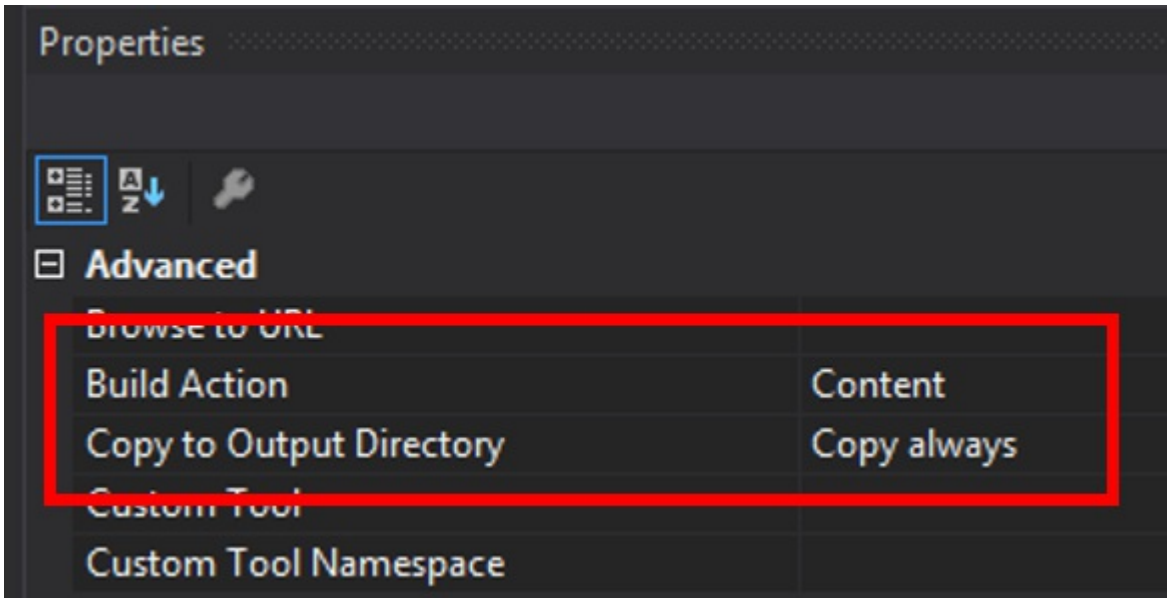
2. Put all powerhsell scripts, modules, and the agent installer (`.msi`) into this folder.

3. Add a new startup script to your solution that calls `scripts/InstallAgentsInAzureWebRole.ps1`.
   For example:

```
@echo off
DATE /T >> c:\Dynatrace-InstallAgentsInAzureWebRole.log 2>&1
TIME /T >> c:\Dynatrace-InstallAgentsInAzureWebRole.log 2>&1
cd App_Data\DynatraceSetup\scripts
powershell -NoProfile -ExecutionPolicy unrestricted -File InstallAgentsInAzureWebRole.
exit /B 0
```


Dynatrace WebRole
Properties
References
App_Data
DynatraceSetup
modules
AzureCloudServiceAgentSetup.psm1
InstallDotNETAgent.psm1
InstallWebserverAgent.psm1
Util.psm1
WindowsAgentSetup.psm1
scripts
InstallAgentsInAzureWebRole.ps1
InstallDotNetAgent.ps1
InstallMSI.ps1
InstallWSAgentModuleIIS.ps1
dynatrace-agent-6.2.0.1239.msi
Default.aspx
startup-dynatrace.cmd
Web.config
1.
2.
3.

4. Ensure your files to be copied to the output directory, setting their properties:
   Copy-To-Output Directory: ”Copy Always”
   Build-Action: ”Content”



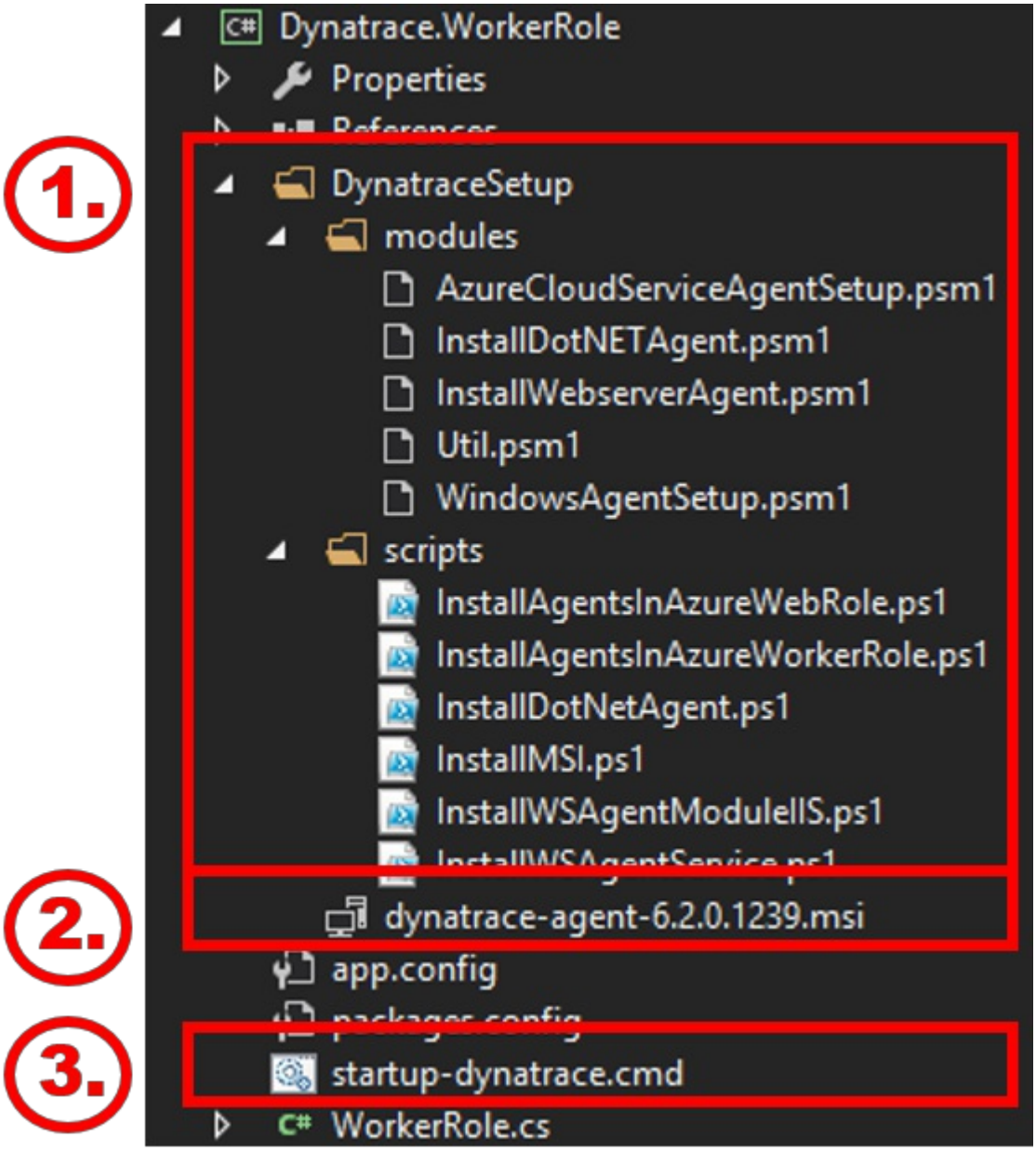
WorkerRole

1. Within your solution create a new folder with a descriptive name. For example, ‘DynatraceSetup’.

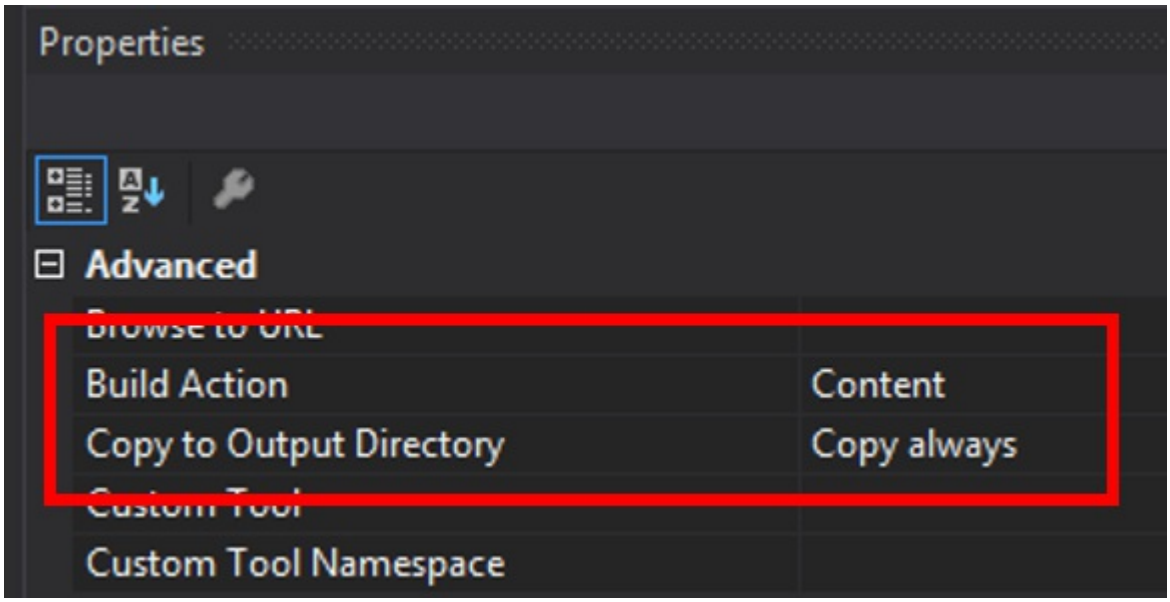
2. Put all powerhsell scripts, modules, and the Dynatrace `.msi` into this folder.

3. Add a new startup script to your solution that calls `scripts/InstallAgentsInAzureWorkerRole.ps1`. For example:

```
@echo off
DATE /T >> c:\Dynatrace-InstallAgentsInAzureWorkerRole.log 2>&1
TIME /T >> c:\Dynatrace-InstallAgentsInAzureWorkerRole.log 2>&1
cd DynatraceSetup\scripts
powershell -NoProfile -ExecutionPolicy unrestricted -File InstallAgentsInAzureWorkerRo
exit /B 0
```


Dynatrace.WorkerRole
Properties
DynatraceSetup
modules
AzureCloudServiceAgentSetup.psm1
InstallDotNETAgent.psm1
InstallWebserverAgent.psm1
Util.psm1
WindowsAgentSetup.psm1
scripts
InstallAgentsInAzureWebRole.ps1
InstallAgentsInAzureWorkerRole.ps1
InstallDotNetAgent.ps1
InstallMSI.ps1
InstallWSAgentModuleIIS.ps1
dynatrace-agent-6.2.0.1239.msi
app.config
startup-dynatrace.cmd
WorkerRole.cs
1.
2.
3.

4. Set the following properties on the files to be copied to the output folder:
   Copy-To-Output Directory: ”Copy Always”
   Build-Action: ”Content”

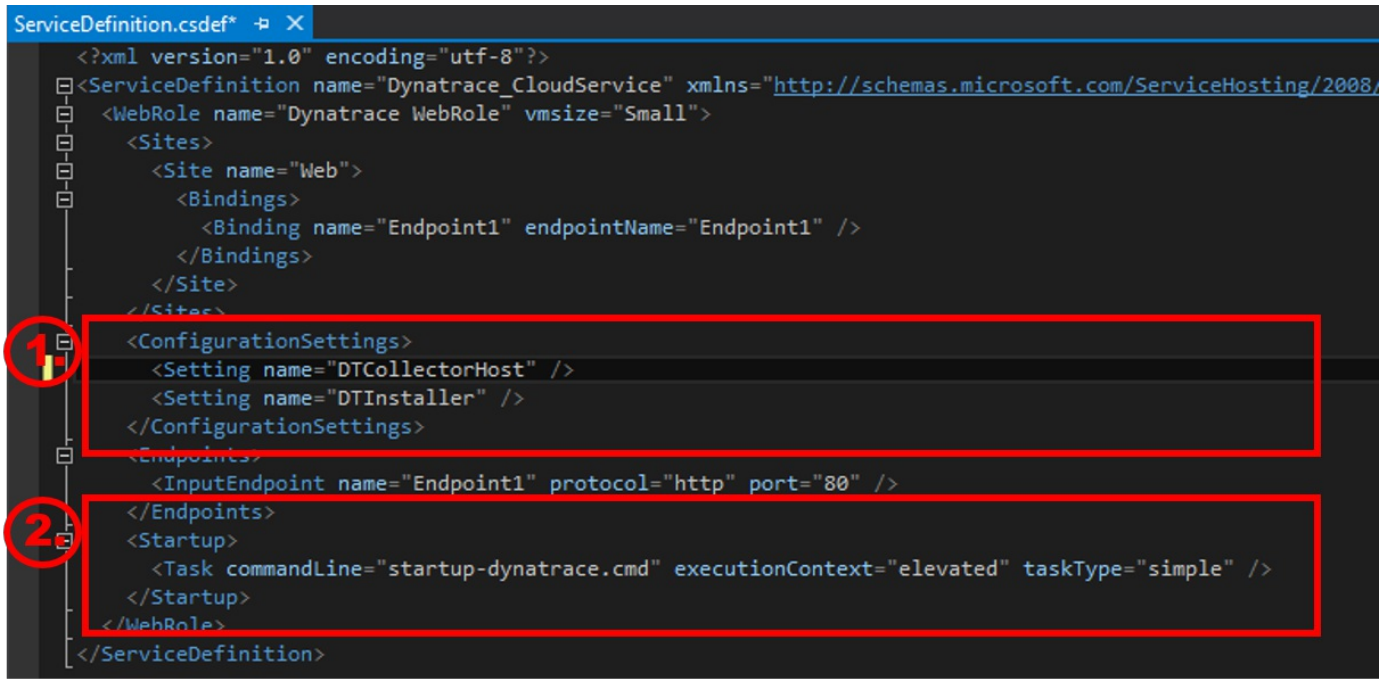


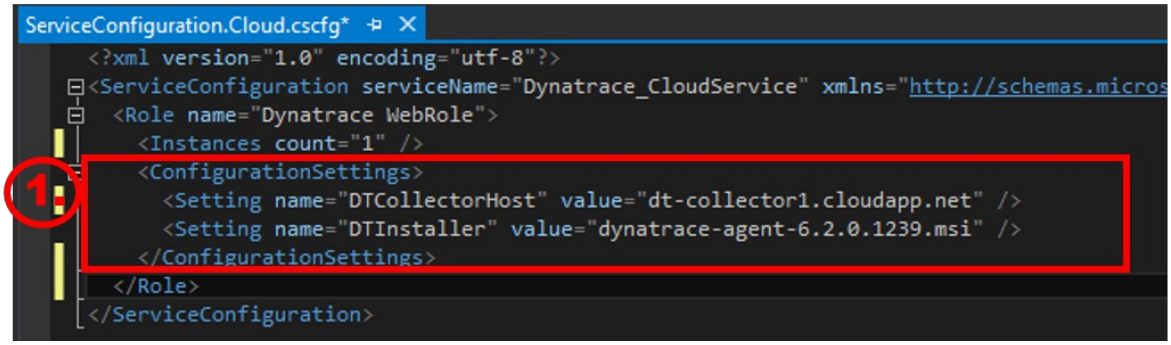
Configure cloud-service’s ServiceDefinition file

1. Add configuration keys
   For minimum configuration at least 2 keys are required:
   a. DTCollectorHost
   b. DTInstaller

For further details on additional configuration options, see description within either `scripts/InstallAgentsInAzureWebRole.ps1` or `scripts/InstallAgentsInAzureWorkerRole.ps1`.

1. Add the startup-task.





Deploy AppMon .NET Agent in MS Azure web apps

On this page

Before you begin

Instrumentation

Update

Uninstall

Related pages

Deploy AppMon agents in MS Azure cloud service

Monitor Microsoft Azure

Monitor specific applications

Monitor cloud applications

Troubleshooting .NET Agent in MS Azure web apps

Azure Web Apps (formerly Azure Websites) is a service provided by Microsoft Azure that gives you the option of deploying and auto-scaling applications and services. Using a predefined Azure site extension you can modify your deployment by supplying additional resources or packages. For more information, see the Azure Web Sites documentation. For maximum ease of installation, Dynatrace provides an Azure site extension that you can download from the Azure Site Extensions gallery.

This guide describes the steps necessary to inject the AppMon .NET Agent into an Azure Web application. For troubleshooting tips, please click here.

The .NET Agent connects to a running instance of a Collector that must be accessible from your Azure Web application.

Before you begin

Download **Agents (including web server agents)** installer, and install agents.

Instrumentation

Configure the Dynatrace Site Extension via the Azure portal

1. Open portal.azure.com in a new browser window.
2. Navigate to the web app you want to monitor.
3. From **Settings**, select **Application Settings**. Scroll down to the App Settings area and add two new Key/Value pairs:
   - `DT_CONNECTION_POINT`: full address, including port, of your AppMon Server.
   - `DT_DOWNLOAD_URL`: link to the distribution package of the OneAgent. You can use this one: https://files.dynatrace.com/downloads/OnPrem/dynaTrace/7.0/7.0.0.2469/dynatrace-one-agent-distribution-loader-7.0.0.2469-windows-x86.zip.

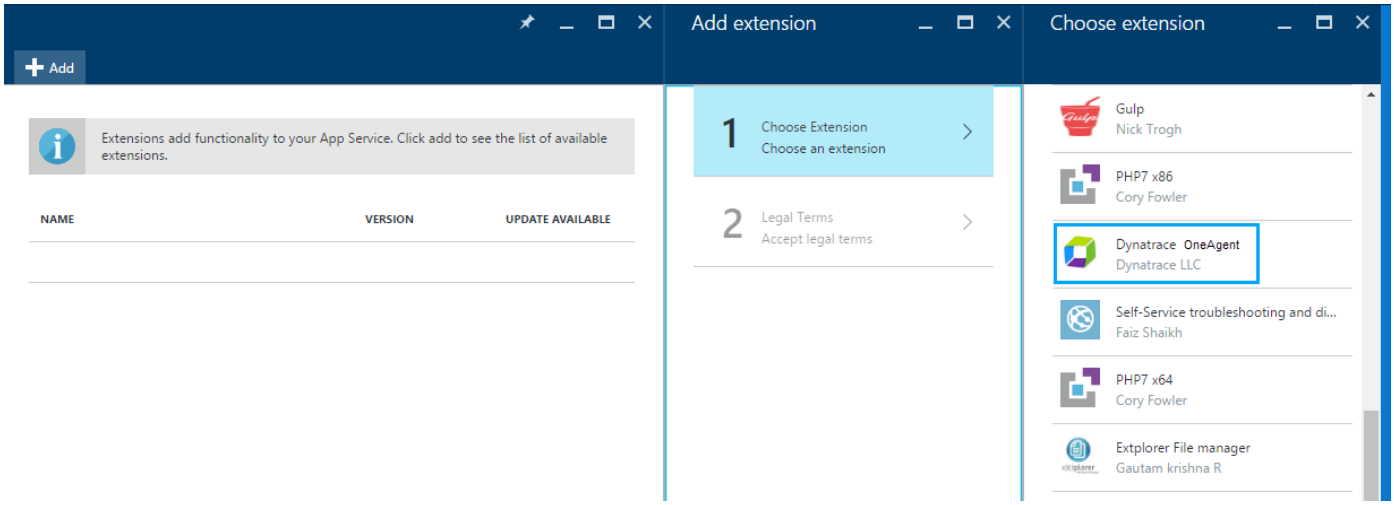
Install the Dynatrace Azure site extension

You can do this either via the Azure Portal, or through Kudo.

Azure portal

1. Open portal.azure.com in a new browser window.
2. Navigate to the web app you want to monitor.
3. Select **Extensions** from the list of options. You'll find this in the **Development tools** subsection (note the Search field at the top of the page in case you have trouble finding this option).
4. Within the new pane (i.e., "blade" in Azure terminology) that appears on the right-hand side, click **Add**.

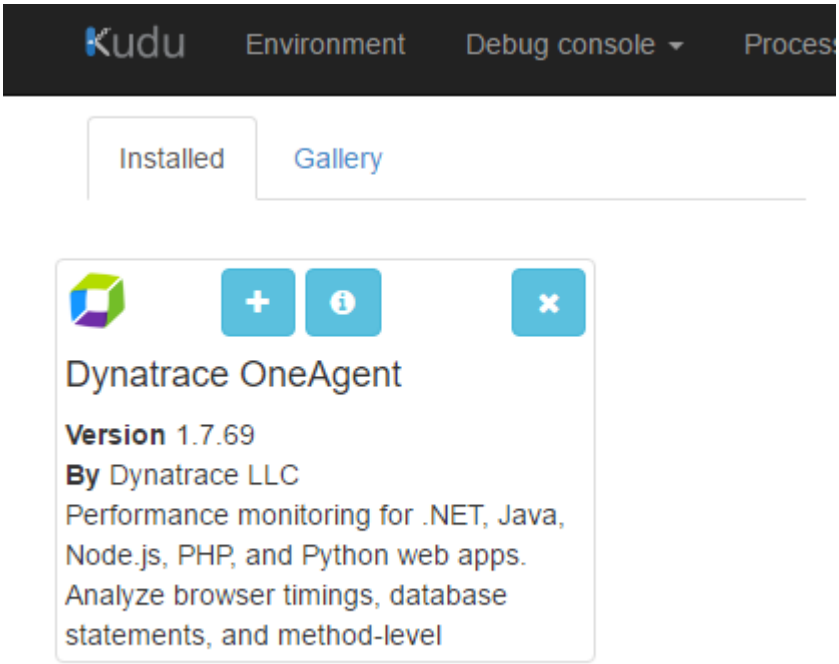
5. Scroll through the list until you find **Dynatrace OneAgent**. Note that entries are not ordered alphabetically.



6. Click **OK** to start monitoring your Azure website.
7. Restart your website so that AppMon begins to receive monitoring data.

Kudo

Alternatively, you can install the Dynatrace site extension via Kudu, which is available at this URL `<website-name>.scm.azurewebsites.net`.



Update

Update the Azure Site Extension

If there is an available extension update, an **Update** button will be visible in the Azure UI. Click it to update your extension.

Update OneAgent

If there is an update available for the OneAgent, it will be installed automatically by the bootstrap agent at the application restart.

Uninstall

Just as with the installation process described above, uninstalling the Dynatrace site extension can be done either via the Azure portal or by using Kudo.

Troubleshooting .NET Agent in MS Azure web apps

On this page

Not receiving agent data

503 Service Unavailable for Web App and Kudo

Related pages

Deploy AppMon agents in MS Azure cloud service

Monitor Microsoft Azure

Monitor specific applications

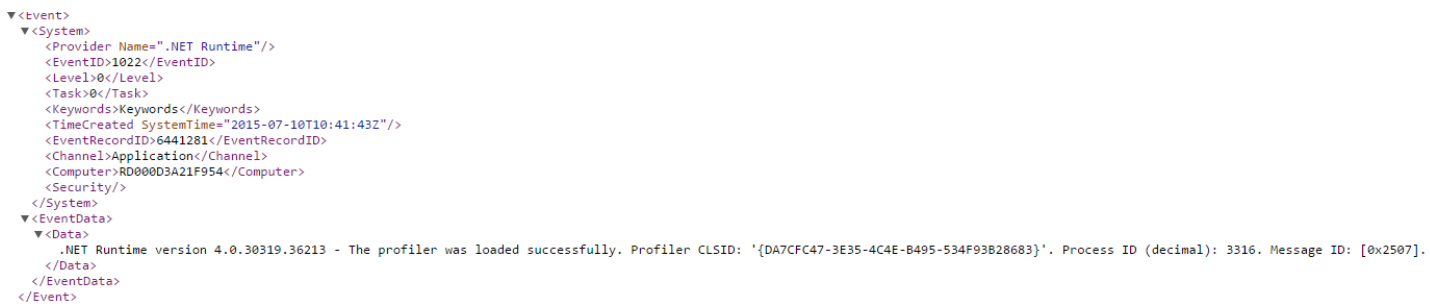
Monitor cloud applications

Not receiving agent data

Check the event log if the agent is properly loaded

You can view the event log via the SCM-Dashboard using the Event-Log-Viewer Site Extension. *https://{site name}.scm.azurewebsites.net* (in other words, add **.scm** in front of the site name), or just directly access the event log. *https://{site name}.scm.azurewebsites.netapi/vfs/LogFiles/eventlog.xml*

You should see an entry like the following



If you do not see this, check your `COR_XXXXXX` application settings.

503 Service Unavailable for Web App and Kudo

The Dynatrace site extension registers IIS modules at installation time. If registration is complete but the actual IIS module files aren't in place (or corrupt), the Web App may return a `503 Service Unavailable` message. Also, the Kudo AppPool will no longer be available, which effectively means that you can no longer fix your app. If this occurs, please file a support case. Fortunately, there is a way to recover from such a situation by adding the following setting in Application Settings to effectively deactivate all site extensions:

```
WEBSITE_PRIVATE_EXTENSIONS=0
```

This setting will give you access to Kudo once again. Please then re-install the site extension and set `WEBSITE_PRIVATE_EXTENSIONS=1` (or remove the setting entirely). Following application restart, everything should work again, including AppMon monitoring.

Diagnose AWT and swing applications

On this page

Download

Overview

AWT / swing demo application

Agent injection

Default AWT sensor pack

Creating your own AWT sensor pack

Catching PurePaths

Business Transactions

Dashboard

Troubleshooting

Related pages

Monitor specific applications

Deploy AppMon Java Agents on Cloud Foundry sample app

Deploy AppMon Node.js Agents on Cloud Foundry sample app

.NET Agent configuration

Download

System Profile	AWT Swing Example.profile.xml
Dashboard	AWT Swing Example Dashboard.dashboard.xml

Stored session	AWT Swing Example Session.dts
Modified SwingSet demo application	ModifiedSwingSetDemoApp.zip

Overview

This page describes how to instrument AWT and Swing applications with AppMon. The goal is to start PurePaths on every user interaction such as mouse clicks or keyboard events.

The standard Java library contains out of the box the following packages:

- AWT (Abstract Widget Toolkit, the first approach from Sun Microsystems to standardize user interface components)
- Swing (Based on AWT, contains many improved classes and extended functionalities)



It is very important to understand that Swing is built on top of AWT, since you do not want AppMon instrumentation based entirely on the Swing level. You must hook into selected basic AWT classes to fully instrument both AWT and Swing.
More information about Swing is available here: https://docs.oracle.com/javase/tutorial/uiswing/

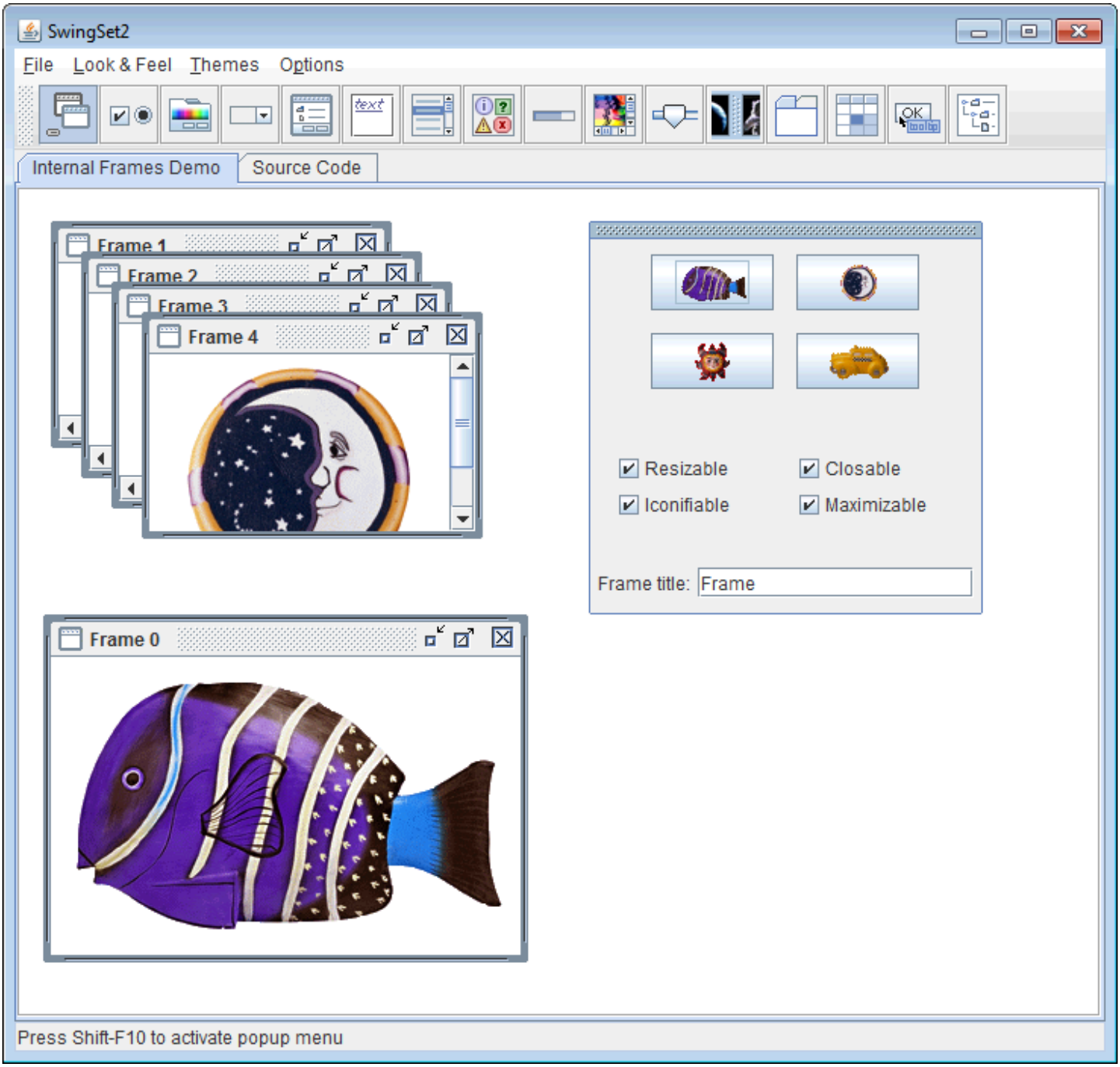
AWT / swing demo application

The AWT/Swing sample application used in this walkthrough ships with the Java Development Kit (JDK). The sample application provides a good overview over most out of the box Swing user interface controls, but it lacks any business logic behind them. It's still a good example for configuring AppMon to start PurePaths on user interface actions.

The sample application is located in your local %JDK_HOME%/demo/jfc/SwingSet2 folder. For example: `"C:\Program Files\Java\jdk1.6.0_21\demo\jfc\SwingSet2"`.

To run the demo application, start a command prompt and enter the following:
`java -jar SwingSet2.jar`

After a quick startup displaying a splash screen, the application should look similar to the following.



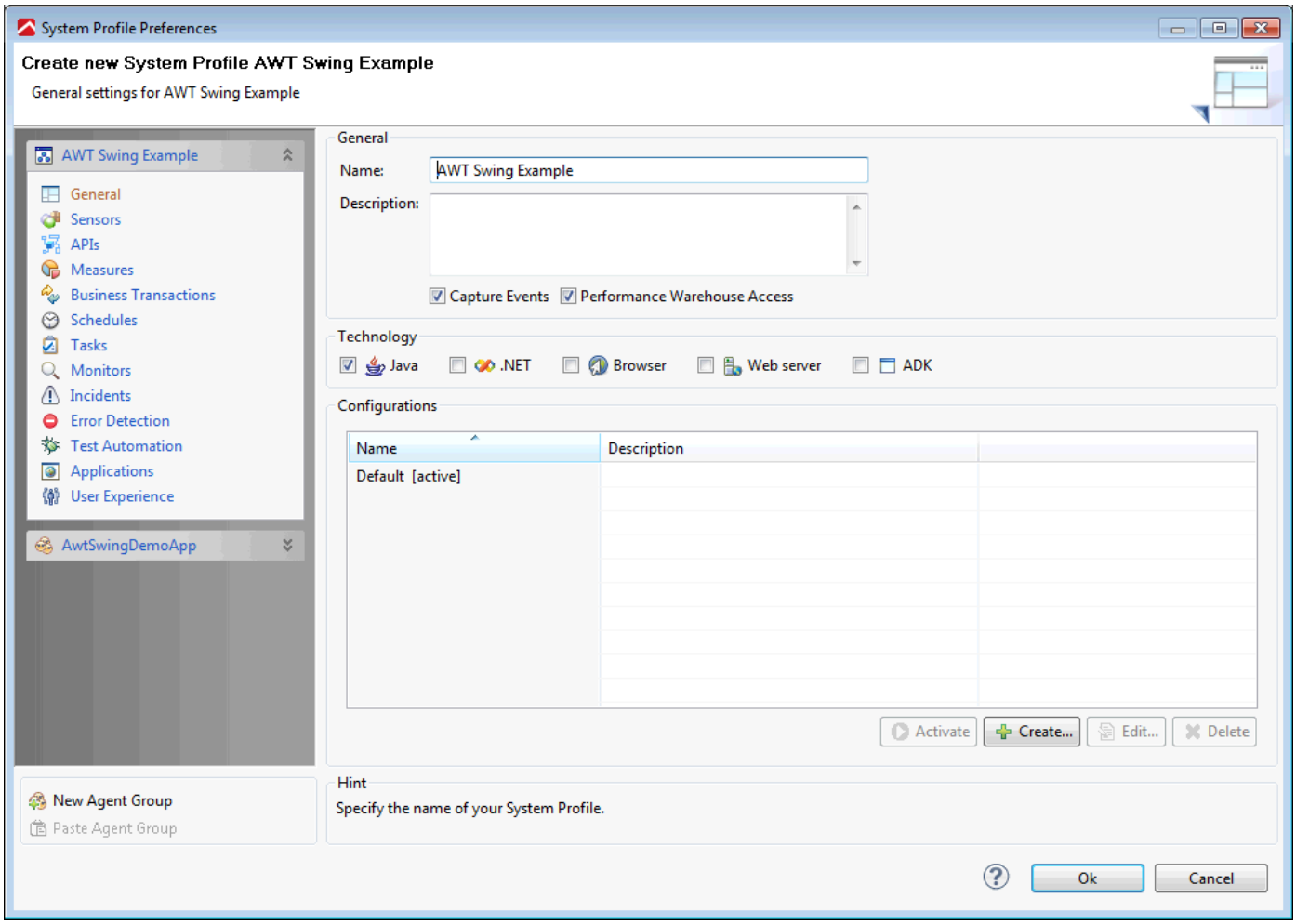
Agent injection

With the demo AWT/Swing sample application running, inject the AppMon agent. See Application Environment Configuration for more about instrumenting Java or .NET applications.

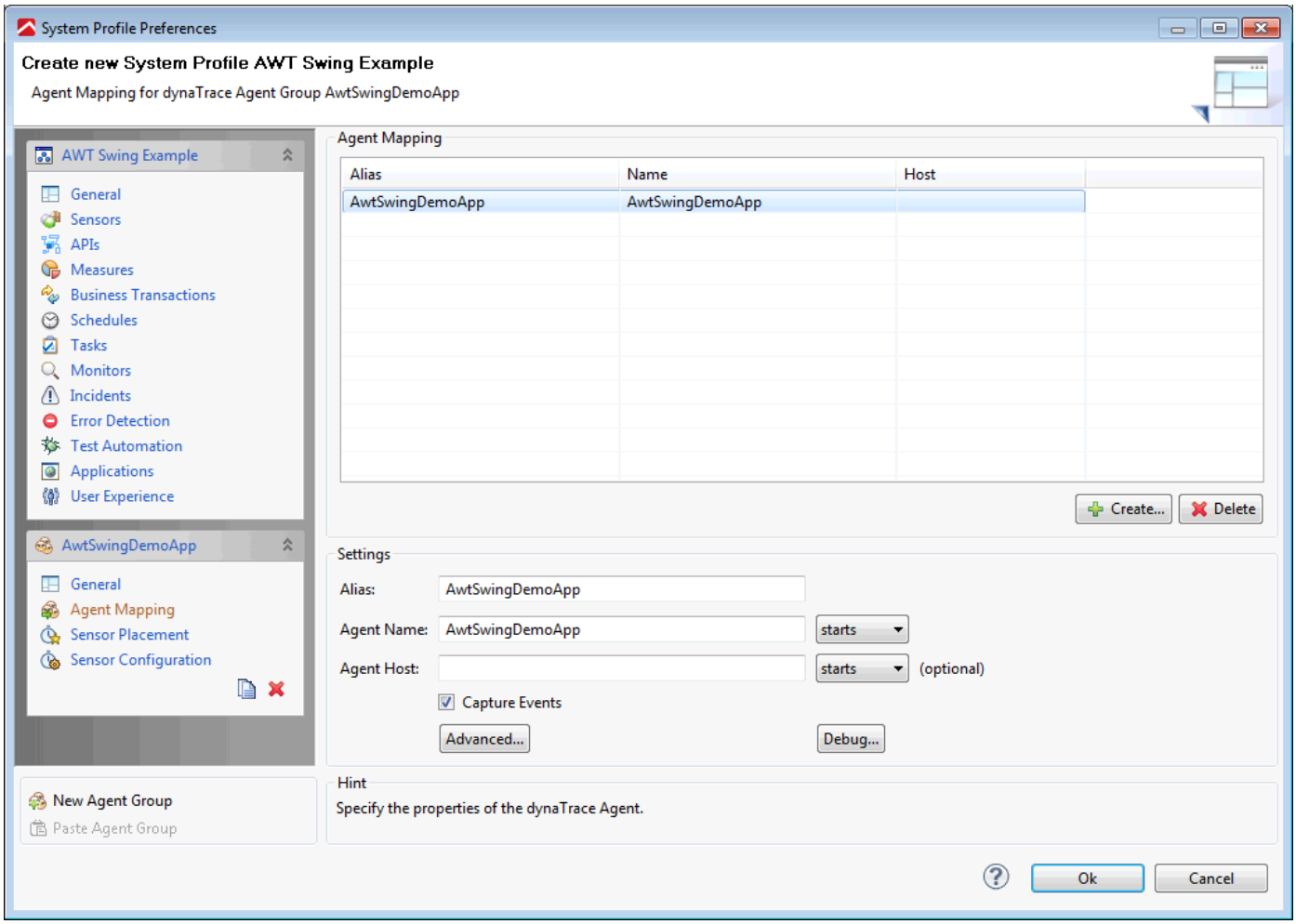
Instead of creating a new System Profile from scratch, you can download the System Profile from the table in the Download section on this page. The prepared System Profile contains the Agent Mapping as well as all measures and Business Transactions needed for this walkthrough.

Perform the following to inject the agent.

1. Create a System Profile for the Swing Demo application, as shown in the following.



2. Create an agent group/mapping that connects an agent with name **AwtSwingDemoApp**.

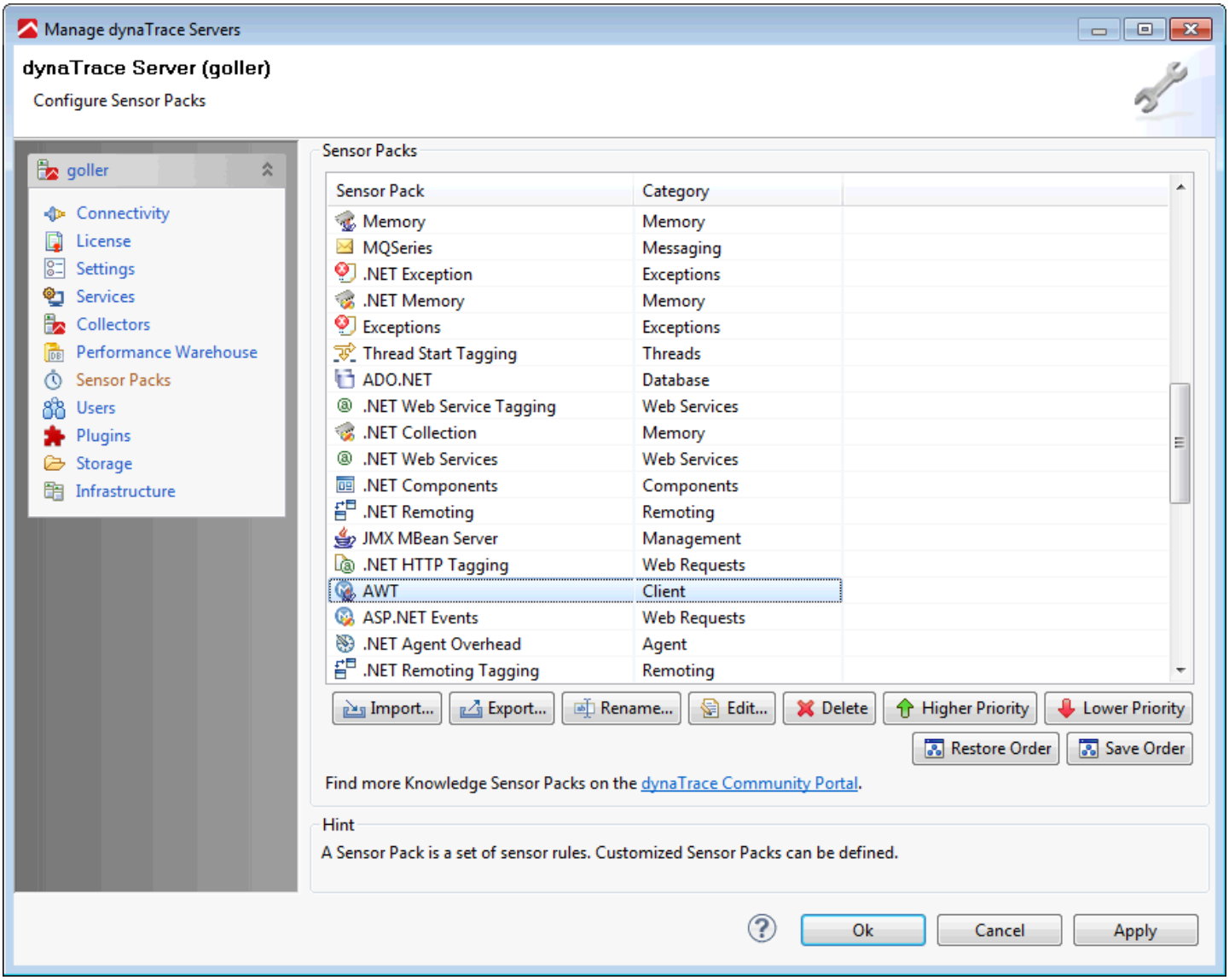


3. Enter the following to inject the AppMon Agent into the Swing Demo application.
`java -agentpath:"C:\Program Files (x86)\dynaTrace\dynaTrace 7.0\agent\lib64\dtagent.dll"=name="AWTSwingDemoApp",server=localhost:9 -jar SwingSet2.jar`.

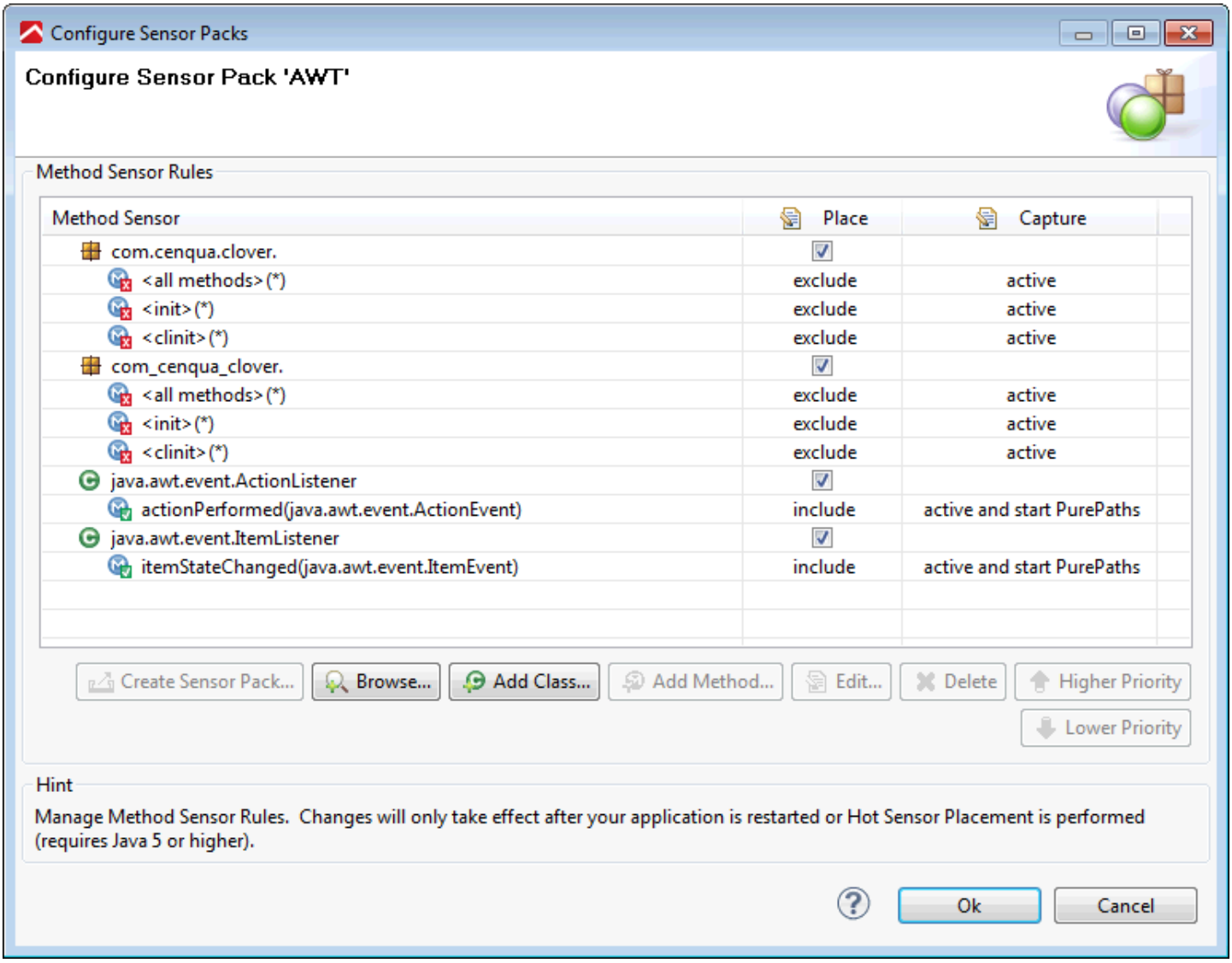
The agent `lib` directory must match your platform and JVM architecture (x86 / x64).

Default AWT sensor pack

A default AppMon AWT Sensor Pack is already activated for the new System Profile. This AWT Sensor Pack is activated by default for all System Profiles using JAVA technology, as shown in the following graphic.


Manage dynaTrace Servers
dynaTrace Server (goller)
Configure Sensor Packs
goller
Connectivity
License
Settings
Services
Collectors
Performance Warehouse
Sensor Packs
Users
Plugins
Storage
Infrastructure
Sensor Packs
Sensor Pack
Category
Memory
Memory
MQSeries
Messaging
.NET Exception
Exceptions
.NET Memory
Memory
Exceptions
Exceptions
Thread Start Tagging
Threads
ADO.NET
Database
.NET Web Service Tagging
Web Services
.NET Collection
Memory
.NET Web Services
Web Services
.NET Components
Components
.NET Remoting
Remoting
JMX MBean Server
Management
.NET HTTP Tagging
Web Requests
AWT
Client
ASP.NET Events
Web Requests
.NET Agent Overhead
Agent
.NET Remoting Tagging
Remoting
Import...
Export...
Rename...
Edit...
Delete
Higher Priority
Lower Priority
Restore Order
Save Order
Find more Knowledge Sensor Packs on the dynaTrace Community Portal.
Hint
A Sensor Pack is a set of sensor rules. Customized Sensor Packs can be defined.
Ok
Cancel
Apply

The default AWT Sensor Pack attends to Action events and ItemSelection events. The action event can be considered a universal event, which may be attached to multiple user interface elements such as buttons, menu items, combo boxes, or other controls. The event does specify an action for the Action event, but it is typically used for mouse clicks or other user interface actions. The ItemSelection event is mostly used for combo boxes, lists, and other controls.



The problem with the default AWT Sensor Pack starts when taking a closer look on how most developers typically work with AWT and Swing. There are many more event types than just the ActionListener and ItemListener. For example, a developer may use an ActionListener to get a simple notification when a button is clicked. Alternatively the MouseListener (which is not part of the AppMon default AWT Sensor Pack) can be notified about a button click. The MouseListener is much more specific than the ActionListener. It provides additional data such as X / Y coordinates, single / double / right button click, and more.

How the application is programmed determines the listeners to track with AppMon. The ActionListener and the ItemListener may not be sufficient for some real world applications, so you may want to create your own AWT Sensor Pack to cover more user interface event types than what the default AWT Sensor Pack provides.

Creating your own AWT sensor pack

You can create your own AWT Sensor Pack to define a wider range of AWT user interface entry points. The following table lists available AWT events. Use this to determine which events you need for instrumenting real user interactions.

Package: java.awt.event	ActionEvent	ActionListener	AdjustmentEvent
AdjustmentListener	AWTEventListener	AWTEvenListenerProxy	ComponentAdapter
ComponentEvent	ComponentListener	ContainerListener	FocusAdapter
FocusEvent	FocusListener	HierarchyBoundsAdapter	HierarchyBoundsListener
HierarchyEvent	HierarchyListener	InputEvent	InputMethodEvent
InputMethodListener	InvocationEvent	ItemEvent	ItemListener
KeyAdapter	KeyListener	MouseAdapter	MouseEvent
MouseListener	MouseMotionAdapter	MouseMotionListener	MouseWheelEvent
MouseWheelListener	NativeLibLoader	PaintEvent	TextEvent
TextListener	WindowAdapter	WindowEvent	WindowFocusListener
WindowListener	WindowStateListener		

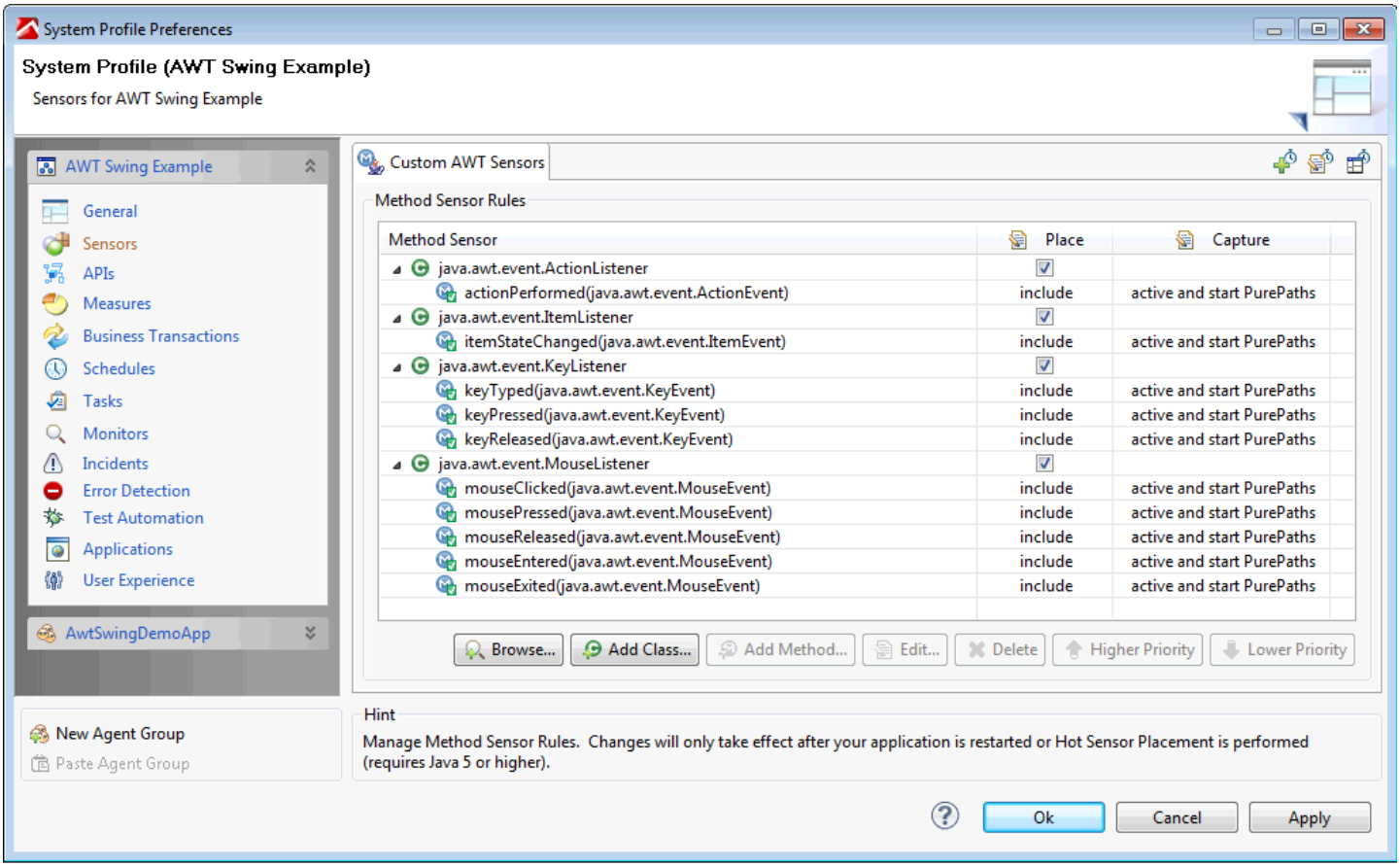
A Listener class provides the hooking machanism to fetch the Event itself. Listener classes (like MouseListener) are usually interfaces and may contain one or more hook methods such as `mouseClicked(EventObject)`, `mousePressed(EventObject)`, or `mouseReleased(EventObject)`. An Event object class such as MouseEvent is a simple object containing event attributes such as X/Y coordinates. This Event object procides event details when passed to the hook method.

There are also some abstract Adapter classes such as MouseAdapter), but they are usually only empty implementations of the corresponding Listener class. This was mainly done to help developers avoid implementing the whole interface, and just inherit from the Adapter class and override the required methods.

From the list of event types above, the following listeners are typically used most often:

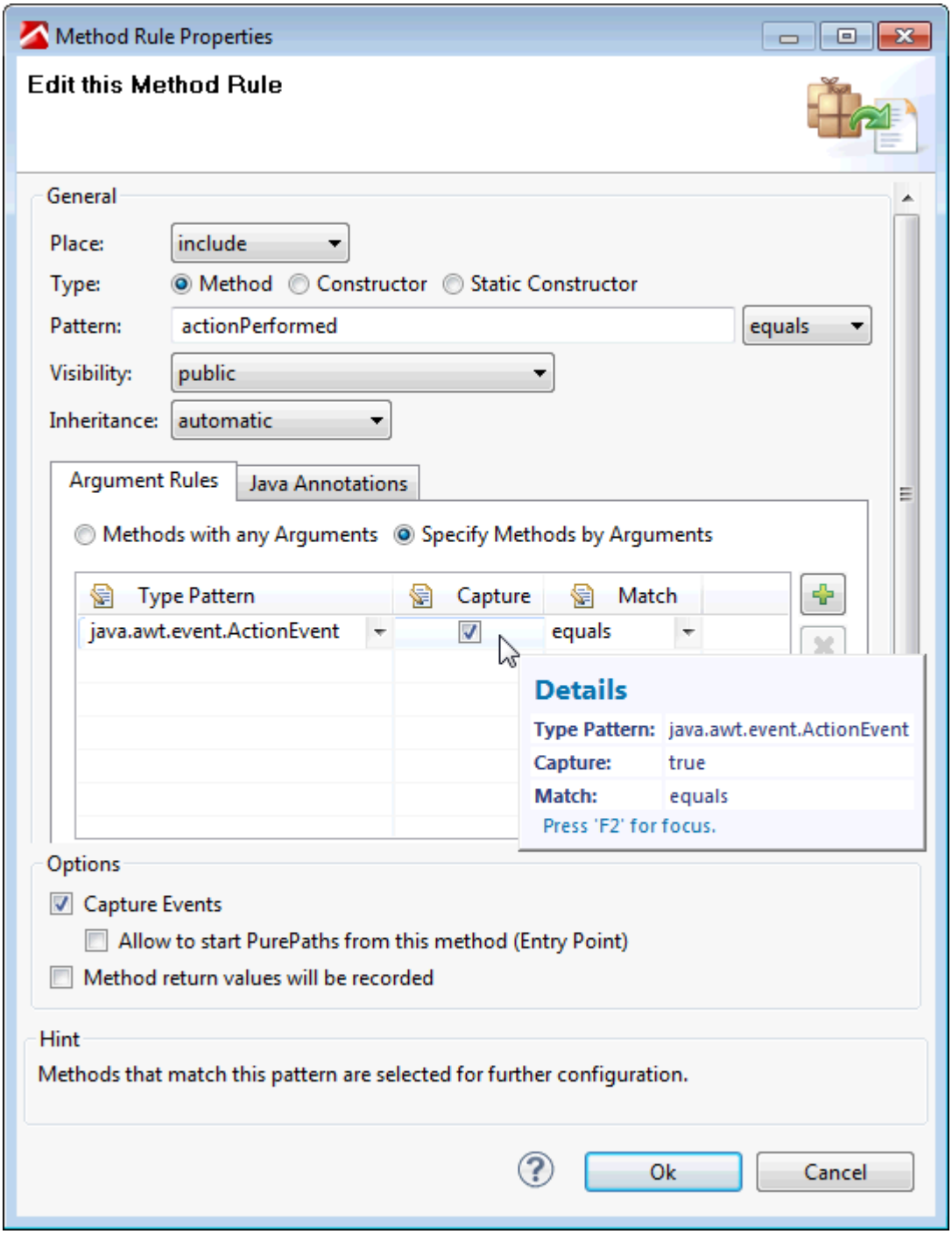
Listener Class	Method
ActionListener	public void actionPerformed(ActionEvent e)
ItemListener	void itemStateChanged(ItemEvent e)
MouseListener	public void mouseClicked(MouseEvent e) public void mouseClicked(MouseEvent e) public void mousePressed(MouseEvent e) public void mouseReleased(MouseEvent e)
KeyListener	public void keyTyped(KeyEvent e) public void keyTyped(KeyEvent e) public void keyPressed(KeyEvent e) public void keyReleased(KeyEvent e)

After creating a custom Sensor Group within the System Profile, it should look like this:

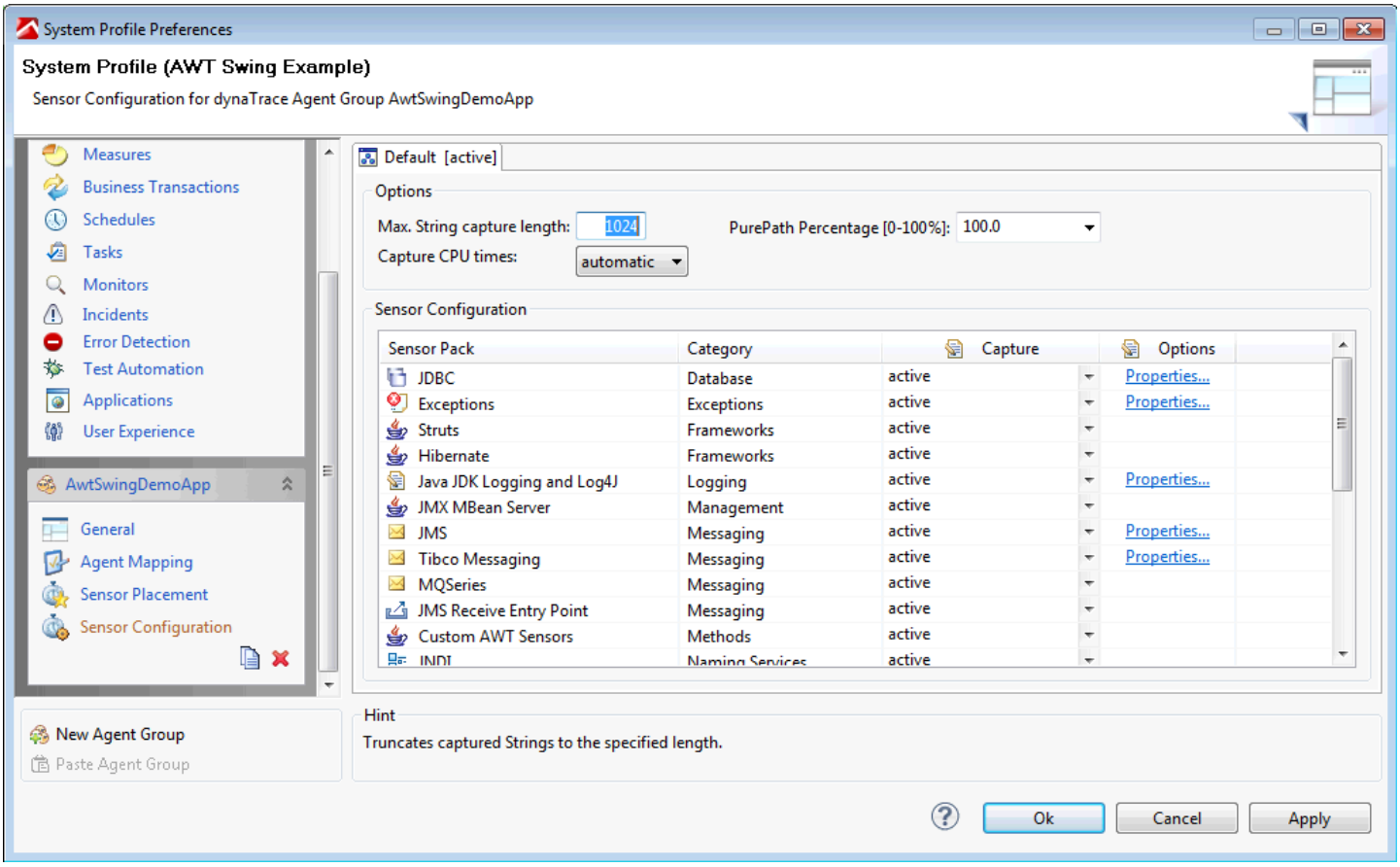


You have now created basic AWT sensors. It is important to turn on argument capturing for the listeners arguments (such as ActionEvent e, ItemEvent e, etc.). Doing this lets you extract helpful additional information, like the name of the pressed UI button, or the pressed key on the keyboard. Use this extra information to set up some business transactions. These business transactions help transform the technical events (PurePaths) into something human readable and understandable.

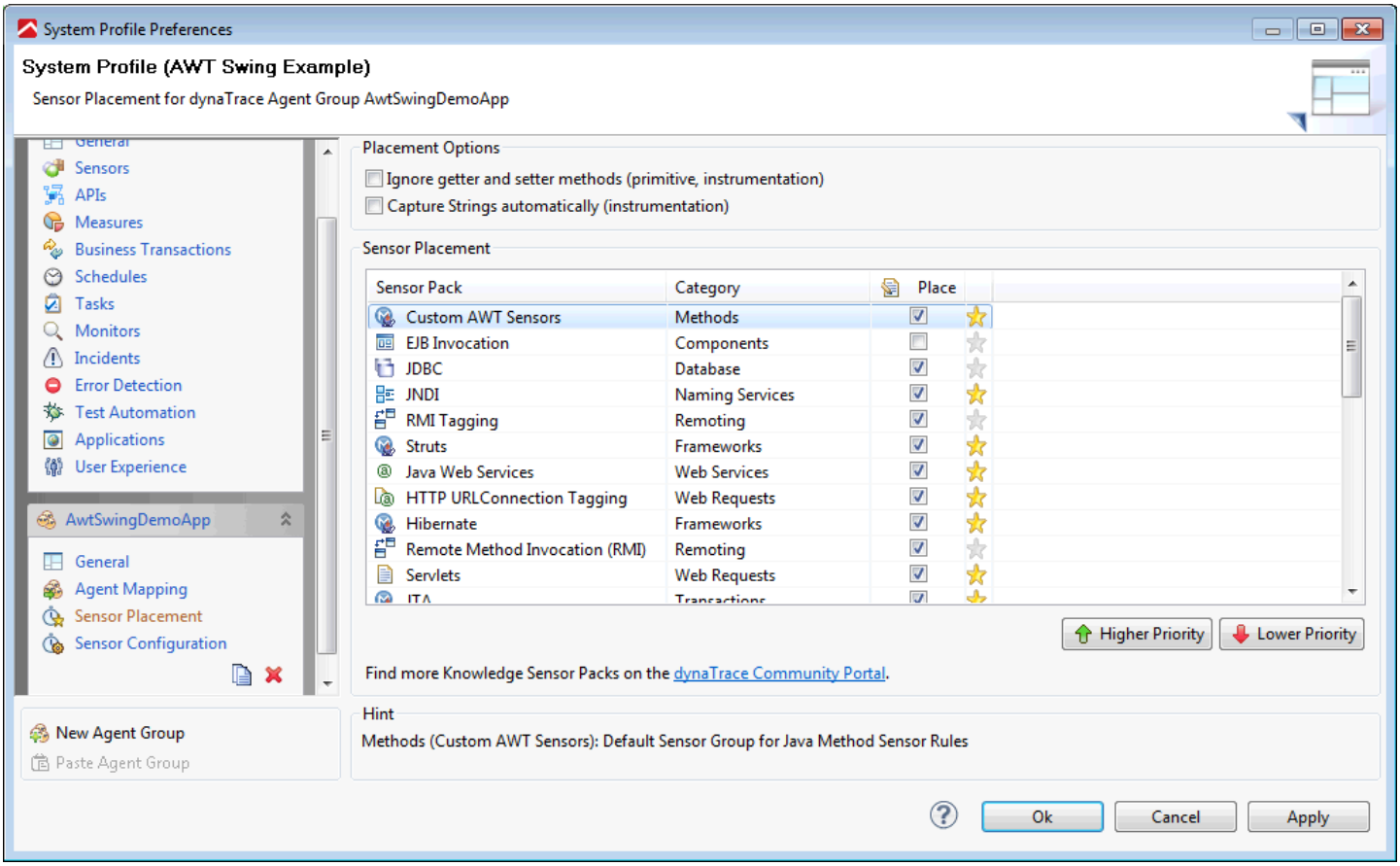
Enable argument capturing for every method sensor rule defined in the new Sensor Group. The following graphic shows an example



Increase the size of the captured strings from 256 to 1024 characters since the toString() method of the events may deliver fairly long strings.

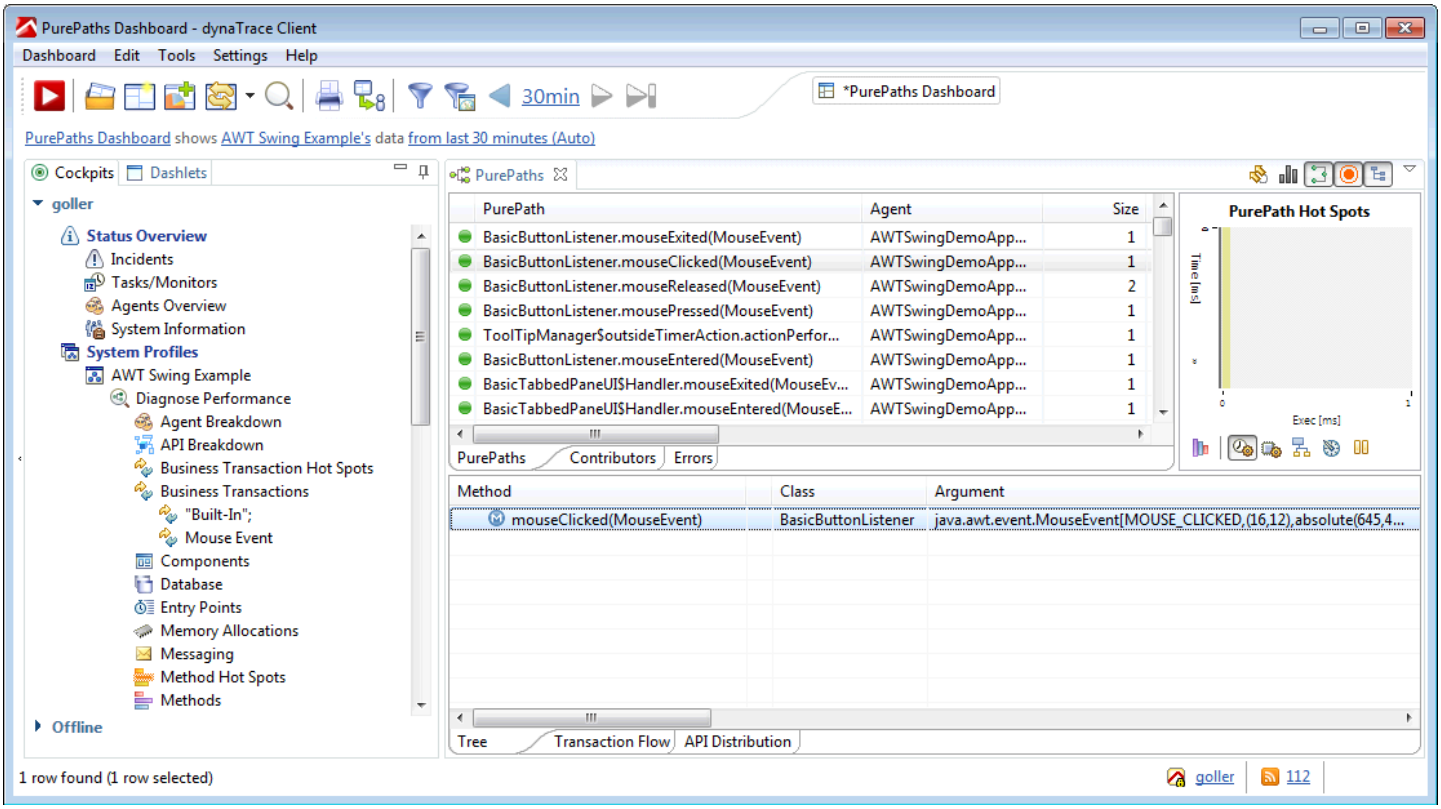


Deactivate the default AWT Sensor Pack from the System Profile and activate the new Sensor Group.



Catching PurePaths

Next, check the PurePaths being captured. Restart the swing demo application and click some buttons, then open the PurePaths dashboard for the System Profile. It should look similar to the following.



You may see many PurePaths with size “1”. This means the listener was triggered, but there was no business logic behind that listener. Unfortunately, the Swing Demo Application does not have much code behind the user interface controls anyway. So at this point, it’s good to have many PurePaths, and to see seeing the event object as string in the argument column as shown above.

Business Transactions

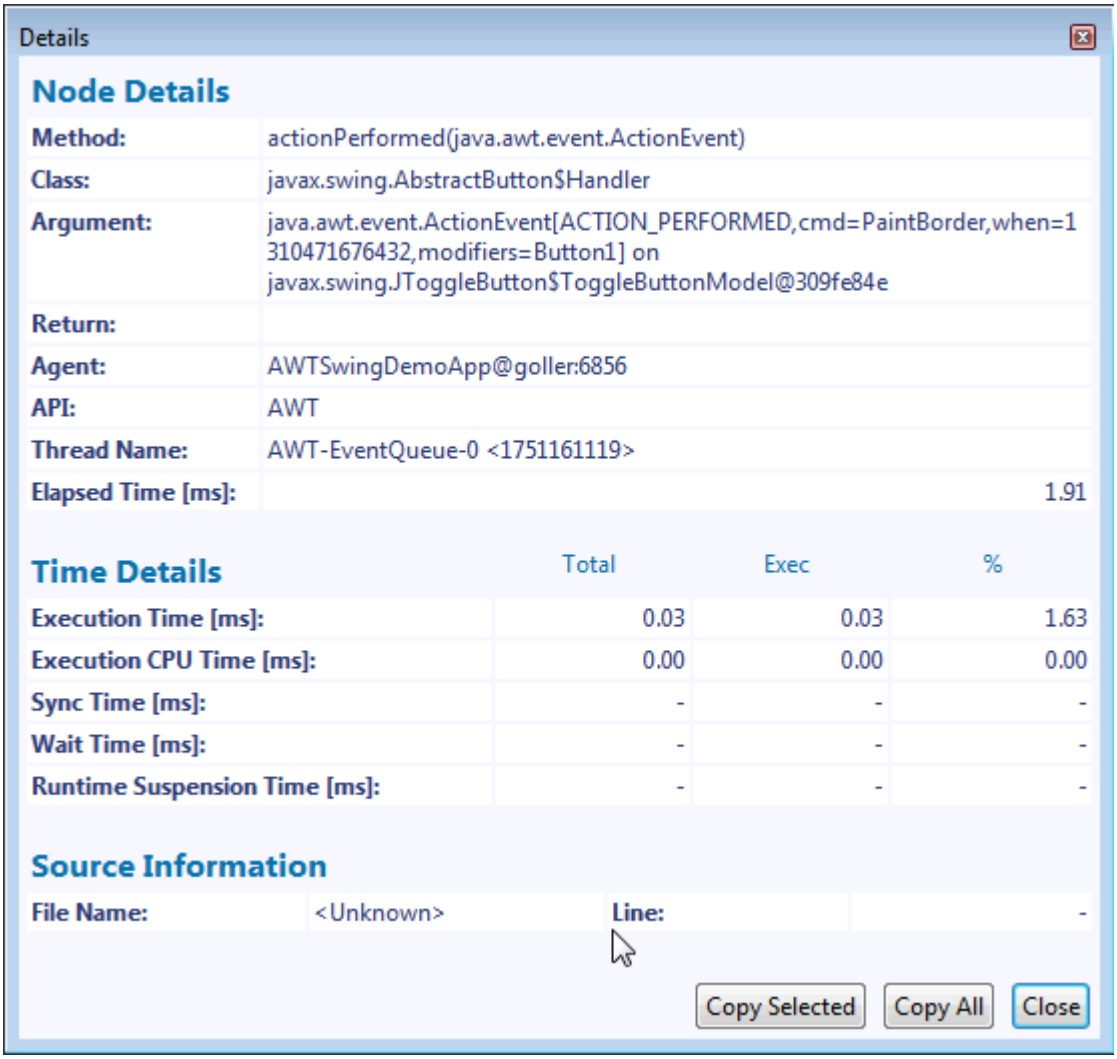
At this point in the walkthrough, many user interface events are captured. For a better overview, you can define some business transactions for the different event types. The following sections describe how to set up these business transaction and how they work.

User interface action events

As previously stated, the ActionEvent is a common universal event you can attach to many user interface controls. What defines an “action” and when it is triggered is handled by the component providing support for Action event Listeners. This means an “action” definition may vary from component to component. For buttons, it’s a simple button click.

Event are primitive but flexible. It can’t provide additional meta information about the triggered action such as mouse coordinates. Because of this, your use of ActionEvents may vary from application to application, or from business case to business case, depending on whether you desire simple listeners or more specific listeners.

The following is an example how an ActionEvent may look when captured.



The example’s argument looks like this:

```
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=**PaintBorder**
,when=**1310471676432** ,modifiers=Button1]
on**javax.swing.JToggleButton** $ToggleButtonModel@309fe84e
```

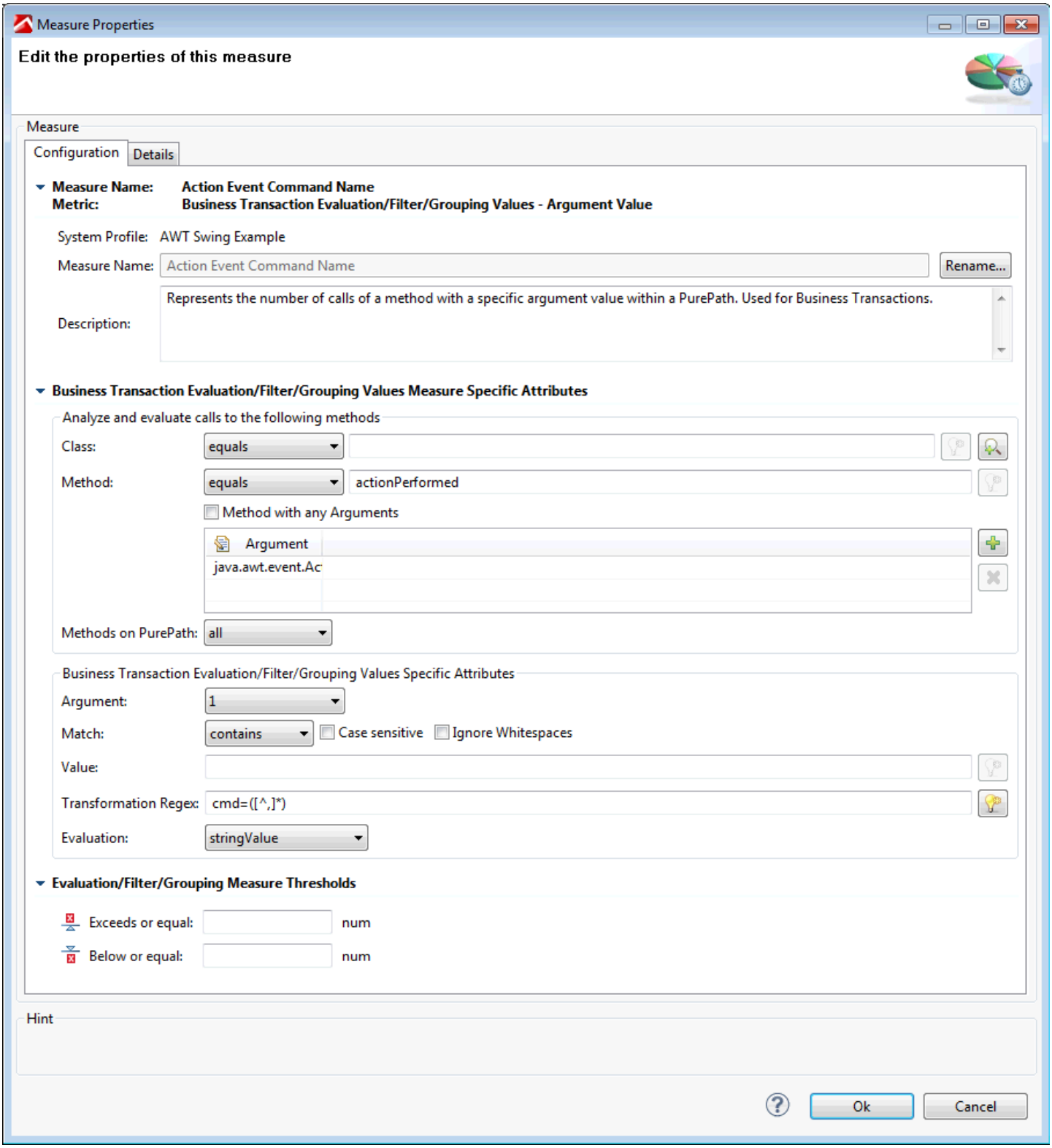
The most important part of an action event is the action command ("**PaintBorder**" parameter). This usually describes the command. It is not mandatory to implement, so its value may be "null" for some applications. Another interesting parameter is the timestamp, which is when the action event was triggered ("**1310471676432**"). The timestamp provides a chronological view of all actions. This may lead to an unintended number of measures. Another interesting parameter is the UI control (**javax.swing.JToggleButton** in the screenshot above) which triggered the action event.

Define a Business Transaction

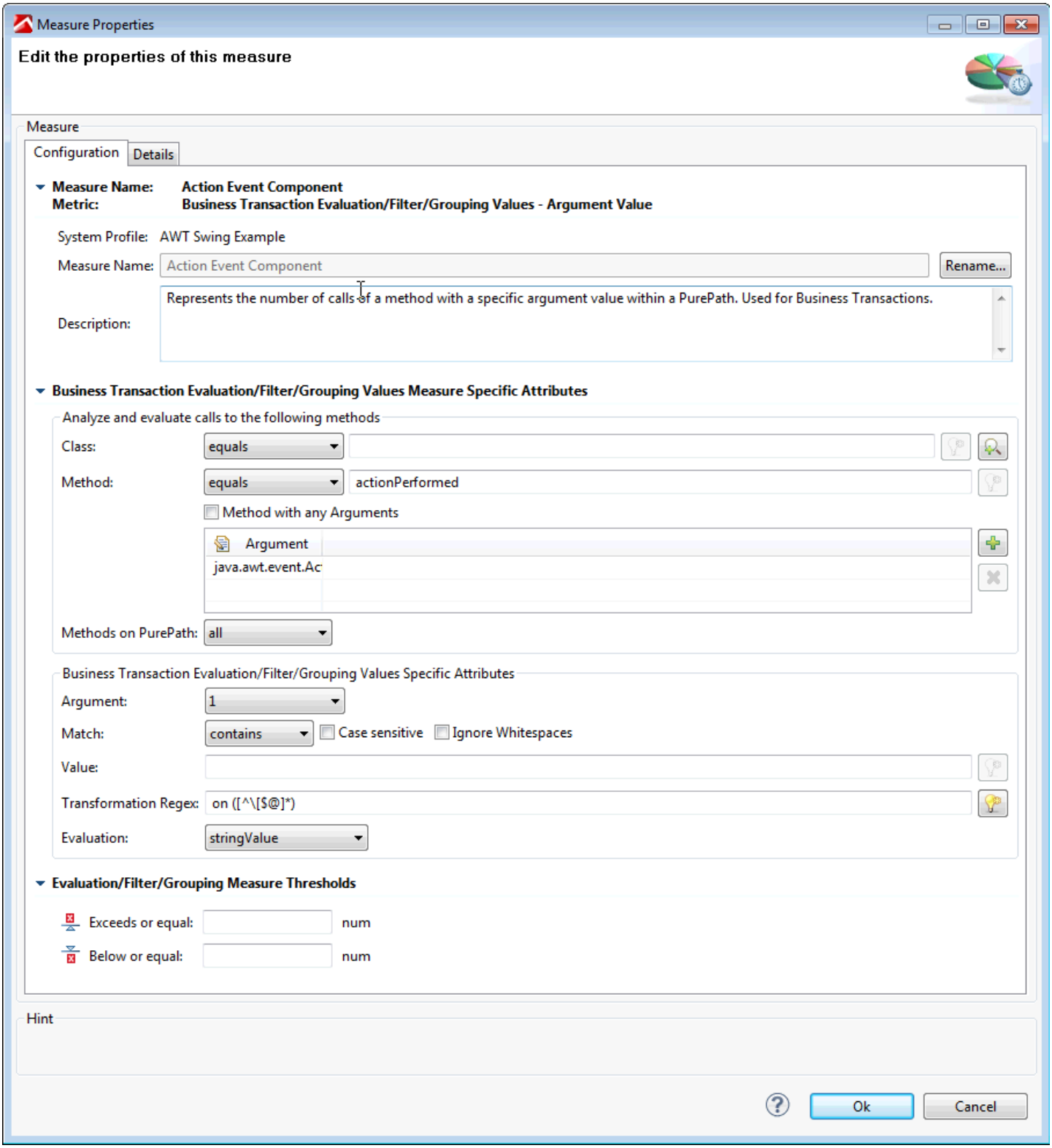
Measures are required to define a business transaction. In this case, Measures are split, which filters only these PurePaths which contain the above described ActionEvent. The result is then further split to list of all PurePaths containing only the ActionEvent by the individual values described above. That result defines this business transaction.

Learn more about business transactions.

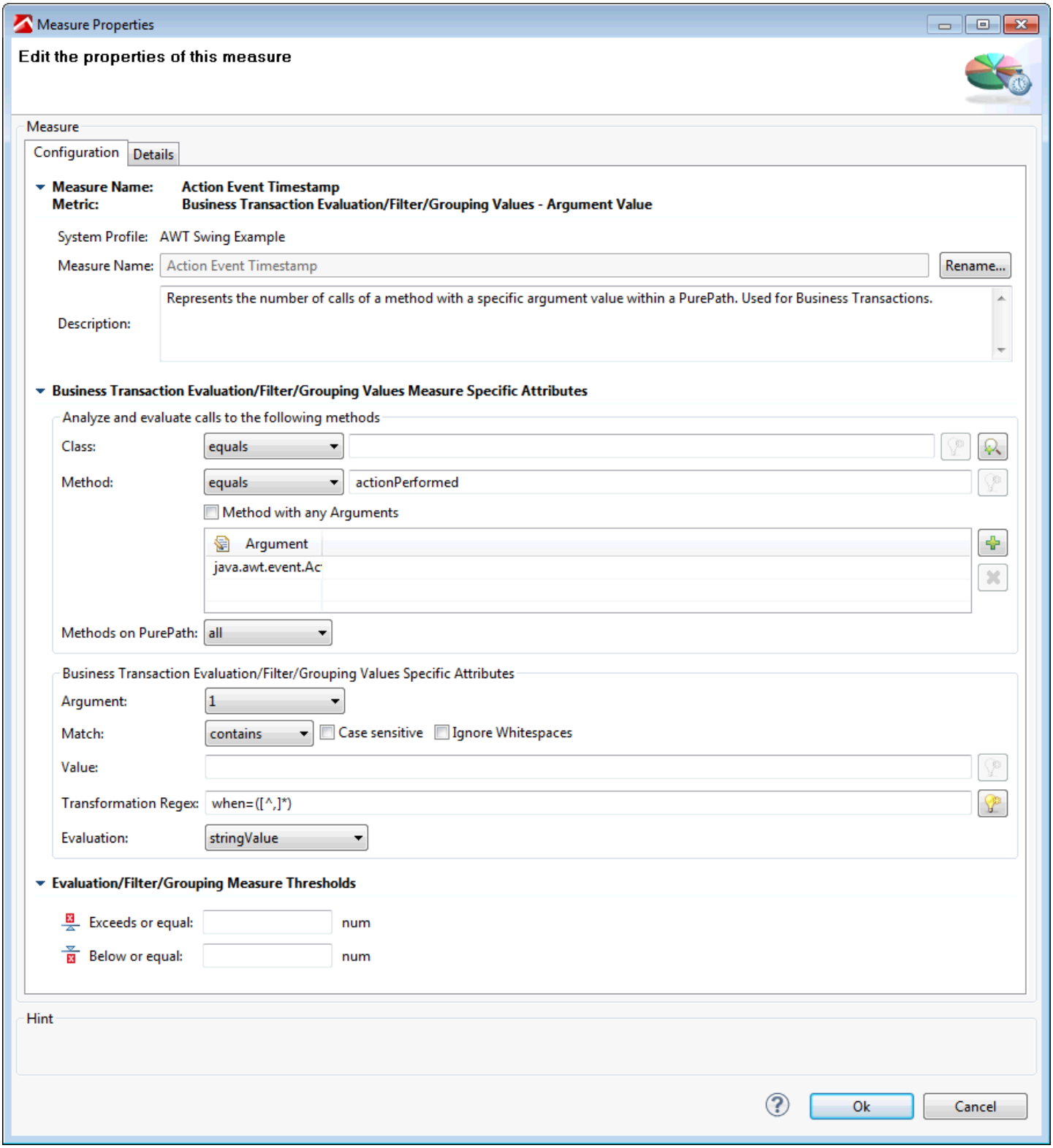
Splitting Measure for the Action Event Command Name:



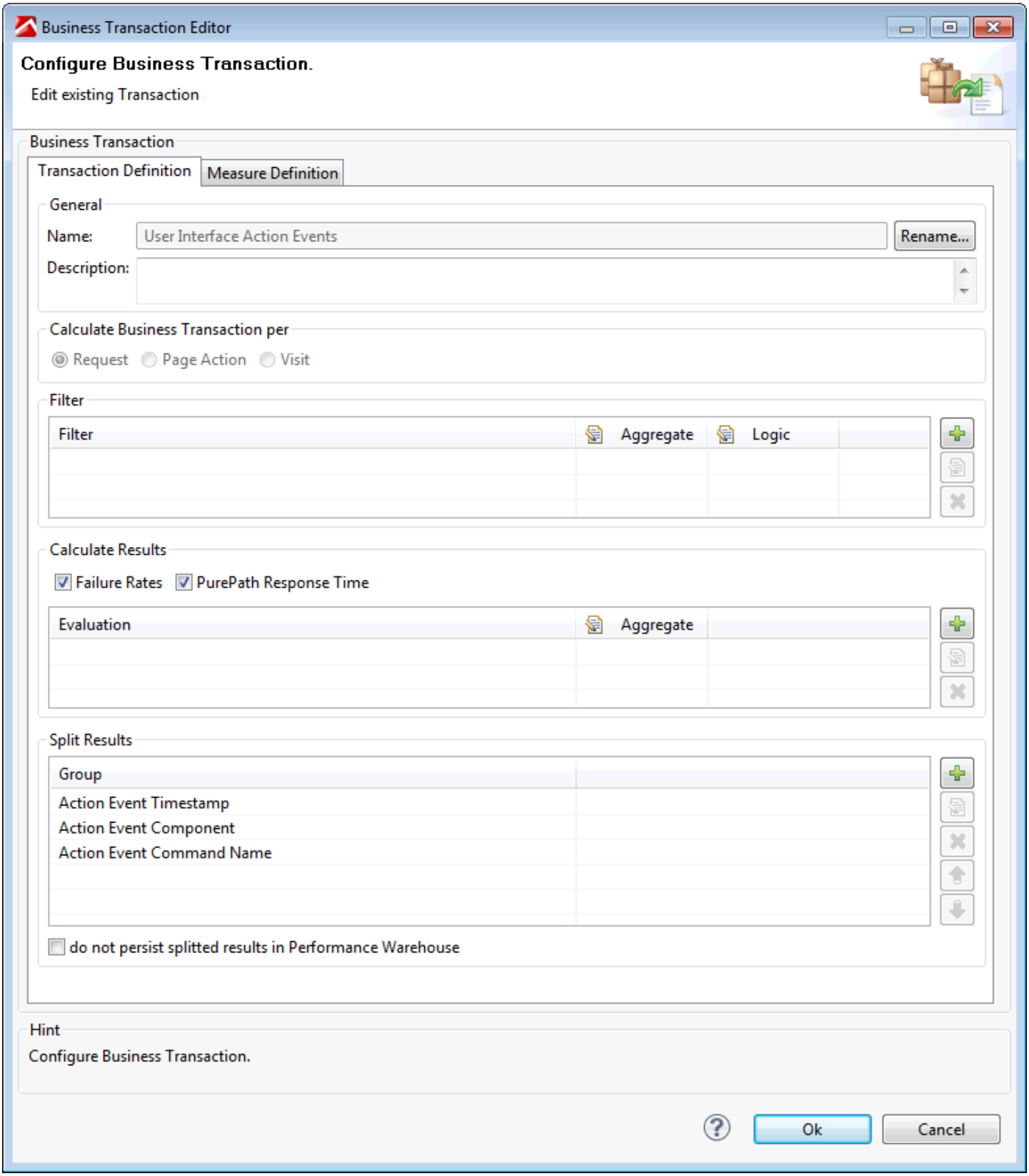
Splitting Measure for the Action Event Component:



Splitting Measure for the Action Event Timestamp:

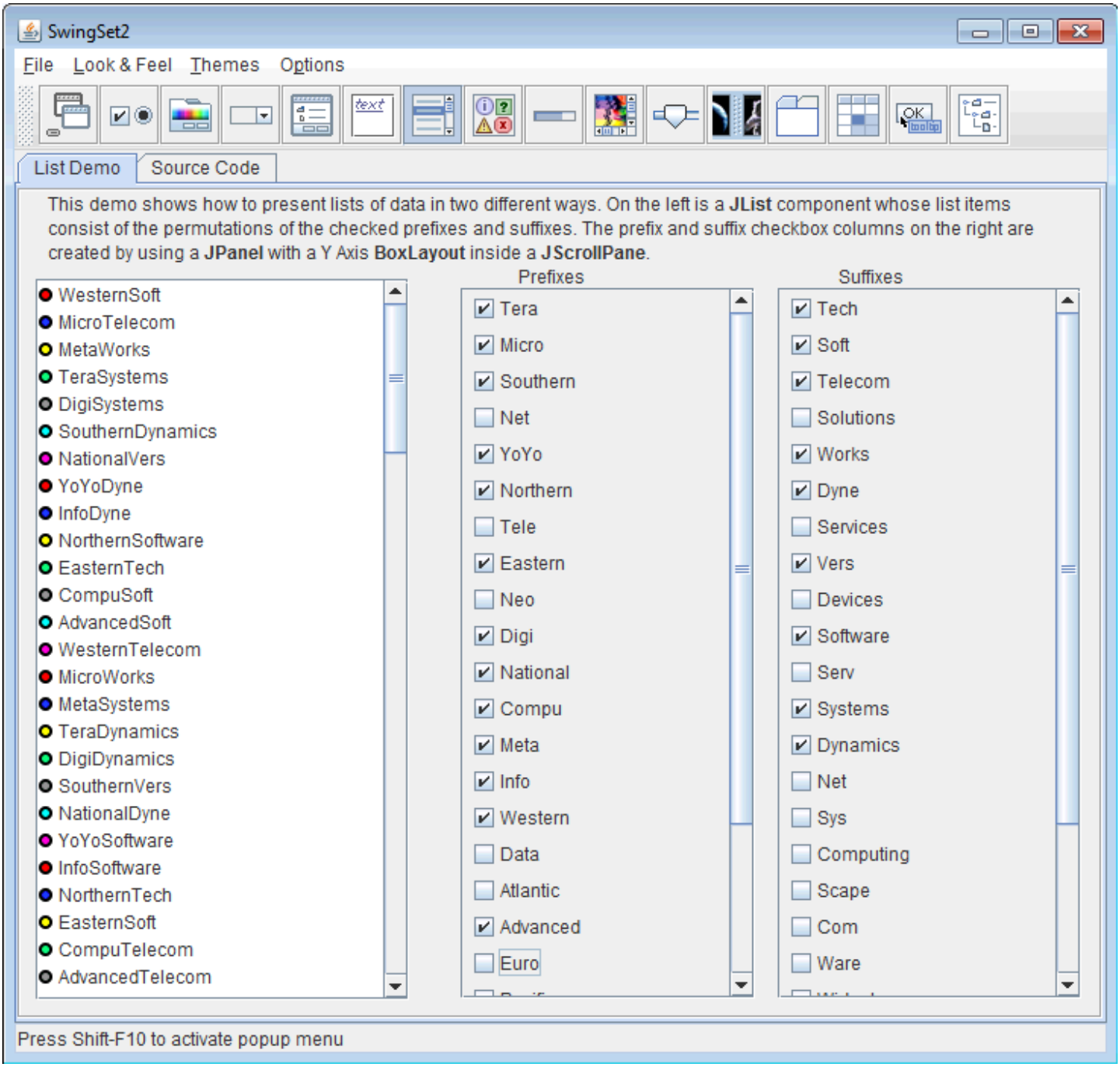


This is the definition of the Action Event business transaction, which uses all splitting measures for grouping:

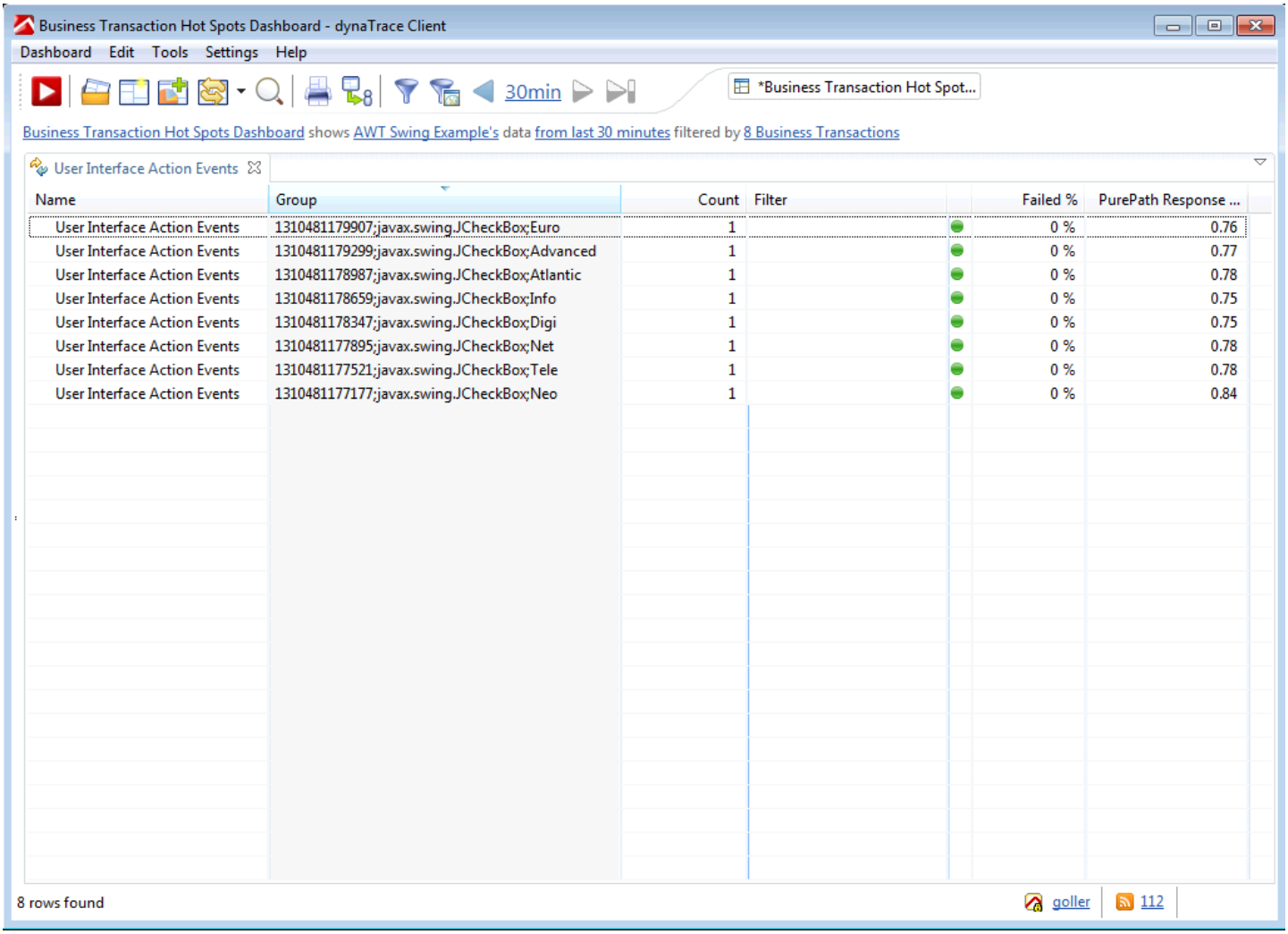

Business Transaction Editor
Configure Business Transaction.
Edit existing Transaction
Business Transaction
Transaction Definition
Measure Definition
General
Name:
User Interface Action Events
Rename...
Description:
Calculate Business Transaction per
Request
Page Action
Visit
Filter
Filter
Aggregate
Logic
Calculate Results
Failure Rates
PurePath Response Time
Evaluation
Aggregate
Split Results
Group
Action Event Timestamp
Action Event Component
Action Event Command Name
do not persist splitted results in Performance Warehouse
Hint
Configure Business Transaction.
Ok
Cancel

Example

After the business transaction is defined, the result should look like the screenshot below when clicking around in the Swing Demo Application. The List Example was used for clicking, as this provides at least some command names. The Swing Demo Application is only a showcase for user interface controls, so developers did not spend time on setting proper command action names.



… looks like this in AppMon:

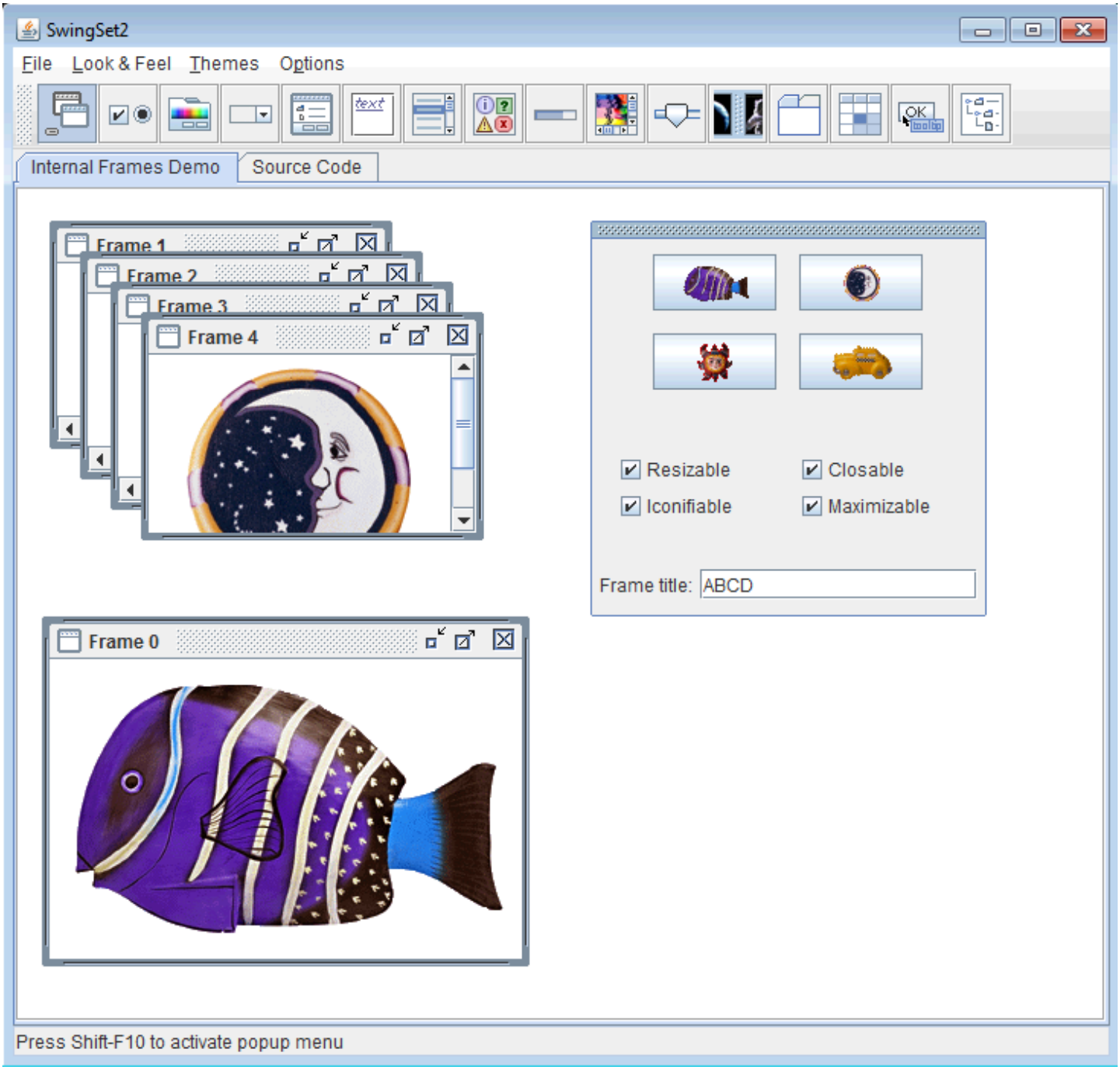


The first value in the “group” column is the timestamp (in numeric format) from our Action Event Timestamp measure. Sorting the “group” column gives you a chronological list of what the user has clicked. It’s sometimes very helpful seeing what the user has clicked in the correct order/timeline. The second value in the “group” column shows the component type which the user has clicked. JCheckBox in this example. This value comes from our Action Event Command Name measure.

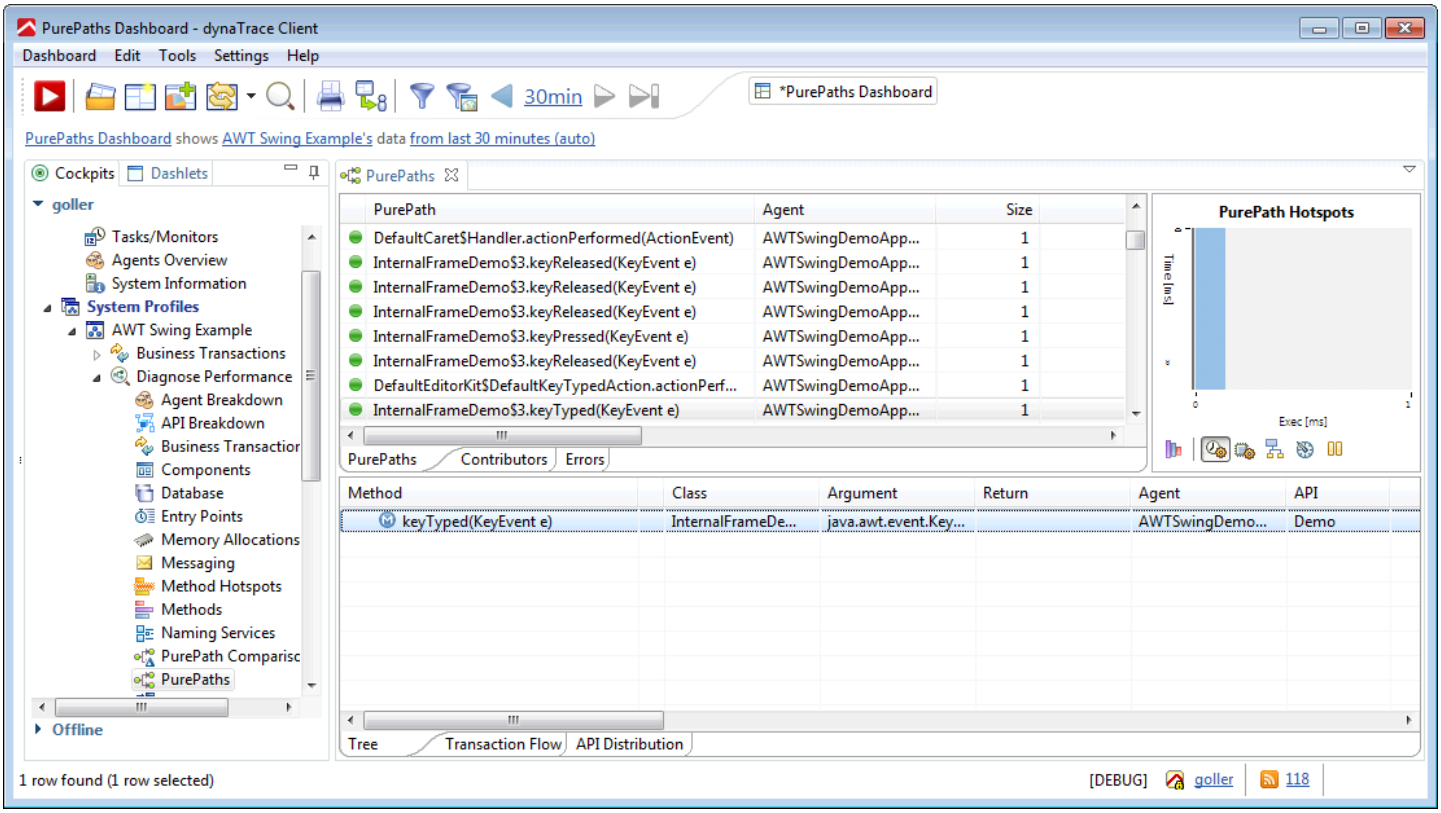
The last value of the “group” column shows the action command name, in the example above, the name of the clicked CheckBox is shown. This value comes from the Action Event Command Name measure we defined for the business transaction.

User interface keyboard events

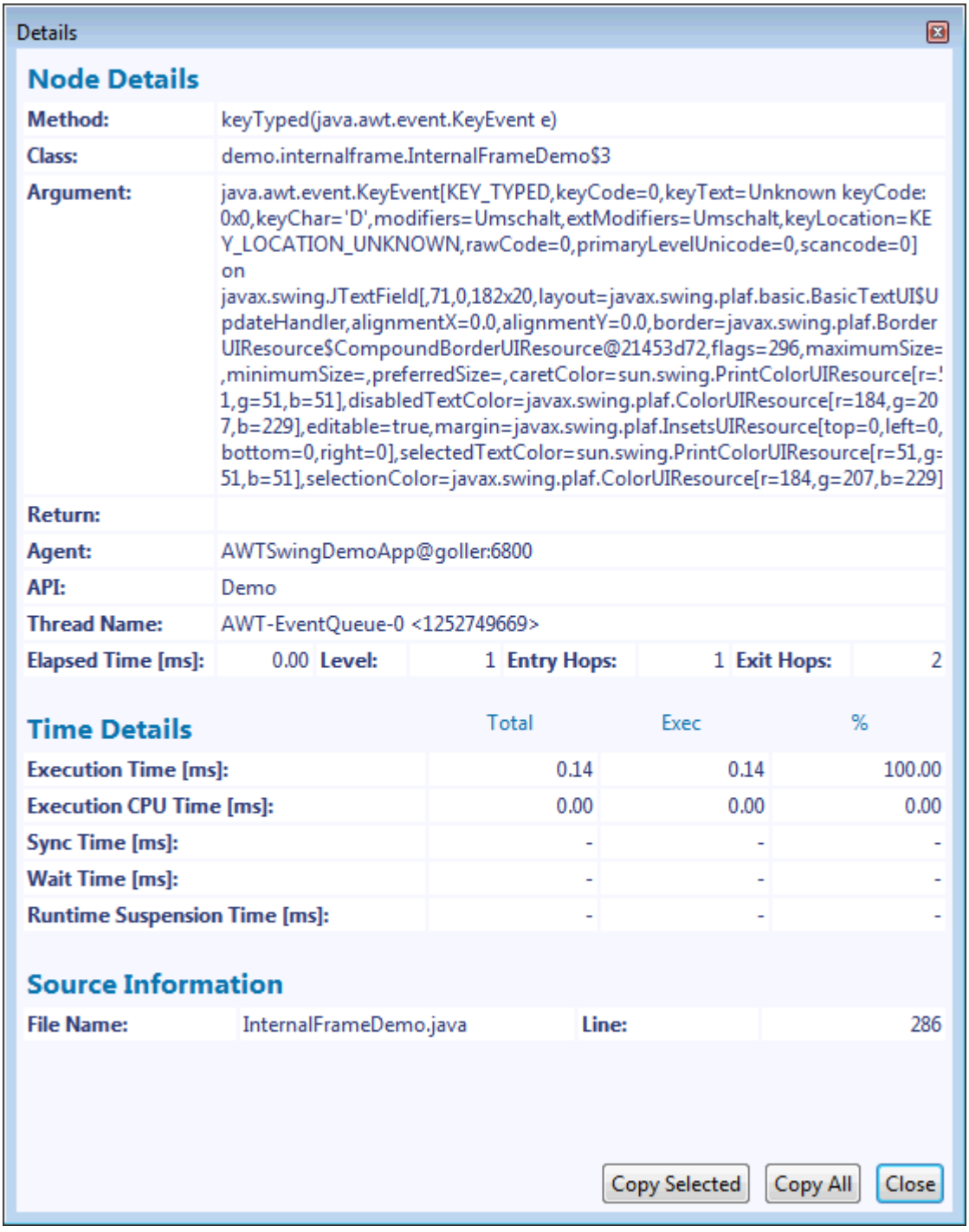
Unfortunately, the Swing demo app does not use key listeners. Most functionality is covered with action listeners. After a quick source code modification and attaching a KeyListener to the first TextField (“Frame Title”), you can see the following PurePaths:



The following screenshot shows a PurePath for the keyReleased Event. The business transaction uses keyTyped() as this is probably the most common event. In this example, A-B-C-D was typed in the Frame Title input field, then ESC and RETURN were hit:



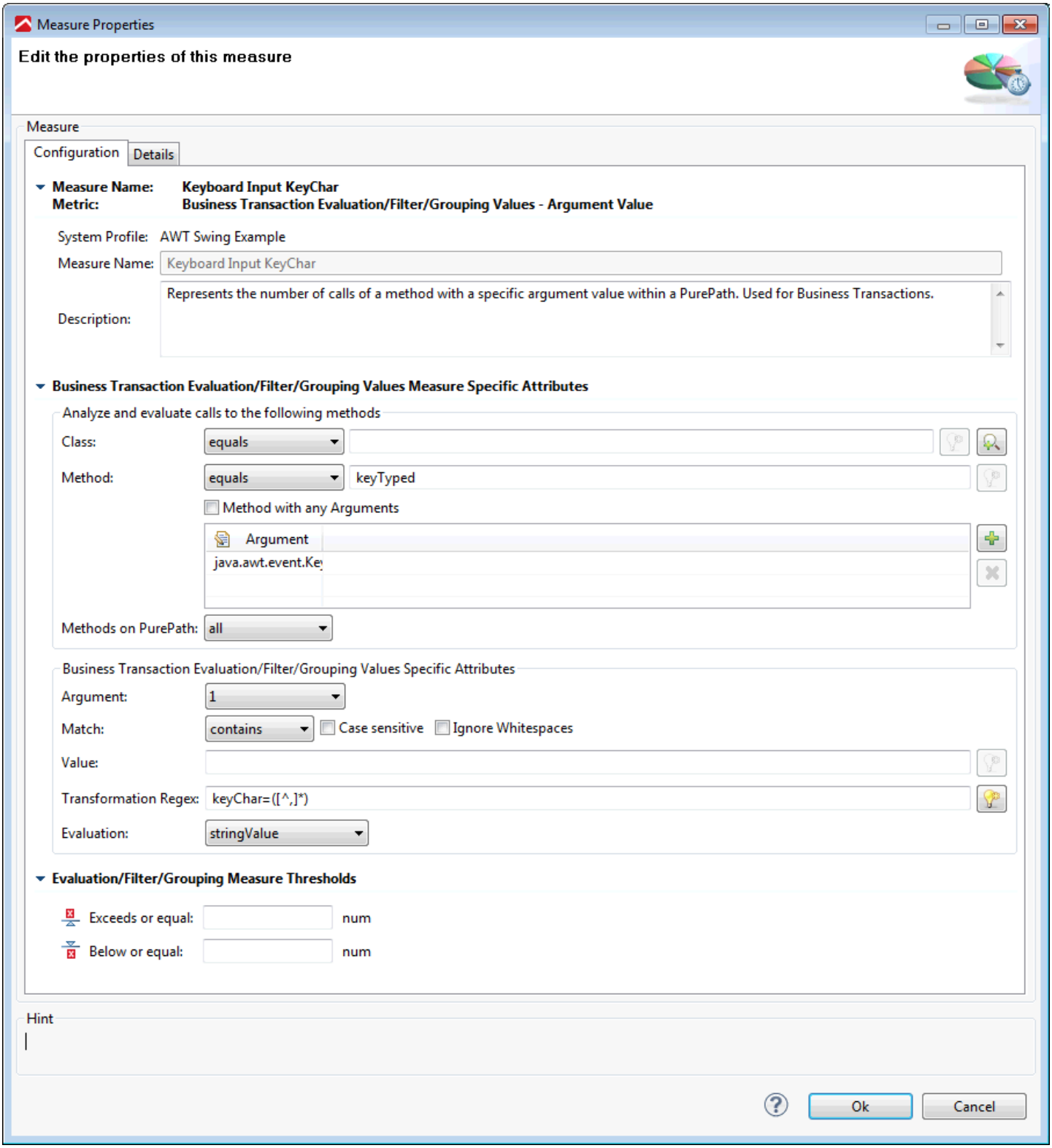
The argument of this example appears in the node details.



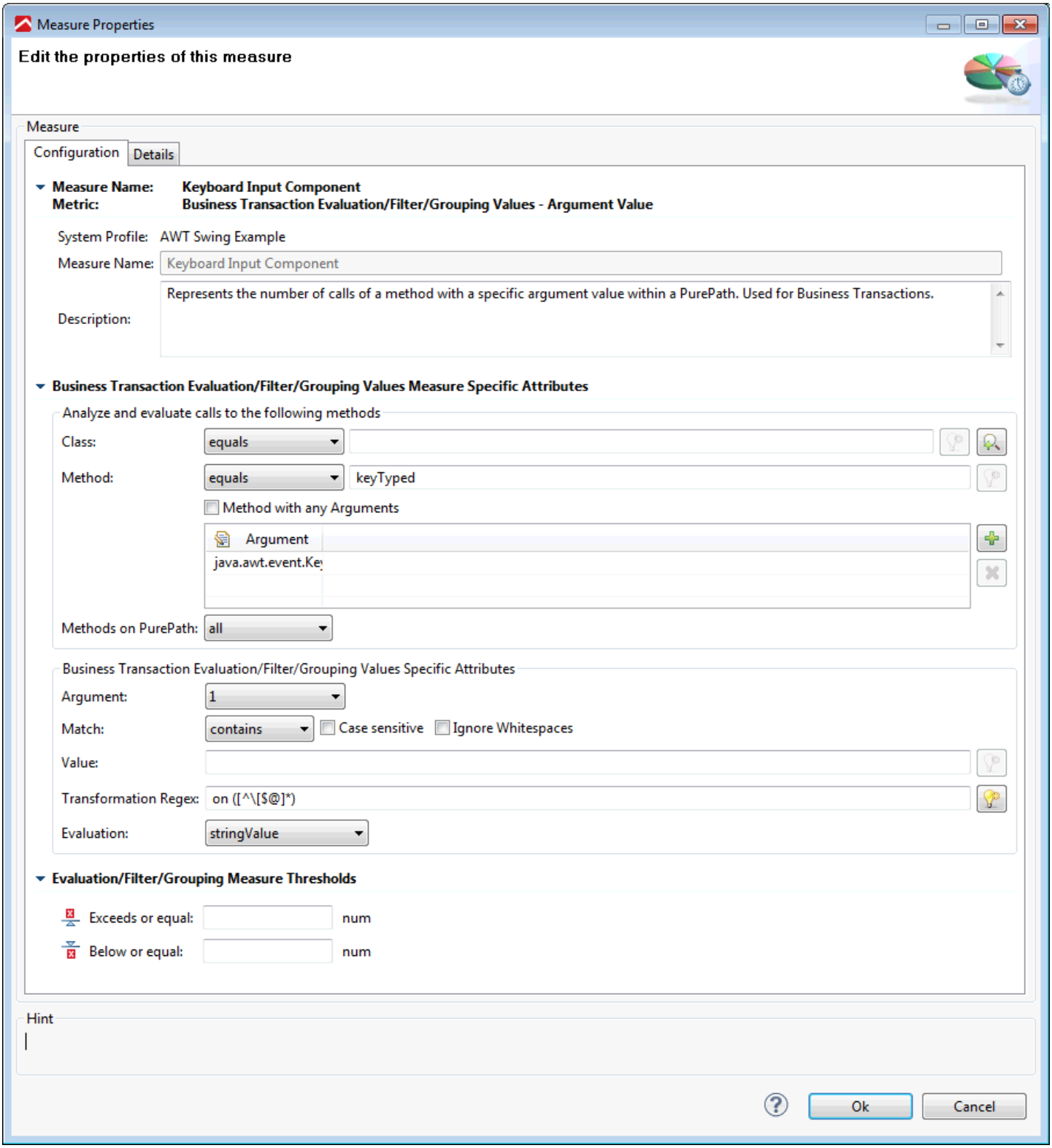
The most important part of the keyboard event is the pressed keyboard character ("**keyChar=**" parameter). In the example above, the 'D' key was pressed on the keyboard. The second interesting parameter is the component, which received the keyboard input (on **javax.swing.JTextField**). If the component has a proper name, it displays here instead of the internal toString() representation.

Definition

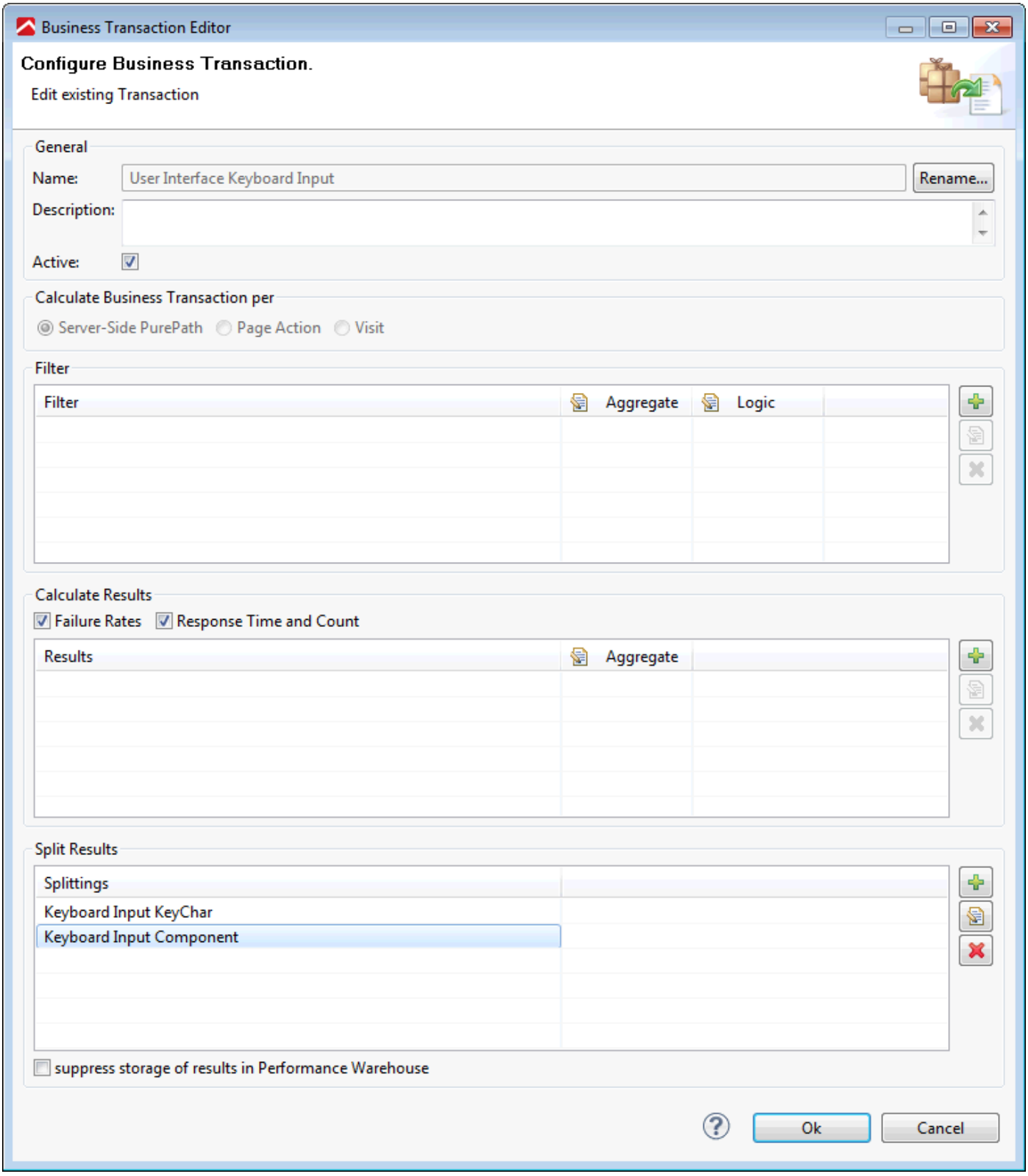
Splitting Measure for the pressed keyboard character:



Splitting Measure for the keyboard input component:

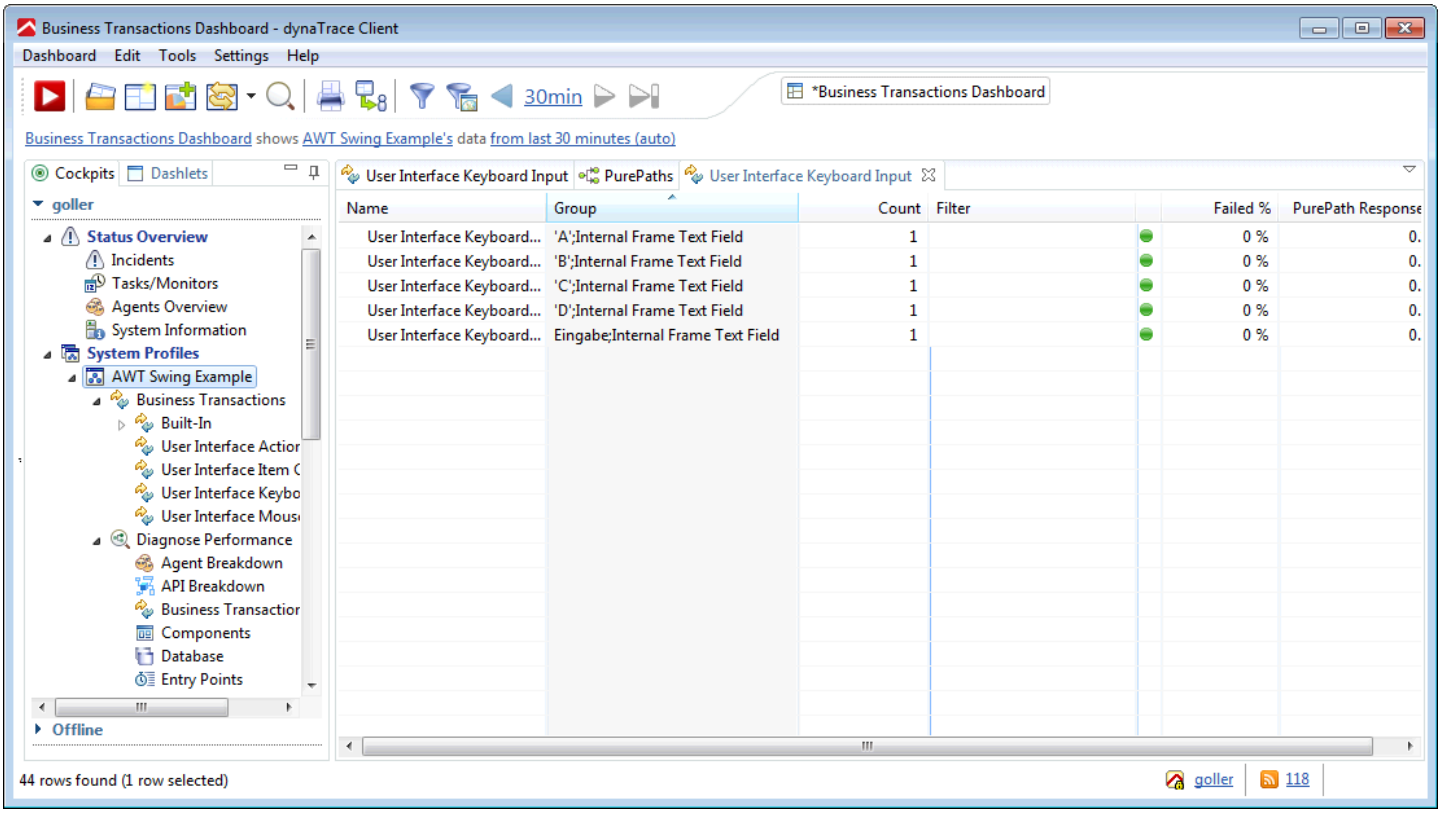


This is the definition of the Keyboard Input business transaction, which uses all splitting measures for grouping:



Example

After doing some other keyboard action in the “Frame Title” text box we defined business transaction retult appears. In addition to regular keystrokes such as “A”, “B”, “C” and “D”, other ley presses such as ESC, RETURN, and BACKSPACE key presses appear.

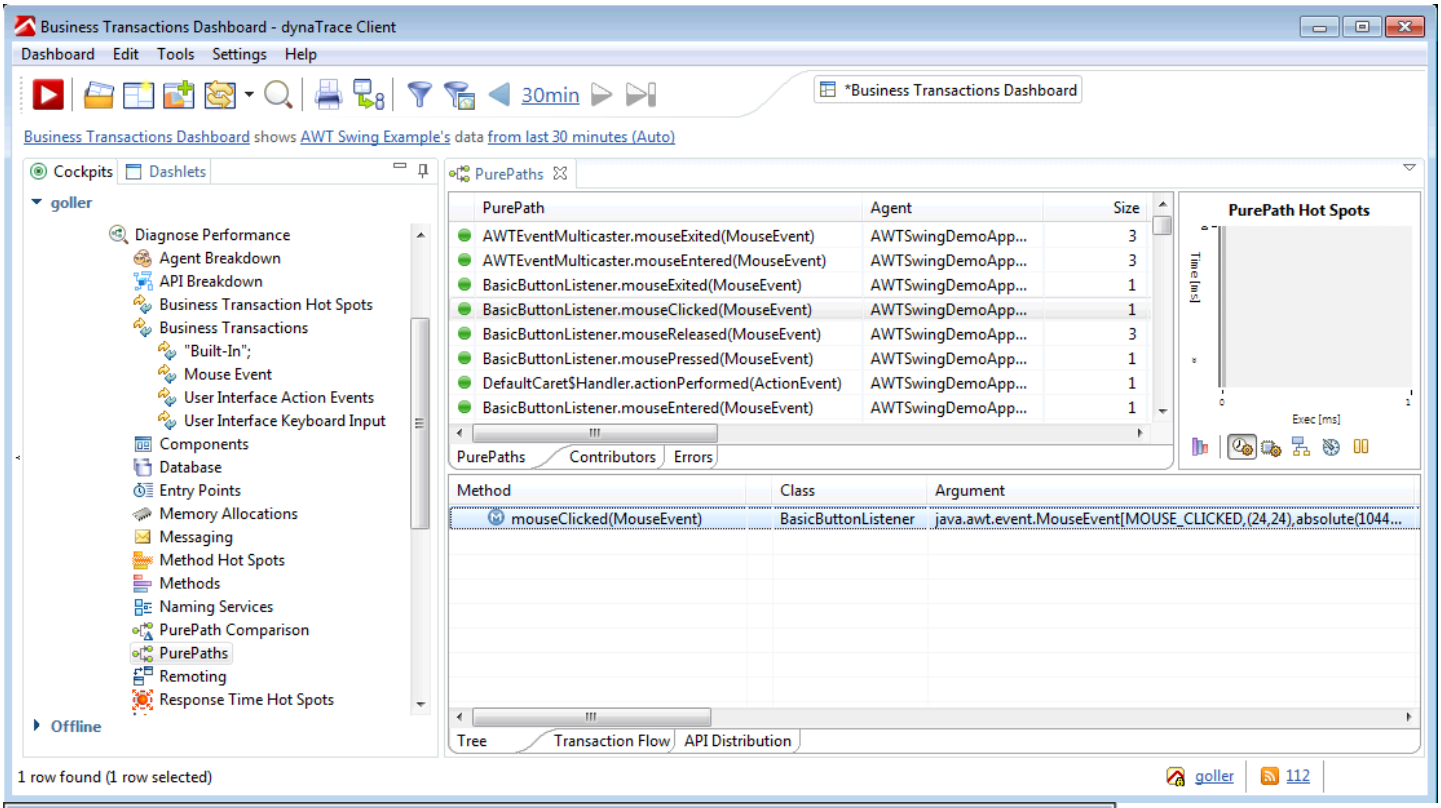


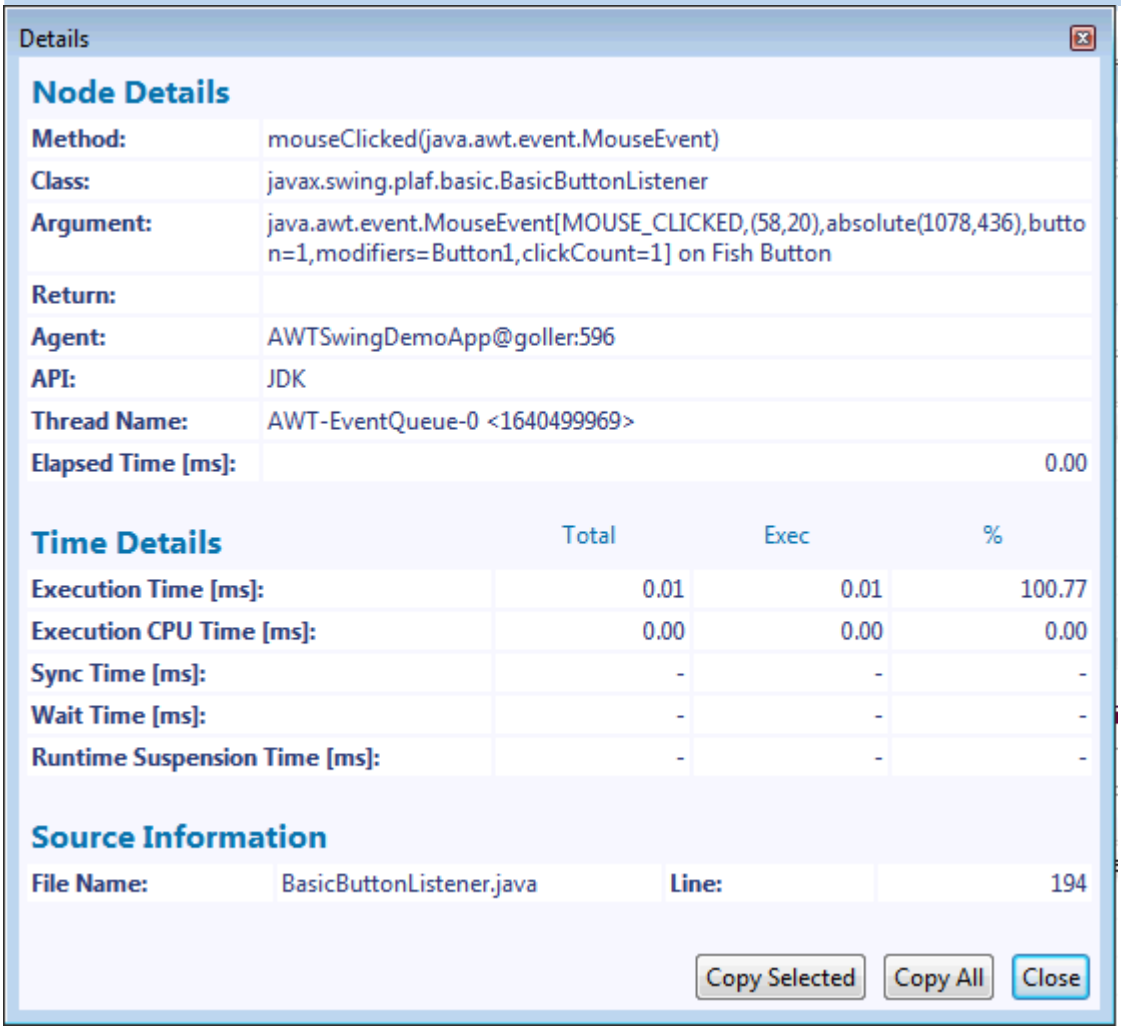
The keyboard events do not deliver a timestamp, so it is not possible to view them chronologically in the business transaction dashlet. However, there is still enough information to pinpoint slow keyboard input processing in the monitored application.

User interface mouse events

The swing demo application also registers some mouse events. Unfortunately there is not much business logic behind the clicks, so there is not much application code running behind these clicks. As already mentioned, the swing demo application mostly uses the action event for triggering functional code. While this is true to the swing demo application, it may be different on other applications. In the real world, it is mainly the developer who decides to either use action, mouse, keyboard or other types of events.

Here's how a PurePath with a mouse event might look:





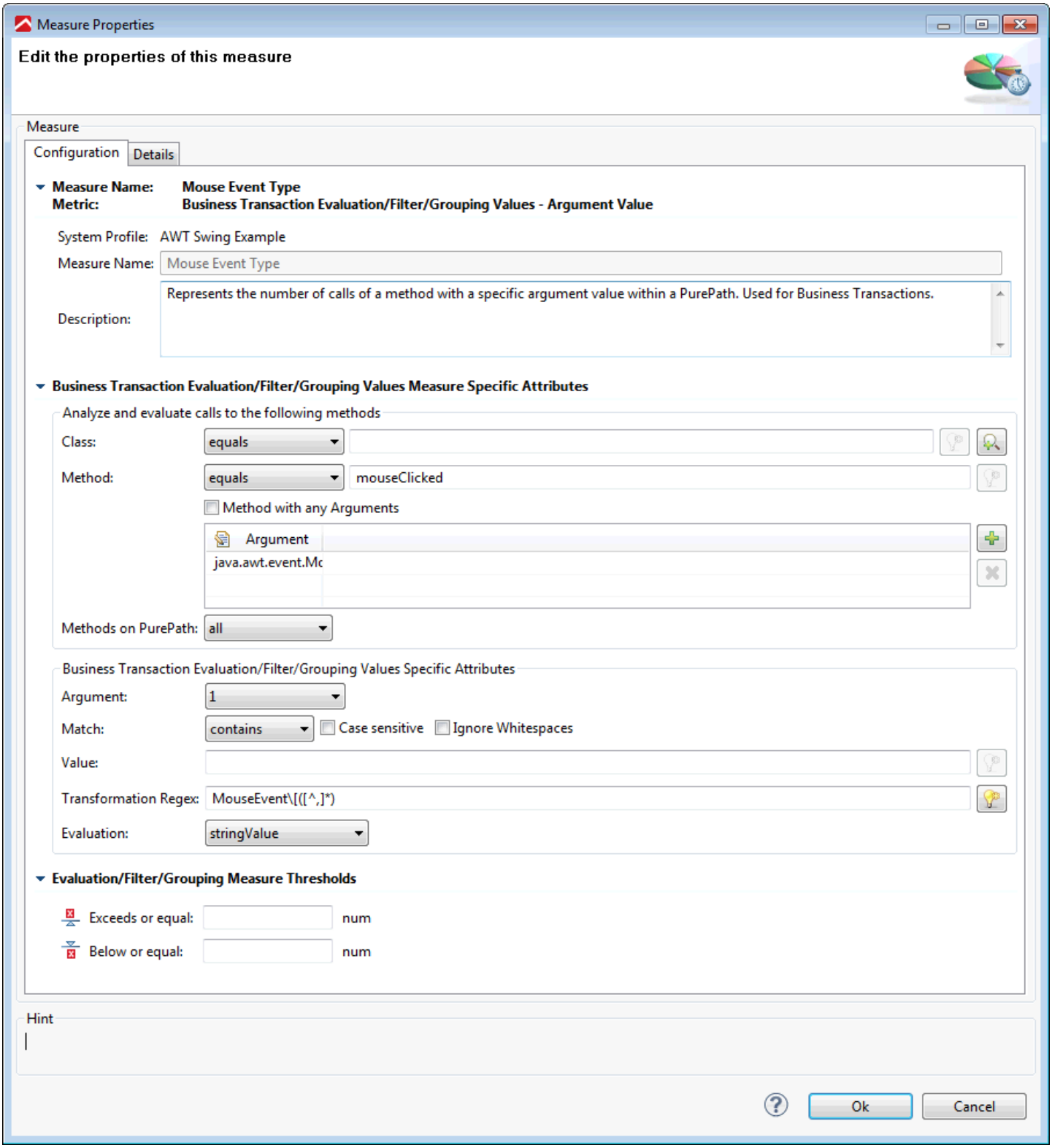
The argument looks like this:

java.awt.event.MouseEvent[**MOUSE_CLICKED**,(58,20),absolute(**1078,436**),button=1,modifiers=Button1,clickCount=1] on **Fish Button**

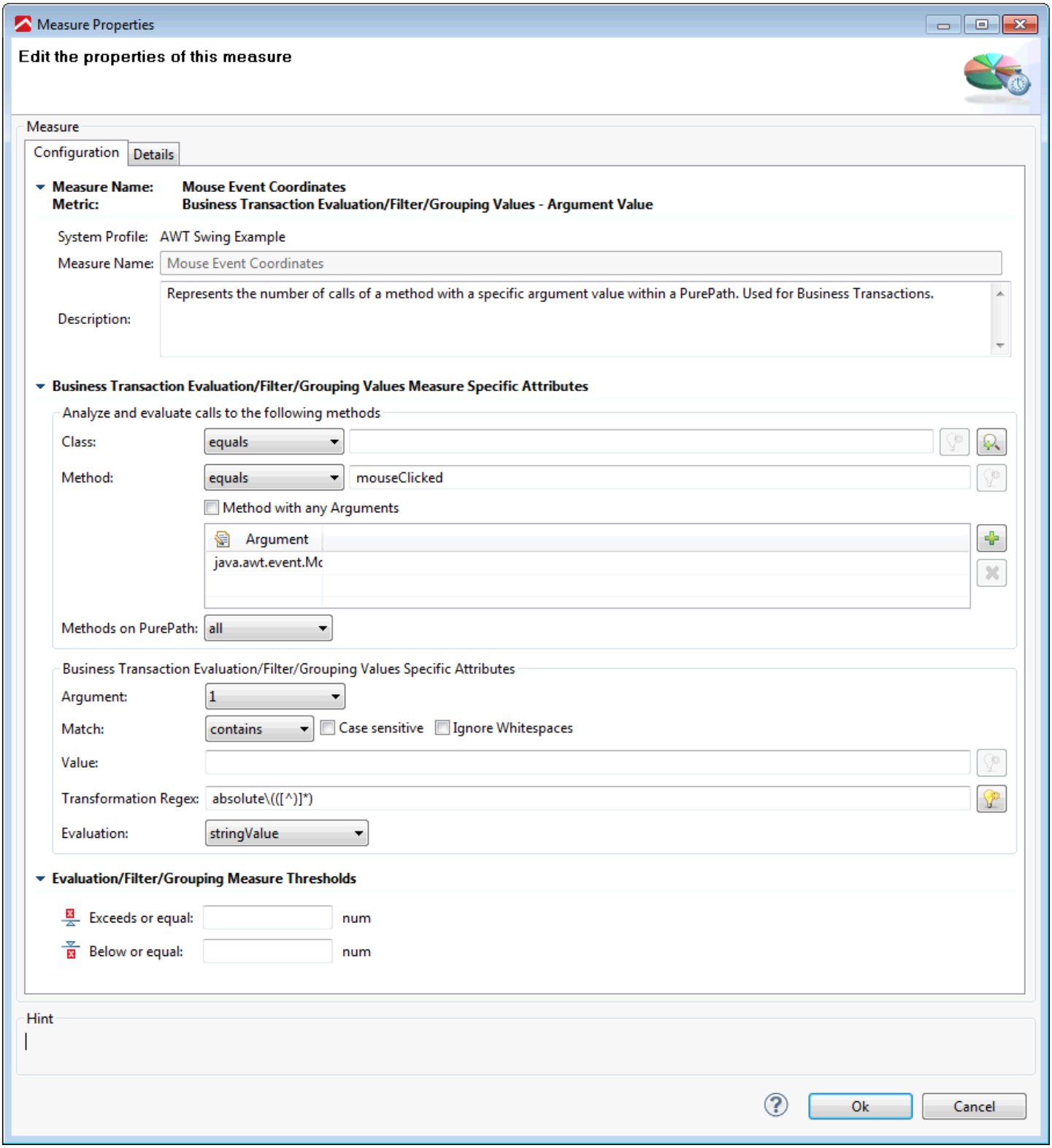
Three arguments could be valuable. First, the event type (**MOUSE_CLICKED**), which identifies the mouse event type (such as clicked or released). Second the clicked X/Y coordinates (**1078, 436**) and third the name of the clicked component (**FishButton**).

Definition

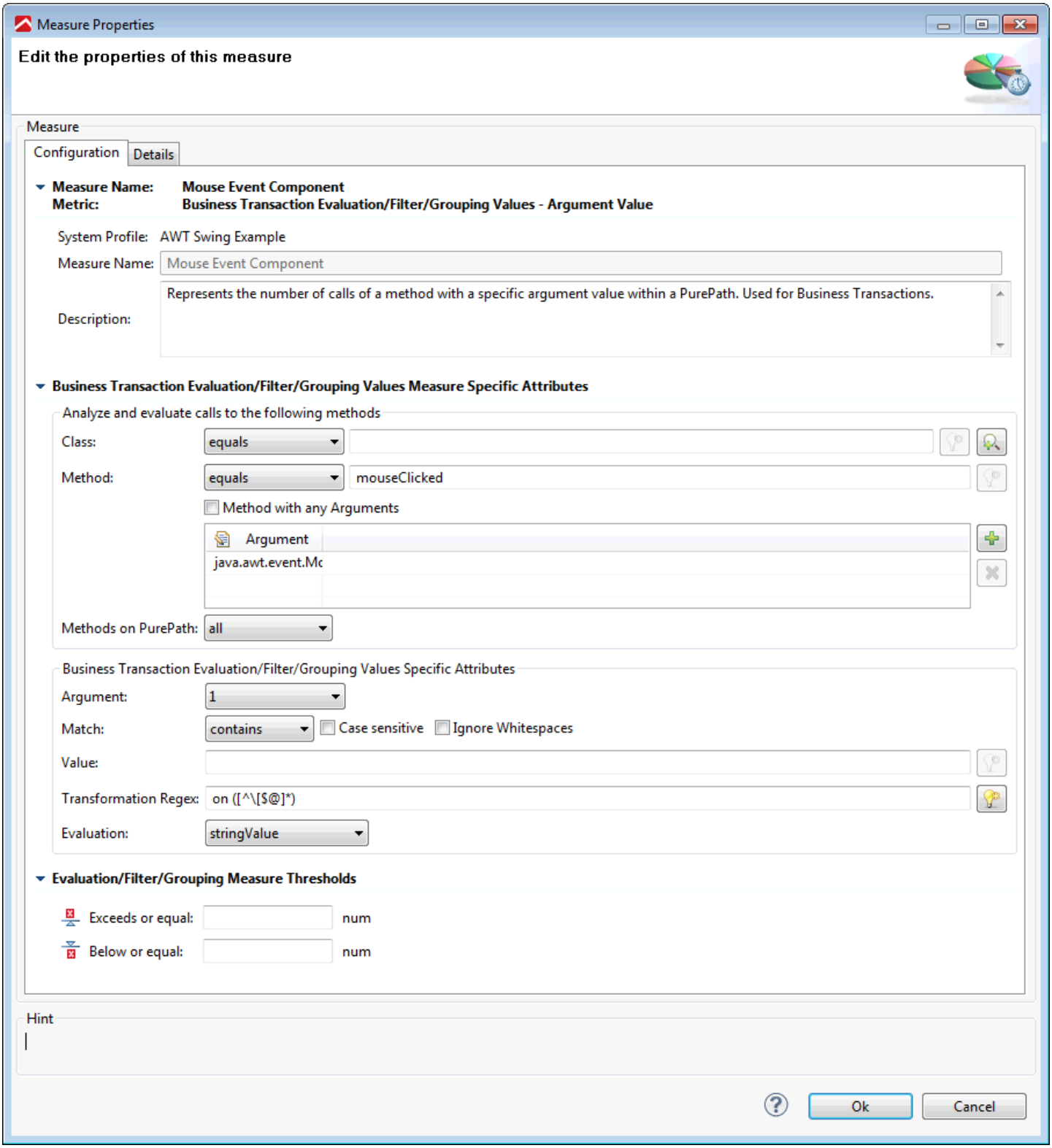
Splitting Measure for the Mouse Event Type:



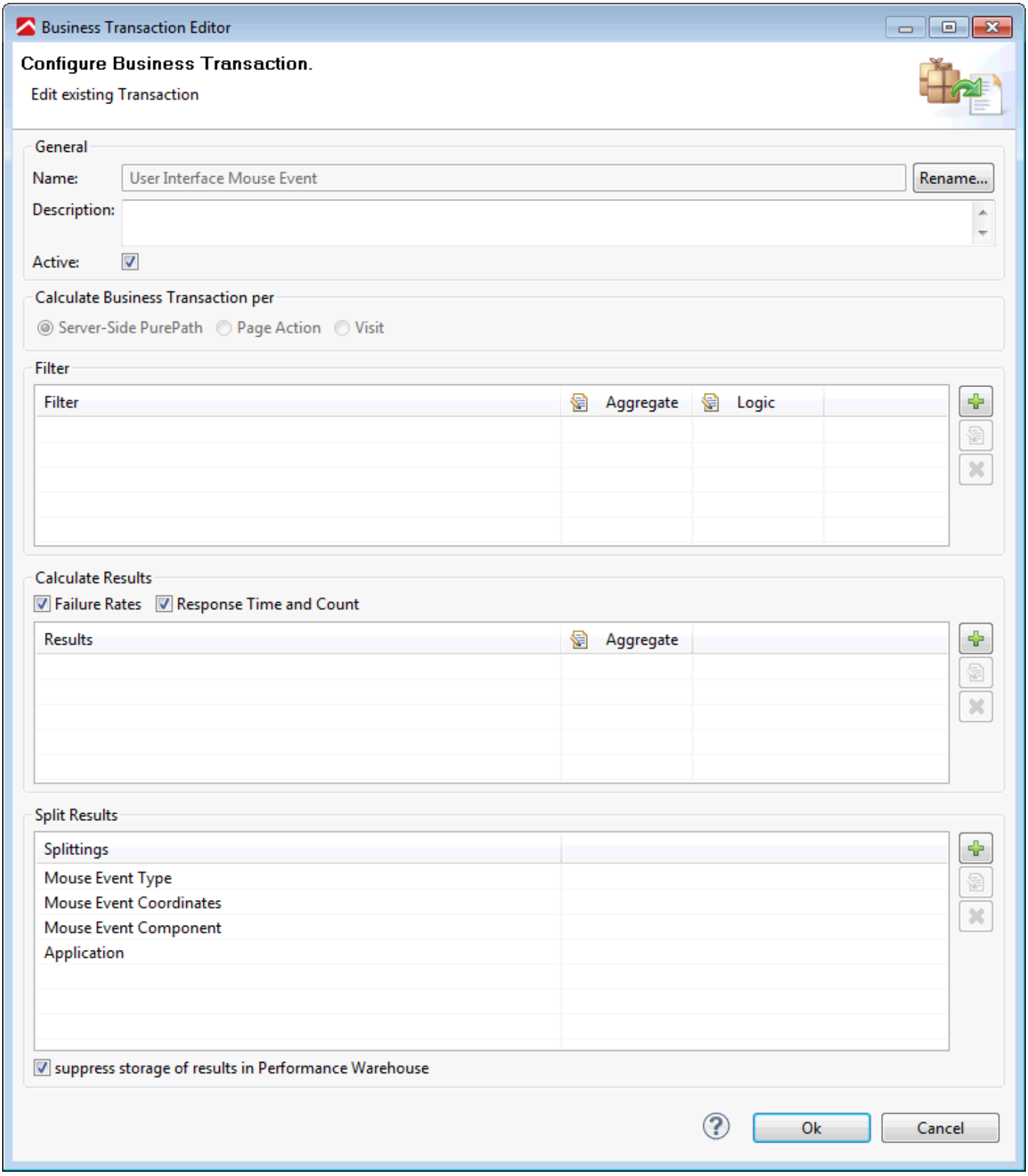
Splitting Measure for the absolute Mouse Event Coordinates:



Splitting Measure for the Mouse Event Component:

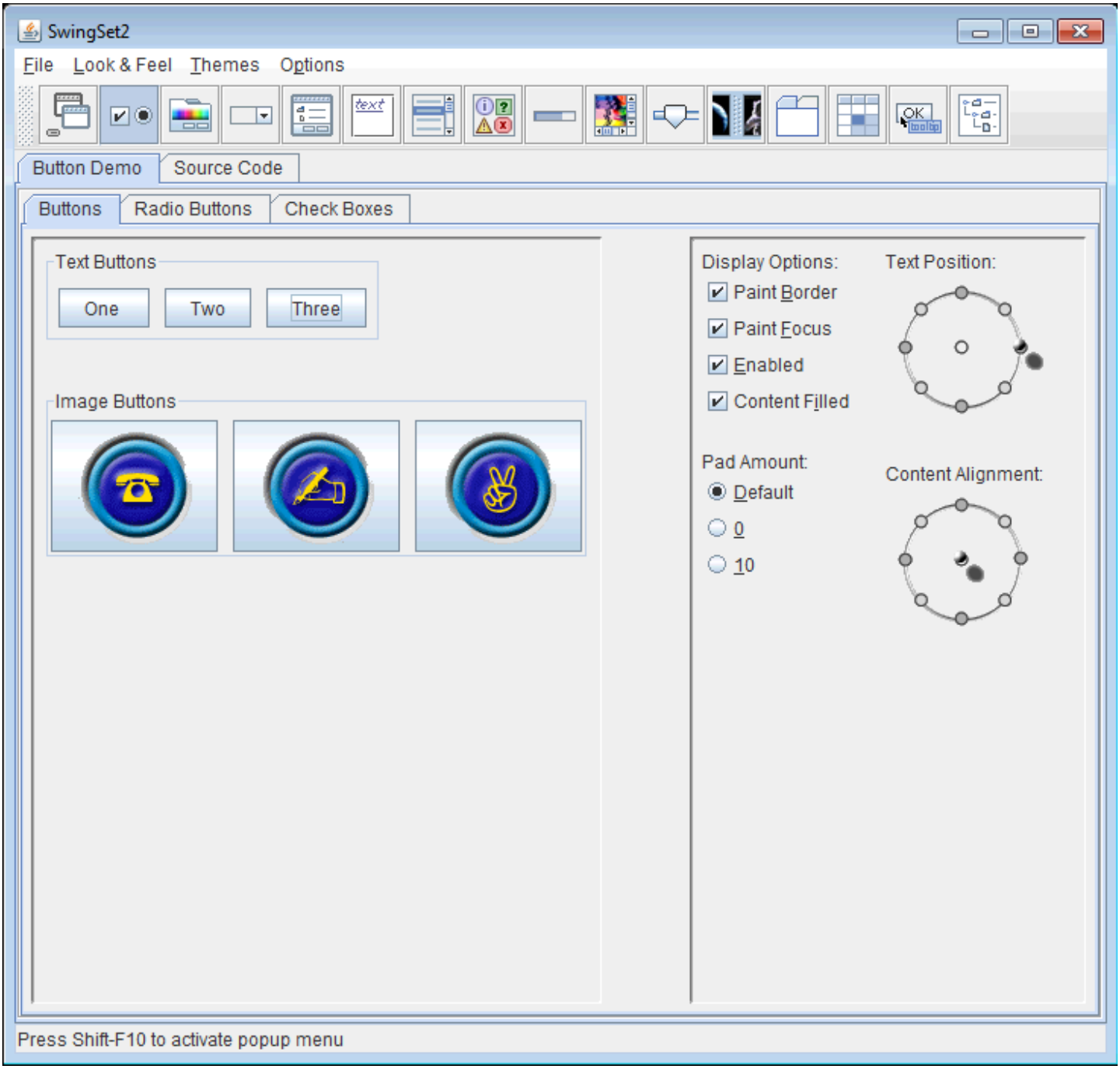


This is the definition of the Mouse Event business transaction, which uses all Splitting Measures from above:

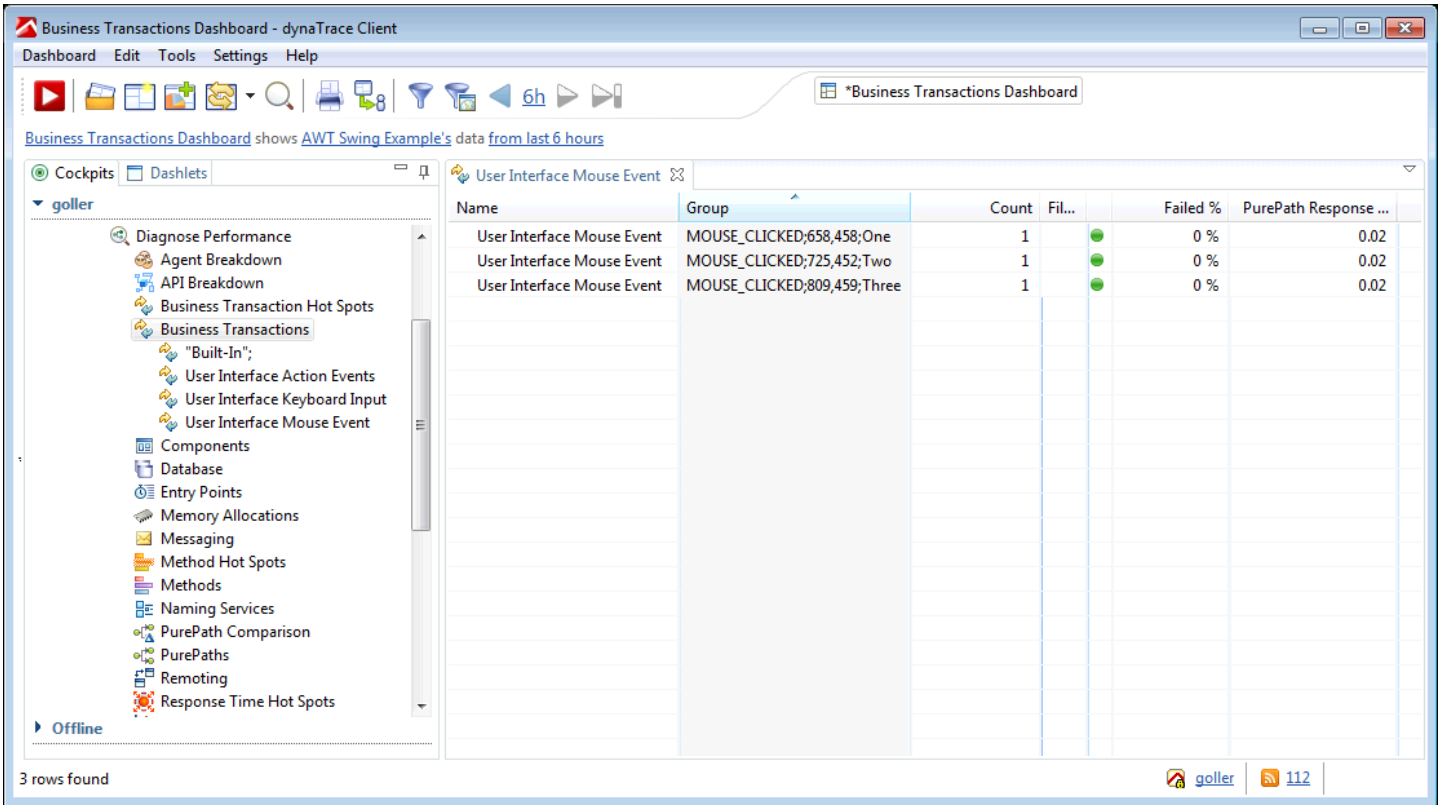


Example

Clicking the three (“One”, “Two”, “Three”) buttons in the Swing demo application



… looks like this in AppMon:

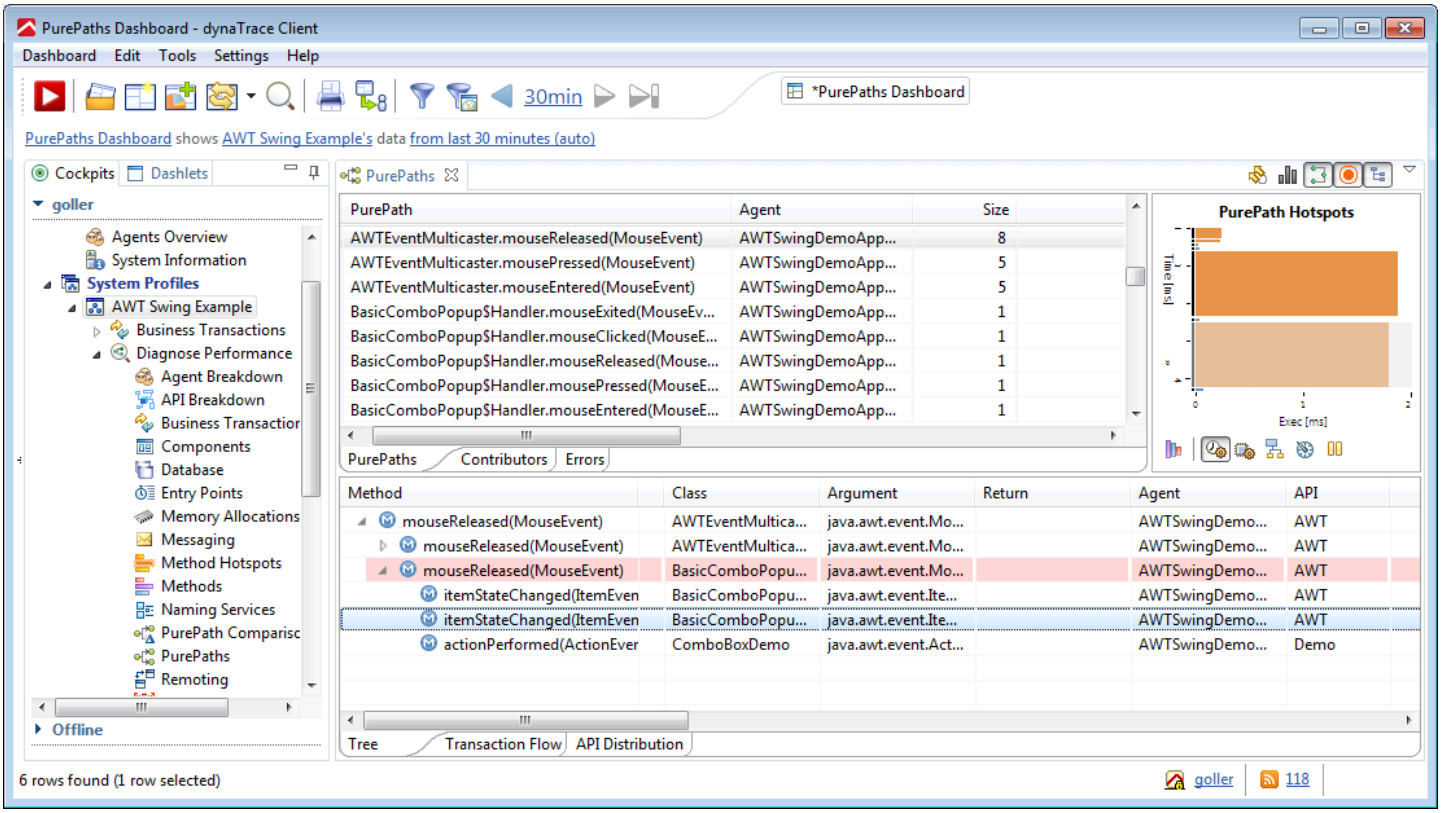


In this example, the Swing Demo Application's source code is modified to give the buttons some names (by using setName("One") etc.) for the buttons.

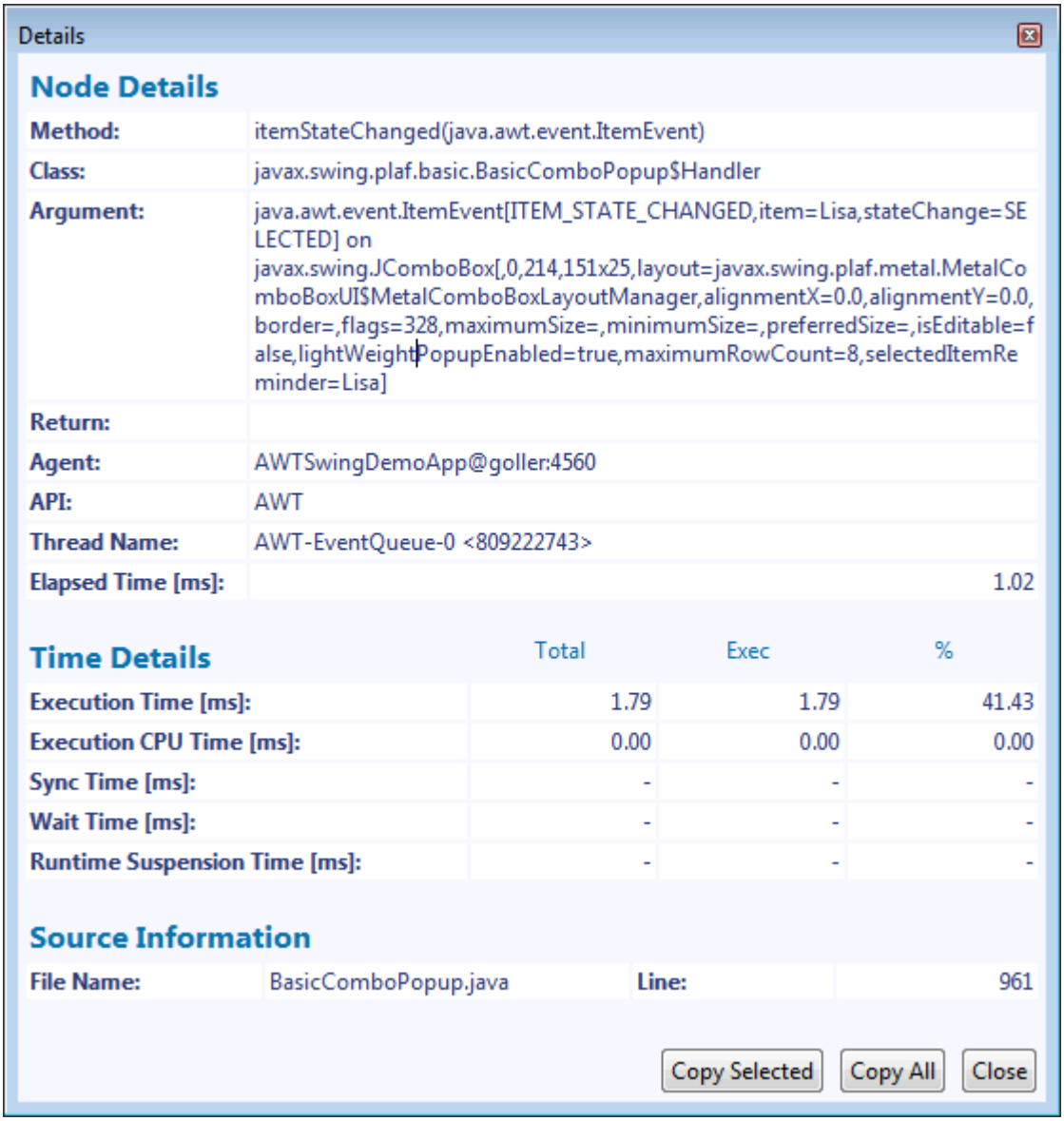
User interface item changed events

The swing demo application also registers some item state events. There is not much business logic behind these events, so there is not much application code running behind these clicks. The event is mostly used for lists and combo boxes to get notified about selection changes.

The following screenshot show an example PurePath for an Item Changed event. In this case, the Mouse event has triggered the Item event. Constellations like this are quite normal for AWT/Swing:



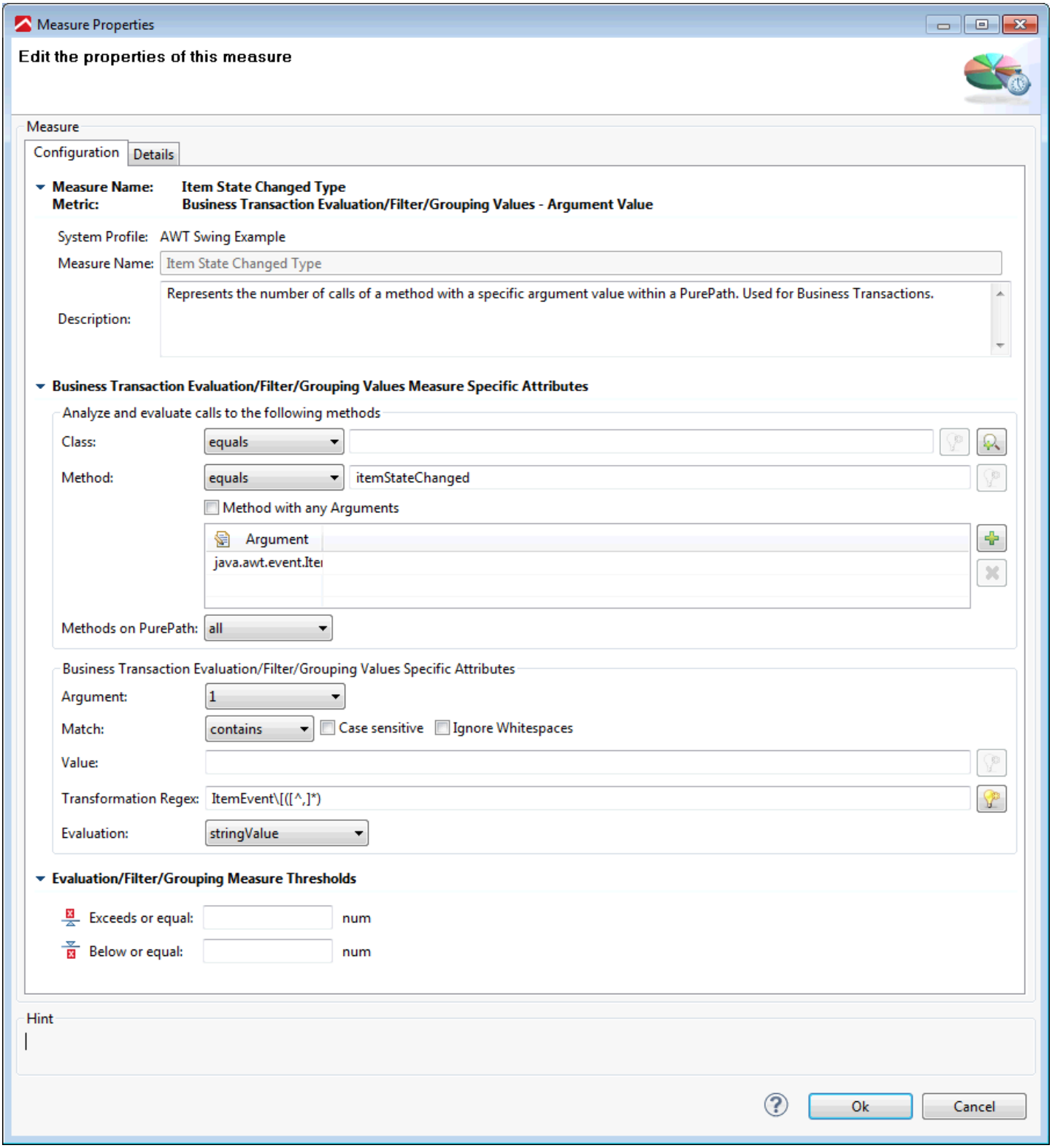
The argument appears in the node details.



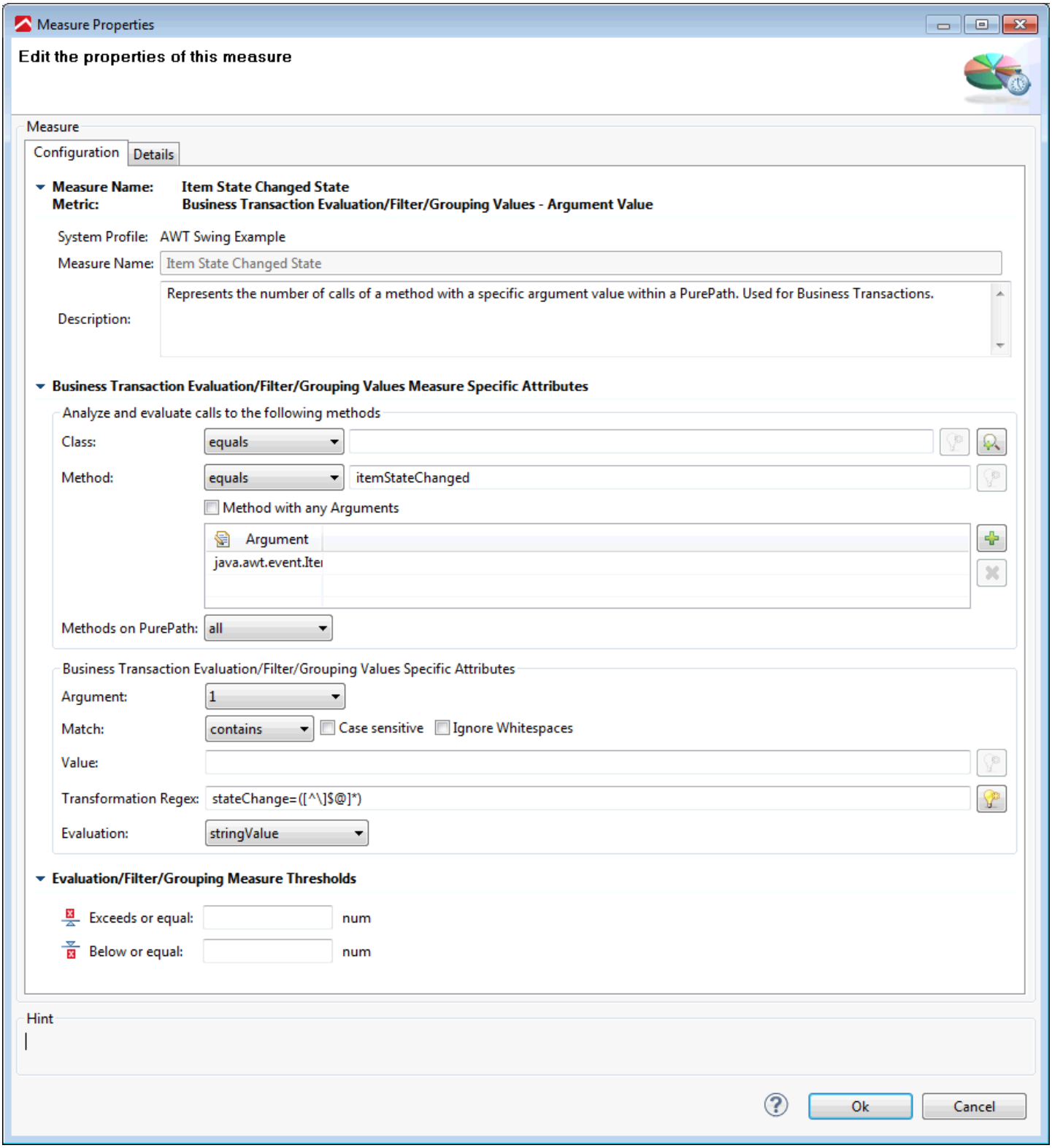
Four arguments could be valuable. First, the item event type (**ITEM_STATE_CHANGED**), which identifies event itself. Second the new state(**SELECTED**) of the item (selected/deselected). Third, the selected/deselected combo box or list item (**Larry**) and fourth, the name of component which raised the event (**javax.swing.JComboBox**). Please note that if the component has a name (by using setName("MyButton") for example), the name is shown here instead of the default toString() representation.

Definition

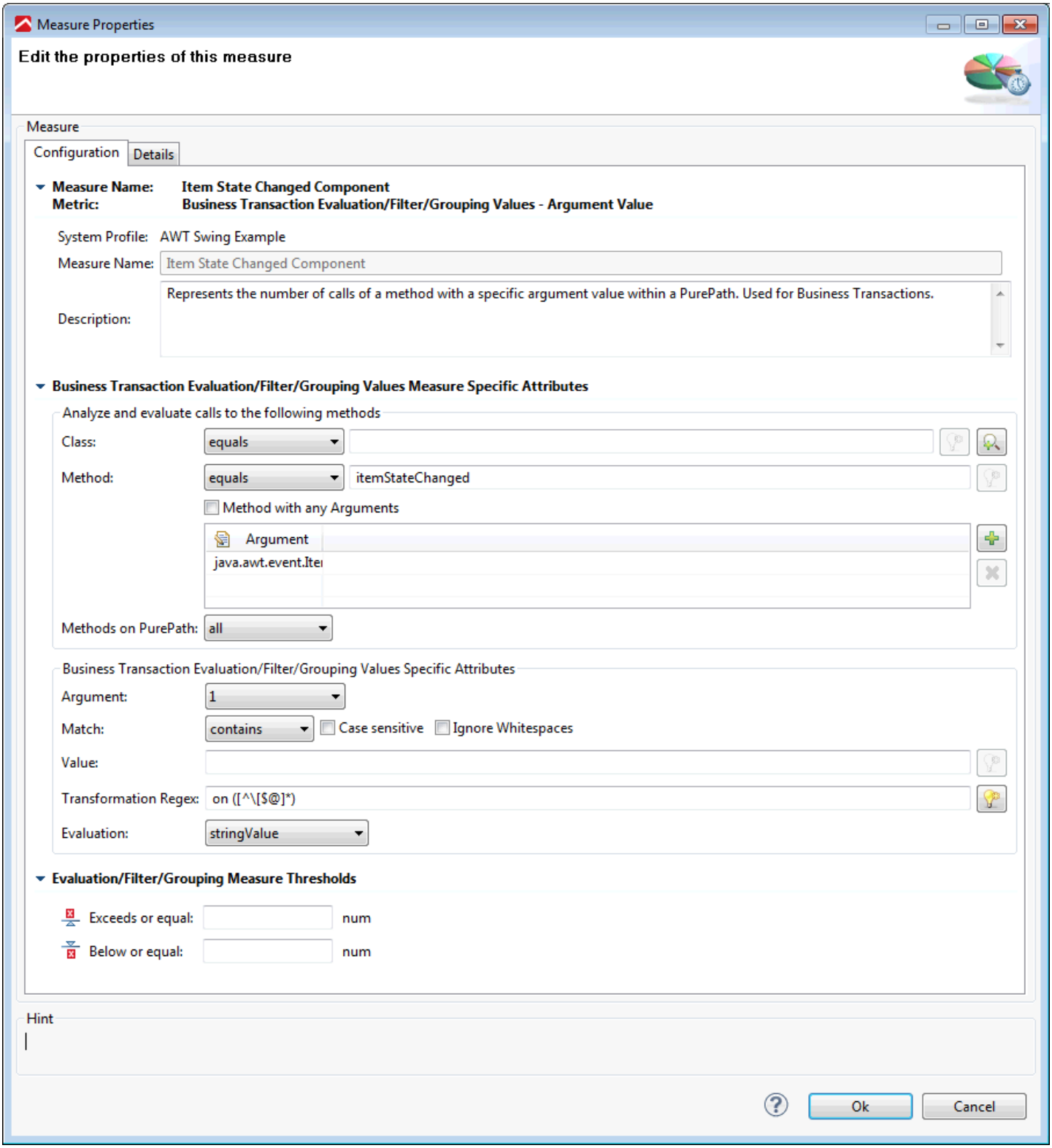
Splitting Measure for the item event Type:



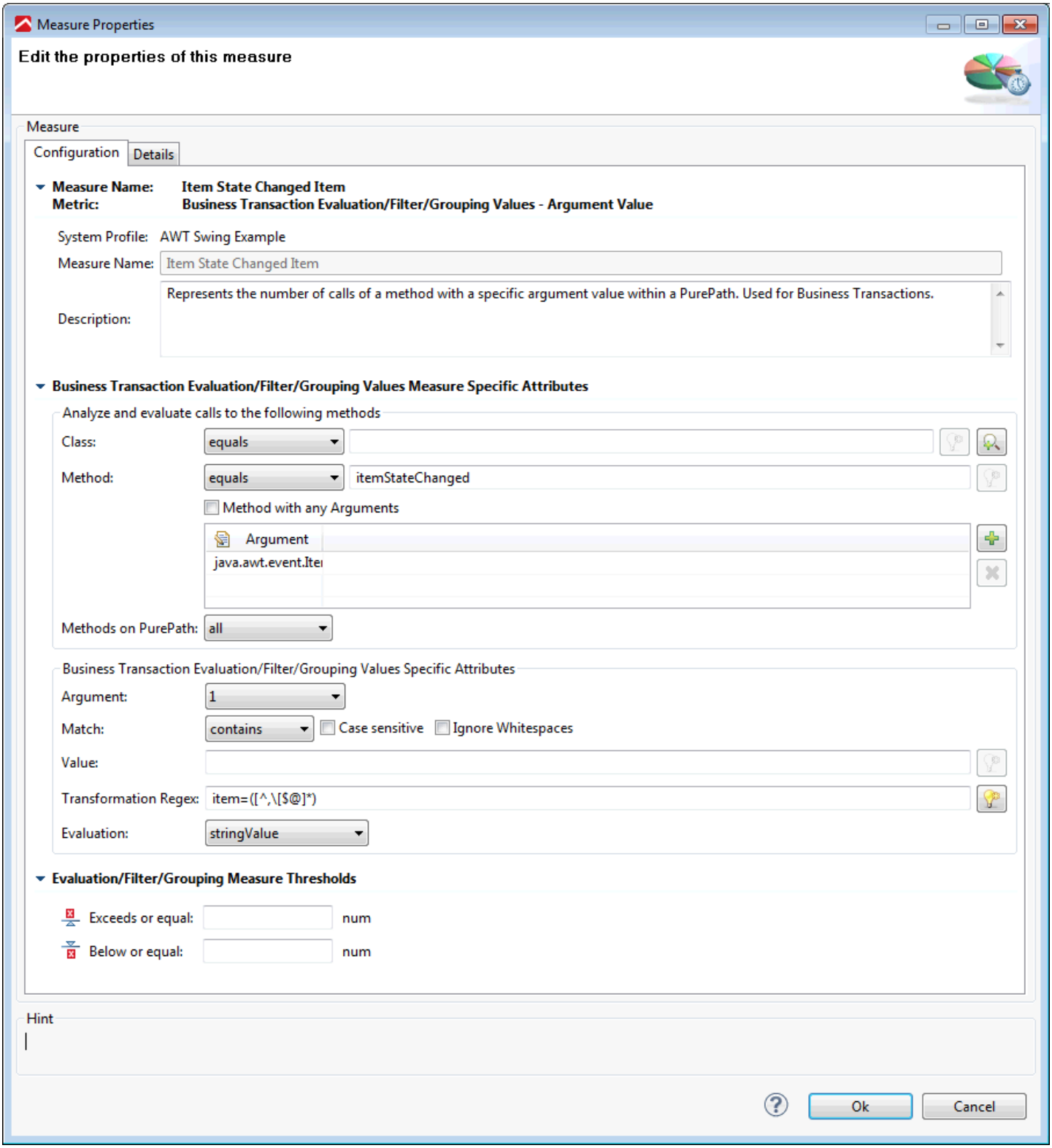
Splitting Measure for the changed State:



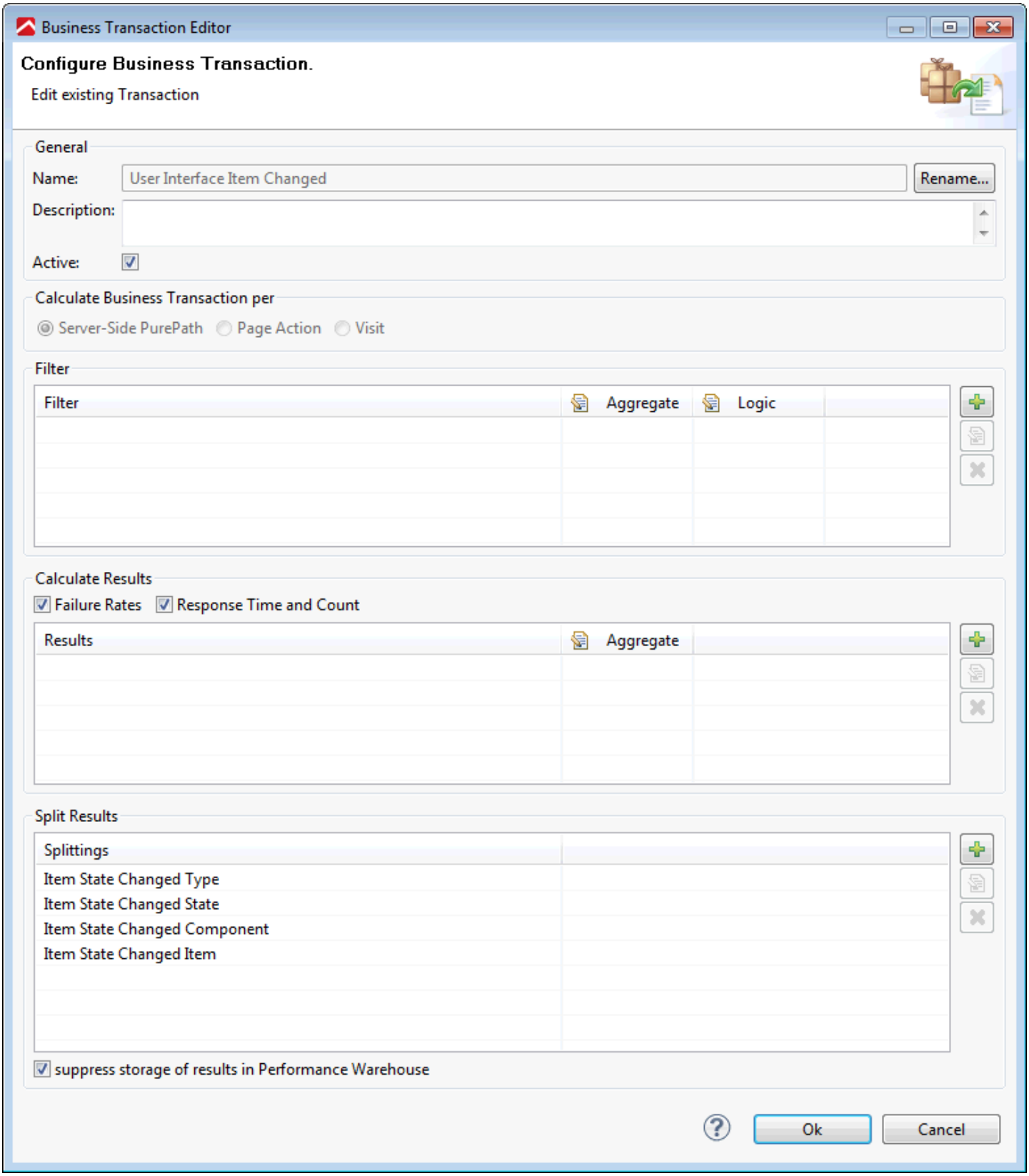
Splitting Measure for changed Component:



Splitting Measure for the changed Item:

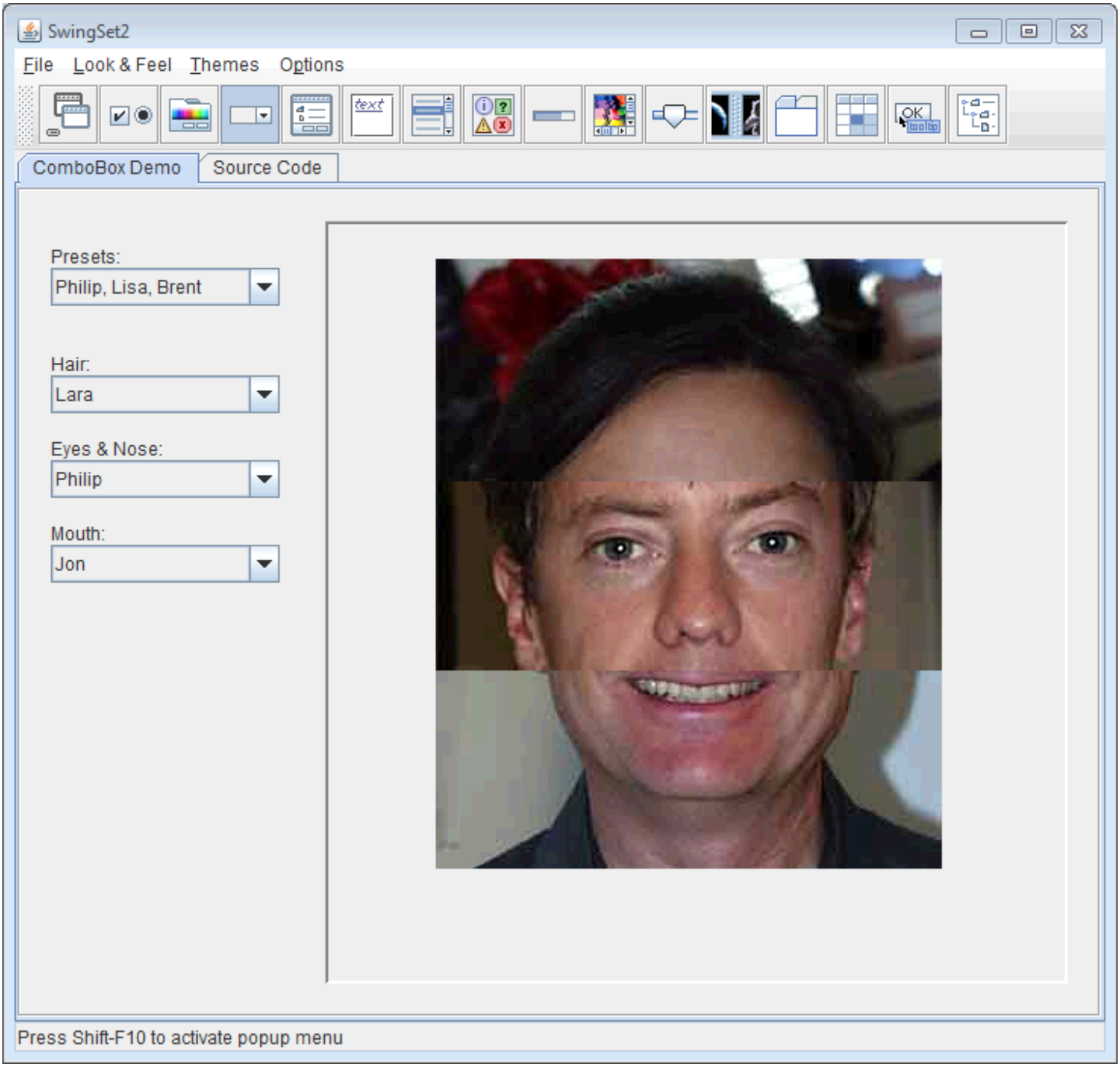


This is the definition of the item Changed Item business transaction, which uses all splitting Measures from above:

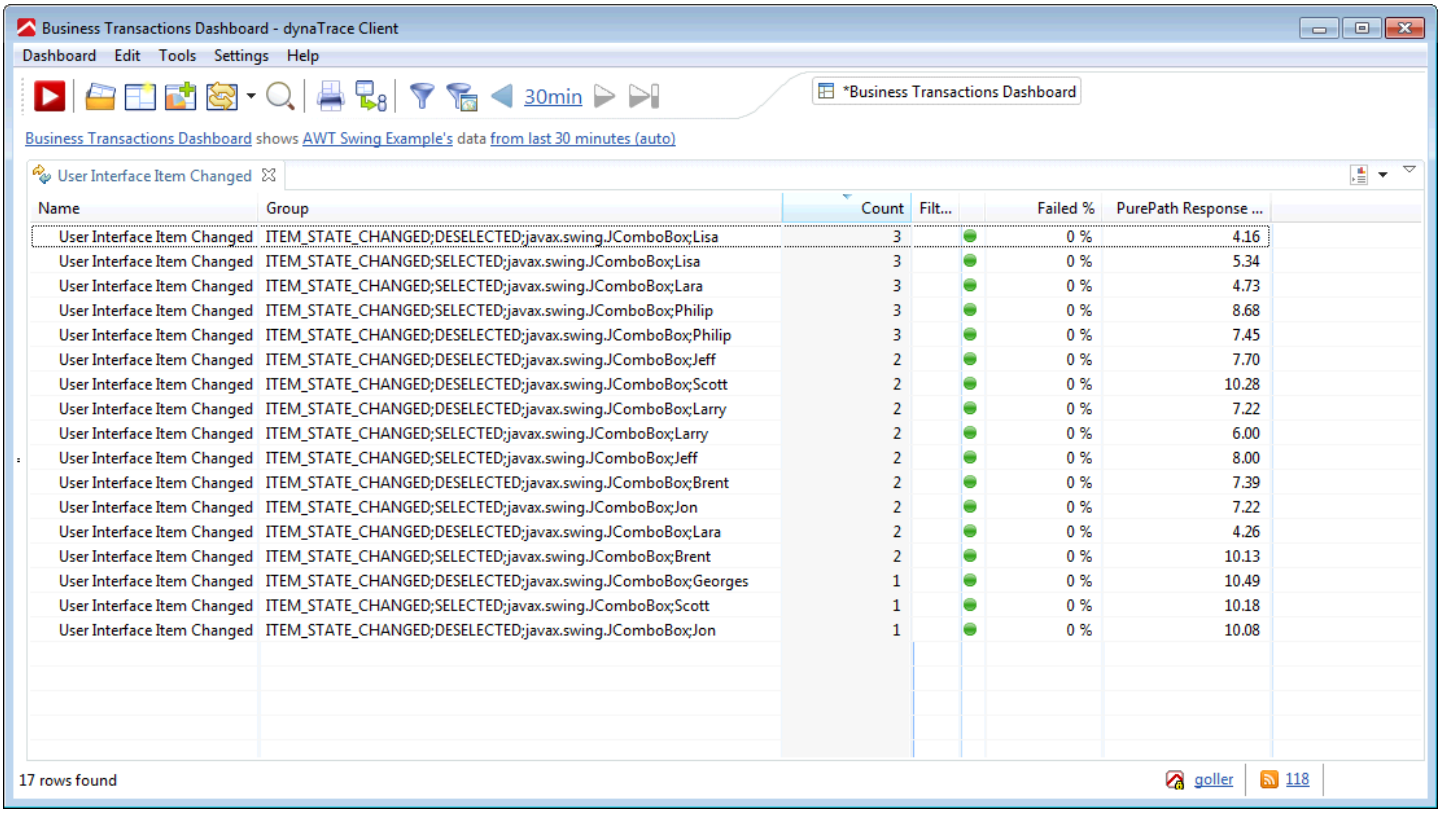


Example

Changing a combo box selection within the Swing demo application …

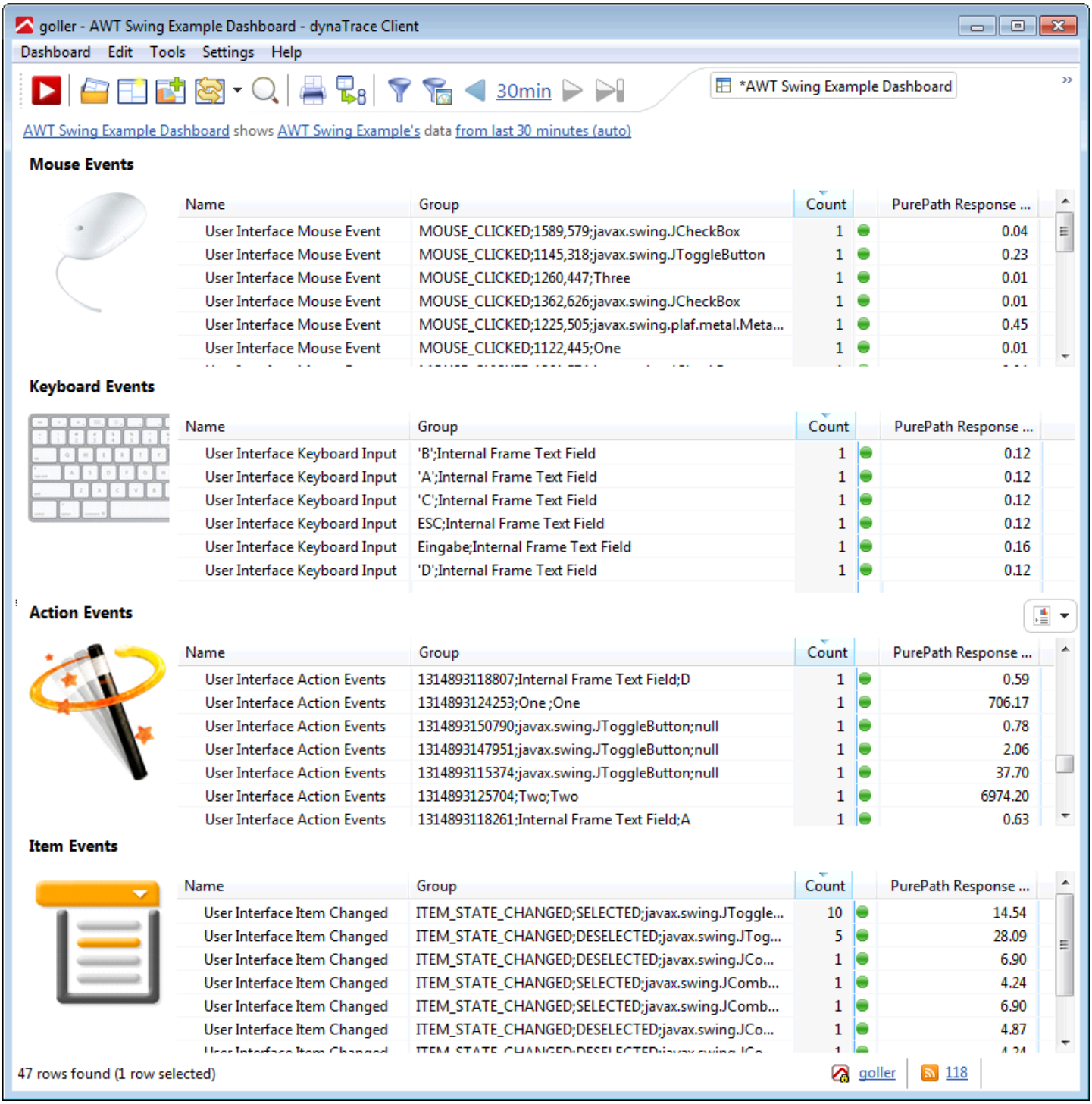


… looks like this in AppMon:



Dashboard

After defining business transactions, you can build a dashboard for easy AWT/Swing analysis. The following dashboard gives an example. The shown dashboard can be downloaded from here



Troubleshooting

Wrong listener type

Instrumenting AppMon Client applications more challenging than instrumenting regular request or response based applications. Because of this, you may miss the listener causing a performance problem. In the case where you may have designed your own listeners, you can find and instrument the corresponding listener. One approach could be to instrument the AWT Event Dispatcher. However, this can lead to a very high quantity of PurePaths and probably should only be used to find out which listener should finally be instrumented.

Too many PurePaths

The defaults used in this walkthrough can lead to quite a lot of PurePaths. Adjust the custom created AWT Sensor Group and remove listeners from which you know that they do not trigger any business logic to reduce the amount of captured PurePaths. You can also define an API based filter for the entire dashboard. This filters out everything that does not call certain business logic code, so the PurePath dashlet only shows relevant transactions.

Modal dialog box problematic

This is challenging to resolve with AppMon. The problem is as follows:

In AWT and Swing you can display modal dialog boxes. Modal means, that a new child window is opened on top of the parent window and the parent window is inactive until the child window is closed. For AWT and Swing this means that the calling method, which opens the modal dialog box, does not finish until the modal dialog box is closed by the user. This might sound like a bad design in AWT / Swing. Some companies even developed their own modal dialog box handling because of this. Unfortunately, AppMon has no chance to overcome this design weakness. AppMon also does not allow to define custom transaction exit points, which could be used to overcome the issue, so the transaction (PurePath) keeps going until the modal dialog box closes. Everything that happened in the modal child window is then in context of one single PurePath. This currently is a known problem.

Possible workarounds:

- Look at the PurePath entry method call which starts the long modal window transaction. Try to skip it by defining a method exclusion sensor rule. Speaking technically, this means if you have a PurePath which contains the setVisible(true) method of a modal window, the PurePath is affected by this problem.
- Look at the CPU time of the modal dialog box PurePath instead of the PP Duration. Duration tells you only how long the modal dialog box was open. CPU Duration gives you at least some hints about the processing hotspots while the modal dialog box was open.

Gatling/AppMon 6.2 integration

On this page

Use hooks to start/stop session recording and switch configuration

Register test run

Use header tags to convey context information with your requests

Analyze results in AppMon

Related pages

Continuous delivery and test automation

Continuous integration builds

Build server



Gatling is an open-source load testing framework. Using the Automation Library for AppMon 6.2, it is fairly easy to drive integration from any Gatling scenario written in Scala.

AppMon uses the Java automation library in the context of a Gatling/Scala script. According to the Scala Web Site: “Scala runs on the JVM, so Java and Scala stacks can be freely mixed for totally seamless integration.”

To do this, add the `com.dynatrace.diagnostics.automation.jar` automation library to your project's classpath. This lets you use the **RESTEndpoint** object provided by the library.

```
// AppMon automation library

import com.dynatrace.diagnostics.automation.rest.sdk.RESTEndpoint
```

```
// AppMon Server REST URL
val dt_server = "http://localhost:8020"

// AppMon User
val dt_user = "admin"

// AppMon Password
val dt_pwd = "admin"

// Create an endpoint to send commands to AppMon Server
// using server address and credentials.

val endPoint = new RESTEndpoint(dt_user, dt_pwd, dt_server)
```

Use hooks to start/stop session recording and switch configuration

Before actually starting the test, you can use the **before** hook to:

- Switch to a given configuration (optional) so that AppMon captures the code execution with a specific level of details.
- Start session recording to create a session on disk to store the recorded traffic.

When the test is done, use the **after** hook to:

- Stop session recording to close the session.
- Switch back to default configuration to revert DTAM to its defaults settings regarding transaction capturing details.

```
val dt_profile = "easyTravel" // System Profile

  // if you are not sure about the System Profile name, you can use the following call
// to get a list of created System Profiles:
//
//    endPoint.getProfiles()
//
val dt_presentableName = "Session1"          // User-readable presentable name for the ses
val dt_description = "My Load Test Session" // Description for the session to be stored
val dt_recordingOption = "all"               // all : All PurePaths including time series,
                                             // violations : Only PurePaths marked as viol
                                             // timeseries : Time series only.
val dt_isSessionLocked = false               // true to lock the session that is recorded,
val dt_isTimestamp = true                    // true to append <u>timestamp</u> informatic
```

```
before {
  // Optional: switch to a specific configuration
  // to allow for specific details level in captured data
  // Note: make sure you created such configuration in AppMon before using it.
  endPoint.activateConfiguration(dt_profile, "myCustomConfigurationName")

// If you are not sure about the Configuration name,
// you can use the following call to get a list of
// created Configurations for this System Profile:
  //
// endPoint.getConfigurations(dt_profile)

// start AppMon session recording for this test
   endPoint.startRecording(  dt_profile
                           , dt_presentableName
                           , dt_description
                           , dt_recordingOption
                           , dt_isSessionLocked
                           , dt_isTimestamp
                           )
  }
```

```
after {
    // stop AppMon session recording for this test
    endPoint.stopRecording(dt_profile)

    // switch back to default AppMon configuration
    endPoint.activateConfiguration(dt_profile, "Default")
}
```

Register test run

When using Gatling to run Web API tests, this interface creates a new test run with metadata. When this method is called, the response contains a unique ID for your test metadata on the AppMon Server. This ID is often referred to as `testRunId`. The AppMon Server uses the ID to distinguish test executions. Specify it in the request headers (see below) when executing your tests.

```
val dt_versionMajor = "1"          // Major version
val dt_versionMinor = "0"          // Minor version
val dt_versionRevision  = "0"      // Revision
val dt_versionBuild  = "1"         // Build number
val dt_versionMilestone  = "1"     // Milestone
val dt_marker  = "1.0.0.1.1"       // Special version marker,
                                   // used in the heat-field of the Test Results dashlet
val dt_category = "webapi"         // Category of the test. Valid value is one of
                                   // "unit", "uidriven", "performance", "load" or "weba
val dt_platform = "Linux"          // The platform the test runs on.
                                   // If not defined here, the platform is detected by t
val dt_loadTestName = "MyLoadTest" // The name of the load test.
                                   // This field is only used for the "load" category.

/* start a new test and register it with AppMon */
val dt_testrunid=endPoint.startTest(
                        dt_profile
                      , dt_versionMajor
                      , dt_versionMinor
                      , dt_versionRevision
                      , dt_versionBuild
                      , dt_versionMilestone
                      , dt_marker
                      , dt_category
                      , dt_platform
                      , dt_loadTestName
                      , null
                      )
```

Use header tags to convey context information with your requests

By default, AppMon labels recorded web requests from their URIs. In order to give them meaningful names (such as step names in a test scenario), a specific header tag can be added to the web requests.

This tag can also include a reference to the `testrunid` to indicate the test run a particular web request is part of.

Assemble the tag

The following helper function builds a AppMon header tag with the expected format, from the provided context data.

See Integration with Web Load Testing and Monitoring Tools and Web API tests for more information.

```
/*
NA      The timer Name of the request.
          This can be the timer or transaction name used in the load test script to identi
          NA=<timername>
          Required

TR      TestRunId, from a registered test run
        TR=<TestRunId>
        Required

PC      The Page Context contains information about what document in the currently process
          The following syntax is recommended, though not required:
          - If it is a named frame, then the value starts with the frame name.
          - The document number, unique for the page, is appended after a period.
          If embedded documents are cached, this number need not be progressive.
          PC=<FrameName>.<DocId>
          optional

VU      The unique number of the Virtual User that sends the request.
        VU=<Id>
        optional

ID      The unique request ID (serial number).
        This string should be unique for one web request or a set of web requests that toc
        ID=<Id>
        optional

SI      The Source ID can be used to identify the product that triggered the request:
        For example WLT (Web Load Testing), SYM  (dynaTrace Synthetic Monitoring), BB (Bac
```

```
        optional

GR      Geographic Region, useful only for the Synthetic Monitoring solution.
        This contains arbitrary text.
        optional

AN      Agent Name: the logical name of the Agent from which the request originated.
        This information is used by Synthetic Monitoring.
        optional

SN      Script Name.
        This groups a set of requests that make up a <u>multi</u>-step transaction, for ex
        optional

TE      Test Name is the name of the entire load test.
        It uniquely identifies a test run.
        optional
*/

def dt_tag(
          NA:String
          , TR:String
          , PC:String=""
          , VU:String=""
          , ID:String=""
          , SI:String=""
          , GR:String=""
          , AN:String=""
          , SN:String=""
          , TE:String=""
          )
  : Map[String, String] =
  {
    var value:String = "NA="+NA+";TR="+TR+";RC=200;VU="+VU+";PC="+PC+";ID="+PC+";SI="+SI+

    return Map("x-dynaTrace" -> value)
  }
```

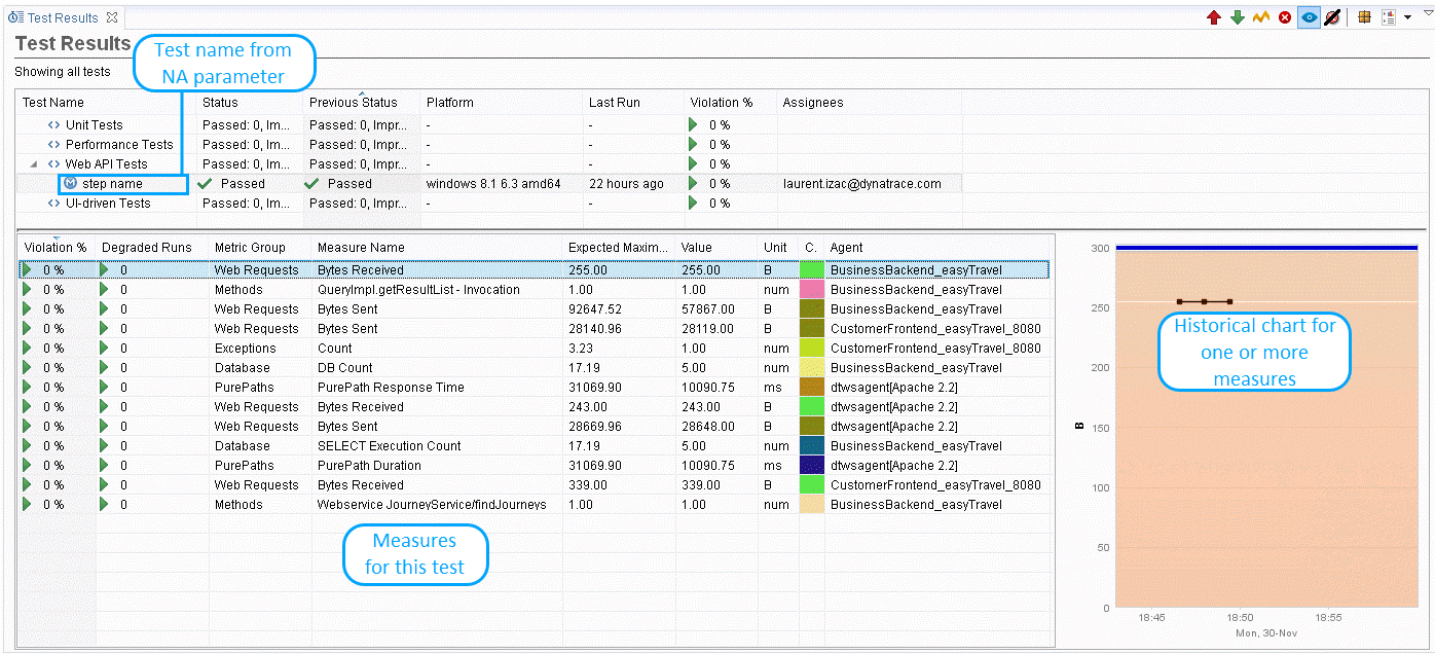
Use the tag

When issuing HTTP requests, add the AppMon tag in the header by calling the aforementioned helper function. Then provide the name, testRunId and other optional contextual data:

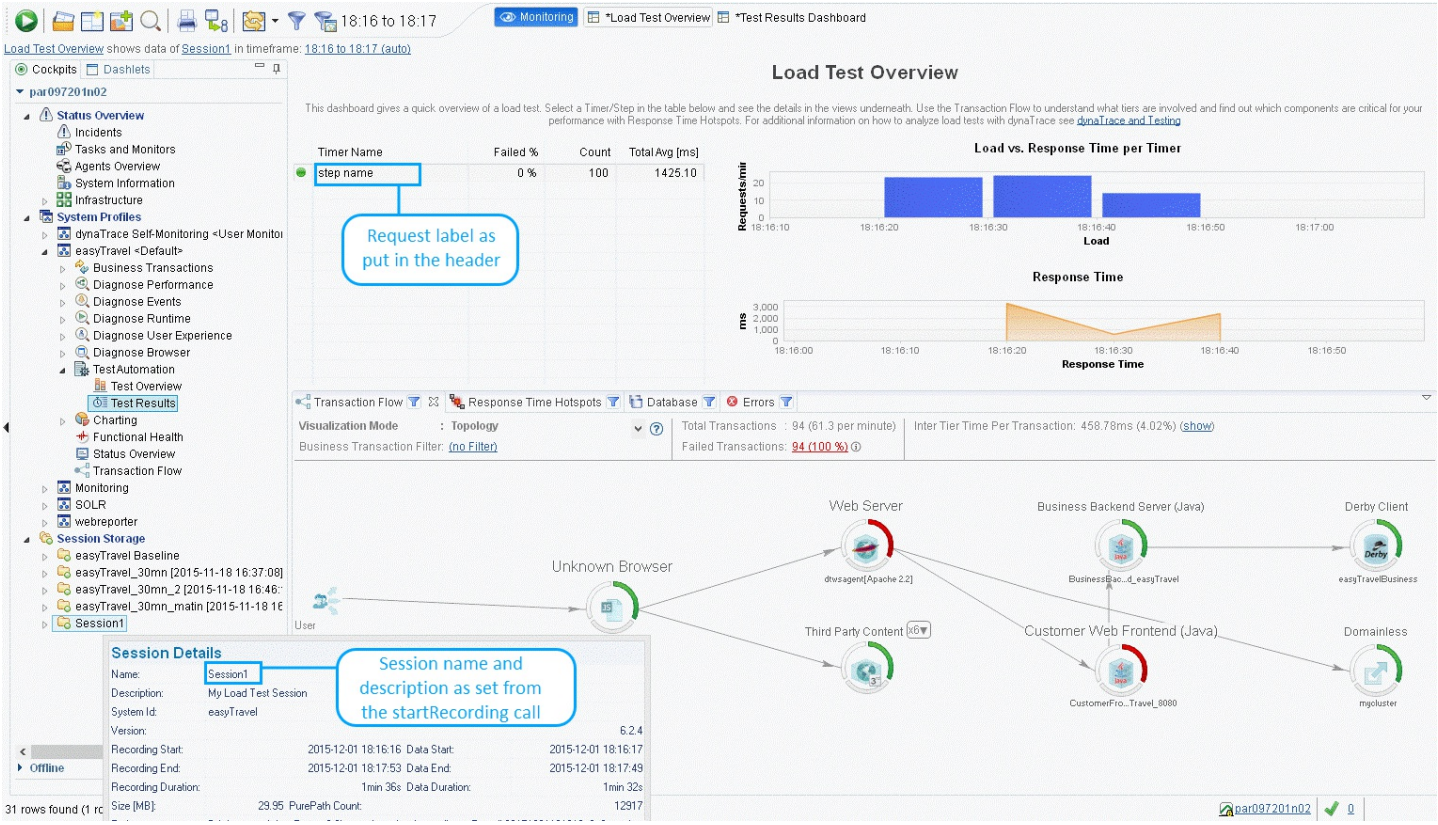
```
  val chain_0 = exec(http("request_0")
                .get("/")
                .headers(dt_tag("step name",dt_testrunid, ""))
                )
```

Analyze results in AppMon

Web API test



Load test


18:16 to 18:17
Monitoring
*Load Test Overview
*Test Results Dashboard
Load Test Overview shows data of Session1 in timeframe: 18:16 to 18:17 (auto)
Cockpits
Dashlets
par09720 1n02
Status Overview
Incidents
Tasks and Monitors
Agents Overview
System Information
Infrastructure
System Profiles
dynaTrace Self-Monitoring <User Monitor
easyTravel <Default>
Business Transactions
Diagnose Performance
Diagnose Events
Diagnose Runtime
Diagnose User Experience
Diagnose Browser
Test Automation
Test Overview
Test Results
Charting
Functional Health
Status Overview
Transaction Flow
Monitoring
SOLR
webreporter
Session Storage
easyTravel Baseline
easyTravel_30mn [2015-11-18 16:37:08]
easyTravel_30mn_2 [2015-11-18 16:46:
easyTravel_30mn_matin [2015-11-18 16
Session1
Offline
31 rows found (1 rc
Load Test Overview
This dashboard gives a quick overview of a load test. Select a Timer/Step in the table below and see the details in the views underneath. Use the Transaction Flow to understand what tiers are involved and find out which components are critical for your performance with Response Time Hotspots. For additional information on how to analyze load tests with dynaTrace see dynaTrace and Testing
Timer Name
Failed %
Count
Total Avg [ms]
step name
0 %
100
1425.10
Request label as put in the header
Load vs. Response Time per Timer
Requests/mi
Load
Response Time
ms
Response Time
Transaction Flow
Response Time Hotspots
Database
Errors
Visualization Mode : Topology
Business Transaction Filter: (no Filter)
Total Transactions : 94 (61.3 per minute)
Failed Transactions: 94 (100 %)
Inter Tier Time Per Transaction: 458.78ms (4.02%) (show)
Web Server
Business Backend Server (Java)
Derby Client
Unknown Browser
Third Party Content
Customer Web Frontend (Java)
Domainless
User
Session Details
Name: Session1
Description: My Load Test Session
System Id: easyTravel
Version: 6.2.4
Recording Start: 2015-12-01 18:16:16
Data Start: 2015-12-01 18:16:17
Recording End: 2015-12-01 18:17:53
Data End: 2015-12-01 18:17:49
Recording Duration: 1min 36s
Data Duration: 1min 32s
Size [MB]: 29.95
PurePath Count: 12917
Session name and description as set from the startRecording call
par09720 1n02
0

Gatling/AppMon 7.0 integration

On this page

WEB API test run

LOAD test run

Sample DSL scripts

Related pages

Continuous delivery and test automation

Continuous integration builds

Build server



Gatling is an open-source load testing framework. Using the Automation Library for AppMon 7.0, it is fairly easy to drive integration from any Gatling scenario written in Scala.

AppMon uses the Java automation library in the context of a Gatling/Scala script. According to the Scala Web Site: “Scala runs on the JVM, so Java and Scala stacks can be freely mixed for totally seamless integration.”

To do this, add the `server-rest-sdk-0.0.2.jar` automation library to your project’s classpath. It allows you to use the objects provided by the library.

```
// AppMon automation library

import com.dynatrace.sdk.server._, exceptions._, sessions._, sessions.models._, testautom
```

```
// --------- Context -----------------

val dt_server_host = "localhost" // Server Hostname
val dt_server_port = 8021        // Server Port
val dt_ssl = true                // use SSL ?
val dt_user = "admin"            // User ID
val dt_pwd = "admin"             // User password
val dt_profile = "easyTravel"    // System Profile

// get a client configuration object
  val dc = new DynatraceClient(new BasicServerConfiguration(dt_user, dt_pwd, dt_ssl, dt_s
```

WEB API test run

When using Gatling to run Web API tests, this interface creates a new test run with metadata. When this method is called, the response contains a unique ID for your test metadata on the AppMon Server. This ID is often referred to as `testRunId`. The AppMon Server uses the ID to distinguish test executions. Specify it in the request headers (see below) when executing your tests.

```
// initialize the context for our Test Run
  val ctrr = new CreateTestRunRequest()
  ctrr.setSystemProfile(dt_profile)       // System Profile.
  ctrr.setCategory(TestCategory.WEB_API) // Category of the test. Valid value is one of U
  ctrr.setVersionMajor("1")               // Major version.
  ctrr.setVersionMinor("0")               // Minor version.
  ctrr.setVersionRevision("0")            // Revision.
  ctrr.setVersionBuild("1")               // Build number.
  ctrr.setVersionMilestone("1")           // Milestone.
  ctrr.setMarker("1.0.0.1.1")             // Special version marker, used in the heat-fiel
  ctrr.setPlatform("Linux")               // The platform the test runs on. If not defined

  // Register the Test run and get the ID
  val dt_testrunid = new TestAutomation(dc).createTestRun(ctrr).getId()
```

Use header tags to convey context information with your requests

By default, AppMon labels recorded web requests from their URIs. In order to give them meaningful names (such as step names in a test scenario), a specific header tag can be added to the web requests.

This tag can also include a reference to the `testrunid` to indicate the test run a particular web request is part of.

Assemble the tag

The following helper function builds a AppMon header tag with the expected format, from the provided context data.

See Web API tests for more information.

```
def dt_web_api_tag(TN:String, TR:String) : Map[String, String] =
  {
       /*  Web API test attributes for request header :

TN      Test Name (TN) should be provided using this field.
               Metrics for requests with the same name are aggregated.
               You can execute as many tests as desired within the same test run,
               but give them different names if they are not testing the same thing.

TR      TestRunId (TR), from a registered test run.
               See REST API POST testruns for more information.
               * */

      var value:String = "TN="+TN+";TR="+TR

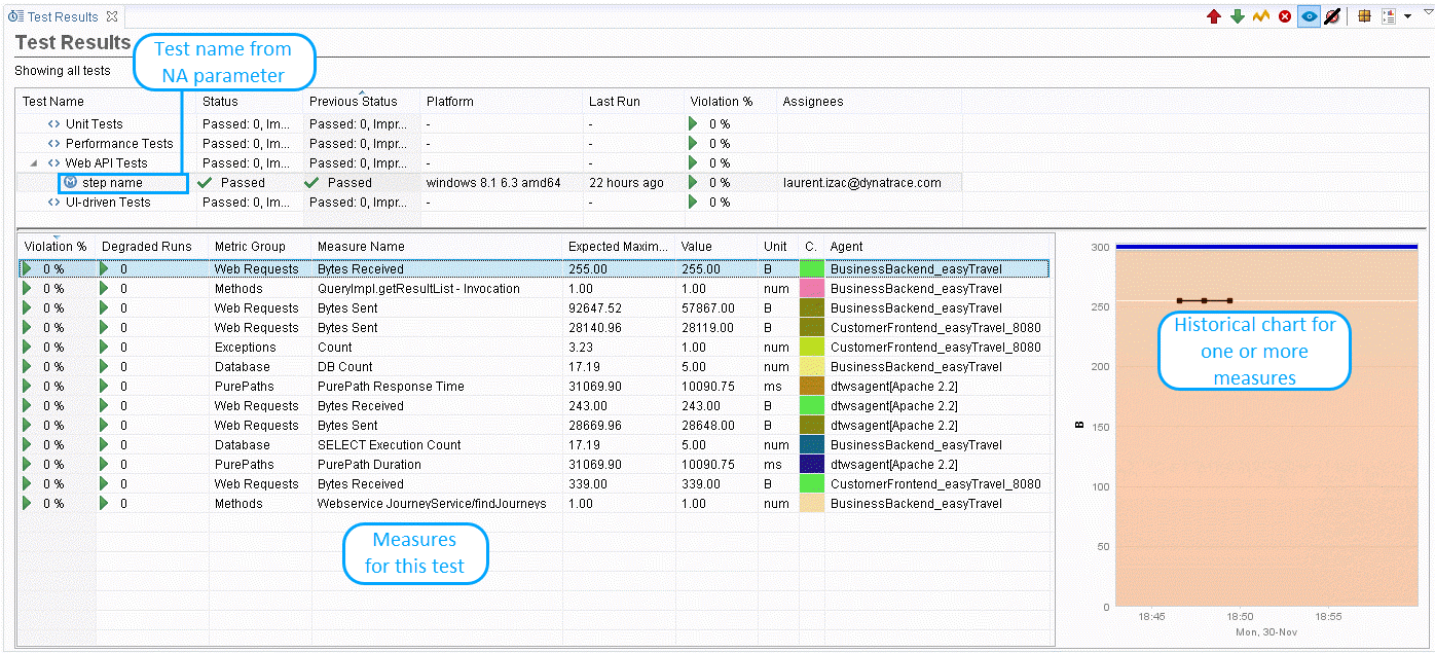
      return Map("x-dynaTrace" -> value)
   }
```

Use the tag

When issuing HTTP requests, add the AppMon tag in the header by calling the aforementioned helper function with a name for the test and the testRunId:

```
    val chain_1 = exec(http("request")
                  .get("/REST/SomeFunction")
                  .headers(dt_web_api_tag("My Web API test", dt_testrunid))
                  )
```

Analyze results in AppMon



LOAD test run

When using Gatling to run Load Tests, you can :

Use hooks to start/stop session recording and switch configuration

Before actually starting the test, you can use the **before** hook to:

- Switch to a given configuration (optional) so that AppMon captures the code execution with a specific level of details.
- Start session recording to create a session on disk to store the recorded traffic.

When the test is done, use the **after** hook to:

- Stop session recording to close the session.
- Switch back to default configuration to revert DTAM to its defaults settings regarding transaction capturing details.

```
// --------- Initialization ------------------


// get a SystemProfiles object from the client configuration to be able to interact with
  val sp = new SystemProfiles(dc)


  // initialize the context for recording a session with our test run
  val srr = new StartRecordingRequest(dt_profile)
  srr.setPresentableName("presentableName")    // User-readable presentable name for the s
  srr.setDescription("description")            // Description for the session to be storec
  srr.setRecordingOption(RecordingOption.ALL) //   ALL : All PurePaths including time ser
                                              //   VIOLATIONS : Only PurePaths marked as
                                              //   TIME_SERIES : Time series only.
  srr.setSessionLocked(false)                 // true to lock the session that is recorde
  srr.setTimestampAllowed(true)               // true to append timestamp information to


// get a Sessions object from the client configuration for the given parameters
  val s = new Sessions(dc)
```

```
before {
  // Optional: switch to a specific configuration
  // to allow for specific details level in captured data
  // Note: make sure you created such configuration in AppMon before using it.
  sp.activateProfileConfiguration(dt_profile, "myCustomConfigurationName")


  // start session recording for this test with given parameters
  s.startRecording(srr)}
```

```
after {
    // stop AppMon session recording for this test
      s.stopRecording(dt_profile)


    // switch back to default AppMon configuration
    sp.activateProfileConfiguration(dt_profile, "Default")
 }
```

Use header tags to convey context information with your requests

By default, AppMon labels recorded web requests from their URIs. In order to give them meaningful names (such as step names in a test scenario), a specific header tag can be added to the web requests.

This tag can also include context information about the load test (script name, platform, virtual user, …).

Assemble the tag

The following helper function builds a AppMon header tag with the expected format, from the provided context data.

See Integration with Web Load Testing and Monitoring Tools for more information.

```
def dt_load_tag(
               NA:String
             , PC:String=""
             , VU:String=""
             , ID:String=""
             , GR:String=""
             , AN:String=""
             , SN:String=""
             , TE:String=""
             )
  : Map[String, String] =
  {

       /*  Load test attributes for request header :
```

```
NA      The timer NAme of the request.
                This can be the timer or transaction name used in the load test script to
                or the document/page title, or any other human-readable URL encoded identi
              NA=<timername>
              Required
PC      The Page Context contains information about what document in the currently process
               The following syntax is recommended, though not required:
               - If it is a named frame, then the value starts with the frame name.
               - The document number, unique for the page, is appended after a period.
               If embedded documents are cached, this number need not be progressive.
                PC=<FrameName>.<DocId>
              optional
VU      The unique number of the Virtual User that sends the request.
              VU=<Id>
              optional
ID      The unique request ID (serial number).
                This string should be unique for one web request or a set of web requests
              ID=<Id>
              optional
SI      The Source ID can be used to identify the product that triggered the request:
                For example WLT (Web Load Testing), SYM  (dynaTrace Synthetic Monitoring),
            optional
GR      Geographic Region, useful only for the Synthetic Monitoring solution.
              This contains arbitrary text.
              optional
AN      Agent Name: the logical name of the Agent from which the request originated.
              This information is used by Synthetic Monitoring.
              optional
SN      Script Name.
                This groups a set of requests that make up a multi-step transaction, for e
              optional
TE      TEst Name is the name of the entire load test.
              It uniquely identifies a test run.
              optional
*/

    var value:String = "NA="+NA+";VU="+VU+";PC="+PC+";ID="+PC+";SI=Gatling;GR="+GR+";AN

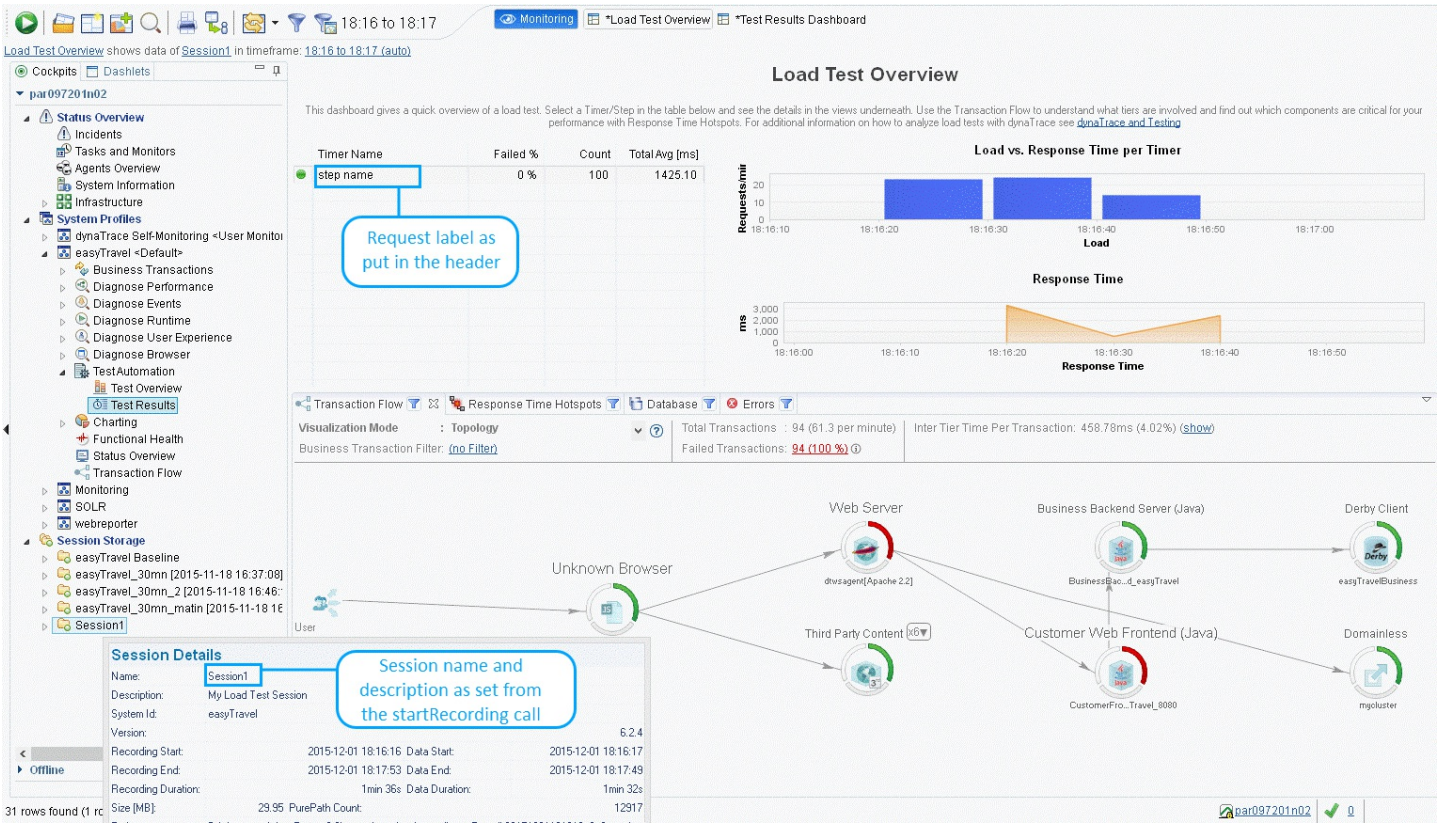
    return Map("x-dynaTrace" -> value)
  }
```

Use the tag

When issuing HTTP requests, add the AppMon tag in the header by calling the aforementioned helper function. Then provide the name, testRunId and other optional contextual data:

```
val chain_0 = exec(http("request")
                  .get("/")
                  .headers(dt_load_tag("step name"))
                  )
```

Analyze results in AppMon



Sample DSL scripts

In the attached .zip file, you will find 2 sample simulations, one for a Web API test context, the other for a Load test context.

Monitor Cloud Foundry

- Java agent in Cloud Foundry
- .NET agent in Cloud Foundry
- Node.js agent in Cloud Foundry

You can configure various AppMon agents to monitor Cloud Foundry environments:

Monitor cloud applications

On this page

Cloud monitoring
Amazon AWS
Microsoft Azure
Docker
Cloud Foundry
VMware
Cloud technology support

Related pages

Monitor Amazon applications
Monitor Microsoft Azure
Monitor Docker applications
Monitor Cloud Foundry
VMware

Cloud monitoring

Cloud is a vague term, but the core concept of on-demand and automated provisioning turns infrastructure into commodity. This lets you focus on what matters most to you: application performance. With AppMon you have chosen an APM (Application Performance Management) solution that exactly follows this approach: AppMon actually monitors your application rather than the environment in which it is running. This enables you to identify whether or not any application failure draws from your infrastructure or any of the cloud services your application uses or from the application itself. Additionally, AppMon comes with integrated UEM (User Experience Management), which is amongst others important for geographic optimization of your global deployment. Thus, we define cloud monitoring as application monitoring plus end user experience management.

While AppMon supports and runs in all cloud environments, there is special support for Amazon's AWS, Microsoft's Windows Azure and VMWare-based Clouds:

Amazon AWS

- Deploying AppMon with your Application in EC2
- Monitoring your Application on Amazon Cloud
- Analyzing Impact of AWS Infrastructure on Performance

Microsoft Azure

- How to deploy Dynatrace on Microsoft Azure

Docker

- Monitor Docker applications

Cloud Foundry

- Deploy Appmon Agents on Cloud Foundry

VMware

- Monitoring your Application and the virtual Infrastructure
- Running the AppMon Server in VMWare
- Analyzing Impact of VMWare on Performance

Cloud technology support

In addition to supporting a wide range of technologies, AppMon also provides special support and analytic capabilities for the important field of Big Data.

Monitor Docker applications

Related pages

Monitor dockerized apps with AppMon - Kubernetes and OpenShift

Monitor dockerized apps with AppMon

You can configure AppMon to monitor applications wrapped in docker containers:

- Docker applications (basic)
- Docker applications (Kubernetes and OpenShift)

Monitor dockerized apps with AppMon

On this page

Overview

Option A: inheritance-based approach

Option B: composition-based approach

Q&A

Related pages

Monitor dockerized apps with AppMon - Kubernetes and OpenShift

Overview

This page describes two methods for integrating an Application Monitoring agent with a Dockerized application. These methods, referred to as the “inheritance-based” and “composition-based” approach on this page, have proven to be most effective by our customers. Depending on your particular situation, you may find the one or the other more suitable. To make your decision easier, we have listed known pros and cons.

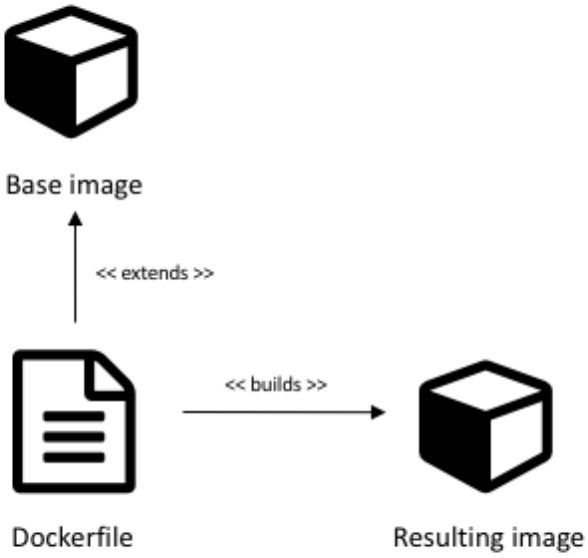
Option A: inheritance-based approach

In technical terms, Dockerizing an application process typically involves two parts:

- A base image, such as `java:8` or `node:7`, which provides the basic execution environment
- A `Dockerfile` that augments selected base image with application-specific setup instructions.

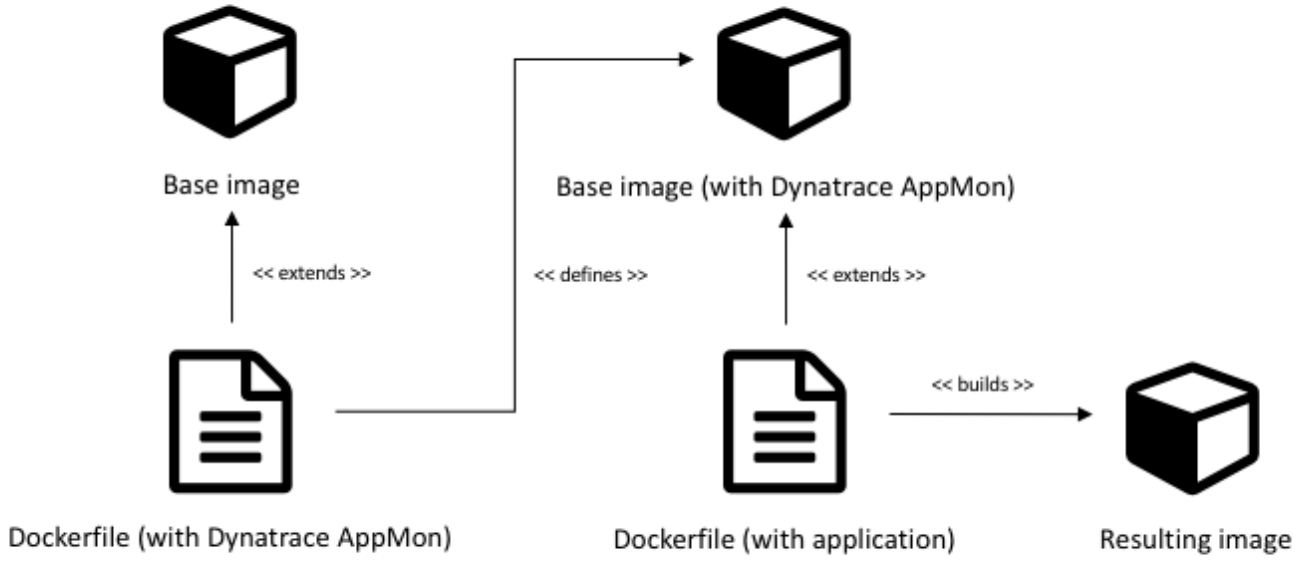
Building your `Dockerfile` using the `docker build` command creates the desired Docker image.

A comprehensive set of base images is provided on the Docker Hub. You can also read Using Dockerfiles to Automate Building of Images and Best Practices for Writing Dockerfiles for more information.



Examples

You can create base images for your application that are ready for monitoring. Weaving the agent into base images auto-enables monitoring by making AppMon part of the “platform”.



Example: Java

This example shows a `Dockerfile` that extends the official openJDK Docker image base image and downloads and extracts the agent. A number of environment variables are set for convenience, such as `DT_AGENT_NAME` and `DT_AGENT_COLLECTOR`, which you can populate with your own values. Additionally, `JAVA_OPTS` is initialized with a proper `-agentpath` that picks up these variables and points to the 64-bit agent in `DT_AGENT_LIB64`.

```
FROM openjdk:8

ENV DT_AGENT_INSTALLER_URL "http://files.dynatrace.com/downloads/OnPrem/dynaTrace/6.5/6.5

ENV DT                     "/dynatrace"
ENV DT_AGENT_LIB32         "${DT}/agent/lib/libdtagent.so"
ENV DT_AGENT_LIB64         "${DT}/agent/lib64/libdtagent.so"

ENV DT_AGENT_NAME          "java"
ENV DT_AGENT_COLLECTOR     "127.0.0.1:9998"

ENV JAVA_OPTS              "-agentpath:${DT_AGENT_LIB64}=name=${DT_AGENT_NAME},collector=

# Install the AppMon Agent
RUN curl -L -o /tmp/`basename ${DT_AGENT_INSTALLER_URL}` ${DT_AGENT_INSTALLER_URL} && \
    java -jar /tmp/`basename ${DT_AGENT_INSTALLER_URL}` -t ${DT} && \
    rm -f /tmp/`basename ${DT_AGENT_INSTALLER_URL}`
```

Building this Dockerfile using `docker build . -t openjdk:8-dtappmon -f ./Dockerfile` for example creates a new Docker image with name `openjdk` and tag `8-dynatrace-appmon` in your local Docker registry. With each application build, you can now create a “latest” application image that extends `openjdk:8-dtappmon` as shown in the following example (where `repo.internal` refers to a fictitious binary repository and `my-app` is a fictitious application). This also gives you the possibility to override the `DT_AGENT_NAME` environment variable to be more specific than its definition in the base image.

```
FROM openjdk:8-dtappmon

ENV DT_AGENT_NAME "my-app"
ADD https://repo.internal/my-app/builds/latest.tar.gz /app

CMD java ${JAVA_OPTS} -jar /app/my-app.jar
```

Example: Nginx

This example shows a `Dockerfile` that extends the official NGINX Docker image base image and downloads and extracts the agent. A number of environment variables are set for convenience, such as `DT_WSAGENT_NAME` and `DT_WSAGENT_COLLECTOR`, which you can populate with your own values. Additionally, `DT_WSAGENT_INI` provides a proper agent configuration and overrides the container's `CMD` to load the agent with NGINX.

```
FROM nginx:1.9

ENV DT_WSAGENT_INSTALLER64_URL "http://files.dynatrace.com/downloads/OnPrem/dynaTrace/6.5

ENV DT                         "/dynatrace"
ENV DT_WSAGENT_INI             "${DT}/agent/conf/dtwsagent.ini"
ENV DT_WSAGENT_BIN64           "${DT}/agent/lib64/dtwsagent"
ENV DT_WSAGENT_LIB64           "${DT}/agent/lib64/libdtagent.so"

ENV DT_WSAGENT_NAME            "nginx"
ENV DT_WSAGENT_COLLECTOR       "127.0.0.1:9998"
ENV DT_WSAGENT_LOG_LEVEL       "info"

# Install the AppMon Agent
ENV DT_INSTALL_DEPS            "curl"
RUN apt-get update && apt-get install -y --no-install-recommends ${DT_INSTALL_DEPS} && \
    curl -L -o /tmp/`basename ${DT_WSAGENT_INSTALLER64_URL}` ${DT_WSAGENT_INSTALLER64_URI
    tar -C /tmp -xf /tmp/`basename ${DT_WSAGENT_INSTALLER64_URL}` && sh /tmp/dynatrace-ws
    rm -f /tmp/dynatrace-* && apt-get remove --purge -y ${DT_INSTALL_DEPS} && rm -rf /var

# Configure the AppMon WebServer Agent
RUN sed -i -r "s/^#?Name.*/Name ${DT_WSAGENT_NAME}/;s/^#?Server.*/Server ${DT_WSAGENT_COI
CMD sh -c "(${DT_WSAGENT_BIN64} &) && LD_PRELOAD ${DT_WSAGENT_LIB64} nginx -g 'daemon off
```

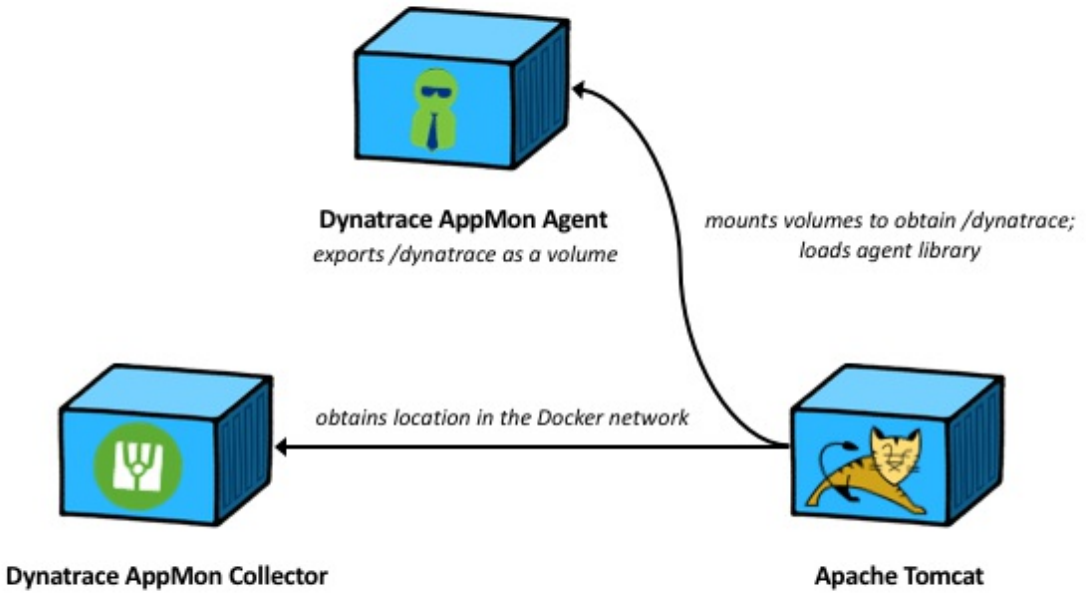
Analysis

- On the **pros** side, this approach reduces the integration of a monitoring solution like AppMon to less frequent, preparatory work that does not add any overhead to the much more frequent process of building, shipping and running Dockerized applications.
- On the **cons** side, depending on your particular use-case and on the technologies you use, you must integrate with each of these technologies by hand. And since this approach tightly ties together the agent with a particular technology in the same base image, these base images may have to be recreated when switching to a new version of either the technology or of AppMon.

Option B: composition-based approach

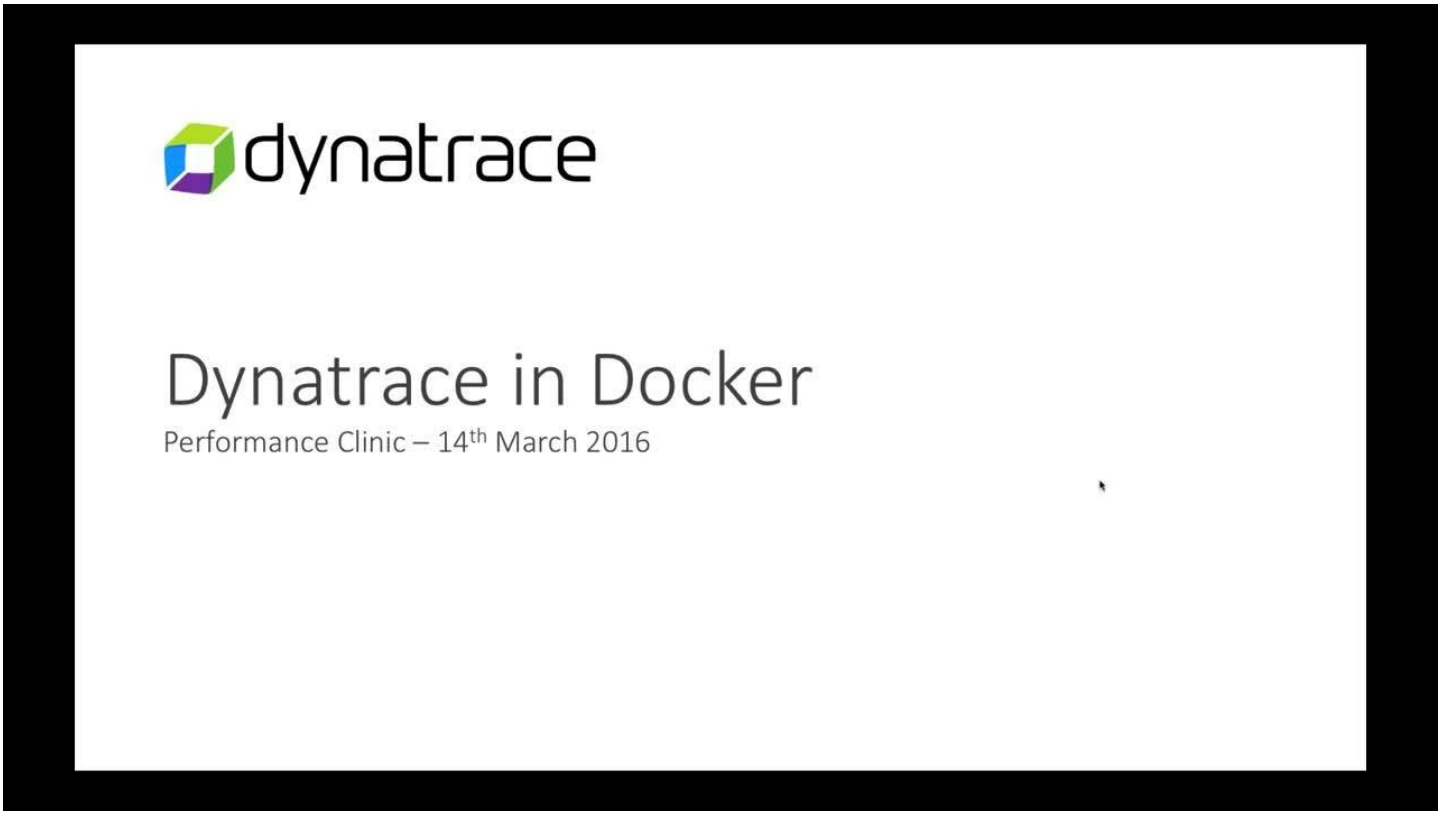
With the “composition-based” approach, you use the AppMon/agent Docker image that includes all variants of the AppMon agent – that you can configure to attach to your existing Docker containers.

Technically speaking, this approach uses a feature of Docker that allows a Docker container to export parts of its file system as Docker volumes and thereby make them obtainable by other, interested containers. THis looks similar to the following.



Examples

The following examples assume that you are already running a `dynatrace/agent` Docker container by name `dtagent` that exports the `/dynatrace` installation folder as a volume. The AppMon in Docker project on GitHub includes scripts to do exactly that and even allows you to conveniently set up an entire AppMon environment in Docker. More information can be found in the following Performance Clinic.



Example: Apache Tomcat

This `docker-compose.yml` file mounts the exported volumes of a container `dtagent` and initializes the `CATALINA_OPTS` environment variable with a proper `-agentpath`.

```
tomcat:
  image: tomcat
  ports:
  - 8080
  volumes_from:
  - dtagent
  environment:
    CATALINA_OPTS: "-agentpath:/dynatrace/agent/lib64/libdtagent.so=name=tomcat,collector
  command: catalina.sh run
```

Example: NGINX

This `docker-compose.yml` file mounts the exported volumes of a container `dtagent` and overrides the container's `CMD` to load the agent with NGINX.

```
nginx:
  image: nginx
  ports:
  - 80
  volumes_from:
  - dtagent
  environment:
    DT_AGENT_NAME: "nginx"
    DT_AGENT_COLLECTOR: "127.0.0.1:6698"
    LD_PRELOAD: "/dynatrace/agent/lib64/libdtagent.so"
  command: sh -c "/dynatrace/run-wsagent.sh && nginx -g 'daemon off;'"
```

Analysis

- On the **pros** side, this approach neatly contributes to a clean separation of concerns, which is a design principle in the containers world. Also, you don't have get agents into your base images. A simple configuration at run time is all you need to get your containers monitored.
- On the **cons** side, while the Docker run time has great support for exchanging volumes between containers, doing so on a container orchestration platform such as Kubernetes or OpenShift can render your application configurations overly complex.

Q&A

Can I monitor applications running on alpine in docker?

As of now, the AppMon agent does not support monitoring applications which are statically or dynamically linked against musl standard C library (instead of the more widely distributed and common glibc). As a temporary workaround that will also save a considerable amount of disk space, you can use a vanilla Alpine Docker image and install *glibc* together with your runtime environment, such as *Java*, on top. This means that you cannot install Alpine's OpenJDK using *apk* since it, like all Alpine packages, depends on musl. Refer to anapsix/alpine-java as your base image, which at the time of writing combines *Alpine 3.4*, *glibc 2.23-r3* and *Oracle Java 1.8.0_71* out of the box.

```
FROM anapsix/alpine-java

ENV DT_AGENT_INSTALLER_URL "http://files.dynatrace.com/downloads/OnPrem/dynaTrace/6.5/6.5

ENV DT                     "/dynatrace"
ENV DT_AGENT_LIB32         "${DT}/agent/lib/libdtagent.so"
ENV DT_AGENT_LIB64         "${DT}/agent/lib64/libdtagent.so"

ENV DT_AGENT_NAME          "java"
ENV DT_AGENT_COLLECTOR     "127.0.0.1:9998"

ENV JAVA_OPTS              "-agentpath:${DT_AGENT_LIB64}=name=${DT_AGENT_NAME},collector=

# Install the AppMon Agent
RUN wget -O /tmp/`basename ${DT_AGENT_INSTALLER_URL}` ${DT_AGENT_INSTALLER_URL} && \
    java -jar /tmp/`basename ${DT_AGENT_INSTALLER_URL}` -t ${DT} && \
    rm -f /tmp/`basename ${DT_AGENT_INSTALLER_URL}`
```

If you need an application server like Apache Tomcat on top, see one of their (glibc-based) Dockerfiles and augment the above example as needed, for example with Apache Tomcat 7 on Java 8.

Can I monitor dockerized applications on kubernetes or OpenShift?

Yes. See the second part of this series.

Can I run easyTravel in docker?

EasyTravel has been fully Dockerized and is available at the easyTravel in Docker GitHub project. You can easily instrument it using Dynatrace in Docker.

Monitor dockerized apps with AppMon - Kubernetes and OpenShift

On this page

Option A: inheritance-based approach

Option B: composition-based approach

Related pages

Monitor dockerized apps with AppMon

The page How to monitor dockerized apps with AppMon describes how to monitor Dockerized apps in vanilla Docker environments with AppMon.

This page elaborates on monitoring Dockerized applications in Kubernetes and Red Hat OpenShift(v3) environments. If you are unsure how these two container platform technologies relate to each other, OpenShift is Kubernetes for the enterprise.

As described in How to monitor dockerized apps with AppMon, depending on your particular situation, you may find the one or the other of the following approaches more suitable. Again, to make your decision easier, we have listed known pros and cons.

Option A: inheritance-based approach

The goal of the “inheritance-based approach” is to put the AppMon agent into your Docker base images. Since Kubernetes and OpenShift are both container platforms, this approach allows you to reuse your monitoring enabled images on any of these platforms. However, since OpenShift is a secure container platform running containers, processes executed as “root” (which is how most Docker images are built) is prohibited by default. See the OpenShift Container Image Guidelines to learn how to prepare your Docker images for OpenShift. See How to monitor dockerized apps with AppMon for information on how to apply this approach to your Docker images.

Example: Java

Since the particular technologies inside your base images have are being monitored by AppMon, you only need simple runtime configuration settings to bind the agent to the AppMon Collector.

The following example defines a ReplicationController with a container named *catalog* inside an inline Pod. The environment variables `DT_AGENT_NAME` and `DT_AGENT_COLLECTOR` (as defined in our How to monitor dockerized apps with AppMon) override the respective values provided in a fictitious `acmeco/my-app` base image.

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: my-app
spec:
  template:
    spec:
      containers:
      - name: my-app
        image: acmeco/my-app
        env:
        - name: DT_AGENT_NAME
          value: "my-app"
        - name: DT_AGENT_COLLECTOR
          value: "dtappmon-collector.acmeco.com:9998"
        ports:
        - containerPort: 8080
```

Example: NGINX

Since the particular technologies inside your base images have are being monitored by AppMon, you only need simple runtime configuration settings to bind the agent to the AppMon Collector.

The following example defines a ReplicationController with a container named *api-gateway* inside an inline Pod. The environment variables `DT_WSAGENT_NAME` and `DT_WSAGENT_COLLECTOR` (as defined in How to monitor dockerized apps with AppMon) override the respective values provided in a fictitious `acmeco/my-app` base image .

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: my-app
spec:
  template:
    spec:
      containers:
      - name: my-app
        image: acmeco/my-app
        env:
        - name: DT_WSAGENT_NAME
          value: "my-app"
        - name: DT_WSAGENT_COLLECTOR
          value: "dtappmon-collector.acmeco.com:9998"
        ports:
        - containerPort: 80
```

Analysis

- On the **pros** side, once the agent has been put into your Docker base images, it doesn't matter much on which container platform runs your applications. Also, this approach reduces the integration of AppMon to less frequent, preparatory work that does not add any overhead to the frequent process of building, shipping and running Dockerized applications.
- On the **cons** side, depending on your particular use-case and on the technologies you use, you must manually integrate each of these technologies. And since this approach tightly ties together the agent with a particular technology in the same base image, these base images may altogether have to be recreated when switching to a new version of either the technology or of AppMon.

Option B: composition-based approach

With the composition-based approach, you use the AppMon/agent Docker image that includes all variants of the AppMon agent – that you can configure to attach to your existing Docker containers. For information on how this approach actually works, refer to Monitor dockerized apps with AppMon.

Examples

The following examples integrate the agent Docker image with a fictitious container `acmeco/my-app` inside a ReplicationController. To mimic the Docker capability of exchanging volumes between containers (as described in Monitor dockerized apps with AppMon), manual steps are required to achieve the same behavior on Kubernetes-based platforms. The following shows what the replication controller looks like without the agent in place.

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: my-app
spec:
  template:
    spec:
      containers:
      - name: my-app
        image: acmeco/my-app
        ports:
        - containerPort: 8080
```

Make the agent available inside the application container

To make the agent available inside the application container, declare a shared Volume inside the Pod definition. In the following example, volume is set to an empty directory on the container's underlying host.

```
apiVersion: v1
kind: ReplicationController
metadata:
 name: my-app
spec:
 template:
   spec:
     containers:
     - name: my-app
       image: acmeco/my-app
       ports:
       - containerPort: 8080
     volumes:
     - name: dtappmon-agent-volume
       emptyDir: {}
```

Next, add the agent Docker image to the list of containers, have it mount the volume and use the Container Lifecycle Hook`postStart` to copy the agent installation present in `/dynatrace` inside `dynatrace/oneagent` to `/srv/dynatrace` (which equals the mount path of the volume).

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: my-app
spec:
  template:
    spec:
      containers:
      - name: dtappmon-agent
        image: dynatrace/agent
        volumeMounts:
        - name: dtappmon-agent-volume
          mountPath: /srv/dynatrace
        lifecycle:
          postStart:
            exec:
              command: [cp, -R, /dynatrace, /srv]
      - name: my-app
        image: acmeco/my-app
        ports:
        - containerPort: 8080
      volumes:
      - name: dtappmon-agent-volume
        emptyDir: {}
```

The `dtappmon-agent` container must be defined before any application containers that want make use of it. If this order is not maintained, the application container may not be able to eventually pick up the agent once it's started. Finally the application container mounts the agent installation directory from the shared volume.

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: my-app
spec:
  template:
    spec:
      containers:
      - name: dtappmon-agent
        image: dynatrace/agent
        volumeMounts:
        - name: dtappmon-agent-volume
          mountPath: /srv/dynatrace
        lifecycle:
          postStart:
            exec:
              command: [cp, -R, /dynatrace, /srv]
      - name: my-app
        image: acmeco/my-app
        ports:
        - containerPort: 8080
        volumeMounts:
        - name: dtappmon-agent-volume
          mountPath: /srv/dynatrace
      volumes:
      - name: dtappmon-agent-volume
        emptyDir: {}
```

Load the agent into the application’s process: Java

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: my-app
spec:
  template:
    spec:
      containers:
      - name: dtappmon-agent
        image: dynatrace/agent
        volumeMounts:
        - name: dtappmon-agent-volume
          mountPath: /srv/dynatrace
        lifecycle:
          postStart:
            exec:
              command: [cp, -R, /dynatrace, /srv]
      - name: my-app
        image: acmeco/my-app
        env:
        - name: DT_AGENTNAME
          value: "my-app"
        - name: DT_COLLECTOR
          value: "dtappmon-collector.acmeco.com:9998"
        - name: JAVA_OPTS
          value: "-agentpath:/srv/dynatrace/agent/lib64/libdtagent.so"
        ports:
        - containerPort: 8080
        volumeMounts:
        - name: dtappmon-agent-volume
          mountPath: /srv/dynatrace
      volumes:
      - name: dtappmon-agent-volume
        emptyDir: {}
```

Load the agent into the application’s process: NGINX

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: my-app
spec:
  template:
    spec:
      containers:
      - name: dtappmon-agent
        image: dynatrace/agent
        volumeMounts:
        - name: dtappmon-agent-volume
          mountPath: /srv/dynatrace
        lifecycle:
          postStart:
            exec:
              command: [cp, -R, /dynatrace, /srv]
      - name: my-app
        image: acmeco/my-app
        env:
        - name: DT_AGENT_NAME
          value: "my-app"
        - name: DT_AGENT_COLLECTOR
          value: "dtappmon-collector.acmeco.com:9998"
        - name: DT_WSAGENT_BIN64
          value: "/srv/dynatrace/agent/lib64/dtwsagent"
        - name: DT_WSAGENT_INI
          value: "/srv/dynatrace/agent/conf/dtwsagent.ini"
        - name: LD_PRELOAD
          value: "/srv/dynatrace/agent/lib64/libdtagent.so"
        command: [/bin/sh, -c, /srv/dynatrace/run-wsagent.sh "${DT_WSAGENT_BIN64}" "${DT_
        ports:
        - containerPort: 80
        volumeMounts:
        - name: dtappmon-agent-volume
          mountPath: /srv/dynatrace
      volumes:
      - name: dtappmon-agent-volume
        emptyDir: {}
```

This finalizes the composition-based approach.

Analysis

- On the **pros** side, this approach neatly contributes to a clean separation of concerns, which is a design principle in the containers world. Also, you don’t have to care about getting agents into your base images, a simple configuration at runtime is all you need to get your containers monitored.
- On the **cons** side, while the Docker runtime has great support for exchanging volumes between containers, doing so on a container orchestration platform such as Kubernetes or OpenShift can render your application configurations overly complex.

Monitor a WAS liberty profile server

On this page

Overview

Add classes

Additional info

Related pages

WebSphere Liberty Profile

Monitor specific applications

WebSphere Application Server

Overview

AppMon includes two classes added to the ClassLoader sensor pack. These classes let AppMon monitor WAS Liberty Profile Server. The classes was work for WAS Liberty profile server 8.5.5.4 and later.

Add classes

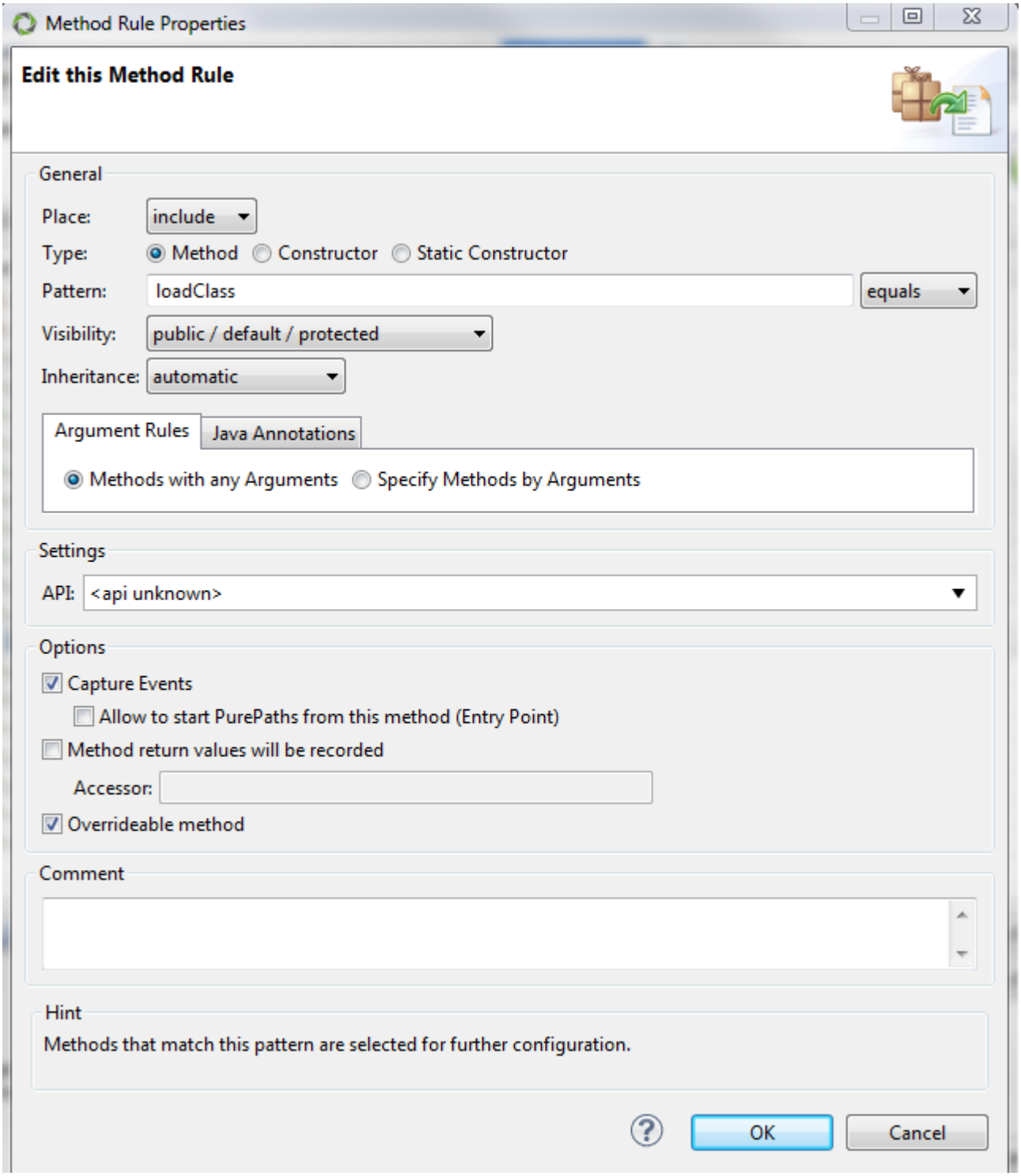
To add these classes, go into debug mode by pressing Ctrl + Shift + F9 in the AppMon Client. Select Sensor Packs, edit the ClassLoader sensor pack. Click add class and add one at a time.

org.eclipse.osgi.internal.loader.EquinoxClassLoader

org.eclipse.osgi.internal.loader.ModuleClassLoader

After adding each class edit the method and change them from all methods to loadClass(*).

The following screenshot specifies all the changes needed. Visibility needs to change to public / default / protected.



Apply your changes and then restart the JVM.

Additional info

There was an initial support ticket SUPDT-6650 with additional details on an error message that you might see. It effected the initial log in to the application with null pointer exceptions and NoClassDef Errors. This was resolved with the addition of the class loaders.

Here is the where the agentpath is added to the WAS Liberty Profile Server.

The command line is: `/opt/IBM/WebSphere/AppServer/java/jre/bin/java -javaagent:/opt/IBM/WebSphere/Liberty/bin/tools/ws-javaagent.jar -Djava.awt.headless=true -XX:MaxPermSize=256m -Xmx1024m -verbose:class -verbose:gc -Xverbosegclog:logs/verbosegc.log -agentpath:/opt/dynatrace-6.1.0/agent/lib64/libdtagent.so=name=WebSphere_App,server=ipaddress:9998 -jar /opt/IBM/WebSphere/Liberty/bin/tools/ws-server.jar ApplicationName`.

This was installed on a Linux 64-bit computer with IBM J9 VM 2.6.

Dynatrace 6.1.0.7880 build was used with fix pack 8054 applied.

WAS liberty Profile Server sensor pack attached.

The attachments contains a sensor pack. Import this sensor pack into your Dynatrace server. Click **Tools** in the top right hand corner and then select the attachments. From there you can download the sensor pack and import it into AppMon.

Set up Performance Warehouse on Oracle Database (separate tablespace for indexes)

Related pages

Oracle Application Server

Database Agent configuration

When setting up the Performance Warehouse on Oracle database, some users may use a dedicated tablespace for indexes, while the actual data is stored in a separate tablespace. The AppMon default specified in the Oracle DB schema creation script (which can be found in `DT_HOME\server\conf\repository\oracle\create.ddl`) is that the indexes are in the same tablespace. To have the indexes in a separate tablespace:

- Edit the schema creation script (`DT_HOME\server\conf\repository\oracle\create.ddl`) for the indexes. make a copy of the default schema creation script and edit the copy rather than overwrite the existing script file.
- Execute it manually on the Oracle Database.
- Once manually connected, connect the Performance Warehouse to the AppMon server by selecting **Server settings > Performance Warehouse** in the AppMon client.

The recommended setting for the different tablespaces (both data and index tablespace) are:

```
CREATE TABLESPACE XYZ DATAFILE

LOGGING
ONLINE
EXTENT MANAGEMENT LOCAL AUTOALLOCATE
BLOCKSIZE 8K
SEGMENT SPACE MANAGEMENT AUTO
FLASHBACK ON;
```

Use AspectJ to connect PurePaths in Tibco BusinessWorks

On this page

Example 1: Tibco BusinessWorks hosting a WebService

Example 2: Tibco BusinessWorks queue receiver

Example 3: connecting PurePaths for outbound SOAP requests

Instrument BusinessWorks with AspectJ and ADK

Related pages

Tibco Messaging sensor

Monitor specific applications

This page describes how to use AspectJ for technologies where PurePaths are not supported end-to-end out of the box. AppMon provides full end-to-end visibility, but you can use this page to understand how to use AspectJ for other technologies where PurePaths are not supported end-to-end out of the box. It includes examples of what can be done in Tibco BusinessWorks with AppMon. It also shows what could be done using AspectJ to join broken PurePaths(PPs) together using ADK calls.

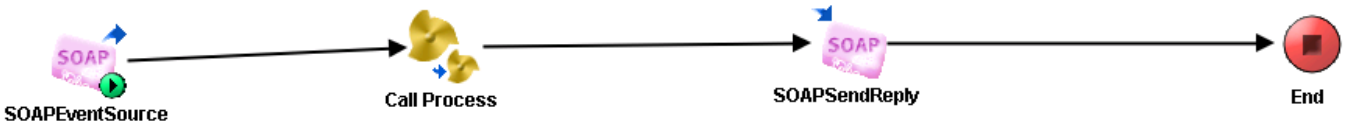
Note

Examples on this page include downloadable session files containing the sensors used in the examples. In some cases, the sensors include more that what is needed to capture required PurePath data. At a minimum, the following should be captured:

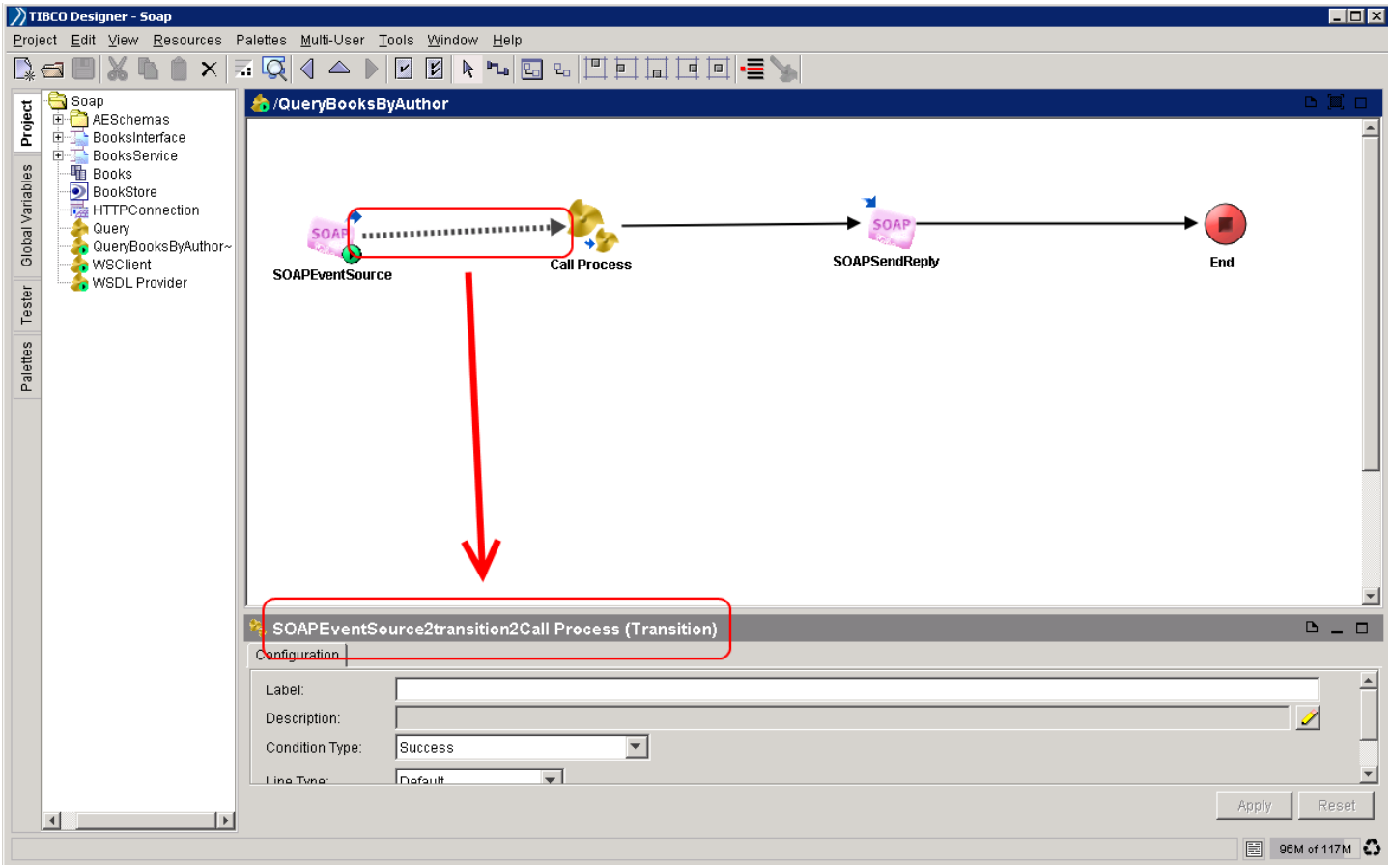
- JobDispatcher.schedule(Job) - Capture the Job parameter as it provides the Job-Id.
- JobDispatcher$JobCourier.a(Job) - Capture the Job paramater as it also provides the Job-ID.
- HawkJobListener.transitionEvaluated(String, String, String, boolean) - Capture the string values as they contain the Process name, the last complete activity in the process, and the transition of what activity is being called next.
- TaskImpl.eval(ProcessContext) - This method is the timing of how long the activity that was transitioned to (from the previous transitionEvaluated() call) took.

Example 1: Tibco BusinessWorks hosting a WebService

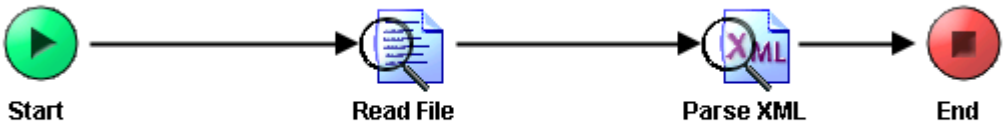
The following graphic shows a process called QueryBooksByAuthor that implements a WebService.



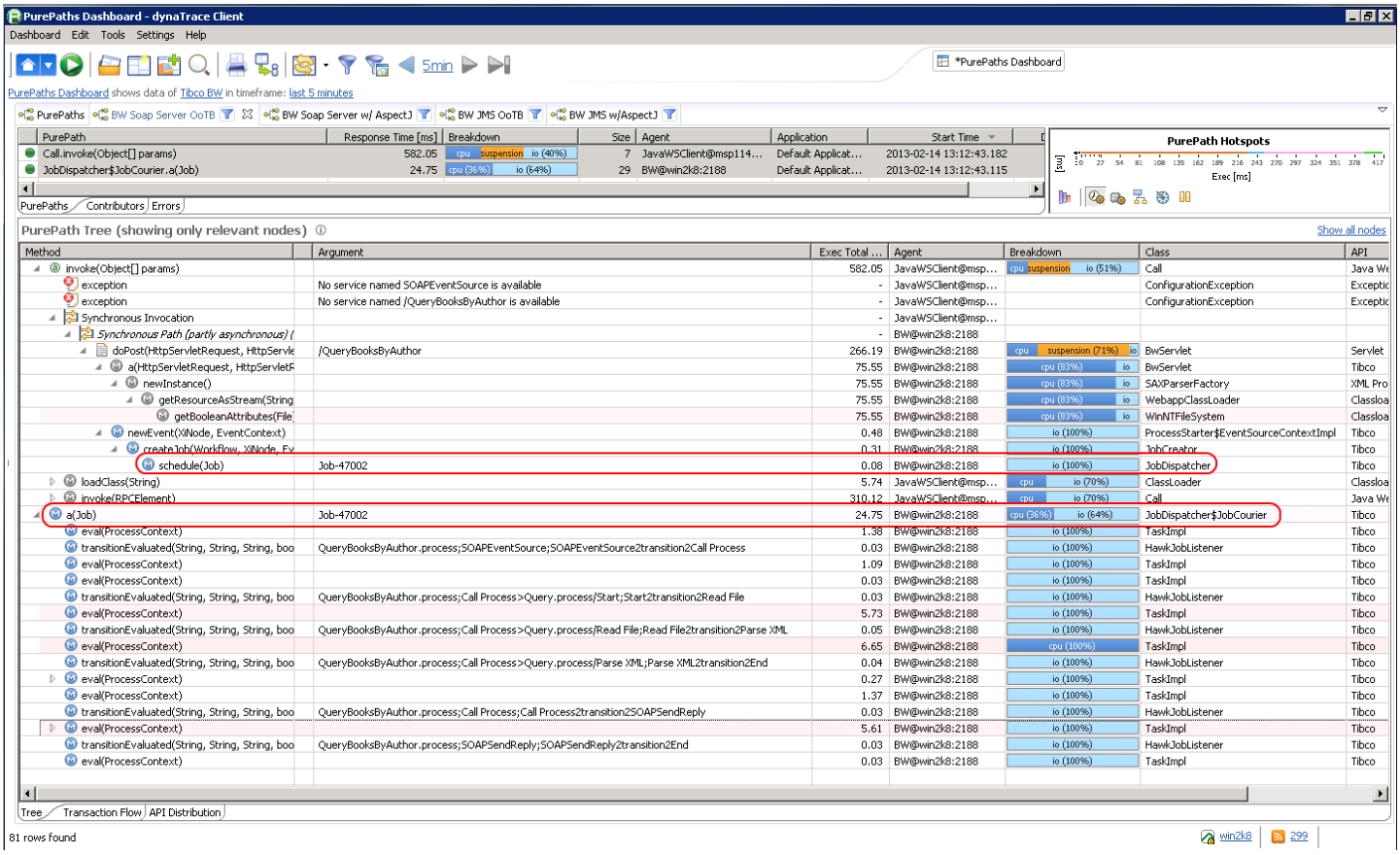
Transitions between nodes in the process flow are labeled “[srcNode]2transition2[dstNode]”.



The Call Process node calls out to the Query Process shown below.



The following screenshot shows what can be captured using AppMon:

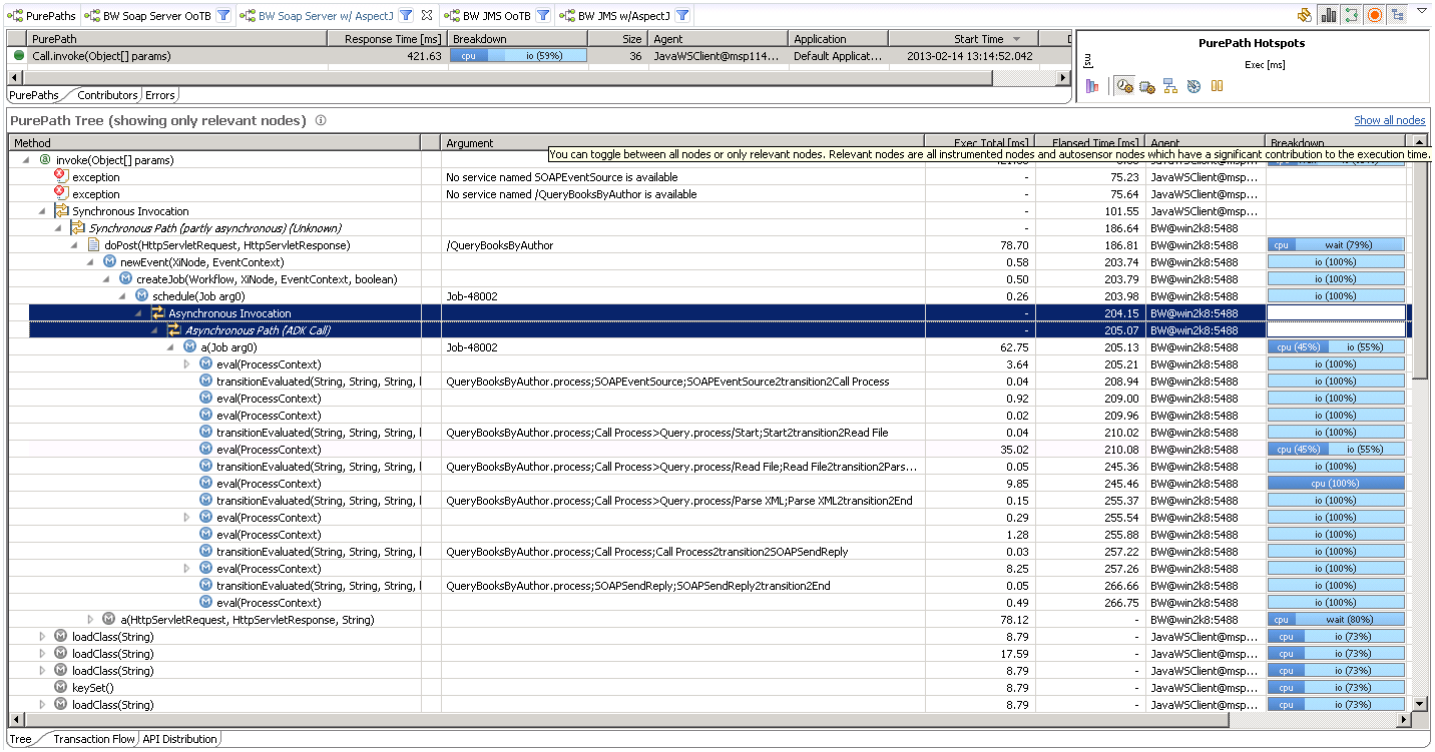


For the next example, download the following AppMon Session File: bwsoapserver-ootb.dts

This example shows two disconnected PurePaths. Manually placing some sensors can capture the Job Names (Job-47002) from both PurePaths. However, you must place two different sensors, which means a single Business Transaction (BT) can't bring these PurePaths together.

In the transitionEvaluated() method, capturing some of the arguments provides the ProcessName, the Current node, and the transition being executed. The eval() method/s after the transitionEvaluated() call is really the invocation of the next node in the Process.

Using AspectJ to weave in the ADK calls can connect these PurePaths. Below is a screenshot of what the PurePaths looks like when this is done.



The following AppMon Session file bwsoapserver-with-aspectj-adk.dts shows these two PurePaths are now joined together into a single PurePath.

The following screenshot shows the code for bwtagging.aj aspect.

```
package com.tibco.pe.core;

import com.dynatrace.adk.DynaTraceADKFactory;
import com.dynatrace.adk.Tagging;

privileged aspect BWTagging {
    //Add a field to com.tibco.pe.core.Job to hold the PP id.
    public String Job.dtTag=null;

    // Add a field to hold the Tagging object
    private Tagging JobDispatcher.tagging = DynaTraceADKFactory.createTagging();
    private Tagging JobDispatcher.JobCourier.tagging = DynaTraceADKFactory.createTagging();


    // Add a static initializer to both classes to make sure the ADK is initialized
    before () : staticinitialization(JobDispatcher) {
        DynaTraceADKFactory.initialize();
    }

    before () : staticinitialization(JobDispatcher.JobCourier) {
        DynaTraceADKFactory.initialize();
    }

    //Instrument the schedule(Job) method
    pointcut schedule(Job job, JobDispatcher jobDispatcher) : execution (public void schedule(Job)) && args(job) && target (jobDispatcher);

    void around (Job job, JobDispatcher jobDispatcher) : schedule(job, jobDispatcher) {
        if (null == job.dtTag) {
            job.dtTag =jobDispatcher.tagging.getTagAsString();
            jobDispatcher.tagging.linkClientPurePath(true);
//          System.out.println("+++JDF+++ Linked PurePath: " + job.dtTag + " Workflow: " + job.getWorkflow().getName());
        } else {
//          System.out.println("+++JDF+++ SubSchedule for: " + job.dtTag);
        }
        proceed(job, jobDispatcher);
    }

    // Instrument the a(Job) meethod
    pointcut a(Job job, JobDispatcher.JobCourier jobCourier) : call (private boolean a(Job)) && args(job) && target(jobCourier);

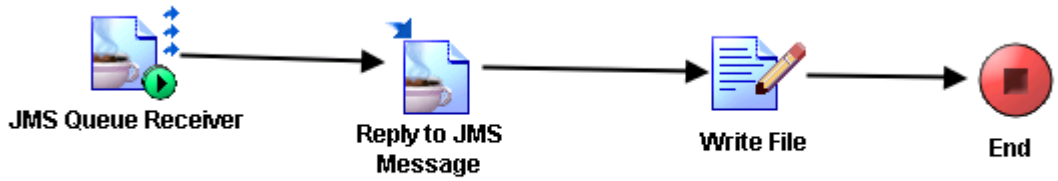
    before (Job job, JobDispatcher.JobCourier jobCourier) : a(job, jobCourier) {
//      System.out.println("+++JDF+++ Serverside:      "+ job.dtTag);
        jobCourier.tagging.setTagFromString(job.dtTag);
        jobCourier.tagging.startServerPurePath();
    }

    after (Job job, JobDispatcher.JobCourier jobCourier) : a(job, jobCourier) {
        jobCourier.tagging.endServerPurePath();
    }
}
```

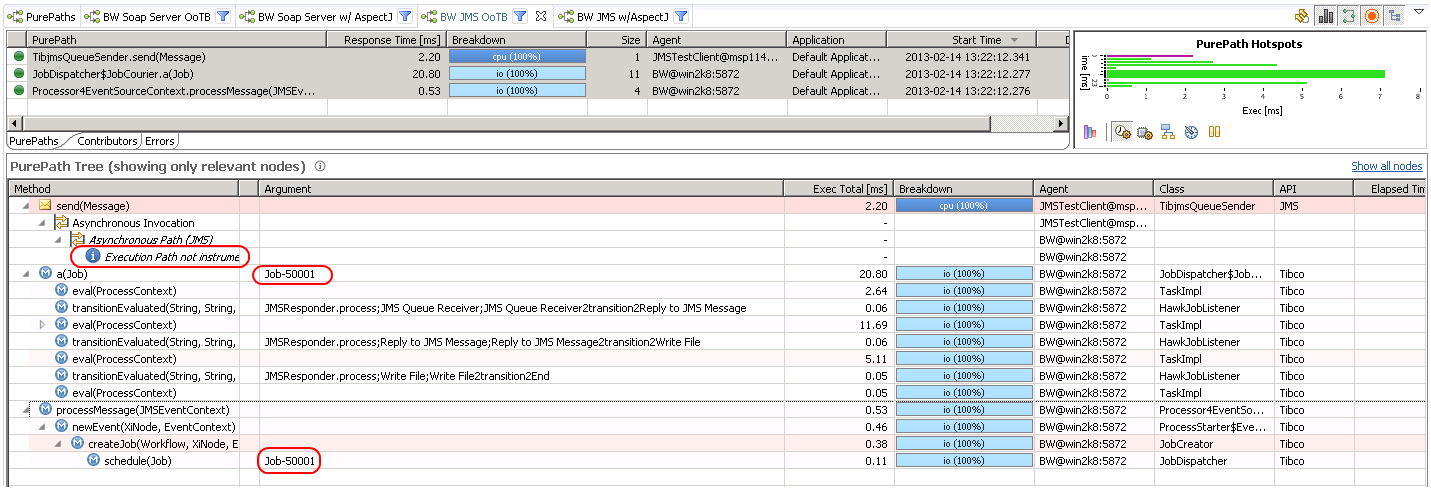
Download the AspectJ Download: bwtagging.aj

Example 2: Tibco BusinessWorks queue receiver

The following screenshot shows a process that listens to a queue, echoes the message back on the response queue and then writes to a log file.



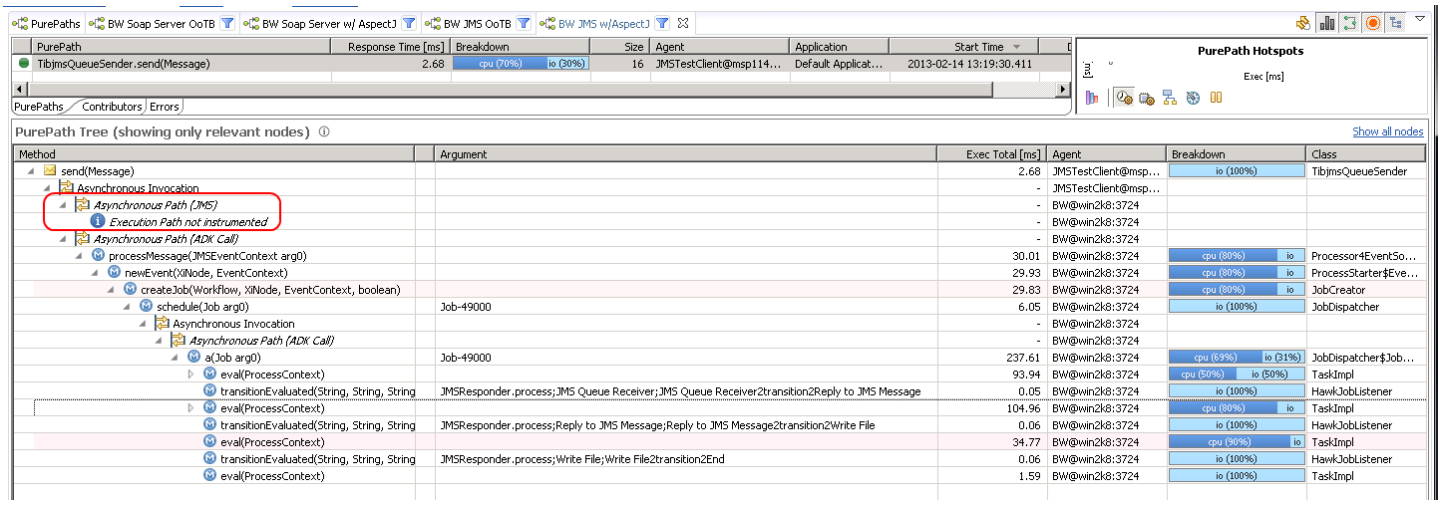
AppMon shows the following disconnected PurePath for this process.



This AppMon Session File bwjms-ootb.dts

Shows three separate PurePaths.

The Aspect defined above joins the PurePaths running in BusinessWorks together, but the PurePaths are still disconnected from the JMSClient. It appears BusinessWorks uses methods to handle JMS that isn't currently tagged in AppMon. Adding another Aspect to extract the AppMon PurePath ID from the JMS property and passing that to the ADK connects the PurePaths, as shown in the following screenshot.



It also shows in this AppMon Session file. bwjms-with-aspectj-adk.dts

Here's the code to the bwjmstagging.aj aspect:

```
package com.tibco.pe.core;

import java.util.Enumeration;

import javax.jms.JMSException;
import javax.jms.Message;

import com.dynatrace.adk.DynaTraceADKFactory;
import com.dynatrace.adk.Tagging;
import com.tibco.plugin.share.jms.impl.JMSEventContext;
import com.tibco.plugin.share.jms.impl.Processor4EventSourceContext;

public aspect BWJMSTagging {
    // Add a field to hold the Tagging object
    public Tagging Processor4EventSourceContext.tagging = DynaTraceADKFactory
            .createTagging();

    // Add a static initializer to both classes to make sure the ADK is initialized
    before() : staticinitialization(Processor4EventSourceContext) {
        DynaTraceADKFactory.initialize();
    }


    //Instrument the processMessage(JMSEventContext) method
    pointcut processMessage(JMSEventContext jmsEventCTX,
            Processor4EventSourceContext processor4EventSourceContext) : call (public void processMessage(JMSEventContext)) && args(jmsEventCTX) && target(processor4EventSourceContext);

    before(JMSEventContext jmsEventCTX,
            Processor4EventSourceContext processor4EventSourceContext) : processMessage(jmsEventCTX,processor4EventSourceContext) {
        try {
            Message message = jmsEventCTX.getMessage();
            String dtTag = message.getStringProperty("dtdTraceTagInfo");
            System.out.println("++JDF++ Recieved JMS PP - " + dtTag);
            processor4EventSourceContext.tagging.setTagFromString(dtTag);
            processor4EventSourceContext.tagging.startServerPurePath();

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

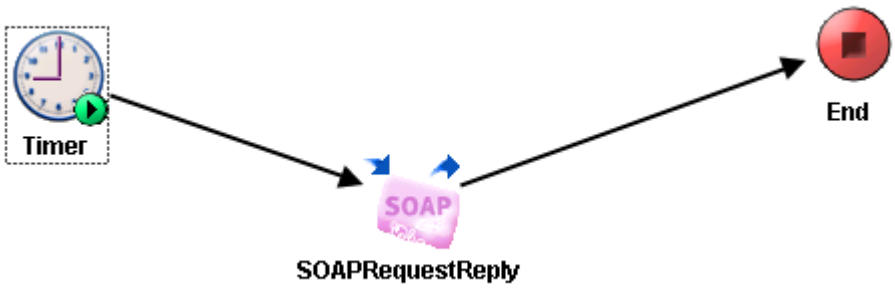
    after(JMSEventContext jmsEventCTX,
            Processor4EventSourceContext processor4EventSourceContext) : processMessage(jmsEventCTX,processor4EventSourceContext) {
        processor4EventSourceContext.tagging.endServerPurePath();
    }

}
```

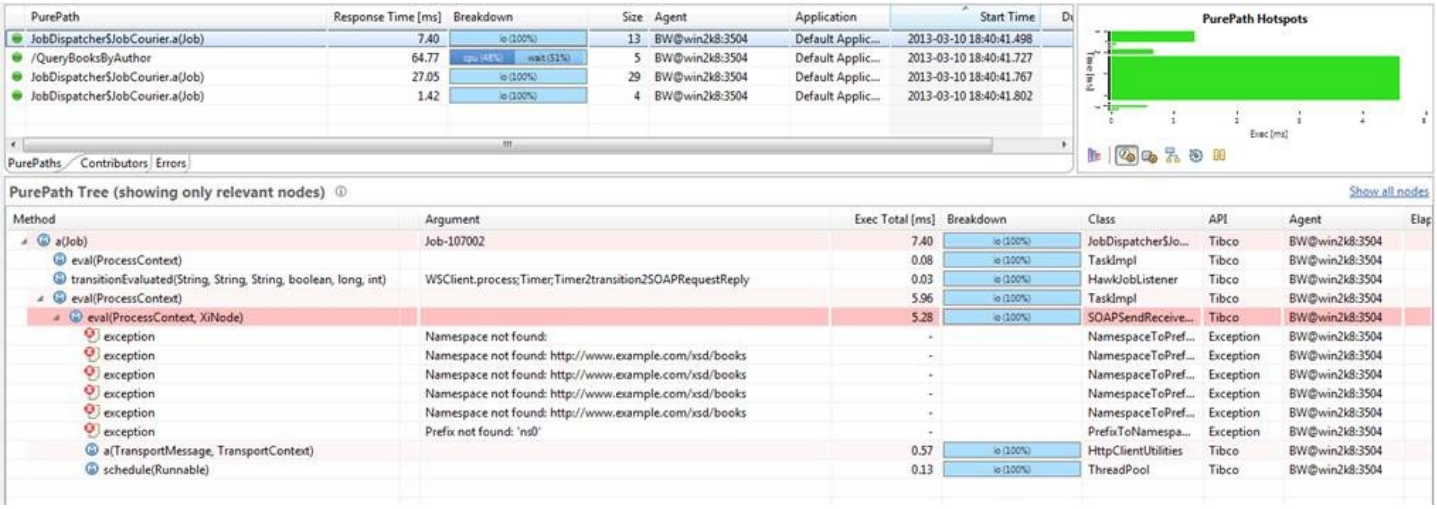
Download the AspjectJ file: bwjmstagging.aj

Example 3: connecting PurePaths for outbound SOAP requests

The following shows a BusinessWorks process that makes a SOAP request every 60 seconds to another process detailed in Example 1 above:

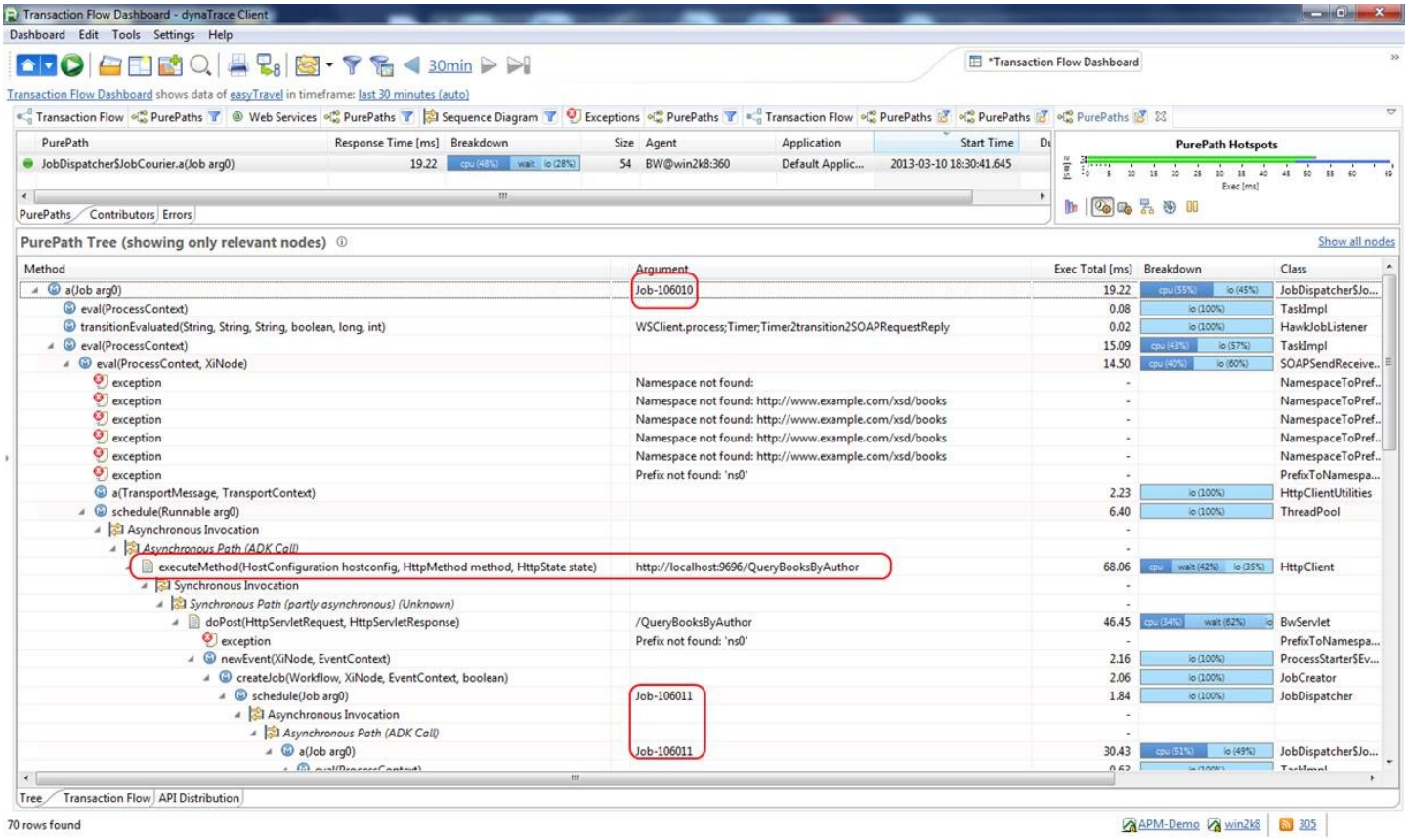


This screenshot shows this in AppMon. With some custom sensors, and four separate PurePaths.



These also display in this AppMon Session File bw-soap-ootb.dts

Using AspectJ, you can use these fixes, which would give two PurePaths. But an additional fix is needed to connect the outbound HTTP request that is part of the SoapRequestReply action.



This is also shown in this AppMon Session file bw-soap-aspectj.dts

The outbound executeMethod is now connected to Job-106010. When the call happens to the SOAP service (Job-106011) it is also connected in the PurePath.

Here's what the AspectJ code looks like:

```
privileged aspect BwHttpTagging {
    // Add a field to hold the Tagging object
    public Tagging ThreadPool.tagging = DynaTraceADKFactory.createTagging();
    public Tagging JakartaHttpTransportDriver.RequestExecutor.tagging = DynaTraceADKFactory
            .createTagging();

    // Add a field to store the Tag
    public String JakartaHttpTransportDriver.RequestExecutor.dtTag;

    // Add a static initializer to both classes to make sure the ADK is initialized
    before() : staticinitialization(ThreadPool) {
        DynaTraceADKFactory.initialize();
    }

    before() : staticinitialization(JakartaHttpTransportDriver.RequestExecutor) {
        DynaTraceADKFactory.initialize();
    }

    pointcut schedule(Runnable runnable, ThreadPool threadPool) : execution (public void schedule(Runnable)) && args(runnable) && target(threadPool) && within (ThreadPool);

    before(Runnable runnable, ThreadPool threadPool) : schedule(runnable, threadPool) {
        try {
            JakartaHttpTransportDriver.RequestExecutor requestExecutor = (JakartaHttpTransportDriver.RequestExecutor) runnable;
            String tag = requestExecutor.tagging.getTagAsString();
            requestExecutor.dtTag = tag;
            System.out.println("+++JDF+++ Adding HTTP PP tag : " + tag);
            requestExecutor.tagging.linkClientPurePath(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    pointcut run(JakartaHttpTransportDriver.RequestExecutor requestExecutor) : execution (public void run()) && target(requestExecutor)
    && within (JakartaHttpTransportDriver.RequestExecutor);

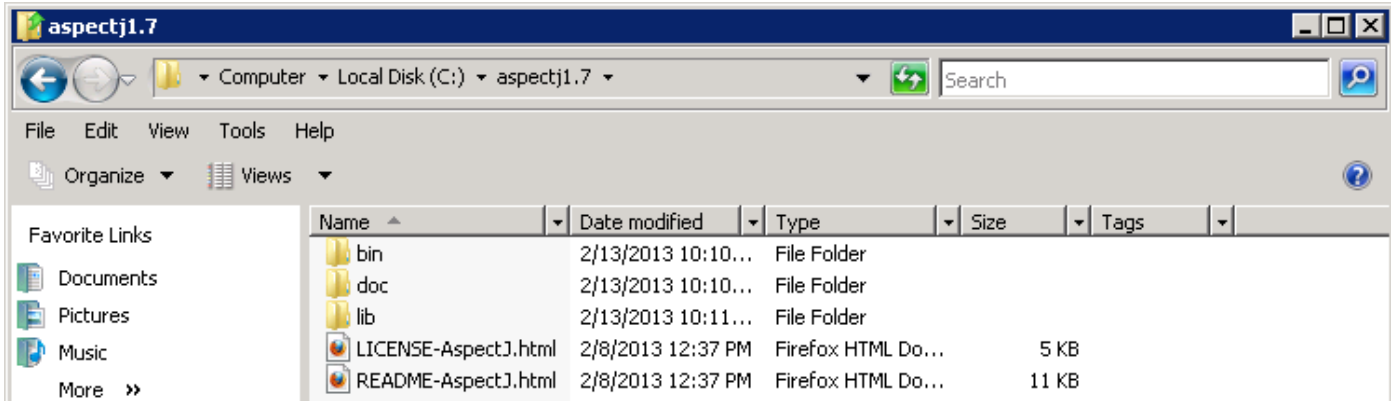
    before(JakartaHttpTransportDriver.RequestExecutor requestExecutor) : run(requestExecutor) {
        System.out.println("+++JDF+++ Linking HTTP PP tag: "
                + requestExecutor.dtTag);
        requestExecutor.tagging.setTagFromString(requestExecutor.dtTag);
        requestExecutor.tagging.startServerPurePath();
    }

    after(JakartaHttpTransportDriver.RequestExecutor requestExecutor) : run(requestExecutor) {
        requestExecutor.tagging.endServerPurePath();
    }
}
```

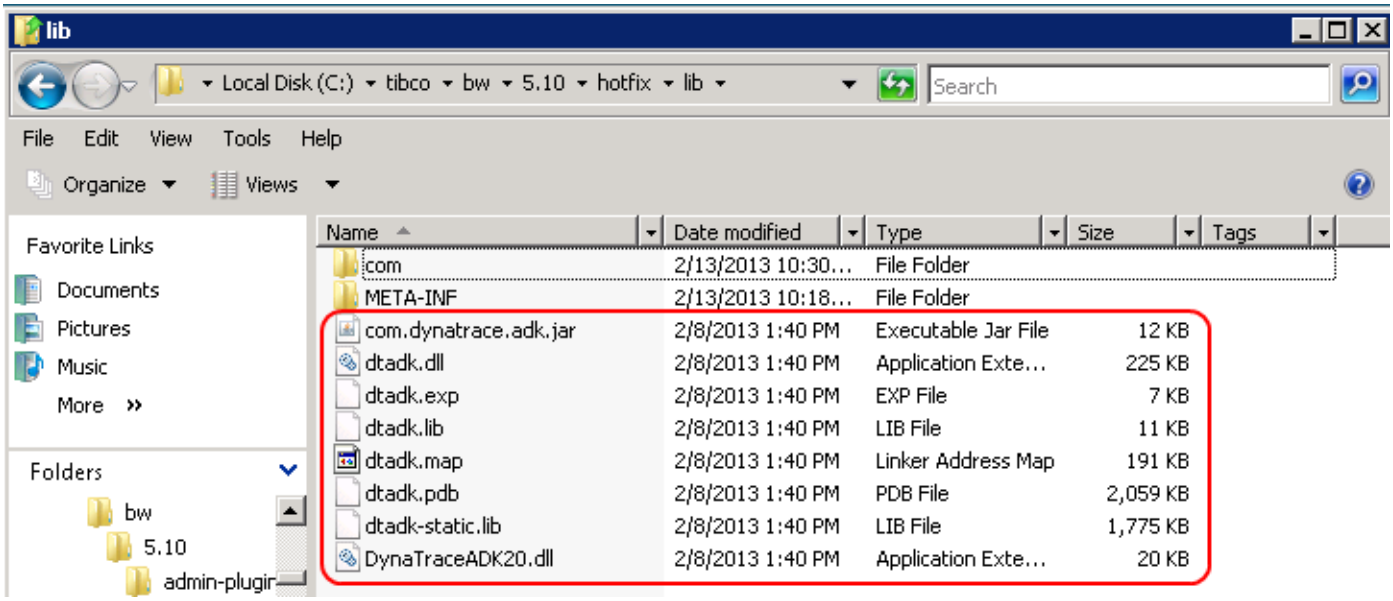
Download the AspjectJ file: bwhttptagging.aj

Instrument BusinessWorks with AspectJ and ADK

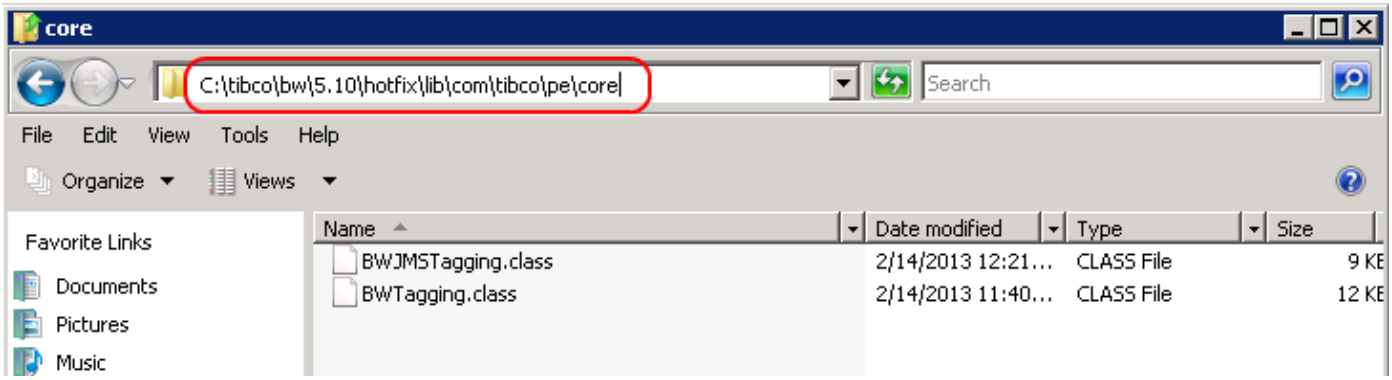
1. Install AspectJ to an easily accessible path on the machine running BusinessWorks. For example `c:\aspectj1.7`



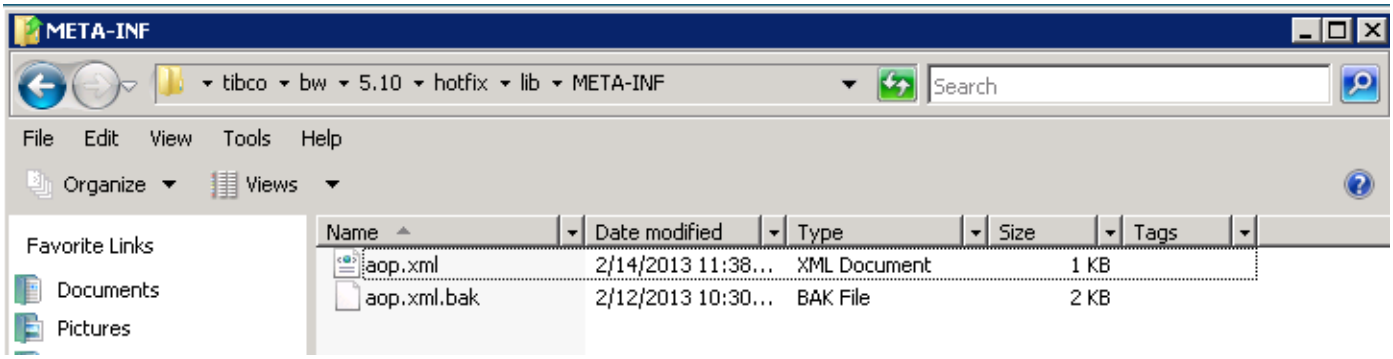
2. Install the ADK. You can copy the ADK files into `<BW_HOME>/hotfix/lib` so they would automatically be added to the BusinessWorks classpath.



3. Copy over the compiled aspect files.



4. Create an `aop.xml` configuration file for AspectJ in `<BW_HOME>/hotfix/lib/META-INF/aop.xml`



The contents of the file should look similar to the following.

```
<aspectj>

  <aspects>
    <!-- declare two existing aspects to the weaver -->
    <aspect name="com.tibco.pe.core.BWTagging"/>
    <aspect name="com.tibco.pe.core.BWJMSTagging"/>


    <!-- of the set of aspects declared to the weaver
         use aspects matching the type pattern "com..*" for weaving. -->
    <include within="com.tibco.pe.core..*"/>


  </aspects>

  <weaver options="-verbose -showWeaveInfo">
                <include within="com.tibco.pe.core..*"/>
                <include within="com.tibco.plugin.share.jms..*"/>
  </weaver>

</aspectj>
```

Download the `aop.xml` file: aop.xml

As an alternative, you can also build a `.jar` file of the Aspects in Step 3 and the `aop.xml` in Step 4 and place that in `<BW_HOME>/hotfix/lib` instead.

5. Add the AspectJ load time weaver to the BusinessWorks' `.tra` file to the same location where you add the dT agent, for example `<BW_HOME>/bin/bwengine.tra`.

The following is an example added the following at the bottom of the BusinessWorks' `.tra` file.

```
java.extended.properties=-javaagent:c:/aspectj1.7/lib/aspectjweaver.jar -agentpath:c:/tib
```

1. Restart the bwengine.

Infrastructure monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Host health monitoring

Infrastructure monitoring walkthroughs

Monitors

Process health monitoring

Host Monitoring Agent

Infrastructure monitoring dashboards

Incidents and alerting

Operations web view

Watch and learn

Monitoring infrastructure health in AppMon Client

Monitoring infrastructure health in AppMon Web

AppMon provides full-blown infrastructure monitoring in a global context, scaling to thousands of hosts across all supported platforms. Detailed infrastructure monitoring per System Profile is also available in the AppMon Web.

Every type of AppMon Agent[1], in addition to the standalone Host Monitoring Agent, provides host metrics for CPU, memory, network, and disk. Automatic health indicators are calculated to warn about host-specific problems that may also affect your application or specific Business Transactions.

Agent-based monitoring of the key host metrics is supplemented by plug-ins for other OS and VM metrics (performance counters), SNMP, etc., that can periodically pull additional data from different types of sources. See Monitors for more info.

[1] exceptions apply. The Node.js Agent e.g. does not collect monitoring data. The Web Server master Agent only does so when a slave Agent is connected.

Process health monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Installation and configuration

Process types

Process Health incidents

Special scenarios

Related pages

Monitors

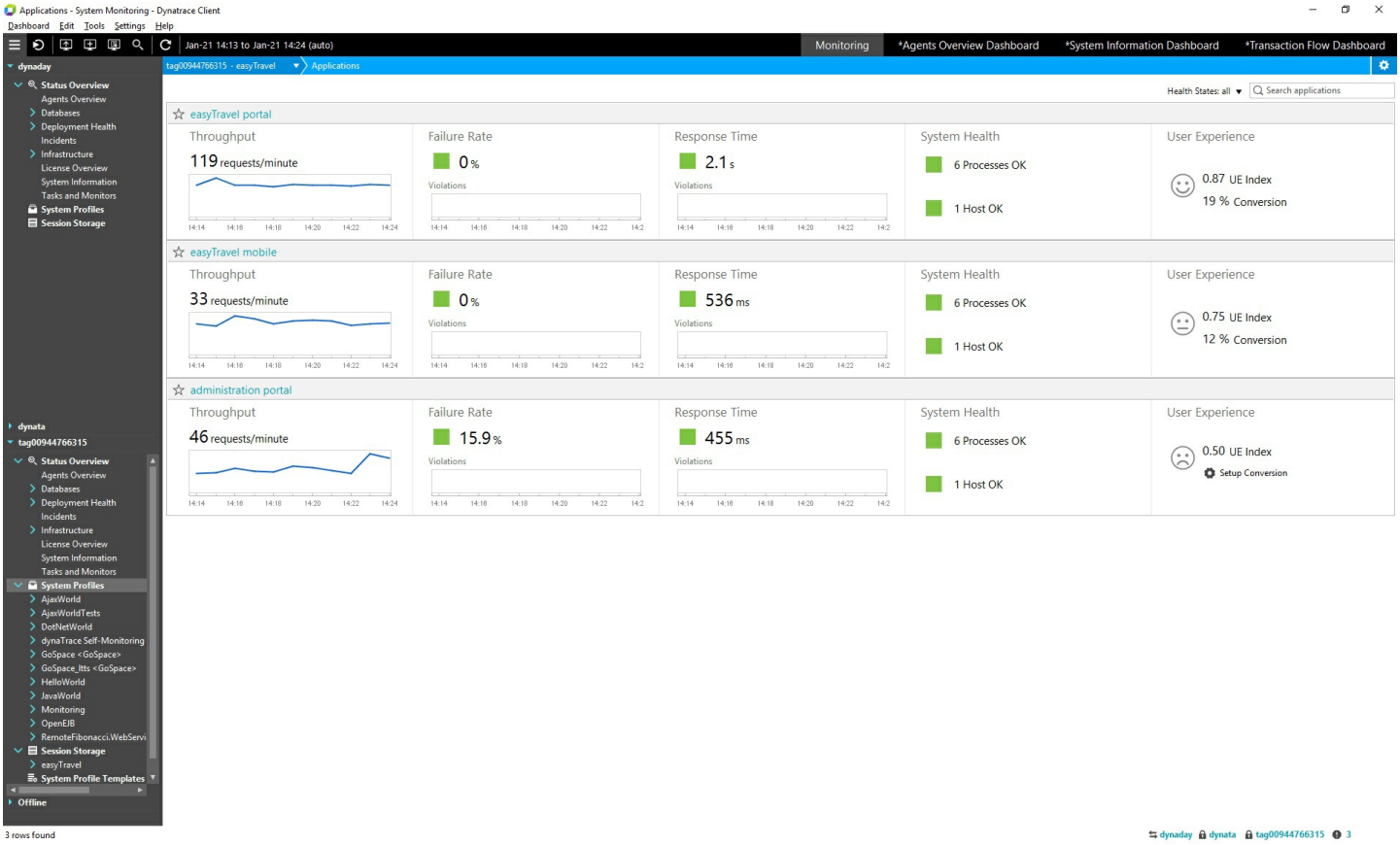
Host health monitoring

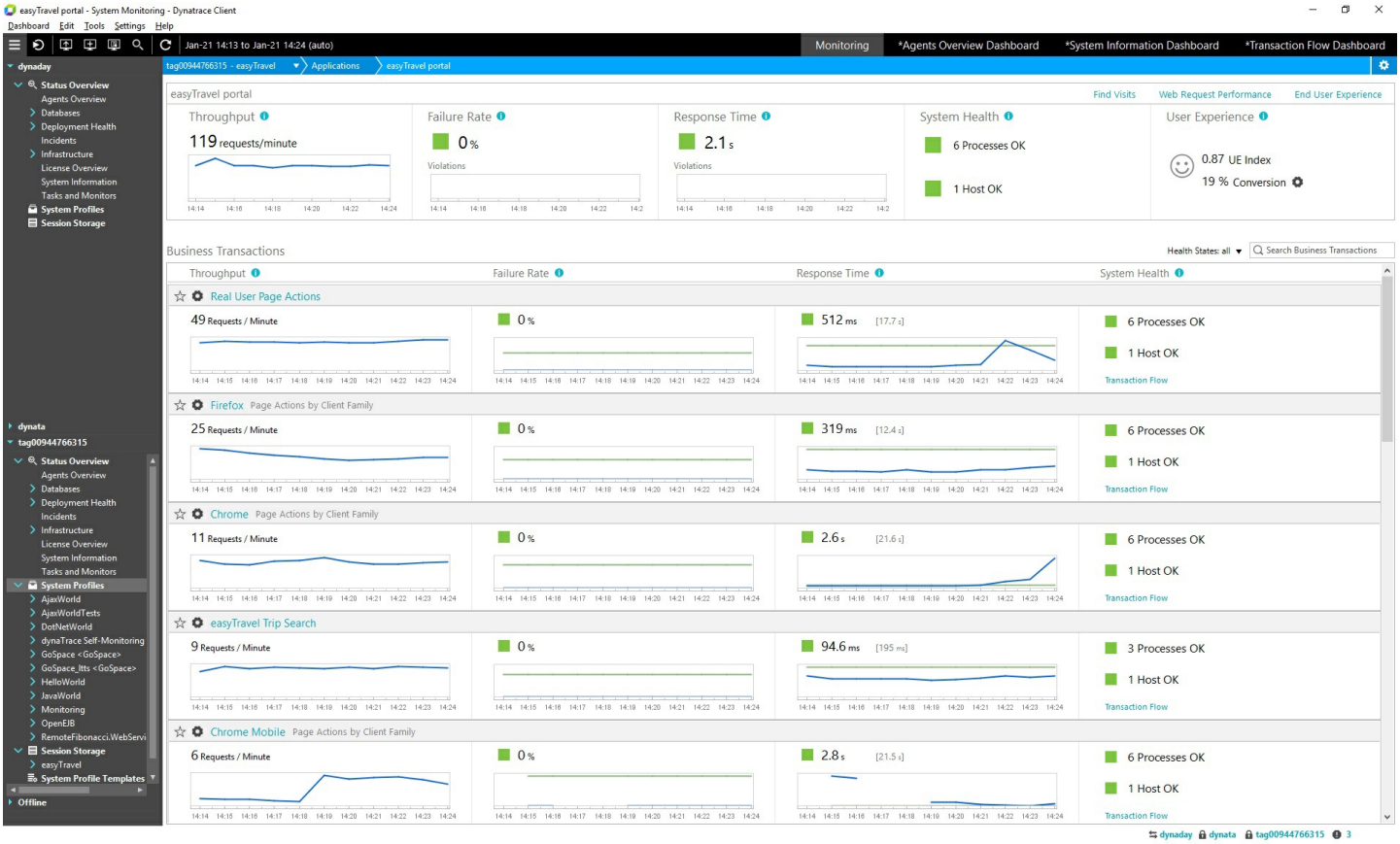
Overview

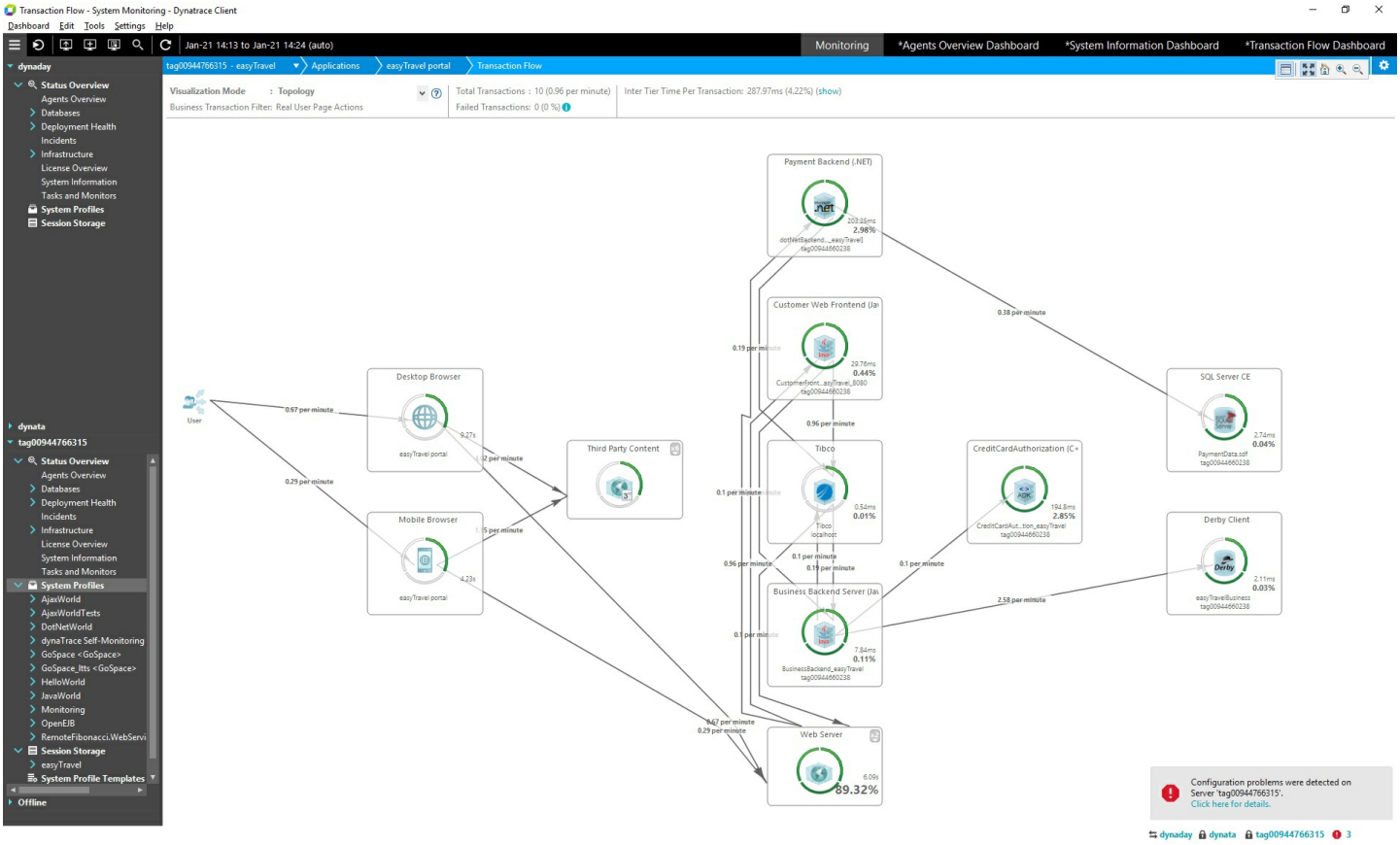
To narrow down the causes of problems and bottlenecks, know the states of the processes that run on your machines. For more information, see Infrastructure Monitoring Walkthroughs and Host Health Monitoring.

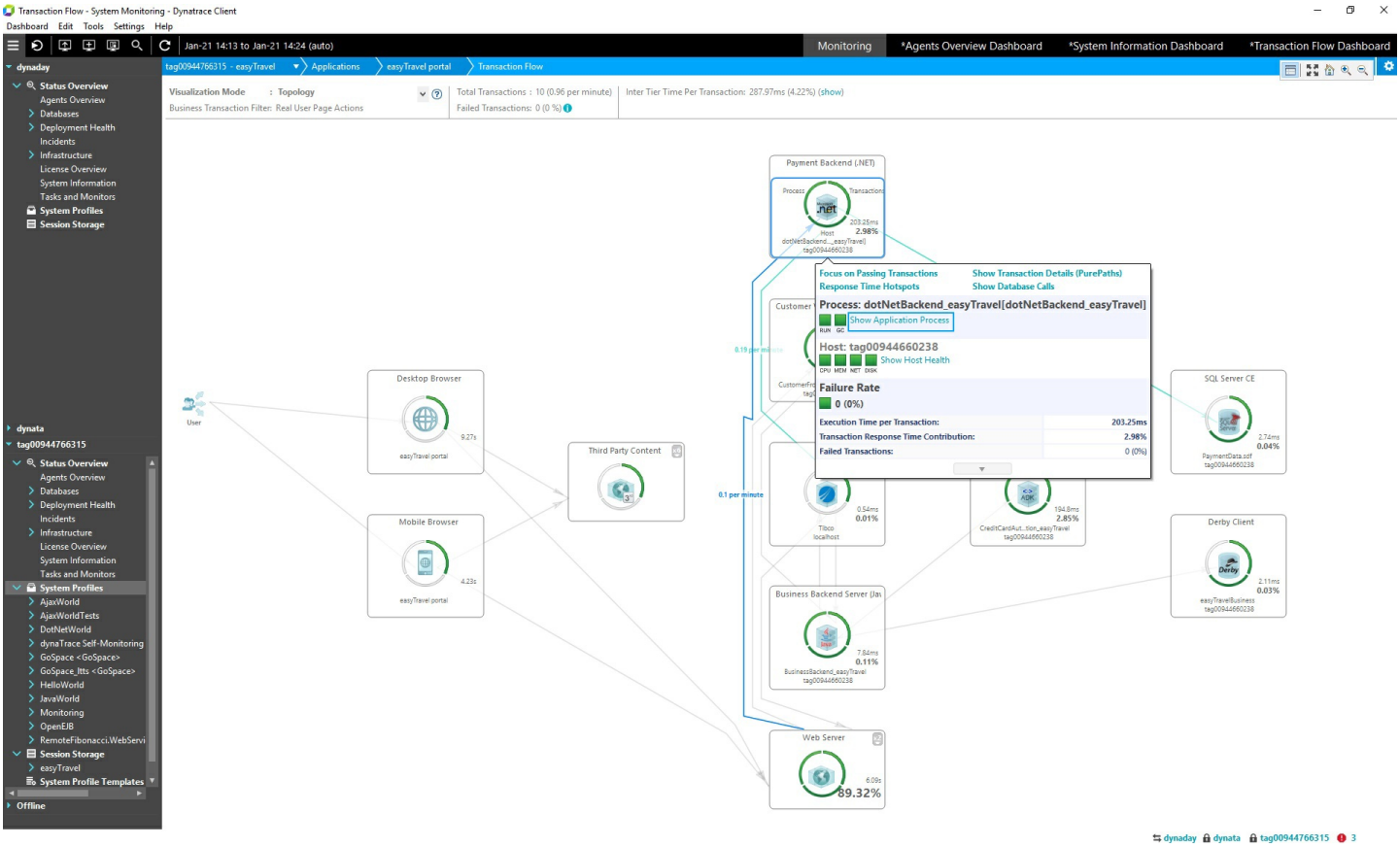
Each running Agent collects important information about the instrumented process such as used memory, GC time, CPU usage (percentage of the CPU capacity currently used by the monitored process), thread count, and transactions that pass through this node.

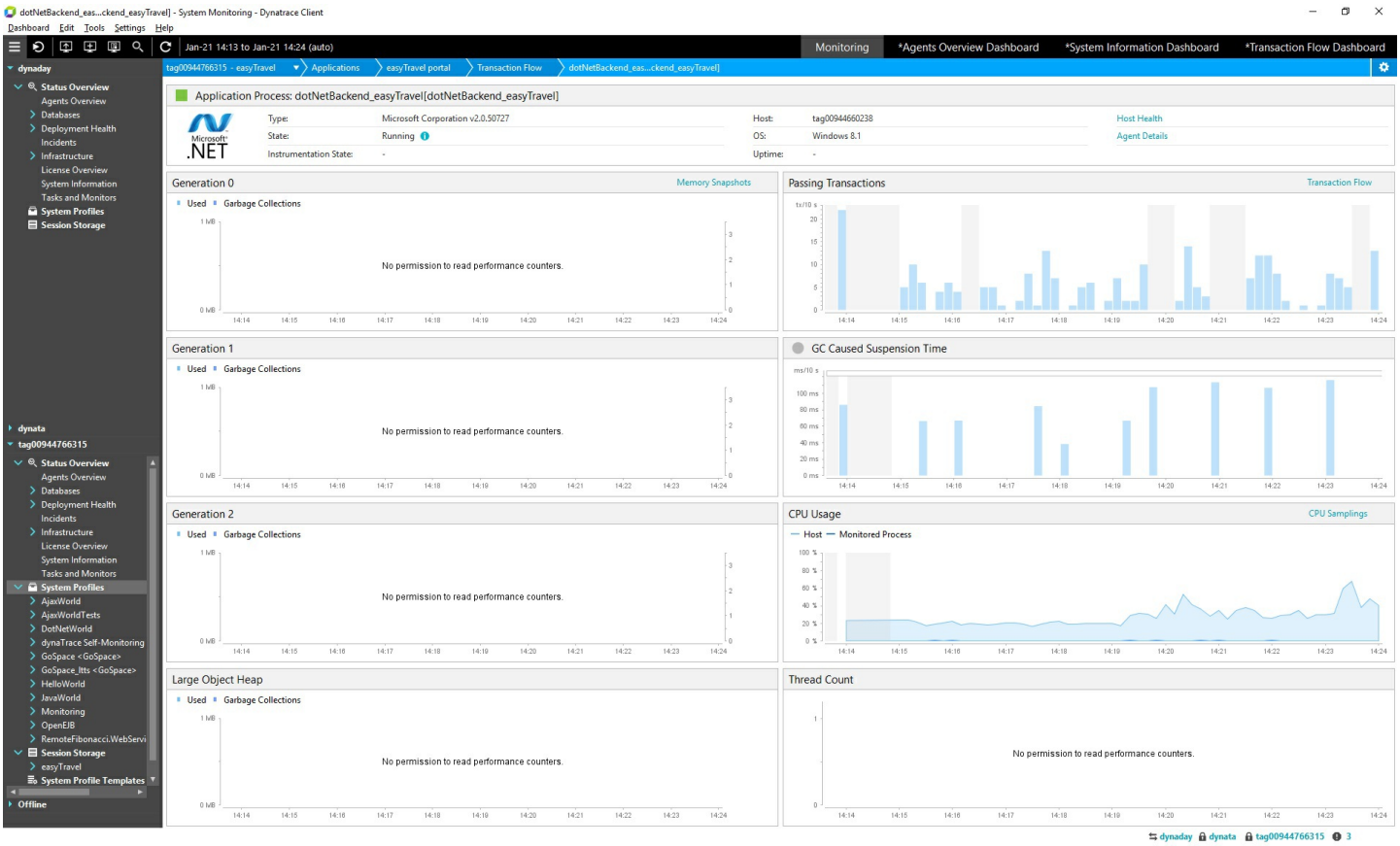
- A process is considered healthy (green label) if it spends less than 15% of its execution time in Garbage Collection (GC). GC naturally doesn't apply to native, just run state.
- The label is red if GC is above threshold or the process was terminated unexpectedly.
- Gray label: Information about the known process is unavailable, e.g. its Agent or Collector is disconnected.


Applications - System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
Jan-21 14:13 to Jan-21 14:24 (auto)
Monitoring
*Agents Overview Dashboard
*System Information Dashboard
*Transaction Flow Dashboard
tag00944766315 - easyTravel
Applications
Health States: all
Search applications
easyTravel portal
Throughput
119 requests/minute
Failure Rate
0 %
Violations
Response Time
2.1 s
System Health
6 Processes OK
1 Host OK
User Experience
0.87 UE Index
19 % Conversion
easyTravel mobile
33 requests/minute
0 %
536 ms
0.75 UE Index
12 % Conversion
administration portal
46 requests/minute
15.9 %
455 ms
0.50 UE Index
Setup Conversion
3 rows found


easyTravel portal - System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
Jan-21 14:13 to Jan-21 14:24 (auto)
Monitoring
*Agents Overview Dashboard
*System Information Dashboard
*Transaction Flow Dashboard
tag00944766315 - easyTravel
Applications
easyTravel portal
Find Visits
Web Request Performance
End User Experience
Throughput
119 requests/minute
Failure Rate
0 %
Response Time
2.1 s
System Health
6 Processes OK
1 Host OK
User Experience
0.87 UE Index
19 % Conversion
Business Transactions
Health States: all
Search Business Transactions
Real User Page Actions
49 Requests / Minute
512 ms
Transaction Flow
Firefox Page Actions by Client Family
25 Requests / Minute
319 ms
Chrome Page Actions by Client Family
11 Requests / Minute
2.6 s
easyTravel Trip Search
9 Requests / Minute
94.6 ms
3 Processes OK
Chrome Mobile Page Actions by Client Family
6 Requests / Minute
2.8 s


Transaction Flow - System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
Jan-21 14:13 to Jan-21 14:24 (auto)
Monitoring
*Agents Overview Dashboard
*System Information Dashboard
*Transaction Flow Dashboard
tag00944766315 - easyTravel
Applications
easyTravel portal
Transaction Flow
Visualization Mode : Topology
Business Transaction Filter: Real User Page Actions
Total Transactions : 10 (0.96 per minute)
Failed Transactions: 0 (0 %)
Inter Tier Time Per Transaction: 287.97ms (4.22%) (show)
Payment Backend (.NET)
Customer Web Frontend (Ja
Desktop Browser
Mobile Browser
Third Party Content
Tibco
Business Backend Server (Ja
Web Server
SQL Server CE
CreditCardAuthorization (C+
Derby Client
89.32%
Configuration problems were detected on Server 'tag00944766315'.
Click here for details.


Transaction Flow - System Monitoring - Dynatrace Client
Dashboard Edit Tools Settings Help
Jan-21 14:13 to Jan-21 14:24 (auto)
Monitoring
*Agents Overview Dashboard
*System Information Dashboard
*Transaction Flow Dashboard
Payment Backend (.NET)
Focus on Passing Transactions
Show Transaction Details (PurePaths)
Response Time Hotspots
Show Database Calls
Process: dotNetBackend_easyTravel[dotNetBackend_easyTravel]
Show Application Process
Host: tag00944660238
Show Host Health
Failure Rate
0 (0%)
Execution Time per Transaction: 203.25ms
Transaction Response Time Contribution: 2.98%
Failed Transactions: 0 (0%)
Desktop Browser
Mobile Browser
Third Party Content
Tibco
Business Backend Server (Ja
Web Server
SQL Server CE
Derby Client
89.32%



Installation and configuration

Most Agents automatically deliver process health information. In addition to full product install files, there are Agent-only files. On Windows, the Agents are registered as a service. On Linux and UNIX, the .jar-installer unpacks a `dthostagent` binary which has to be added to the startup process (for example, init.d script) manually.

On Windows Host Agent services must be in the Performance Monitor Users group to deliver process health information.

Process types

Basic process types

	Java	.NET	Native
Icon	1		

Memory	✓ 2	✓ 3	✓
Passing Transactions	✓	✓	✓
GC Time	✓	✓	
CPU Usage	✓	✓	✓
Thread Count	✓	✓	✓

[1] Generic Java icon is overruled if some special JVM Vendors or Application Servers are detected.

2up> Young Generation and Old Generation are shown (names may differ depending on the JVM vendor).

[3] Generation 0, Generation 1, Generation 2 and Large Object Heap are shown.

JVM vendors

Vendor Name	Icon
Oracle	
IBM	
HP-UX	

Application servers

Detected application servers may overrule the generic icons of the Basic Process Types.

Application Servers detected by AppMon:

Application Server Name	Icon
Apache Tomcat	

JBoss	
Microsoft Sharepoint	
Oracle WebLogic	
IBM WebSphere	

Web servers

Web servers are only shown in the Host Health Monitoring dashlet. Only name, version, and OS are available.

Web servers detected by AppMon:

Webserver Name	Icon
Apache	
Microsoft IIS	

Others

For native z/OS processes, the generic icon is overruled by .

Process Health incidents

The table shows the default values when process health incidents are triggered, and how to change them in `<DT_HOME>/dtserver.ini` with `-D<property>=<value>`.

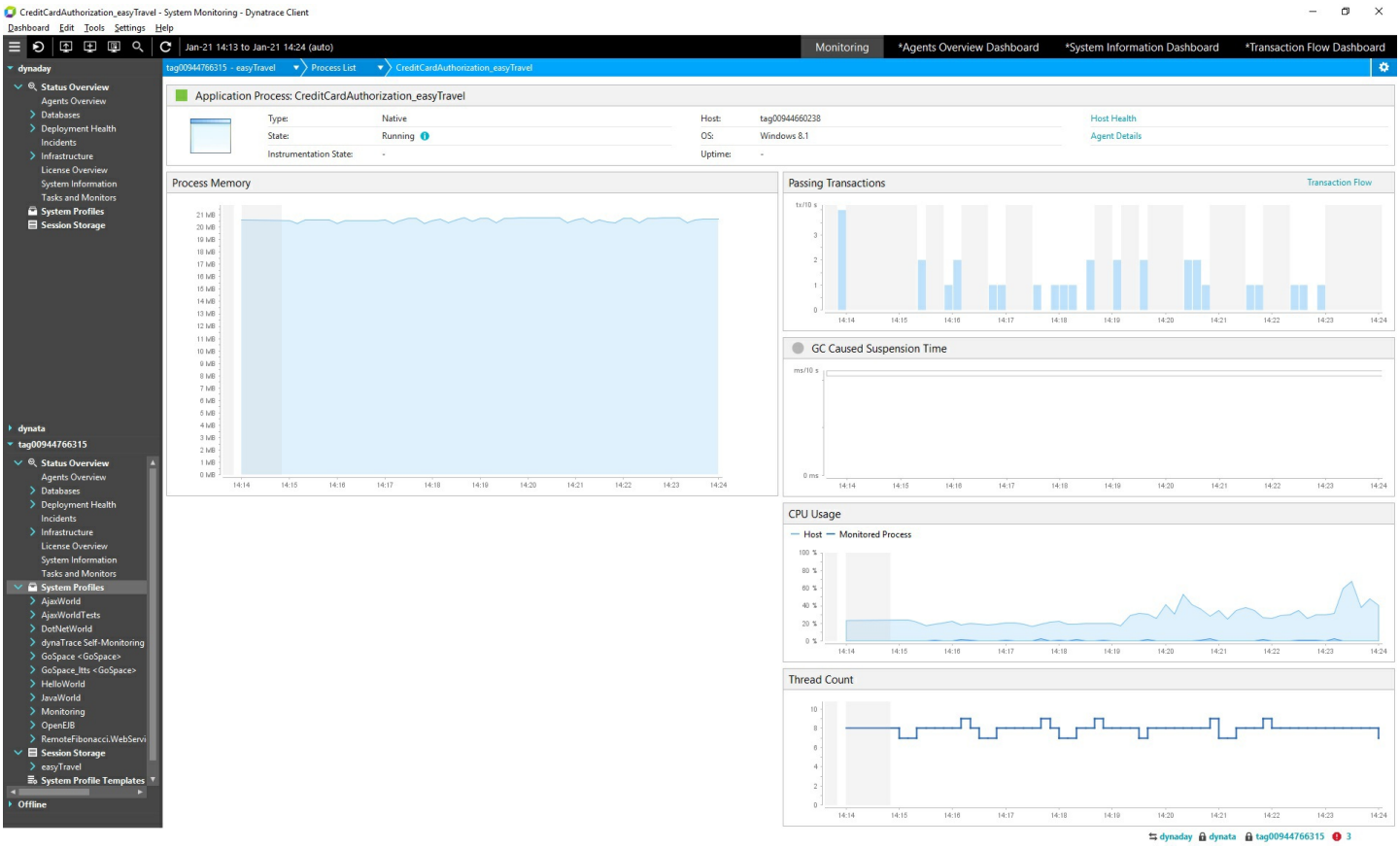
Property	Value	Since	Description

com.dynatrace.diagnostics.maxRelativeGCActivity	int (default = 15) %	4.2	Set custom health threshold for max relative GC activity of a process (percentage of wall clock time)
com.dynatrace.diagnostics.healthWatchPeriodGc	int (default = 5) minutes	4.2	Set custom process health threshold for the watch period
com.dynatrace.diagnostics.healthWatchPeriodViolationThresholdGc	int (default = 4) minutes	4.2	Set custom process health threshold for the watch period threshold to be considered unhealthy

Special scenarios

Native process

For native processes, no GC Time is available. Therefore, the health of this process only depends on its running state (green if the process is running, gray if the process is not running and red if the process was shutdown unexpectedly).
In contrast to other process types, only the total process memory displays.



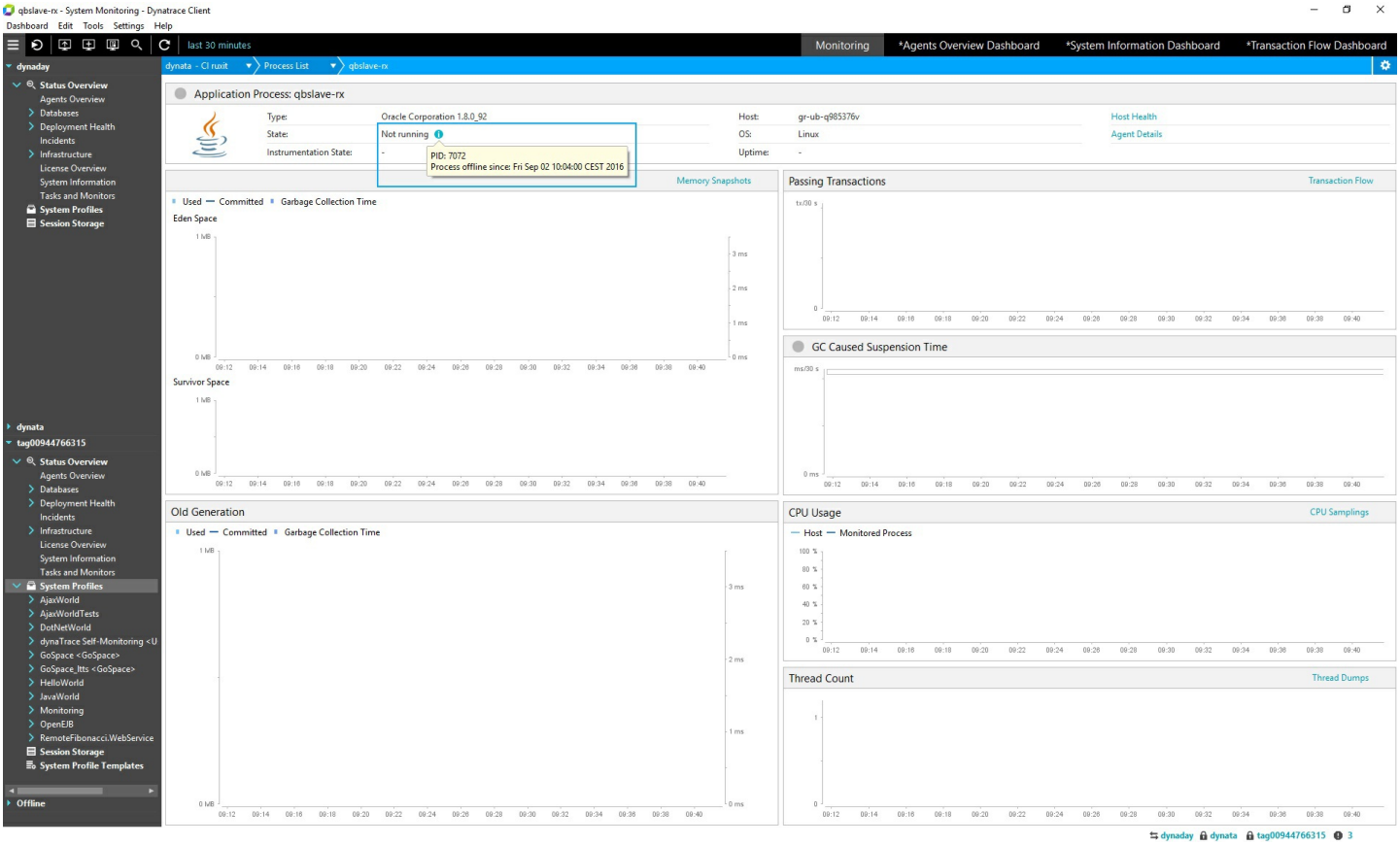
Native process

Offline process

A gray health indicator means that no health state can be determined for this process. The reason for this is shown in the state entry of the overview table. Hover over the small



next to the state. A tooltip shows the process ID and how long the process has been offline.



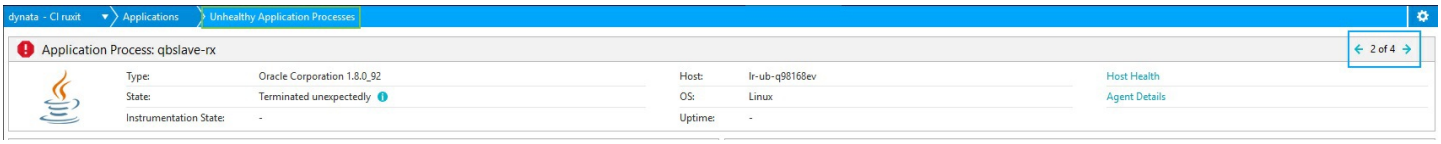
Offline process (gray health indicator)

Several unhealthy processes

If more than one process is unhealthy, use the Application Overview to drill down to the **Process Health** dashlet for all affected processes. Hover over the red icon and click the **Application Processes** link to open the **Unhealthy Application Processes** window, which contains the **Process Health** dashlet and shows information about the first unhealthy process. Click the blue arrows on the top right of the table to switch to the other unhealthy processes.

> If the unhealthy state of the processes in the Application Overview is caused by their host, the **Application Processes** link does not display in the tooltip. Instead you will see a **Host** link that directly drills down to the Host Health Monitoring dashlet.


quickbuild-rx.lab.dynatrace.org
Find Visits
Web Request Performance
End User Experience
Throughput
204 requests/minute
Failure Rate
0 %
Violations
Response Time
24.3 ms
Violations
System Health
4 of 14 Processes affected
4 Processes with health problems detected.
Application Processes
Unhealthy Application Processes
Web Server
User Experience
- UE Index
Setup Conversion


dynata - CI ruxit
Applications
Unhealthy Application Processes
Application Process: qbslave-rx
2 of 4
Type:
Oracle Corporation 1.8.0_92
State:
Terminated unexpectedly
Instrumentation State:
-
Host:
lr-ub-q98168ev
OS:
Linux
Uptime:
-
Host Health
Agent Details

Host health monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Data-center-aware infrastructure monitoring
Installation and configuration
What is monitored on what platform?
Thresholds, alerts and inclusions or exclusions for host metrics
User specific monitors

Related pages

Infrastructure
Infrastructure monitoring dashboards
Infrastructure monitoring
Monitors
Process health monitoring

It is crucial to know the state of your machines. AppMon has been providing zero config host health monitoring for many years. Every host with a running Agent[1] or installed standalone Host (Monitoring) Agent monitors the host system automatically.
The Agent collects information about CPU, memory, disk and network interface utilization to determine the host system´s health state.

Data-center-aware infrastructure monitoring

For an end-to-end overview please have a look at:

- Infrastructure - How to configure a scalable monitoring with sites, hostgroups, labels, alerts and exceptions therefore
- Infrastructure Monitoring Dashboards - How to check the state of your data-centers
- Infrastructure Monitoring - How to utilize different views to be in the picture

The Start Center



now takes you to the System Profile related Topology view of the infrastructure for your applications.

Installation and configuration

Every installed Agent Group automatically delivers Host Health information.[1]
In addition to full product install files, there are Agent-only install files. See the respective Agent configuration pages for details.
See Host Monitoring Agent for its installation and configuration if you have no other Agent on that machine.

What is monitored on what platform?

Category	Measure Name	Description	Windows	Linux	Solaris	AIX	HP-UX	zOS[5]
Processor	CPU User Time		✓	✓	✓	✓	✓	–
	CPU System Time		✓	✓	✓	✓	✓	–
	CPU Idle		✓	✓	✓	✓	✓	–
	CPU I/O			✓	✓	✓	✓	–
	CPU Load	“Unix Load Average” (exponential moving) average of running processes per minute		✓	✓	✓	✓	–
Memory	Memory Usable		✓	✓	✓	✓	✓	–

	Memory Used		✓	✓	✓	✓	✓	–
	Page Faults[2]	Hard page faults only	✓	✓	✓	✓	✓	–
Disk	Disk Storage Utilization[3]	Free disk space and percentage of available space	✓	✓	✓	✓	✓	–
	I/O Statistics[3]	Read/write statistics of physical disks	✓	✓	–	–	–	–
Network	Network Inbound Utilization[4]	Percentage of used inbound bandwidth	✓	✓	✓	✓	✓	–
	Network Outbound Utilization[4]	Percentage of used outbound bandwidth	✓	✓	✓	✓	✓	–

[1]The Web Server Master Agent service does not deliver host monitoring data on its own. There must be at least one Web Server Slave Agent loaded on the web server or a Host Monitoring Agent. The Node.js Agent does so far not deliver host monitoring data. Start the Host Monitoring Agent service if it's the only Agent on the machine.

[2] Requires disk access permissions to the disks that contain the swap partition or swap file.

[3] Requires read and list files permissions to the disks that are to be monitored. Mapped network drives are excluded by default from Disk Storage Utilization.

*[4] On *NIX systems a kernel with version greater or equal to 2.6.33 is required otherwise only the transfer rates are displayed.*

[5] zOS is not supported currently.

Remark on CPU core count on AIX LPAR

CPU core count is only determined during startup of agent.

CPU core count = "Maximum Virtual CPUs" * SMT-Factor according to "lparstat -i".

OS virtualization and container monitoring

VMware, Xen and Hyper-V are detected and supported. For more information, see VMware.

VirtualBox is detected, but not supported.

Some operating system level virtualization techniques on various UNIX / Linux based systems like Solaris Zones, OpenVZ, BSD Jails and VServer are not supported. Agents running within a container report metrics from the operating system and the instrumented applications, but not from the container management (tools).

Thresholds, alerts and inclusions or exclusions for host metrics

Threshold definition, alert configuration and ex / inclusions got a GUI in **Settings menu > Dynatrace Server… > Infrastructure vertical tab**. See Infrastructure.

User specific monitors

For more information, see Monitors, Windows Performance Monitor and Unix System Monitoring.

Monitors

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Configuration

Monitor types

Related pages

Infrastructure monitoring - zero config with agents

SNMP monitor

Unix system monitoring

URL monitor

VMware

Web transaction monitor

Windows Performance Monitor

Watch and learn

Configuring monitors

Overview

A *Monitor* is scheduled for repetitive execution on a Collector to deliver additional performance metrics to the AppMon Server.
Typical examples are windows performance counters that are not specific to a process but host-wide, and SNMP data sources.

Data retrieved by Monitors are stored as Measures, which can then be combined and correlated to metrics captured from within the application. For example, transaction response times, number of exceptions and executed database statements.

You can monitor multiple hosts without creating a separate Monitor for each host. Instead, a master Monitor is configured to run on multiple hosts.

You must configure at least one target host for a Monitor. You can also add specific hostgroups or intersections of hostgroups as targets. Whenever a new host machine is added to a specific hostgroup that is a target of a Monitor, the added host is automatically monitored without any extra configuration. For more information on configuring hosts and hostgroups, see Infrastructure. For a walkthrough on monitoring multiple hosts, see Using Host and Hostgroups for Large System Monitoring.

Configuration

You can create a monitor through the System Profile or the **Tasks/Monitors** dashlet. In the System Profile, select the **Monitors** screen and click the **Create** button. In the dashlet, right-click to open the context menu and select **Create**.

When the **Monitor Type Chooser** is displayed, select the desired type.

The basic configuration is the same procedure for all Monitor types. For information on the built-in Monitors, see Monitor Types. You can add Monitors through Monitoring Plugins available in the Dynatrace Community.

On the **Schedule** tab, define the monitor schedule and select the execution target.

Select whether the monitor should be run manually or run on a schedule.

If you select **Run scheduled**:

- Select the interval from the first list. The selections range from **Every 10 Seconds** through **Every Hour**.

- Click **Add** to use a preconfigured schedule or define a new schedule, or click **Edit** to revise the schedule. For further information on schedules, see System Profile - Schedules.
- Optionally select **Business Hours** from the second list. For further information on Business Hours, see System Profile - Business Hours.

> **Warning**
>
> Monitoring can cause significant load on a Collector. It is recommended that you use a separate Collector for monitoring of multiple hosts.

Monitor types

These Monitor types are provided by default.

Host monitor

See Host Health Monitoring.

SNMP monitor

See SNMP Monitor.

Unix monitor

See Unix System Monitoring.

URL monitor

See URL Monitor.

VMware compute resource performance monitor

This Monitor retrieves performance counters from a Compute Resource within the VMware infrastructure. It provides the following information:

Parameter Name	Description
Compute Resource	Name of the VMware Compute Resource from which to retrieve performance counters.
Username	The username for the VMware host account.
Password	The password for the VMware host account.
VMware webservice URL	The URL to access the VIM webservice on the VMware host.

VMware host system performance monitor

This Monitor retrieves performance counters from a host system within the VMware infrastructure. It provides the following information:

Parameter Name	Description
Username	The username for the VMware host account.
Password	The password for the VMware host account.
VMware webservice URL	The URL to access the VIM web service on the VMware host.

VMware resource pool performance monitor

This Monitor retrieves performance counters from a resource pool within the VMware infrastructure. It provides the following information:

Parameter Name	Description

Resource Pool	Name of the VMware Resource Pool from which to retrieve perfomance counters.
Username	The username for the VMware host account.
Password	The password for the VMware host account.
VMware webservice URL	The URL to access the VIM web service on the VMware host.

VMware virtual machine performance monitor

This Monitor retrieves performance counters from a Virtual Machine within the VMware infrastructure.

Parameter Name	Description
Virtual Machine	Name of the VMware Virtual Machine from which to retrieve performance counters.
Username	The username for the VMware host account.
Password	The password for the VMware host account.
VMware webservice URL	The URL to access the VIM web service on the VMware host.
Compute Virtual CPU on Host basis	Virtual CPU usage is normally calculated for each virtual CPU, which is based on one host core. This can be confusing. Enable this option to calculate the virtual CPU usage based on the total CPU of the host system.

Web transaction monitor

See Web Transaction Monitor.

Windows Performance Monitor

See Windows Performance Monitor.

Infrastructure monitoring - zero config with agents

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Dashboards

Plugin vs. agent-based measures

Overview

As opposed to plug-in-based monitoring: With Agent-based (zero-configuration) infrastructure monitoring, many Measures are available out of the box. In addition to simplifying the configuration process (you don't need to set up passwords or permissions, especially for Windows), retrieving measurements from hosts now has less impact on the system's performance.

The target host must have at least one active and connected AppMon Agent installed.

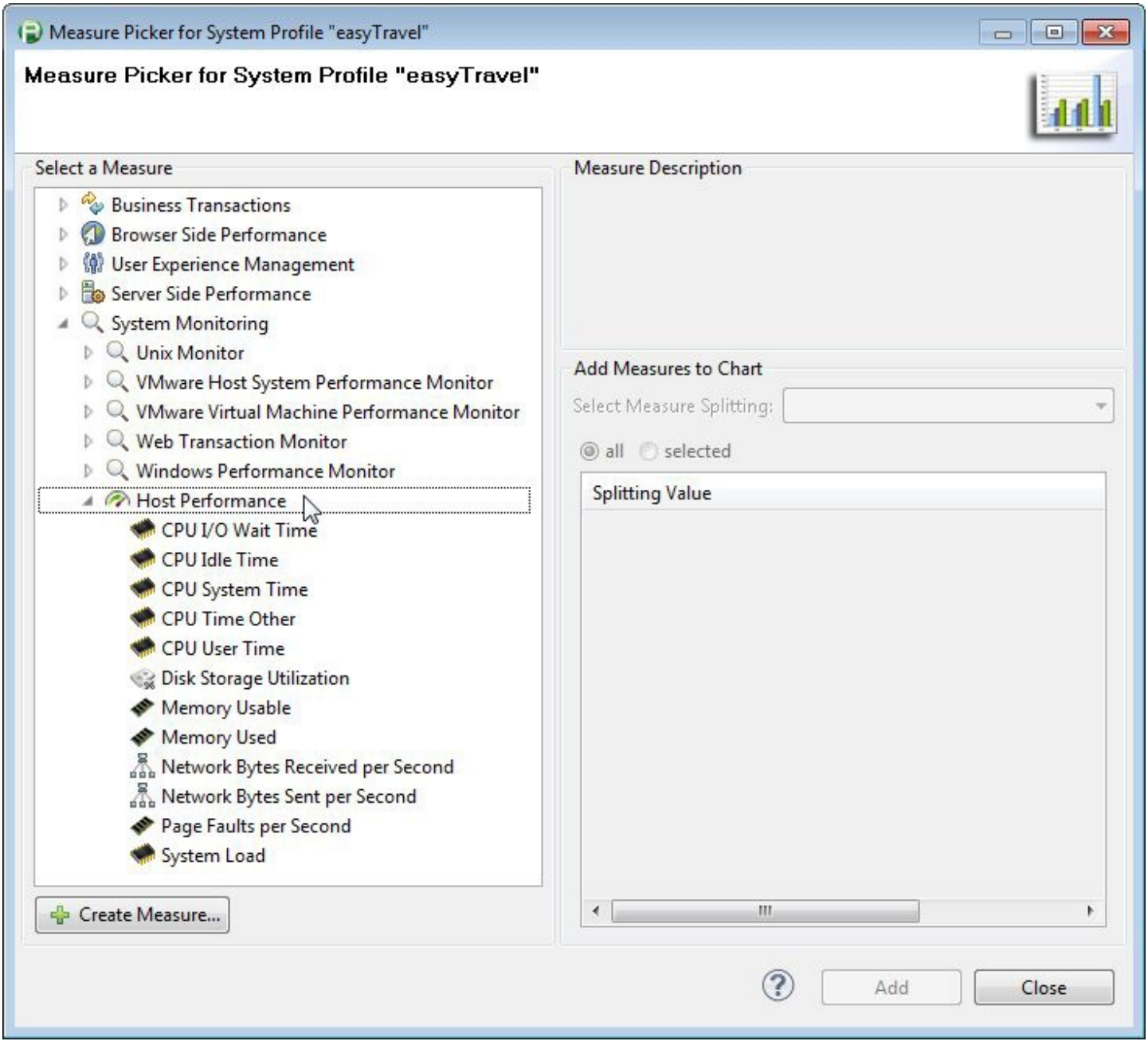
If you migrate AppMon from a release earlier than 4.2, you should review existing configurations of Windows Performance Monitor and Unix Monitor plugins, and remove them if the Measures that are collected via the Agent are sufficient. See Plugin vs. Agent-based Measures for a table of equivalent Measures.

If you need to install and configure a Host Monitoring Agent because there is no other active Agent on the machine, refer to Host Monitoring Agent for details.

Dashboards

Besides supporting pre-configured Monitoring in Application Overview Dashboard and dashboards, Agent-based Measures can be used to build custom charts and dashboards. Use the **Measure Picker** dialog box, as shown below.

1. In the Cockpits pane, expand **System Profiles > <profile> > Charting** and double-click **Custom**.
2. Double-click the **Add Series** button to open the Measure Picker dialog box.
3. Expand the tree as needed to display the desired Measure, then double-click the Measure to list it in the **Add Measures to Chart** area.
4. Click **Add** to add the Measure.
5. Repeat steps 3 and 4 as needed, then click **Close**.



Measure picker

Plugin vs. agent-based measures

It is recommended that you use Agent-based Measures whenever possible.

In the table, Agent-based Measures that are **not** subscribed by default are marked with an asterisk (*).

Windows Performance Monitor	UNIX Monitor	Agent-based Monitoring	Notes
CPU			
User Time	User Time	CPU User Time	
Idle Time	Idle Time	CPU Idle Time	
	System Time	CPU System Time	Also available on Windows. This Measure contains kernel time.
	I/O Wait Time	CPU I/O Wait Time	
Processor Time	Processor Time		
Processor Interrupts/sec			
Disk			
Custom Disk Free Space		Disk Storage Available *	This measure is automatically split per disk (Windows) or mount point (UNIX). It is recommended that you replace this Measure with Disk Storage Utilization, which is subscribed by default.
Free Diskspace	Free Diskspace	Disk Storage Usable *	This measure is automatically split per disk (Windows) or mount point (UNIX).
	Used Diskspace	Disk Storage Used *	This measure is automatically split per disk (Windows) or mount point (UNIX).
Disk Reads/sec	Disk Reads/sec		
Disk Writes/sec	Disk Writes/sec		
	Total Diskspace		

	Var Usage		
	Temp Usage		
	Root Usage		
Current Disk Queue Length			
LogicalDisk Free Percentage			
PhysicalDisk Read			
PhysicalDisk Write			
Memory			
Free Memory	Free Memory	Memory Usable	
	Free Swap	Swap Free *	
	Used Memory	Memory Used	
	Used Swap	Swap Used *	
	Memory Available	Memory Available *	
	Memory Usage	Memory Utilization *	
Pages/Sec		Page Faults per Second	The Agent-based Measure only counts hard faults.
Committed Bytes In Use			

	Total Memory		
	Total Swap		
	Swap Usage		
Pages Input	Pages Input		
Pages Output	Pages Ouput		
Network			
	Collisions		
Custom Network Bytes Total/sec	Bytes Total/sec		
Custom Network Bytes Received/sec	Received Bytes/sec	Network Bytes Received per Second	This measure is automatically split per network interface. Custom configuration is not necessary.
Packets Received Errors	Received Packets Error		
	Received Packets Dropped		
	Received Packets OK		
	Received Packets Total		
Custom Network Bytes Sent/sec	Transmitted Bytes/sec	Network Bytes Sent per Second	This measure is automatically split per network interface. Custom configuration is not necessary.
	Transmitted Packets Dropped		
	Transmitted Packets Error		

	Transmitted Packets OK		
	Transmitted Packets Total		
Network Output Queue Length			
Unix process			
	Blocked		
	Process Count		
	Run Queue, current		
	Run Queue, last 15 minutes		
	Run Queue, last 5 minutes		
	Run Queue, last minute		
Windows-specific measurements			
Process Count			
Process Memory Usage		Current Overall Memory Consumption	This measure is part of the Metric Group **Process Performance**.
Run Queue, current			

SNMP monitor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Configuration

Select host(s) that should be the target(s) of the monitor

Customizing the execution schedule

Centrally deploying a monitor to different global locations

Measures

Overview

The SNMP Monitor Plugin collects Metrics via Simple Network Management Protocol. SNMP is often used for network infrastructure equipment. Any hosts that run an SNMP Agent can be queried. All actions from the Monitor are read-only.

Configuration

After specifying a name and description for the new Monitor, configure the following monitor settings:

Setting name	Description
Agent Port	The Agent port from which to collect SNMP metrics.
SNMP Version	The SNMP version to use for queries. Supported versions are Version 1 and Community-Based SNMP Version 2 (SNMPv2c).
Community	The community string to use when querying the SNMP metrics. Public and private community strings are supported.

Query timeout	Timeout for SNMP queries, in milliseconds. The default value is 500.

Select host(s) that should be the target(s) of the monitor

A Monitor can be configured for multiple target hosts. Only one configuration is necessary; it is executed for all configured machines. This is especially useful in cluster environments that feature many similarly configured hosts.

Add **Single hosts**, **hostgroups**, or **sites** by clicking the **+** icon. Create a new host if the desired host is not shown. If you add a hostgroup or site, the Monitor will be run for all contained hosts.

Optionally, add an **intersection** of hostgroups or sites. The Monitor will be run for all hosts that belong to all selected groups or sites. If the configuration would resolve a single host more than once (if Host1 is also contained in Hostgroup1), the Monitor is only executed once per host.

You can also add target hosts via the Server REST interface. The required POST request is located in the **Tasks and Monitors** group. The request documentation is based on the Swagger specification and embedded to the request form.

For more information how to manage hosts and hostgroups, see Infrastructure.

Examples

Configured Hosts As Shown in Hosts Table	Monitor Will Be Run for…
Host1	Host1
Host1 Host2	Host1 and Host 1
Hostgroup 1 Host2	All hosts from Hostgroup 1 and also Host 2
Hostgroup 1 Hostgroup 2	All hosts from Hostgroup 1 and also Hostgroup 2
Hosts belonging to Hostgroup 1, Hostgroup 2 Host 1	All hosts that are **both** in Hostgroup 1 and Hostgroup 2 (an intersection of Hostgroup 1 and Hostgroup 2), plus Host 1

Customizing the execution schedule

Open the **Schedule** tab, then choose a predefined schedule or create your own schedule if none fits. Any schedules you create will be available for the whole System Profile.

Centrally deploying a monitor to different global locations

Open the **Schedule** tab, then select the desired AppMon Collector or AppMon Server from the **Execute on** list. Only currently connected AppMon Collectors are shown in this list. The selection appears as **Embedded Dynatrace Collector**.

> **Note**
>
> If an AppMon Collector that is connected through a slow network or a Monitor that contains large library files is run on a remote machine, it is possible, though rare, that the first execution fails with a Missing Binaries error. This error should disappear on the next scheduled run of the Monitor.

Measures

Configure the Measure subscription in the **Measures** tab of the **Schedule Monitor Editor**. You can create new measures from the SNMP Metric Group or create a new custom Metric Group via PDE.

Expand the links below for information about the preconfigured Measures.

RFC1213 measures

ID	Description	OID
iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0	The time in hundredths of a second since the network management portion of the system was last reinitialized.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.system.sysServices.0	A value that indicates the set of services that this entity primarily offers. The value is a sum. This sum initially takes the value zero. For each layer **L** , in the range 1 through 7, for which this node performs transactions, 2 raised to (L–1) is added to the sum. For example, a node that performs primarily routing functions would have a value of `4(2^(3-1))`. In contrast, a node that is a host offering application services would have a value of `72(2^(4-1) + 2^(7-1))`. In the context of the Internet suite of protocols, values should be calculated accordingly. Layer functionality: • 1 Physical (e.g., repeaters) • 2 Datalink/subnetwork (e.g., bridges) • 3 Internet (e.g., IP gateways) • 4 End-to-end (e.g., IP hosts) • 7 Applications (e.g., mail relays) For systems including OSI protocols, layers 5 and 6 may also be counted.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifNumber.0	The number of network interfaces present on this system, regardless of their current state.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifIndex.0	A unique value for each interface. Its value ranges between 1 and the value of `ifNumber`. The value for each interface must remain constant at least from one re-initialization of the entity's network management system to the next reinitialization.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifType.0	The type of interface, distinguished according to the physical/link protocol(s) immediately below the network layer in the protocol stack.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifMtu.0	The size of the largest datagram that can be sent/received on the interface, specified in octets. For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifSpeed.0	An estimate of the interface's current bandwidth in bits per second. For interfaces that do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifAdminStatus.0	The desired state of the interface. The testing(3) state indicates that no operational packets can be passed.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOperStatus.0	The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifLastChange.0	The value of `sysUpTime` at the time the interface entered its current operational state. If the current state was entered prior to the last re-initialization of the local network management subsystem, then this object contains a zero value.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifInOctets.0	The total number of octets received on the interface, including framing characters.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifInUcastPkts.0	The number of subnetwork-unicast packets delivered to a higher-layer protocol.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifInNUcastPkts.0	The number of non-unicast (i.e., subnetwork-broadcast or subnetwork-multicast) packets delivered to a higher-layer protocol.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifInDiscards.0	The number of inbound packets that were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifInErrors.0	The number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifInUnknownProtos.0	The number of packets received via the interface that were discarded because of an unknown or unsupported protocol.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOutOctets.0	The total number of octets transmitted out of the interface, including framing characters.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOutUcastPkts.0	The total number of packets that higher-level protocols requested be transmitted to a subnetwork-unicast address, including those that were discarded or not sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOutNUcastPkts.0	The total number of packets that higher-level protocols requested be transmitted to a non-unicast (i.e., a subnetwork-broadcast or subnetwork-multicast) address, including those that were discarded or not sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOutDiscards.0	The number of outbound packets that were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOutErrors.0	The number of outbound packets that could not be transmitted because of errors.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifOutQLen.0	The length of the output packet queue (in packets).	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.at.atTable.atEntry.atIfIndex.0	The interface on which this entry's equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of `ifIndex`.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.ip.ipForwarding.0	The indication of whether this entity is acting as an IP gateway in respect to the forwarding of datagrams received by, but not addressed to, this entity. IP gateways forward datagrams; IP hosts do not (except those source-routed via the host). For some managed nodes, this object may take on only a subset of the values possible. Accordingly, it is appropriate for an agent to return a `badValue` response if a management station attempts to change this object to an inappropriate value.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipDefaultTTL.0	The default value inserted into the Time-To-Live field of the IP header of datagrams originated at this entity, whenever a TTL value is not supplied by the transport layer protocol.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipInReceives.0	The total number of input datagrams received from interfaces, including those received in error.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipInHdrErrors.0	The number of input datagrams discarded because of errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, etc.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipInAddrErrors.0	The number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity. This count includes invalid addresses (e.g., 0.0.0.0) and addresses of unsupported classes (e.g., Class E). For entities that are not IP gateways and therefore do not forward datagrams, this counter includes datagrams discarded because the destination address was not a local address.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.ip.ipForwDatagrams.0	The number of input datagrams for which this entity was not their final IP destination, as a result of which an attempt was made to find a route to forward them to that final destination. In entities that do not act as IP gateways, this counter will include only those packets that were source-routed via this entity, and the source-route option processing was successful.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipInUnknownProtos.0	The number of locally-addressed datagrams received successfully but discarded because of an unknown or unsupported protocol.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipInDiscards.0	The number of input IP datagrams for which no problems were encountered to prevent their continued processing, but that were discarded (e.g., for lack of buffer space). This counter does not include any datagrams discarded while awaiting re-assembly.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipInDelivers.0	The total number of input datagrams successfully delivered to IP user-protocols (including ICMP).	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipOutRequests.0	The total number of IP datagrams that local IP user-protocols (including ICMP) supplied to IP in requests for transmission. This counter does not include any datagrams counted in `ipForwDatagrams`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipOutDiscards.0	The number of output IP datagrams for which no problem was encountered to prevent their transmission to their destination, but that were discarded (e.g., for lack of buffer space). This counter would include datagrams counted in `ipForwDatagrams` if any such packets met this (discretionary) discard criterion.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.ip.ipOutNoRoutes.0	The number of IP datagrams discarded because no route could be found to transmit them to their destination. This counter includes any packets counted in `ipForwDatagrams` that meet the “no route” criterion. This includes any datagrams that a host cannot route because all of its default gateways are down.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipReasmTimeout.0	The maximum number of seconds received fragments are held while they are awaiting reassembly at this entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipReasmReqds.0	The number of IP fragments received that needed to be reassembled at this entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipReasmOKs.0	The number of IP datagrams successfully reassembled.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipReasmFails.0	The number of failures detected by the IP reassembly algorithm (for whatever reason: timeout, errors, etc). This is not necessarily a count of discarded IP fragments, because some algorithms (notably the algorithm in RFC 815) can lose track of the number of fragments by combining them as they are received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipFragOKs.0	The number of IP datagrams that were successfully fragmented at this entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipFragFails.0	The number of IP datagrams that were discarded because they needed to be fragmented at this entity but could not be, e.g., because their **Don’t Fragment** flag was set.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipFragCreates.0	The number of IP datagram fragments that were generated as a result of fragmentation at this entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipAddrTable.ipAddrEntry.ipAdEntIfIndex.0	The index value that uniquely identifies the interface to which this entry is applicable. The interface identified by a particular value of this index is the same interface as identified by the same value of `ifIndex`.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.ip.ipAddrTable.ipAddrEntry.ipAdEntBcastAddr.0	The value of the least-significant bit in the IP broadcast address used for sending datagrams on the (logical) interface associated with the IP address of this entry. For example, when the Internet standard all-ones broadcast address is used, the value will be 1. This value applies to both the subnet and network broadcasts addresses used by the entity on this (logical) interface.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipAddrTable.ipAddrEntry.ipAdEntReasmMaxSize.0	The size of the largest IP datagram that this entity can re-assemble from incoming IP fragmented datagrams received on this interface.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteIfIndex.0	The index value that uniquely identifies the local interface through which the next hop of this route should be reached. The interface identified by a particular value of this index is the same interface as identified by the same value of `ifIndex`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteMetric1.0	The primary routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's `ipRouteProto` value. If this metric is not used, its value should be set to `-1`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteMetric2.0	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's `ipRouteProto` value. If this metric is not used, its value should be set to `-1`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteMetric3.0	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's `ipRouteProto` value. If this metric is not used, its value should be set to `-1`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteMetric4.0	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's `ipRouteProto` value. If this metric is not used, its value should be set to `-1`.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteType.0	The type of route. The values `direct(3)` and `indirect(4)` refer to the concept of direct and indirect routing in the IP architecture. Setting this object to the value `invalid(2)` has the effect of invalidating the corresponding entry in the `ipRouteTable` object. That is, it effectively dissasociates the destination identified with said entry from the route identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant `ipRouteType` object.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteProto.0	The routing mechanism via which this route was learned. Inclusion of values for gateway routing protocols is not intended to imply that hosts should support those protocols.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteAge.0	The number of seconds since this route was last updated or otherwise determined to be correct. No semantics of “too old” can be implied except through knowledge of the routing protocol by which the route was learned.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry.ipRouteMetric5.0	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route’s `ipRouteProto` value. If this metric is not used, its value should be set to `-1`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipNetToMediaTable.ipNetToMediaEntry.ipNetToMediaIfIndex.0	The interface on which this entry’s equivalence is effective. The interface identified by a particular value of this index is the same interface as identified by the same value of `ifIndex`.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.ip.ipNetToMediaTable.ipNetToMediaEntry.ipNetToMediaType.0	The type of mapping. Setting this object to the value `invalid(2)` has the effect of invalidating the corresponding entry in the `ipNetToMediaTable`. That is, it effectively dissasociates the interface identified with said entry from the mapping identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant `ipNetToMediaType` object.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.ip.ipRoutingDiscards.0	The number of routing entries that were chosen to be discarded even though they are valid. One possible reason for discarding such an entry could be to free up buffer space for other routing entries.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInMsgs.0	The total number of ICMP messages that the entity received. This counter includes all those counted by `icmpInErrors`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInErrors.0	The number of ICMP messages that the entity received but determined as having ICMP-specific errors (bad ICMP checksums, bad length, etc.).	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInDestUnreachs.0	The number of ICMP Destination Unreachable messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInTimeExcds.0	The number of ICMP Time Exceeded messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInParmProbs.0	The number of ICMP Parameter Problem messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInSrcQuenchs.0	The number of ICMP Source Quench messages received.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.icmp.icmpInRedirects.0	The number of ICMP Redirect messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInEchos.0	The number of ICMP Echo (request) messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInEchoReps.0	The number of ICMP Echo Reply messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInTimestamps.0	The number of ICMP Timestamp (request) messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInTimestampReps.0	The number of ICMP Timestamp Reply messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInAddrMasks.0	The number of ICMP Address Mask Request messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpInAddrMaskReps.0	The number of ICMP Address Mask Reply messages received.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutMsgs.0	The total number of ICMP messages that this entity attempted to send. This counter includes all those counted by `icmpOutErrors`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutErrors.0	The number of ICMP messages that this entity did not send because of problems discovered within ICMP, such as a lack of buffers. This value should not include errors discovered outside the ICMP layer, such as the inability of the IP to route the resultant datagram. Some implementations may have no types of error that contribute to this counter’s value.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutDestUnreachs.0	The number of ICMP Destination Unreachable messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutTimeExcds.0	The number of ICMP Time Exceeded messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutParmProbs.0	The number of ICMP Parameter Problem messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutSrcQuenchs.0	The number of ICMP Source Quench messages sent.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutRedirects.0	The number of ICMP Redirect messages sent. For a host, this object will always be zero, because hosts do not send redirects.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutEchos.0	The number of ICMP Echo (request) messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutEchoReps.0	The number of ICMP Echo Reply messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutTimestamps.0	The number of ICMP Timestamp (request) messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutTimestampReps.0	The number of ICMP Timestamp Reply messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutAddrMasks.0	The number of ICMP Address Mask Request messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.icmp.icmpOutAddrMaskReps.0	The number of ICMP Address Mask Reply messages sent.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpRtoAlgorithm.0	The algorithm used to determine the timeout value used for retransmitting unacknowledged octets.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpRtoMin.0	The minimum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the LBOUND quantity described in RFC 793.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpRtoMax.0	The maximum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the UBOUND quantity described in RFC 793.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.tcp.tcpMaxConn.0	The limit on the total number of TCP connections the entity can support. In entities where the maximum number of connections is dynamic, this object should contain the value `-1`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpActiveOpens.0	The number of times TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpPassiveOpens.0	The number of times TCP connections have made a direct transition to the SYN-RCVD state from the LISTEN state.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpAttemptFails.0	The number of times TCP connections have made a direct transition to the CLOSED state from either the SYN-SENT state or the SYN-RCVD state, plus the number of times TCP connections have made a direct transition to the LISTEN state from the SYN-RCVD state.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpEstabResets.0	The number of times TCP connections have made a direct transition to the CLOSED state from either the ESTABLISHED state or the CLOSE-WAIT state.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpCurrEstab.0	The number of TCP connections for which the current state is either ESTABLISHED or CLOSE-WAIT.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpInSegs.0	The total number of segments received, including those received in error. This count includes segments received on currently established connections.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpOutSegs.0	The total number of segments sent, including those on current connections but excluding those containing only retransmitted octets.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpRetransSegs.0	The total number of segments retransmitted; that is, the number of TCP segments transmitted containing one or more previously transmitted octets.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.tcp.tcpConnTable.tcpConnEntry.tcpConnState.0	The state of this TCP connection. The only value that may be set by a management station is `deleteTCB(12)`. Accordingly, it is appropriate for an agent to return a `badValue` response if a management station attempts to set this object to any other value. If a management station sets this object to the value `deleteTCB(12)`, then this has the effect of deleting the TCB (as defined in RFC 793) of the corresponding connection on the managed node, resulting in immediate termination of the connection. As an implementation-specific option, a RST segment may be sent from the managed node to the other TCP endpoint (note however that RST segments are not sent reliably).	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpConnTable.tcpConnEntry.tcpConnLocalPort.0	The local port number for this TCP connection.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpConnTable.tcpConnEntry.tcpConnRemPort.0	The remote port number for this TCP connection.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpInErrs.0	The total number of segments received in error (e.g., bad TCP checksums).	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.tcp.tcpOutRsts.0	The number of TCP segments sent containing the RST flag.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.udp.udpInDatagrams.0	The total number of UDP datagrams delivered to UDP users.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.udp.udpNoPorts.0	The total number of received UDP datagrams for which there was no application at the destination port.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.udp.udpInErrors.0	The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.udp.udpOutDatagrams.0	The total number of UDP datagrams sent from this entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.udp.udpTable.udpEntry.udpLocalPort.0	The local port number for this UDP listener.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.egp.egpInMsgs.0	The number of EGP messages received without error.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpInErrors.0	The number of EGP messages received that proved to be in error.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpOutMsgs.0	The total number of locally generated EGP messages.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpOutErrors.0	The number of locally generated EGP messages not sent because of resource limitations within an EGP entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighState.0	The EGP state of the local system with respect to this entry's EGP neighbor. Each EGP state is represented by a value that is one greater than the numerical value associated with said state in RFC 904.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighAs.0	The autonomous system of this EGP peer. Zero should be specified if the autonomous system number of the neighbor is not yet known.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighInMsgs.0	The number of EGP messages received without error from this EGP peer.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighInErrs.0	The number of EGP messages received from this EGP peer that proved to be in error (e.g., bad EGP checksum).	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighOutMsgs.0	The number of locally generated EGP messages to this EGP peer.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighOutErrs.0	The number of locally generated EGP messages not sent to this EGP peer because of resource limitations within an EGP entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighInErrMsgs.0	The number of EGP-defined error messages received from this EGP peer.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighOutErrMsgs.0	The number of EGP-defined error messages sent to this EGP peer.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighStateUps.0	The number of EGP state transitions to the UP state with this EGP peer.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighStateDowns.0	The number of EGP state transitions from the UP state to any other state with this EGP peer.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighIntervalHello.0	The interval between EGP Hello command retransmissions (in hundredths of a second). This represents the t1 timer as defined in RFC 904.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighIntervalPoll.0	The interval between EGP poll command retransmissions (in hundredths of a second). This represents the t3 timer as defined in RFC 904.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighMode.0	The polling mode of this EGP entity, either passive or active.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry.egpNeighEventTrigger.0	A control variable used to trigger operator-initiated Start and Stop events. When read, this variable always returns the most recent value that `egpNeighEventTrigger` was set to. If it has not been set since the last initialization of the network management subsystem on the node, it returns a value of `stop`. When set, this variable causes a Start or Stop event on the specified neighbor, as specified on pages 8-10 of RFC 904. A Start event causes an Idle peer to begin neighbor acquisition and a non-Idle peer to reinitiate neighbor acquisition. A Stop event causes a non-Idle peer to return to the Idle state until a Start event occurs, via `egpNeighEventTrigger` or otherwise.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.egp.egpAs.0	The autonomous system number of this EGP entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInPkts.0	The total number of messages delivered to the SNMP entity from the transport service.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutPkts.0	The total number of SNMP messages that were passed from the SNMP protocol entity to the transport service.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInBadVersions.0	The total number of SNMP messages that were delivered to the SNMP protocol entity and were for an unsupported SNMP version.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInBadCommunityNames.0	The total number of SNMP messages delivered to the SNMP protocol entity that used a SNMP community name not known to said entity.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.snmp.snmpInBadCommunityUses.0	The total number of SNMP messages delivered to the SNMP protocol entity that represented an SNMP operation that was not allowed by the SNMP community named in the message.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInASNParseErrs.0	The total number of ASN.1 or BER errors encountered by the SNMP protocol entity when decoding received SNMP Messages.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInTooBigs.0	The total number of SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is `tooBig`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInNoSuchNames.0	The total number of SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is `noSuchName`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInBadValues.0	The total number of SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is `badValue`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInReadOnlys.0	The total number of valid SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is `readOnly`. It is a protocol error to generate an SNMP PDU that contains the value `readOnly` in the error-status field, because this object is provided as a means of detecting incorrect implementations of the SNMP.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInGenErrs.0	The total number of SNMP PDUs that were delivered to the SNMP protocol entity and for which the value of the error-status field is `genErr`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInTotalReqVars.0	The total number of MIB objects that were retrieved successfully by the SNMP protocol entity as the result of receiving valid SNMP Get-Request and Get-Next PDUs.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInTotalSetVars.0	The total number of MIB objects that were altered successfully by the SNMP protocol entity as the result of receiving valid SNMP Set-Request PDUs.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.snmp.snmpInGetRequests.0	The total number of SNMP Get-Request PDUs that were accepted and processed by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInGetNexts.0	The total number of SNMP Get-Next PDUs that were accepted and processed by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInSetRequests.0	The total number of SNMP Set-Request PDUs that were accepted and processed by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInGetResponses.0	The total number of SNMP Get-Response PDUs that were accepted and processed by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpInTraps.0	The total number of SNMP Trap PDUs that were accepted and processed by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutTooBigs.0	The total number of SNMP PDUs that were generated by the SNMP protocol entity and for which the value of the error-status field is `tooBig`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutNoSuchNames.0	The total number of SNMP PDUs that were generated by the SNMP protocol entity and for which the value of the error-status is `noSuchName`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutBadValues.0	The total number of SNMP PDUs that were generated by the SNMP protocol entity and for which the value of the error-status field is `badValue`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutGenErrs.0	The total number of SNMP PDUs that were generated by the SNMP protocol entity and for which the value of the error-status field is `genErr`.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutGetRequests.0	The total number of SNMP Get-Request PDUs that were generated by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutGetNexts.0	The total number of SNMP Get-Next PDUs that were generated by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutSetRequests.0	The total number of SNMP Set-Request PDUs that were generated by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutGetResponses.0	The total number of SNMP Get-Response PDUs that were generated by the SNMP protocol entity.	1.3.6.1

iso.org.dod.internet.mgmt.mib-2.snmp.snmpOutTraps.0	The total number of SNMP Trap PDUs that were generated by the SNMP protocol entity.	1.3.6.1
iso.org.dod.internet.mgmt.mib-2.snmp.snmpEnableAuthenTraps.0	Indicates whether the SNMP Agent process is permitted to generate authentication-failure traps. The value of this object overrides any configuration information. It provides a way to disable all authentication-failure traps. It is strongly recommended that this object be stored in non-volatile memory so that it remains constant between re-initializations of the network management system.	1.3.6.1

Sun management center measures

ID
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemUpTime.0
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemUsers.0
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemAvgJobs1.0
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemAvgJobs5.0
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemAvgJobs15.0
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemMaxProc.0
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemFreeMemory.0
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemPhysMemory.0
iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemMaxUserMem.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemSwapConfig.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemEnabledSwap.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemFreeSwap.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemUserCPU.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemSysCPU.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemIdleCPU.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.computerSystem.computerSystemNiceCPU.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.fileSystem.fileSystemMounted.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.fileSystem.fileSystemTable.fileSystemEntry.fileSyst

iso.org.dod.internet.private.enterprises.hp.nm.system.general.fileSystem.fileSystemTable.fileSystemEntry.fileSyst

iso.org.dod.internet.private.enterprises.hp.nm.system.general.fileSystem.fileSystemTable.fileSystemEntry.fileSyst

iso.org.dod.internet.private.enterprises.hp.nm.system.general.fileSystem.fileSystemTable.fileSystemEntry.fileSyst

iso.org.dod.internet.private.enterprises.hp.nm.system.general.fileSystem.fileSystemTable.fileSystemEntry.fileSyst

iso.org.dod.internet.private.enterprises.hp.nm.system.general.fileSystem.fileSystemTable.fileSystemEntry.fileSyst

iso.org.dod.internet.private.enterprises.hp.nm.system.general.fileSystem.fileSystemTable.fileSystemEntry.fileSyst

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processNum.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processPID

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processIdx.

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processUID

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processPP

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processDsi

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processTsi

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processSsi

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processNic

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processMa

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processMin

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processPgr

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processPri

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processAd

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processCP

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processUti

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processStir

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processSta

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processFla

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processSta

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processWc

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processPrc

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processTim

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processCP

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processCP

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processFss

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processPct

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processRss

iso.org.dod.internet.private.enterprises.hp.nm.system.general.processes.processTable.processEntry.processSU

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.interface.ieee8023Mac.ieee8023MacTable.ieee8023MacEntry.iee

iso.org.dod.internet.private.enterprises.hp.nm.icmp.icmpEchoReq.0

iso.org.dod.internet.private.enterprises.hp.nm.icmp.icmpEchoReqTable.icmpEchoReqEntry.icmpEchoReqTime.0

iso.org.dod.internet.private.enterprises.hp.nm.icmp.icmpEchoReqTable.icmpEchoReqEntry.icmpEchoReqPktSize

iso.org.dod.internet.private.enterprises.hp.nm.icmp.icmpEchoReqTable.icmpEchoReqEntry.icmpEchoReqTimeOu

iso.org.dod.internet.private.enterprises.hp.nm.snmp.trap.trapDestinationNum.0

iso.org.dod.internet.private.enterprises.hp.nm.snmp.snmpdConf.snmpdConfRespond.0

iso.org.dod.internet.private.enterprises.hp.nm.snmp.snmpdConf.snmpdReConfigure.0

iso.org.dod.internet.private.enterprises.hp.nm.snmp.snmpdConf.snmpdFlag.0

iso.org.dod.internet.private.enterprises.hp.nm.snmp.snmpdConf.snmpdLogMask.0

iso.org.dod.internet.private.enterprises.hp.nm.snmp.snmpdConf.snmpdVersion.0

iso.org.dod.internet.private.enterprises.hp.nm.snmp.snmpdConf.snmpdStatus.0

iso.org.dod.internet.private.enterprises.hp.nm.snmp.snmpdConf.snmpdSize.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.cluster.isClustered.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.cluster.clusterTable.clusterEntry.clusterID.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.cluster.clusterTable.clusterEntry.clusterSwapServi

iso.org.dod.internet.private.enterprises.hp.nm.system.general.cluster.clusterTable.clusterEntry.clusterKcsp.0

iso.org.dod.internet.private.enterprises.hp.nm.system.general.cluster.clusterCnodeID.0

iso.org.dod.internet.private.enterprises.hp.nm.snmp.authfail.authFailTable.authFailEntry.authTime.0

iso.org.dod.internet.private.enterprises.hp.nm.openView.openViewTrapVars.openViewSourceId.0

HP-UNIX system measures

Id

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmSwap.hmSwapSpac

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmSwap.hmSwapSpac

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmSwap.hmSwapSpac

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmSwap.hmSwapSpac

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmKernelcontention.hr

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmKernelcontention.hr

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmNFS.hmNFSClientR

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmNFS.hmNFSClientR

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmNFS.hmNFSClientR

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmNFS.hmNFSClientR

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmNFS.hmNFSClientR

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmNFS.hmNFSClientR

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmNFS.hmNFSClientR

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmNFS.hmNFSClientR

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmCPU.hmCPUProcIn

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmCPU.hmCPUProcIn

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmCPU.hmCPUProcIn

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmRAM.hmRamMemor

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmRAM.hmRamMemor

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmKMEM.hmKmemSta

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmKMEM.hmKmemSta

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmDNLC.hmDNLCStat

iso.org.dod.internet.private.enterprises.sun.prod.sunsymon.agent.modules.healthMonitor.hmDNLC.hmDNLCStat

Unix system monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Setup and Basic Configuration

Select host(s) that should be the target(s) of the monitor

Customizing the execution schedule

Centrally deploying a monitor to different global locations

Executed commands

Measures

Overview

AppMon supports monitoring of local or remote UNIX systems via the Unix Monitor.

The AppMon Unix Monitor performs agentless system monitoring. The advantage of agentless monitoring is that you do not need to spend time and effort in deploying and maintaining Agents on every monitored system.

The Unix Monitor collects relevant monitoring data from the systems of interest via SSH, via Rexec, or locally. The Monitor can be executed on AppMon Collectors running on Windows or Linux. The Windows Collector does not support the **Local** option.

The Monitor queries common UNIX status commands and returns relevant information filtered from the resulting output stream. If every necessary command is activated on the monitored systems, the Monitor delivers values for every Measure it is configured to deliver. The operating system is automatically identified by executing the `uname` command on the monitored machine.

The Monitor is supported on Linux, Solaris, AIX, and HP-UX platforms.

The Unix Monitor Command List file contains executed commands, expected output, and descriptions of how the values are parsed.

Setup and Basic Configuration

To add a Unix Monitor to the System Profile:

1. In the **System Profile Preferences** , select **Monitors**.
2. Click **Create** to open the **Monitor Type Chooser**.
3. Select **Unix Monitor** and click **OK** to open the **Scheduled Monitor Editor**.
4. Specify a name and an optional description for the Monitor
5. In the **Settings** area, configure the settings as described below.
6. Add hosts as needed. See below for details.

Monitor settings

- **Method** – The connection method can either be **SSH** or **Local**. See below more information on the SSH connection method. Use **Local** only if the target host is the same as the Collector where the Monitor is executed. If you select **Local**, all other settings except **Customize Measurements** (see below) become unavailable.
- **Authentication Method** – For the **SSH** connection method. select either **Password** or **PublicKey**. For **Password** authentication, define these settings:
  - **Port** – The port through which the Monitor connects to the target host.
  - **Username** – The username to log on to the remote host.
  - **Password** – The password for the specified username.

  For **PublicKey** authentication, define these settings:
  - **Port** – The port for the SSH or RExec connection.
  - **Username** – The username to log on to the remote host.
  - **Passphrase** – The optional passphrase for the public key-file.
  - **Key-File** – The private key-file on the AppMon Collector.
- **Customize Measurements** – See below.

Select host(s) that should be the target(s) of the monitor

A Monitor can be configured for multiple target hosts. Only one configuration is necessary; it is executed for all configured machines. This is especially useful in cluster environments that feature many similarly configured hosts.

Add **Single hosts**, **hostgroups**, or **sites** by clicking the **+** icon. Create a new host if the desired host is not shown. If you add a hostgroup or site, the Monitor will be run for all contained hosts.

Optionally, add an **intersection** of hostgroups or sites. The Monitor will be run for all hosts that belong to all selected groups or sites. If the configuration would resolve a single host more than once (if Host1 is also contained in Hostgroup1), the Monitor is only executed once per host.

You can also add target hosts via the Server REST interface. The required POST request is located in the **Tasks and Monitors** group. The request documentation is based on the Swagger specification and embedded to the request form.

For more information how to manage hosts and hostgroups, see Infrastructure.

Examples

Configured Hosts As Shown in Hosts Table	Monitor Will Be Run for…
Host1	Host1
Host1 Host2	Host1 and Host 1
Hostgroup 1 Host2	All hosts from Hostgroup 1 and also Host 2
Hostgroup 1 Hostgroup 2	All hosts from Hostgroup 1 and also Hostgroup 2
Hosts belonging to Hostgroup 1, Hostgroup 2 Host 1	All hosts that are **both** in Hostgroup 1 and Hostgroup 2 (an intersection of Hostgroup 1 and Hostgroup 2), plus Host 1

Customizing the execution schedule

Open the **Schedule** tab, then choose a predefined schedule or create your own schedule if none fits. Any schedules you create will be available for the whole System Profile.

Centrally deploying a monitor to different global locations

Open the **Schedule** tab, then select the desired AppMon Collector or AppMon Server from the **Execute on** list. Only currently connected AppMon Collectors are shown in this list. The selection appears as **Embedded Dynatrace Collector**.

> **Note**
>
> If an AppMon Collector that is connected through a slow network or a Monitor that contains large library files is run on a remote machine, it is possible, though rare, that the first execution fails with a Missing Binaries error. This error should disappear on the next scheduled run of the Monitor.

Customized measurements

Use the **Customize Measurements** to customize the Measurements to a single CPU, NIC, or disk partition. When you select the checkbox in the **Value** column, three settings appear:

- The **CPU** setting can be set to a CPU ID starting with 0. This requires the `mpstat` command on Solaris and the `sar` command on the other operating systems.
- The **NIC** setting restricts the network measures to a single interface. Enter the value as shown in the `netstat` command; for example, `eth0`.
- The **Partition** option restricts the disk measures to a specified partition. Enter the value as shown in the first column of the `df` command.

SSH connection with password authentication

For password authentication between the Monitor and the SSH server, the SSH server has to be configured to allow password authentication (not to be confused with keyboard-interactive authentication, which SSH also supports but which is not used by the Monitor). The following line must be part of the `/etc/ssh/sshd_config` file of the target system.

```
PasswordAuthentication yes
```

If this is not set, password authentication will fail.

SSH connection with public key authentication

If you select **PublicKey** authentication, the public key must be added to the `authorized-keys2` file on the SSH server, in the `.ssh` folder. The path to the key-file must be correct.

Here is an example how to generate an SSH key pair, if not already available, using OpenSSh:

```
Execute "ssh-keygen -t dsa"
```

or

```
Execute "ssh-keygen -t rsa"
```

Copy the `id_dsa` (or `id_rsa`) file to a location available to the AppMon Collector. Specify the file path in the **Key-File** field in the **Monitor** tab.

To add the public key to the `authorized-keys2` file on the server, execute:

```
scp C:/.ssh/id_dsa.pub user@server:~/.ssh/authorized_keys2
```

Depending on the SSH-server setup, it might be necessary to use `authorized_keys` as the target file instead of `authorized_keys2`.

Only OpenSSH-style public keys can be used. Using keys saved by puttygen via **Save private key** (generating a `.ppk` file) will result in an authentication error. To create OpenSSH keys with puttygen, select **Export OpenSSH key** from the puttygen **Conversions** menu.

Executed commands

The attached `.xls` file contains a list of the commands that are executed on the monitored systems. Further information can be found in the man pages.

After the activation of commands from the list, the Monitor should deliver values for each subscribed measure, with exceptions as described below. To collect a subset of the available measures, modify the selections in the **Measures** tab of the **Scheduled Monitor Editor**. You can also create incident rules if that option is available for the selected Measure.

Measures

The following measure can be subscribed.

- Unix Memory Usage
  - Memory Available – Percentage of total memory that is usable (includes caches and buffers).
  - Memory Usage – Percentage of total memory that is being used.
  - Swap Usage – Percentage of total swap that is in use.
- Unix Memory
  - Total Memory – Available total memory.
  - Used Memory – Currently used memory.
  - Free Memory – Currently free memory.
  - Total Swap – Available total swap space.
  - Used Swap – Currently used swap space.
  - Free Swap – Currently free swap space.
  - Pages Input – Paged/swapped in.
  - Pages Ouput – Paged/swapped out.
- Unix CPU Usage
  - Processor Time – Total CPU usage (sum of user and system time).
  - User Time – Time spent running non-kernel code.
  - System Time – Time spent running kernel code.
  - Idle Time – Time idle.
  - I/O Wait Time – Time CPU was blocked while waiting for I/O operations.
- Unix Process
  - Blocked – Number of processes in non-interruptible sleep.
  - Run Queue, current – Number of processes waiting for runtime.
  - Run Queue, last minute – Run Queue Length for the last minute.
  - Run Queue, last 5 minutes – Run Queue Length for the last 5 minutes.
  - Run Queue, last 15 minutes – Run Queue Length for the last 15 minutes.
  - Process Count – Number of processes currently shown in PS.
- Unix Filesystem Usage
  - Temp Usage – Percentage of used space for file system mounted on `/tmp`.
  - Var Usage – Percentage of used space for file system mounted on `/var`.
  - Root Usage – Percentage of used space for file system mounted on `/`.
- Unix Filesystem
  - Free Diskspace – Free disk space.
  - Total Diskspace – Total disk space.
  - Used Diskspace – Used disk space.
  - Disk Reads/sec – Reads per second.
  - Disk Writes/sec – Writes per second.

- Unix Network
  - Bytes Total/sec – Total number of bytes received and transmitted.
  - Transmitted Bytes/sec – Total number of bytes transmitted.
  - Received Bytes/sec – Total number of bytes received.
  - Received Packets Total – Total number of packets received.
  - Received Packets OK – Number of successfully received packets.
  - Received Packets Error – Number of packets received with errors.
  - Received Packets Dropped – Number of dropped packets.
  - Transmitted Packets OK – Number of successfully transmitted packets.
  - Transmitted Packets Error – Number of packets received with errors.
  - Transmitted Packets Dropped – Number of dropped packets.
  - Transmitted Packets Total – Total number of packets transmitted.
  - Collisions – Collisions

Because of incompatibilities of some commands across different UNIX systems, the following measures will not deliver values for the specified UNIX system.

Linux:

- Collisions (in Network Group)

Solaris:

- Transmitted Packets Dropped (in Network Group)
- Received Packets Dropped (in Network Group)
- I/O Wait Time (in CPU Group)

AIX:

- Bytes Total/sec (in Network Group)
- Received Bytes/sec (in Network Group)
- Transmitted Bytes/sec (in Network Group)
- Transmitted Packets Dropped (in Network Group)
- Received Packets Dropped (in Network Group)

HP-UX

- Total Swap (in Memory Group)
- Free Swap (in Memory Group)
- Used Swap (in Memory Group)

- Root Usage (in Filesystem Group)
- Temp Usage (in Filesystem Group)
- Var Usage (in Filesystem Group)
- Bytes Total/sec (in Network Group)
- Received Bytes/sec (in Network Group)
- Transmitted Bytes/sec (in Network Group)
- Transmitted Packets Dropped (in Network Group)
- Received Packets Dropped (in Network Group)
- Transmitted Packets Error (in Network Group)
- Received Packets Error (in Network Group)
- Transmitted Packets OK (in Network Group)
- Received Packets OK (in Network Group)
- Collisions (in Network Group)
- I/O Wait Time (in CPU Group)

Tested systems

The current AppMon release was tested to monitor the following operating systems:

- AIX 5.2
- HP-UX B.11.23
- Red Hat Enterprise Linux AS release 4 (Nahant Update 8)
- SunOS 5.10
- Ubuntu multiple (Ubuntu 12.04.5 LTS, Ubuntu 14.04.5 LTS, and others)

The Unix Monitor does not work in combination with CSH.

URL monitor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Setup

Select host(s) that should be the target(s) of the monitor

The URL Monitor provides a convenient and easy way to monitor availability and performance-related data for a specific HTTP URL. It supports authentication, proxies, and redirection. Rudimentary verification of returned HTTP content is available.

It periodically sends an HTTP request according to its configuration to a specific HTTP server, and waits until it receives the HTTP responses from the server. If a timeout occurs while waiting for the HTTP response, the URL is classified as unreachable. A URL is also reachable if the HTTP server returns a 404 response that indicates the specific URL cannot be resolved. Use content verification features to filter such responses.

Note

See URL Monitor Plugin Community Edition for functionality extension.

Important: The URL Monitor Plugin Community Edition is only supported by the Community, *not* by AppMon.

Setup

To add a new URL Monitor to a System Profile, open the **System Profile Preferences**, select **Monitors**, click **Create**, and select **URL Monitor**. Specify a name and a description for the Monitor, then proceed with the settings as described below.

Basic Configuration

Basic configuration of a URL Monitor consists in the definition of a **path** and **target host(s)**, to point to a specific resource on a specific HTTP server(s). All other configuration parameters are set to default values that match most static HTTP URLs. Use the configuration parameter **Path** to specify the URL that should be monitored. The most convenient way to do this is to navigate to the specific URL in a web browser and paste the path from the address bar.

Select host(s) that should be the target(s) of the monitor

A Monitor can be configured for multiple target hosts. Only one configuration is necessary; it is executed for all configured machines. This is especially useful in cluster environments that feature many similarly configured hosts.

Add **Single hosts**, **hostgroups**, or **sites** by clicking the **+** icon. Create a new host if the desired host is not shown. If you add a hostgroup or site, the Monitor will be run for all contained hosts.

Optionally, add an **intersection** of hostgroups or sites. The Monitor will be run for all hosts that belong to all selected groups or sites. If the configuration would resolve a single host more than once (if Host1 is also contained in Hostgroup1), the Monitor is only executed once per host.

You can also add target hosts via the Server REST interface. The required POST request is located in the **Tasks and Monitors** group. The request documentation is based on the Swagger specification and embedded to the request form.

For more information how to manage hosts and hostgroups, see Infrastructure.

Examples

Configured Hosts As Shown in Hosts Table	Monitor Will Be Run for…
Host1	Host1
Host1 Host2	Host1 and Host 1
Hostgroup 1 Host2	All hosts from Hostgroup 1 and also Host 2

Hostgroup 1 Hostgroup 2	All hosts from Hostgroup 1 and also Hostgroup 2
Hosts belonging to Hostgroup 1, Hostgroup 2 Host 1	All hosts that are **both** in Hostgroup 1 and Hostgroup 2 (an intersection of Hostgroup 1 and Hostgroup 2), plus Host 1

HTTP request configuration

Use the **Method** configuration parameter to specify an HTTP request method: **GET**, **HEAD**, and **POST**. If you select **POST**, an additional configuration parameter, **POST Data**, becomes available. This parameter can be used to simulate HTTP form data.

Use the configuration parameter **HTTP version** to select HTTP 1.1 or HTTP 1.0 and to set the HTTP protocol version that is used to communicate with the HTTP server.

You can specify a user agent string in the **User-Agent** configuration parameter.

The **Max. redirects** configuration parameter defines the number of redirects that are followed to retrieve the HTTP content.

Content verification

The URL Monitor provides basic verification of the received HTTP content.

If the configuration parameter **Match Content** is set to **Success if Match** or **Error if Match**, an additional configuration parameter **Search String** is available. Use **Search String** to specify a string for which the received HTTP content is scanned.

The result of the content verification is provided by **ContentVerified** measures. It is set to 1 if the provided search string is found and **Match Content** is set to **Success if Match**, or if the provided search string is not found and **Match Content** is set to **Error if Match**. Otherwise, **ContentVerified** is set to 0.

If **Match Content** is set to **Expected size in bytes**, an additional configuration parameter **Bytes** is available to specify the expected size of the HTTP response in bytes. If the size of the received response is equal to the expected size, **ContentVerified** is set to 1, otherwise it is set to 0.

Authorization and proxy

The URL Monitor provides configuration parameters to specify HTTP server credentials, use of an HTTP proxy, and HTTP proxy credentials.

If the configuration parameter **Server authorization** is set to **Basic**, three additional configuration parameters, **Server username**, **Server password** and **Preemptive server authentication**, are available to specify user name and password, which are used to log on to the HTTP server. The monitor uses these credentials for Basic HTTP Authentication. If **Preemptive server authentication** is enabled, the URL Monitor will send the authentication Information in the initial request without waiting for an HTTP 401 Server answer. This would be speed up the complete request a little bit and it is preferred to be enabled.

If the configuration parameter **Server authorization** is set to **NTLM**, two additional configuration parameters, **Server username** and **Server password**, are available to specify user name and password, which are used to log on to the HTTP server. The monitor uses these credentials for NTLM HTTP Authentication.

Enabling **Proxy** provides the three additional configuration parameters **Proxy host**, **Proxy port**, and **Proxy authorization**. **Proxy host** and **Proxy port** identify a specific proxy service in the network, and **Proxy authorization** can be used to define a **Proxy username**, **Proxy password** and **Preemptive proxy authentication** if the selected proxy service requires authentication.

Measures

The URL Monitor provides a fixed set of Measure that describe performance, size, and validation of single URL requests.

- **ConnectionCloseDelay** – Required time to close the server connection.
- **ConnectionTimedout** – 1 if a connection timeout occurred (this happens, if the requested server can’t be reached while initiating the connection)
- **ContentVerified** – 1 if content verification was successful, 0 if not. No value is provided if content verification is disabled.
- **FirstResponseDelay** – Time elapsed between sending the URL request and receiving the first byte of the response.
- **HeaderSize** – Size of the response header.
- **HostReachable** – 1 if the HTTP server sent a response in time, 0 if not.
- **HttpStatusCode** – HTTP status code received from the server mapped to a number.

- **ResponseCompleteTime** – Time elapsed between sending the URL request and receiving the last byte of the response.
- **ResponseSize** – Size of the HTTP response.
- **SocketTimedout** – 1 if a socket timeout occurred (this happens, if the connection has been established, but while further communication there was a timeout)
- **Throughput** – Average network throughput achieved during retrieval of HTTP response.

VMware

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

About VMware monitoring

Applications in a virtual machine

Running AppMon in a virtual environment

External links

vSphere management SDK for vSphere

Datacenter Administrator Guide

Overview

To run your application in a virtual environment, you must have a virtualization-aware monitoring tool. Use AppMon to monitor the whole stack, from the application down to the guest OS and to the virtual machine and the physical host.

This page describes how to leverage AppMon's monitoring capability for VMware. In addition, it includes special requirements for running AppMon in a virtual machine.

About VMware monitoring

AppMon's VMware monitoring uses the vSphere Management SDK for vSphere (VMware Infrastructure (VI) SDK for older versions). See System requirements for supported vSphere versions.

Note

vSphere 6.x monitoring is currently not supported.

Monitoring via predefined monitors and dashboard

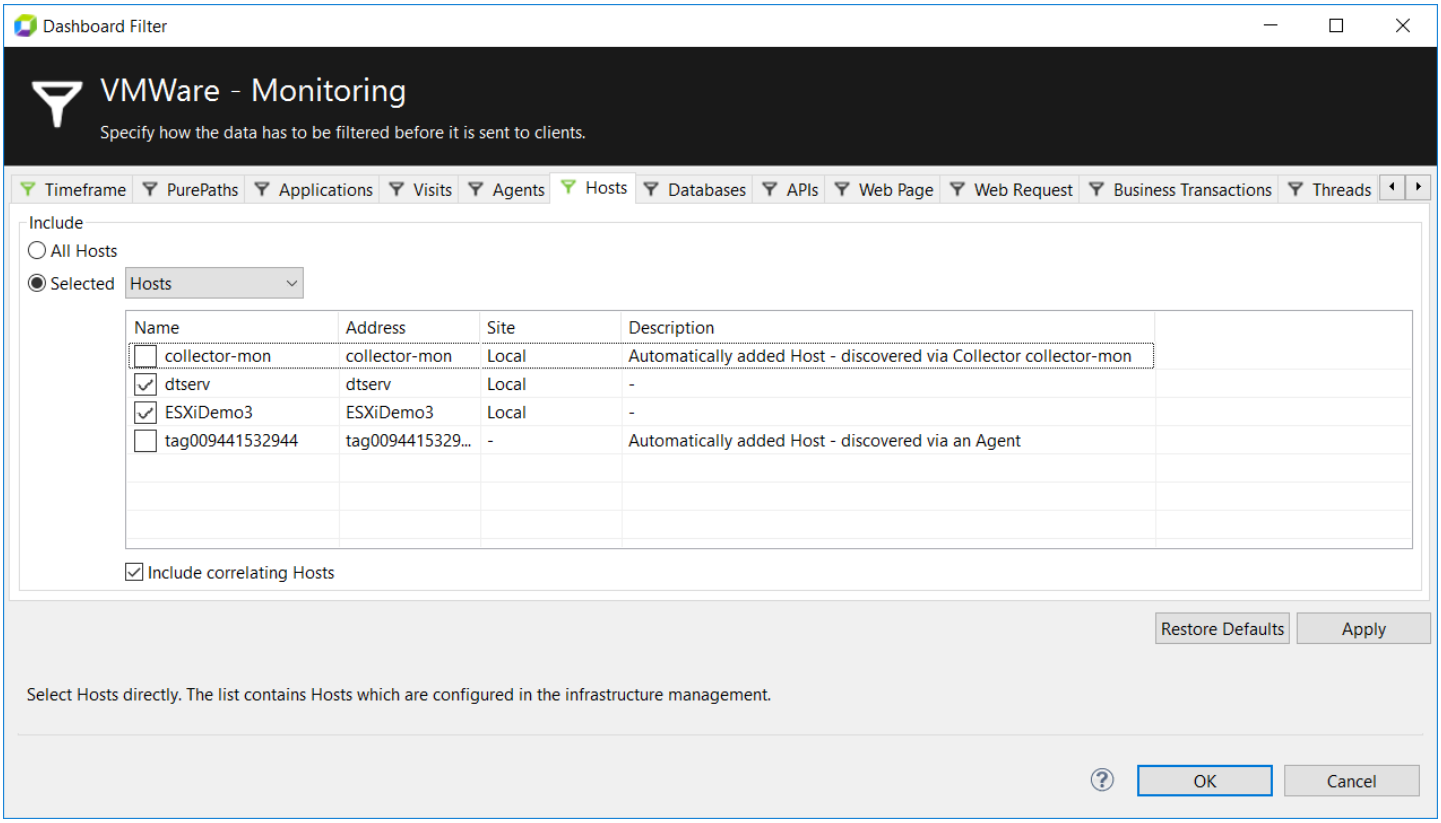
AppMon includes a predefined dashboard for VMware monitoring. In the AppMon Client, select **Dashboard > Open** and select **VMware - Monitoring** to open the VMware Monitoring dashboard.

AppMon automatically installs two monitors for VM monitoring when you set up system monitoring. One is for the VMware virtual machine performance monitor, the other is for the VMware Host system performance monitor. These are named **Virtual Machine** and **VM Host Systems** respectively. This page describes how to use the VMware - Monitoring dashboard and adjust configuration for these monitors.

To use the **VMware - Monitoring** dashboard:

1. Click the **Edit Filter** button in the AppMon Client toolbar to open the Dashboard Filter dialog box.
2. Click the **Hosts** tab, then click the **Selected** radio button make sure *Hosts* is selected in the list box.
3. Select the filters for the physical host and the virtual machine(s) that you want to monitor. For example, for a virtual machine called *dtserv* running on a physical host called *ESXiDemo3*, select those hosts. Click **OK**.

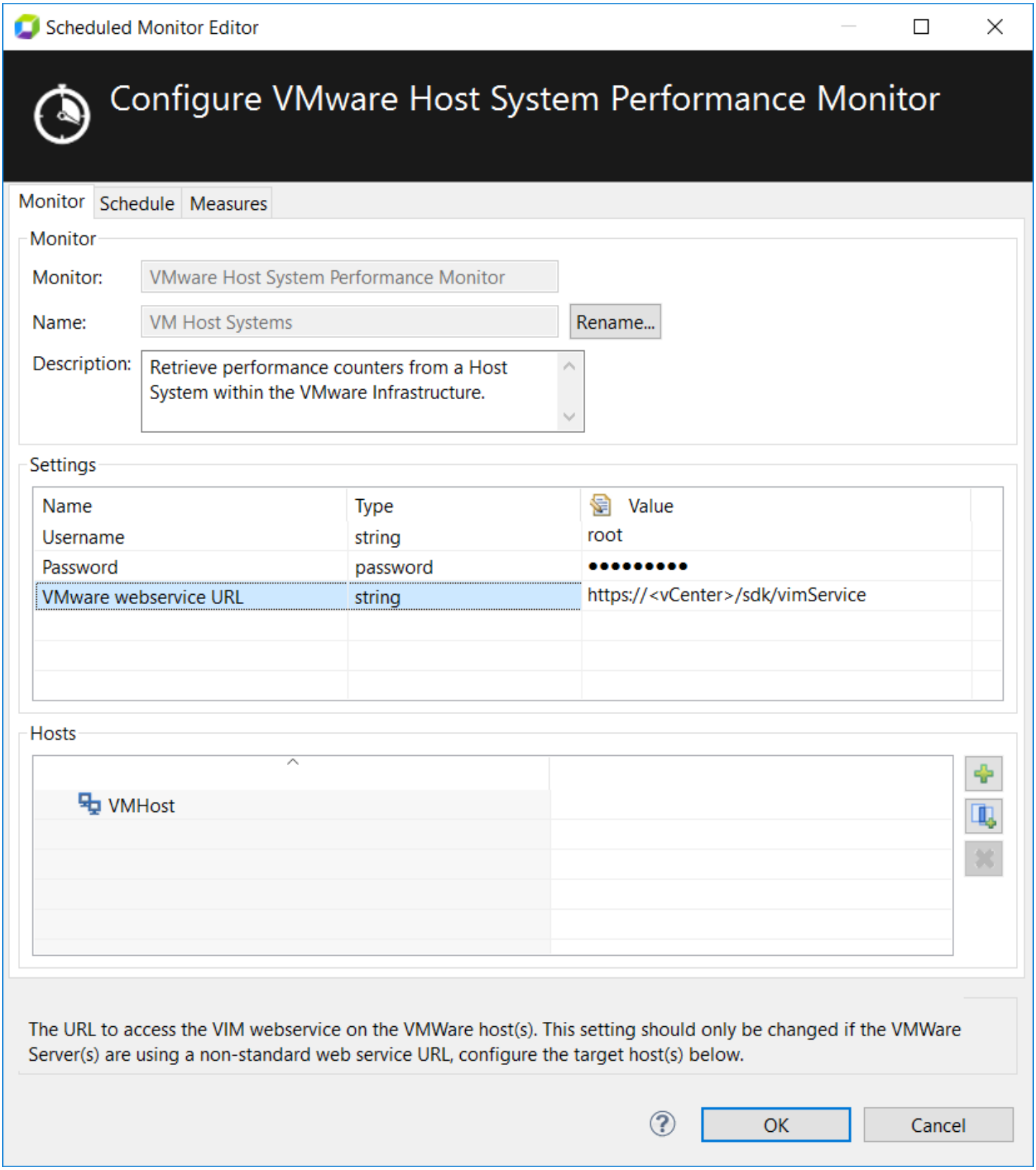
Host filtering on the VMware - Monitoring dashboard

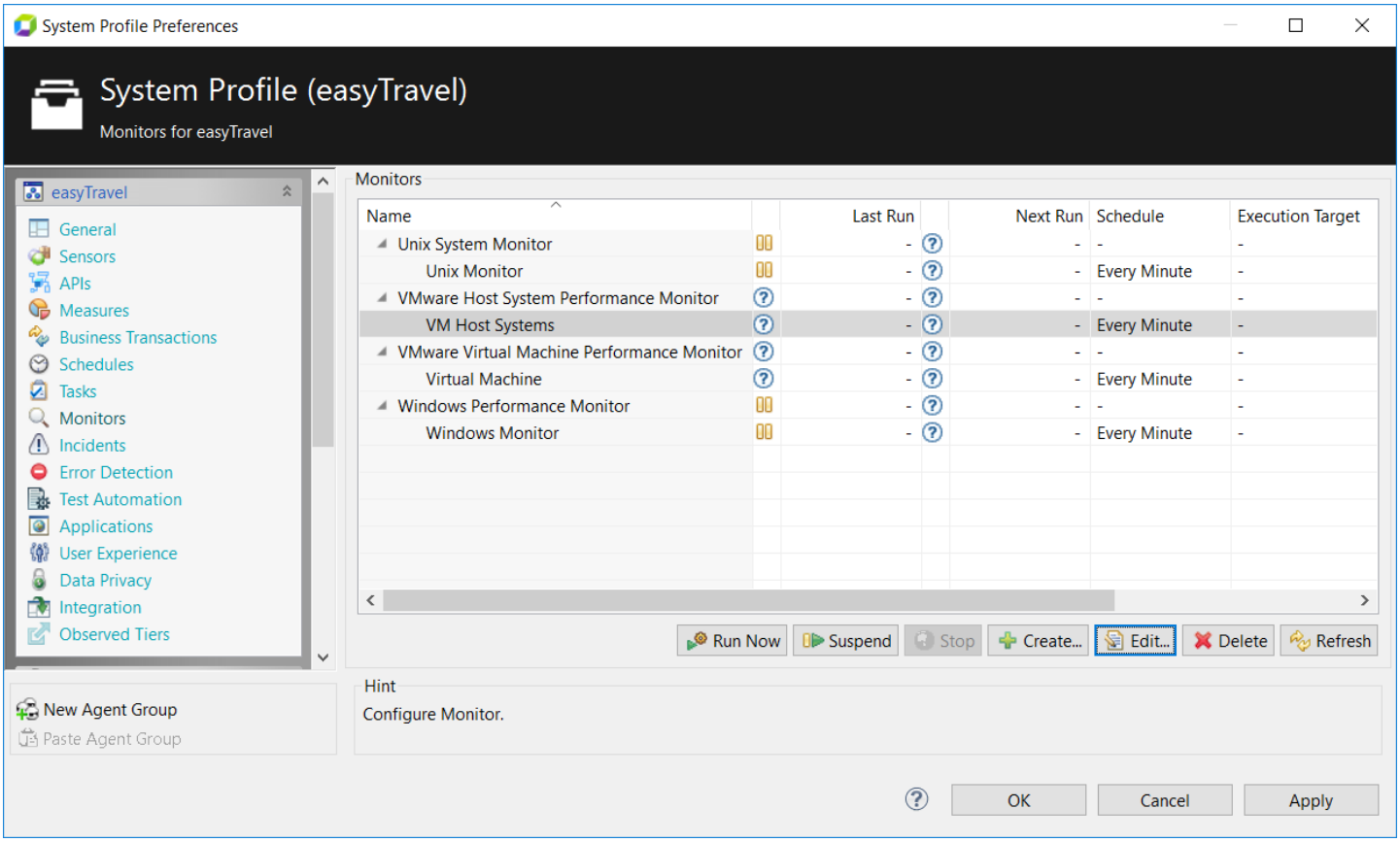


Monitoring virtual hosts

Use the **VMware Host System Performance Monitor** to monitor the physical machines that host your virtual machines.

1. In the AppMon Client, double-click the desired system profile for VMware monitoring and select **Monitors**.
2. Under **VMware Host System Performance Monitor**, select **VM Host Systems** and click **Edit**.
3. In the **Scheduled Monitor Editor** dialog box, enter a username and password for the VMware host account.
4. Decide whether to use a vCenter facility to monitor all your hosts, or monitor each one individually.
   - If you use vCenter, use the vCenter server host name (`https://<vCenterHostName>/sdk/vimService`) or the variable host as the VMware web service URL value and click **OK**. Your URL should look like the following, regardless of how many, and which hosts you monitor: `https://<host>/sdk/vimService`.


Scheduled Monitor Editor
Configure VMware Host System Performance Monitor
Monitor
Schedule
Measures
Monitor
Monitor:
VMware Host System Performance Monitor
Name:
VM Host Systems
Rename...
Description:
Retrieve performance counters from a Host System within the VMware Infrastructure.
Settings
Name
Type
Value
Username
string
root
Password
password
VMware webservice URL
string
https://<vCenter>/sdk/vimService
Hosts
VMHost
The URL to access the VIM webservice on the VMWare host(s). This setting should only be changed if the VMWare Server(s) are using a non-standard web service URL, configure the target host(s) below.
OK
Cancel



- If you want to monitor hosts individually, click the **Add Hosts** button to add them to the **Host** section at the bottom of the **Scheduled Monitor Editor** dialog box. AppMon comes with a predefined **VMHost** host group. Use the host group in the Monitor and add hosts to that host group using the infrastructure management facility.

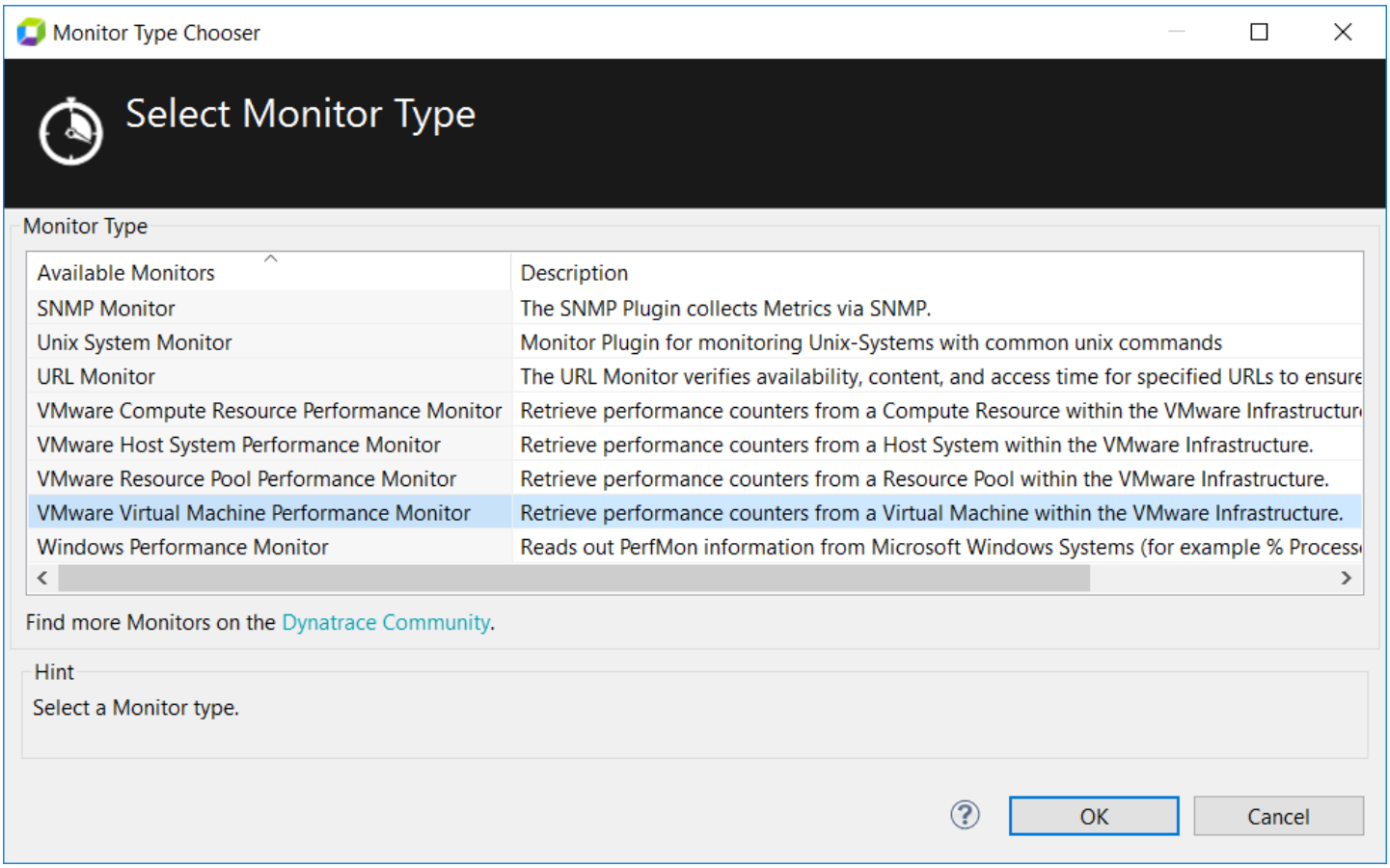
Now you can activate the Monitor and get periodic measures for all of your vHosts. VMware typically has a granularity of 20 seconds, so it does not make sense to have a schedule lower than this. Scheduling once per minute should be sufficient.

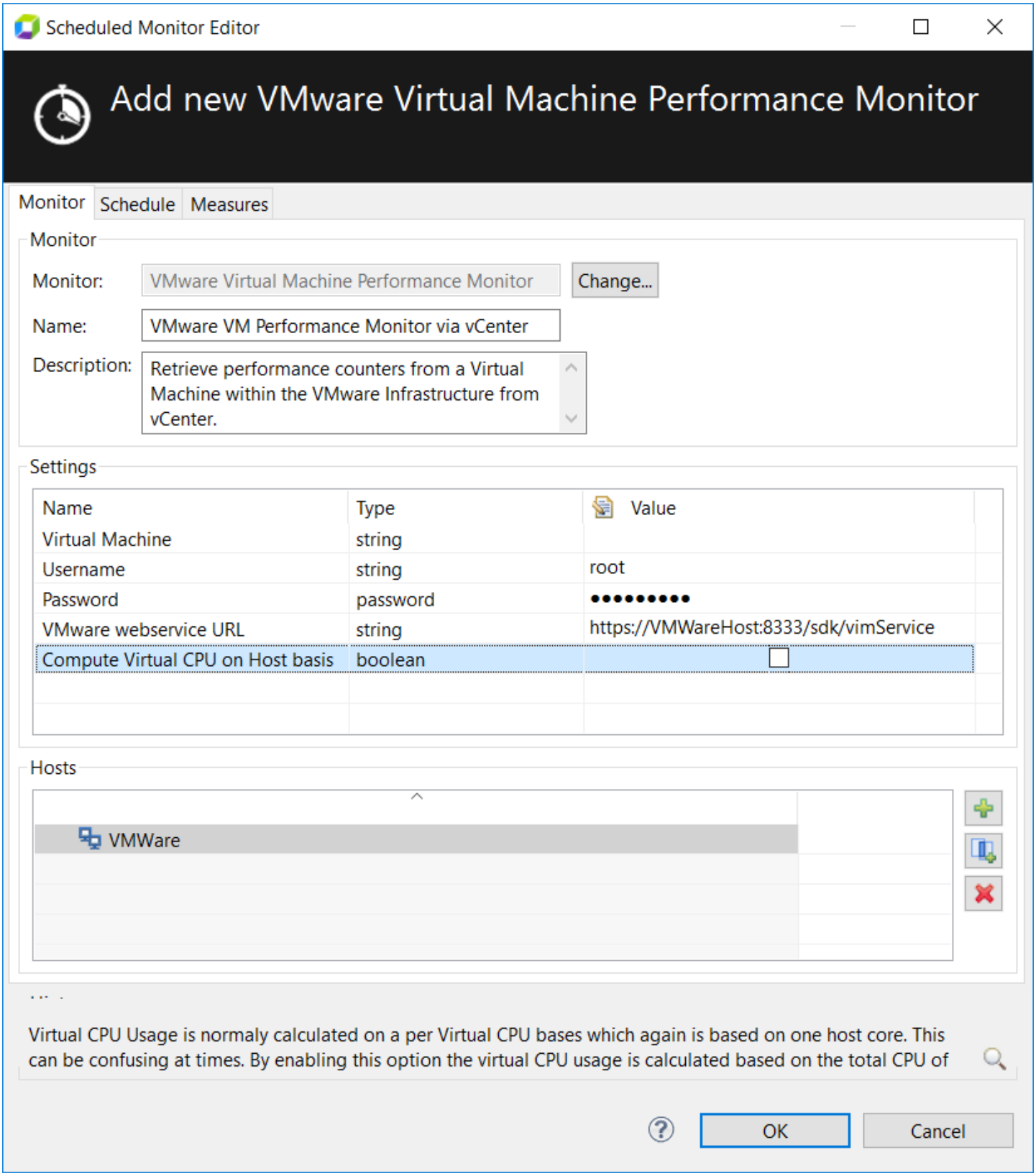
Monitoring virtual machines

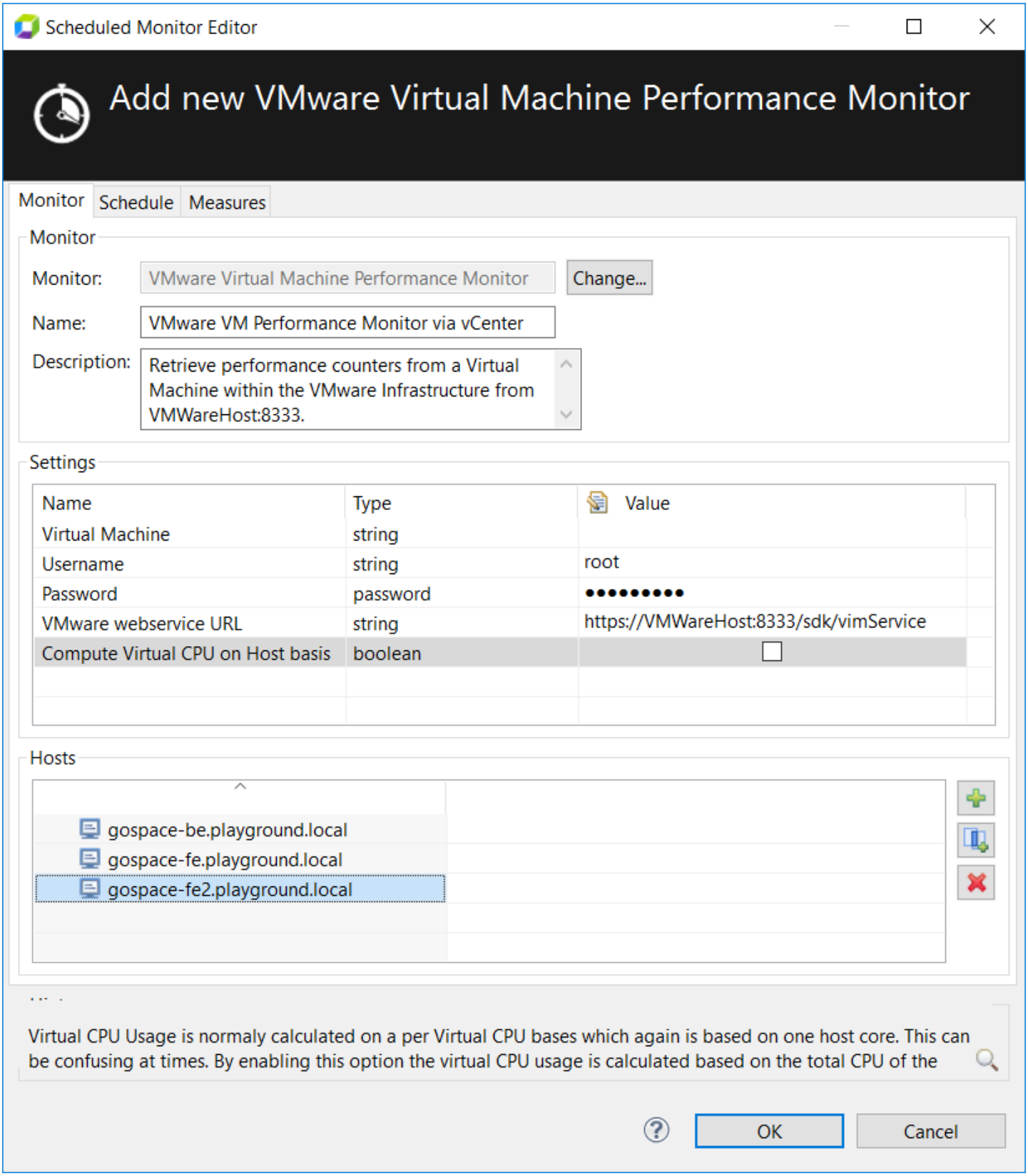
If you use vCenter, simply monitoring the virtual hosts to monitor virtual machines. Use the **vCenter** host in the URL, as described in Monitoring Virtual Hosts. Add the hosts that represent the virtual machines to the host section.

AppMon monitors all hosts designated as virtual machines and gets respective performance counters from vCenter. Use the predefined **VMware** host group in the Monitor and add virtual machines to that host group.

If you can't use vCenter to monitor all hosts, define a separate monitor for each physical vHost that hosts virtual machines. In this case, configure the URL with the real host name of the physical host. Add the virtual machines that are hosted to the host section.




Scheduled Monitor Editor
Add new VMware Virtual Machine Performance Monitor
Monitor
Schedule
Measures
Monitor
Monitor:
VMware Virtual Machine Performance Monitor
Change...
Name:
VMware VM Performance Monitor via vCenter
Description:
Retrieve performance counters from a Virtual Machine within the VMware Infrastructure from vCenter.
Settings
Name
Type
Value
Virtual Machine
string
Username
string
root
Password
password
VMware webservice URL
string
https://VMWareHost:8333/sdk/vimService
Compute Virtual CPU on Host basis
boolean
Hosts
VMWare
Virtual CPU Usage is normaly calculated on a per Virtual CPU bases which again is based on one host core. This can be confusing at times. By enabling this option the virtual CPU usage is calculated based on the total CPU of
OK
Cancel


Scheduled Monitor Editor
Add new VMware Virtual Machine Performance Monitor
Monitor
Schedule
Measures
Monitor
Monitor:
VMware Virtual Machine Performance Monitor
Change...
Name:
VMware VM Performance Monitor via vCenter
Description:
Retrieve performance counters from a Virtual Machine within the VMware Infrastructure from VMWareHost:8333.
Settings
Name
Type
Value
Virtual Machine
string
Username
string
root
Password
password
VMware webservice URL
string
https://VMWareHost:8333/sdk/vimService
Compute Virtual CPU on Host basis
boolean
Hosts
gospace-be.playground.local
gospace-fe.playground.local
gospace-fe2.playground.local
Virtual CPU Usage is normaly calculated on a per Virtual CPU bases which again is based on one host core. This can be confusing at times. By enabling this option the virtual CPU usage is calculated based on the total CPU of the
OK
Cancel

Notes

- You must install guest tools for VMware Management to know the virtual machine host name. Without guest tools installed, you must configure the virtual machine name and use a separate Monitor for each virtual machine.
- If you have guest tools installed, never use the `Virtual Machine` parameter. Instead, use the host section to identify the virtual machines to monitor.

Interpreting VMware measures

The following links show performance counters and categories for each counter data object type (VMware web site):

- CPU Performance
- CPU Resource Utilization
- Memory Performance
- Network Performance
- Disk Performance
- System Performance
- Cluster Services Metrics

For more information on troubleshooting, see Datacenter Administrator Guide.

Applications in a virtual machine

A virtual machine does not run exclusively on a host. It has to share its resources with other virtual machines. This may cause the virtual machine to suspend at any time. The guest OS is usually not aware of this which can cause a timekeeping problem.

As a result, the timings captured in a virtual machine are distorted. The extent of the distortion depends on the following in the guest operating system:

- Response times may appear to be less than they actually are. This is called *apparent time* and means that the time that has passed for the system is less than in reality because it was suspended in between.

- CPU Times might be more or less than they actually are. The reason for this is the same as above. This is a problem particularly on older Linux and UNIX systems; and nearly all Windows systems. This is due to tick based timing.

To fix this, do one of the following:

1. Use an operating system with a tick-less timer. Nearly all new Linux and UNIX Systems use tick-less timers. This reduces both timing problems. For more information, look at the associated documentation.
2. Use the pmc0x1001 Agent Timer. This instructs AppMon to capture the real elapsed time from the VMware infrastructure. This leads to more accurate response times at the cost of a higher overhead. Look in the **Agent** log via the **System Information** dashlet to see the amount of overhead.

The default timer:

```
2010-10-13 09:26:56 info    [java  ] Calibrating overhead calculations...
2010-10-13 09:26:59 info    [java  ] Reference method has a mean CPU execution time of 21
2010-10-13 09:26:59 info    [java  ] Getting CPU time of current thread costs 1714ns
```

The PMC timer:

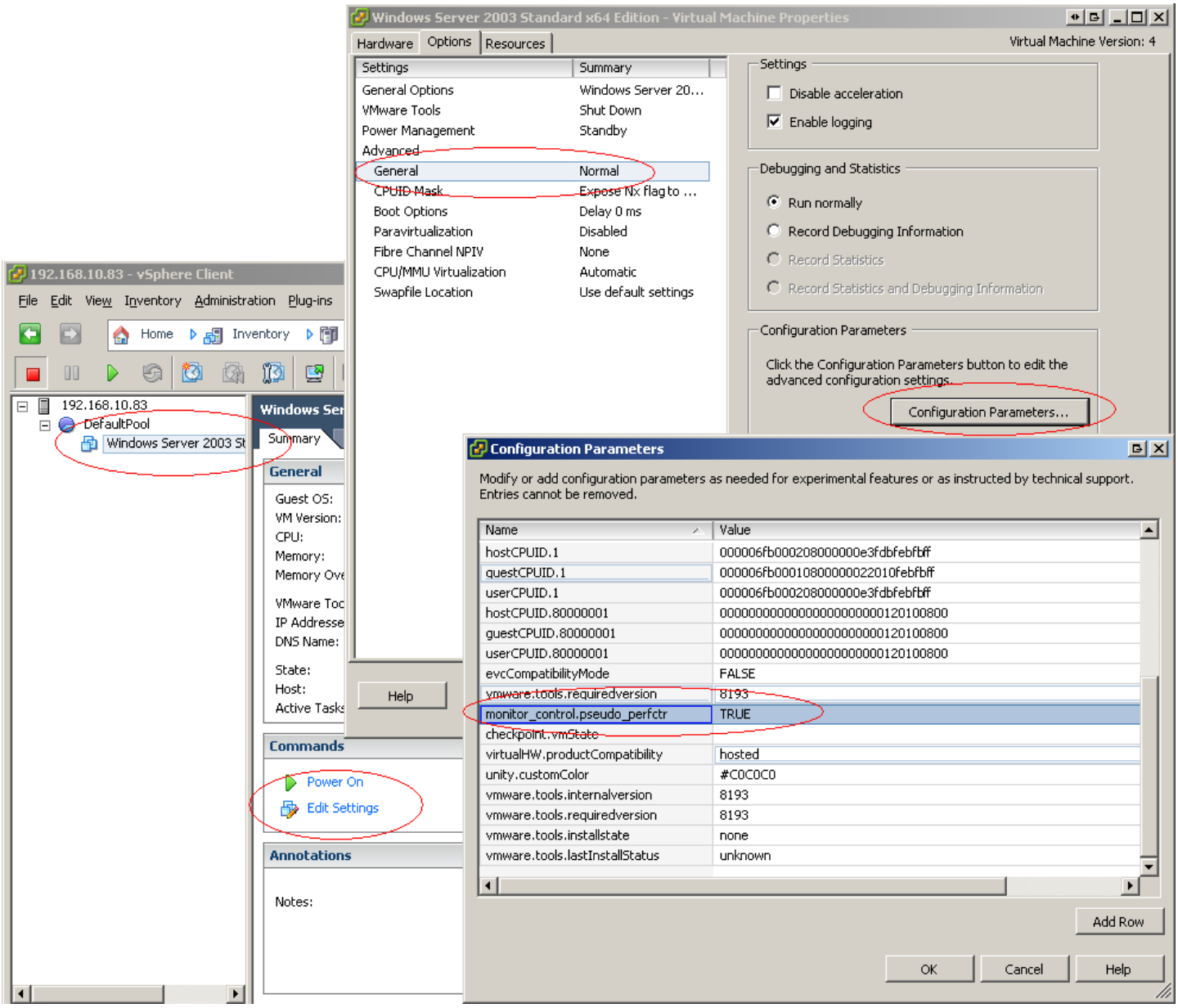
```
2010-10-13 09:28:09 info    [java  ] Calibrating overhead calculations...
2010-10-13 09:28:12 info    [java  ] Reference method has a mean CPU execution time of 44
2010-10-13 09:28:12 info    [java  ] Getting CPU time of current thread costs 1723ns
```

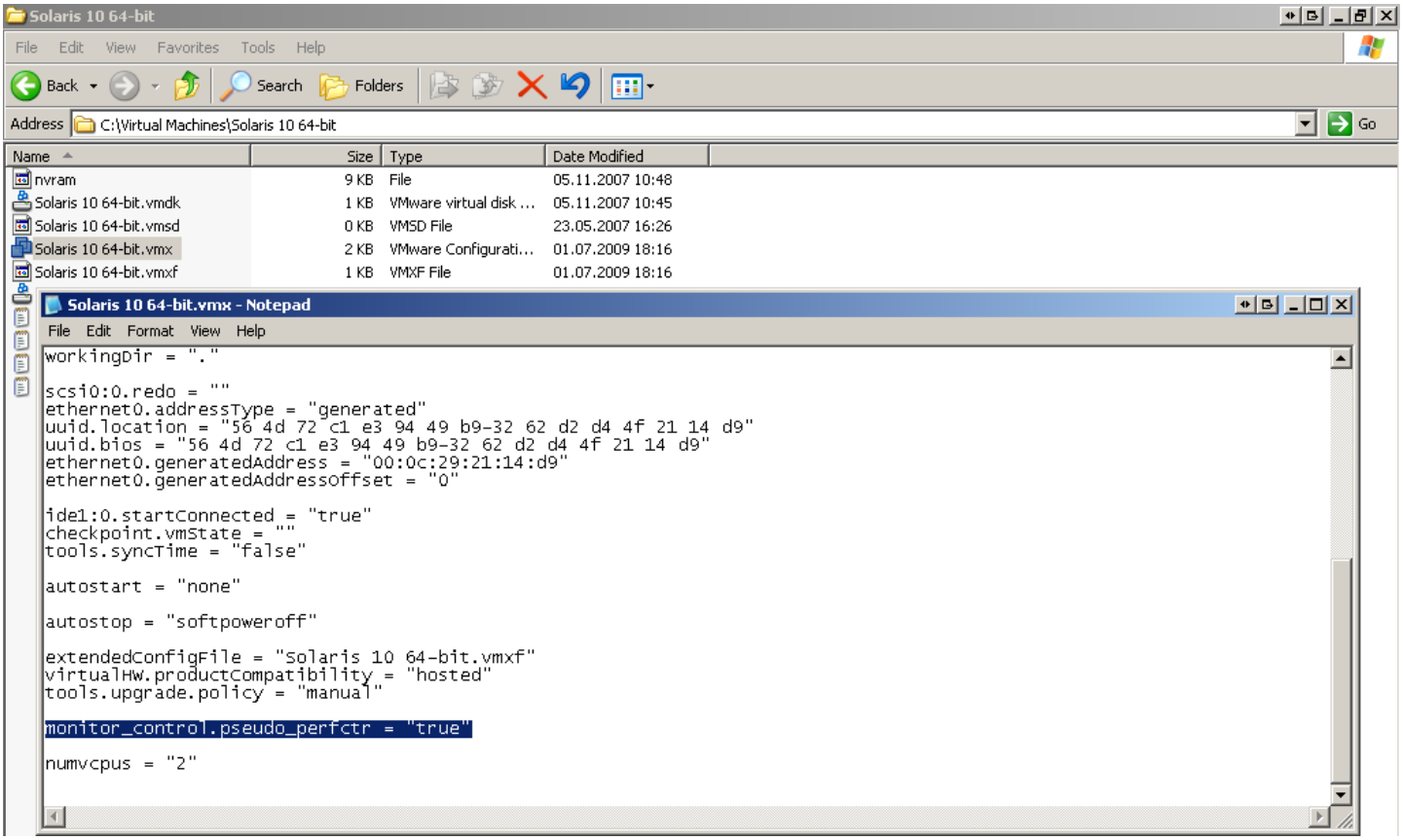
Both of these logs are from the same virtual machine and Java application. The overhead per instrumented method increased from roughly 2000ns to ~4500ns because of the call to the hypervisor.

> **Note**
>
> This demonstrates that the overhead of a PMC timer is higher than the default one. The timings here are not a general baseline for AppMon overhead. This is very specific to the underlying hardware and guest operating system, and was taken in a VM.

To use the PMC Agent timer, you must activate the feature for the virtual machine itself. These special timers must be enabled on the guest instance by setting `monitor_control.pseudo_perfctr = TRUE` for the instance. Depending on the VMware product you use, do this in the management interface or manually edit the instance's .vmx file.

Supported systems include VMware ESX and ESXi 3.5+, VMware Server 2.0+ and VMware Workstation 6.5+. Other systems may support these timers as well.





Running AppMon in a virtual environment

In addition to the items in the Deployment Guide, there are requirements to successfully run AppMon in a virtual environment.

AppMon Server in a virtual machine

The AppMon Server is very demanding on the hardware and periphery. It operates well within a virtualized environment if you follow these rules:

- Memory
  VMware may cause you to over-commit on memory. This means that the sum of memory assigned to virtual machines can exceed the physical memory available on the host. This is managed via memory ballooning. The result is that a virtual machine may not be able to use its assigned memory and may fall back to swapping.
  Swapping is devastating. You must ensure that the AppMon Server itself is never swapped. To ensure that, you must explicitly reserve enough memory for the virtual machine to accommodate for the whole AppMon Server, as well as the necessary parts of the guest OS in physical memory. You can verify this by monitoring the ballooning memory of the AppMon Server's VM. There must be no ballooning.

- Virtual CPUs
  VMware generally states that you should run VMs with a single virtual CPU, unless the running software takes special advantage from multiple CPUs. AppMon does take advantage of multiple CPUs and cores. You should configure the VM with multiple virtual CPUs. You will get the best results if the number of virtual CPUs and cores equals the number of physical cores.
- Disk
  AppMon writes a lot of data to disk if you use session recording. Make sure that the physical disk and the interface underlying the assigned virtual disk can accommodate the load. In addition to the normal self-monitoring, you must monitor the AppMon Server performance via the **VMware - Monitoring** dashboard. Ensure that there is no memory ballooning and that the CPU ready time does not exceed 1000ms. (1 second per 20s interval).

AppMon Collector in a virtual machine

The Collector has similar demands on the virtual environment as the AppMon Server, but does not use a lot of disk. Make sure you meet the demands of CPU and network memory.

Web transaction monitor

Since AppMon 5.5 the **Web Transaction Monitor** plugin that shipped with the product is no longer officially supported, so the content of this page has been moved to Web Transaction Monitor Plugin.

Windows Performance Monitor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Basic Configuration

Select host(s) that should be the target(s) of the monitor

Customizing the execution schedule

Centrally deploying a monitor to different global locations

Authentication and permissions

Measure Configuration

Overview

Through the Windows Performance Monitor, AppMon monitors host system parameters that are running a Windows-based operating system.

Most other Measures are calculated by AppMon. Those retrieved by the Windows Performance Monitor are not retrieved by an AppMon Agent. They are directly calculated by the Monitor. Performance Counters is available on most versions of Windows starting with Windows 95 and Windows NT 4.0. Different Measures may be available for different operating systems.

Basic Configuration

To add a new Windows Performance Monitor to a System Profile follow these steps:

1. Open the **System Profile** preferences.
2. Select the **Monitors** tab.
3. Click **Create > Windows Performance Monitor**.

4. Specify a name and a description for the Monitor and configure the settings as described below.

Translation

The **Translation** check box enables the Performance Counter name translation. Select this check box if you notice missing Measures due to some target machines running a non-English language version of Windows.

Select host(s) that should be the target(s) of the monitor

A Monitor can be configured for multiple target hosts. Only one configuration is necessary; it is executed for all configured machines. This is especially useful in cluster environments that feature many similarly configured hosts.

Add **Single hosts**, **hostgroups**, or **sites** by clicking the **+** icon. Create a new host if the desired host is not shown. If you add a hostgroup or site, the Monitor will be run for all contained hosts.

Optionally, add an **intersection** of hostgroups or sites. The Monitor will be run for all hosts that belong to all selected groups or sites. If the configuration would resolve a single host more than once (if Host1 is also contained in Hostgroup1), the Monitor is only executed once per host.

You can also add target hosts via the Server REST interface. The required POST request is located in the **Tasks and Monitors** group. The request documentation is based on the Swagger specification and embedded to the request form.

For more information how to manage hosts and hostgroups, see Infrastructure.

Examples

Configured Hosts As Shown in Hosts Table	Monitor Will Be Run for…
Host1	Host1
Host1 Host2	Host1 and Host 1
Hostgroup 1 Host2	All hosts from Hostgroup 1 and also Host 2

Hostgroup 1 Hostgroup 2	All hosts from Hostgroup 1 and also Hostgroup 2
Hosts belonging to Hostgroup 1, Hostgroup 2 Host 1	All hosts that are **both** in Hostgroup 1 and Hostgroup 2 (an intersection of Hostgroup 1 and Hostgroup 2), plus Host 1

Note

Use the actual host name on Windows Vista instead of *localhost*.

Customizing the execution schedule

Open the **Schedule** tab, then choose a predefined schedule or create your own schedule if none fits. Any schedules you create will be available for the whole System Profile.

Centrally deploying a monitor to different global locations

Open the **Schedule** tab, then select the desired AppMon Collector or AppMon Server from the **Execute on** list. Only currently connected AppMon Collectors are shown in this list. The selection appears as **Embedded Dynatrace Collector**.

Note

If an AppMon Collector that is connected through a slow network or a Monitor that contains large library files is run on a remote machine, it is possible, though rare, that the first execution fails with a Missing Binaries error. This error should disappear on the next scheduled run of the Monitor.

Note

Execute this Monitor on an AppMon Server or AppMon Collector that runs on Windows.

Authentication and permissions

Remote retrieval of Performance Counter data requires authentication to succeed. This means that the AppMon Collector service that runs the Monitor must run as a user that can query Performance Counters on the target machine. For more information, see How To Retrieve Performance Metrics with Windows Performance Monitor.

Measure Configuration

Select the **Measures** tab to review the Measures currently subscribed by this Monitor. Click **Add** to add other pre-configured Measures. You can edit a newly added Measure to match a desired performance counter. Note that the parameters are case-sensitive.

How to configure additional metrics

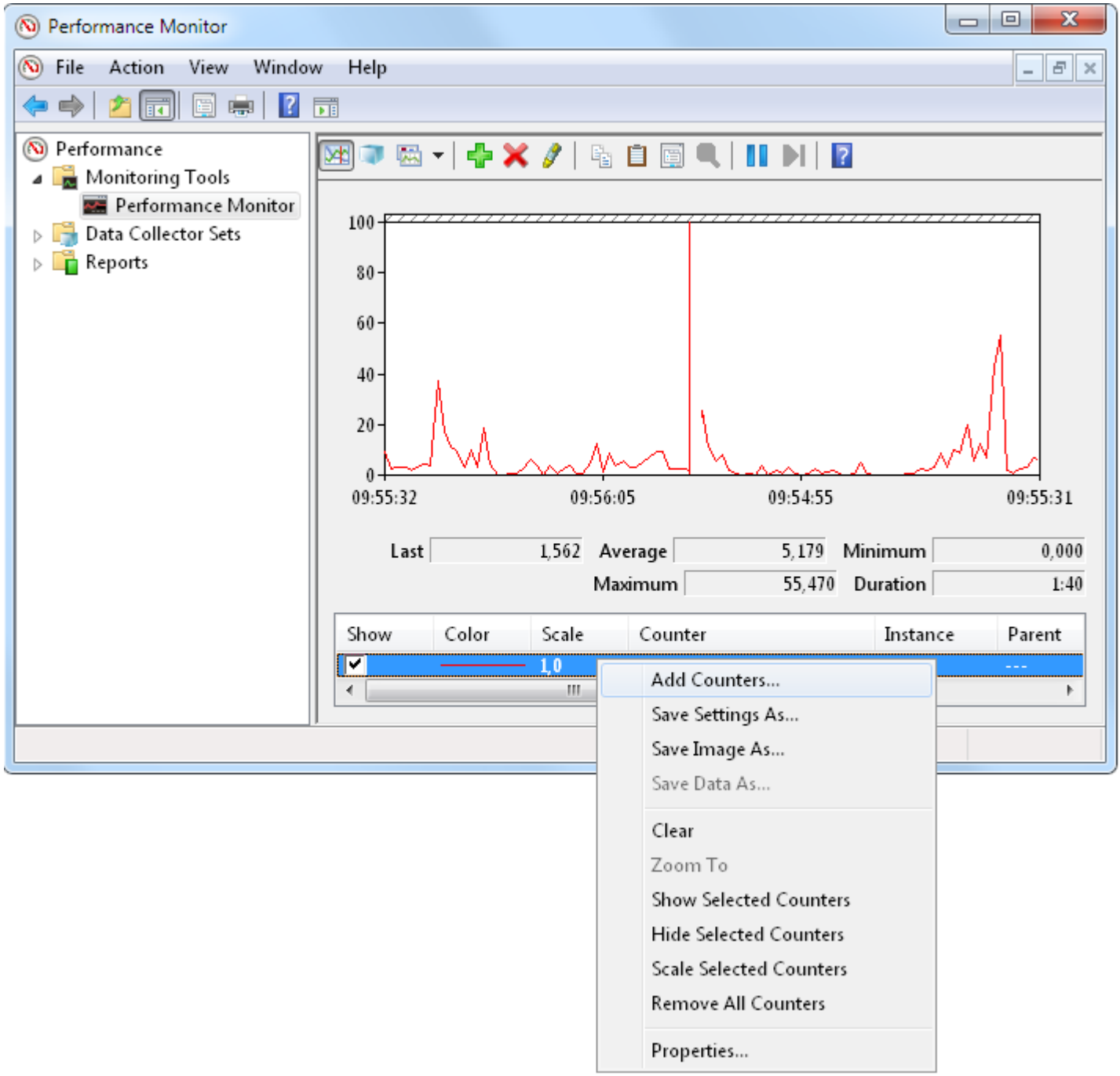
To add Measures to a Monitor that is not predefined, you must define a matching metric first. To do this, follow these steps:

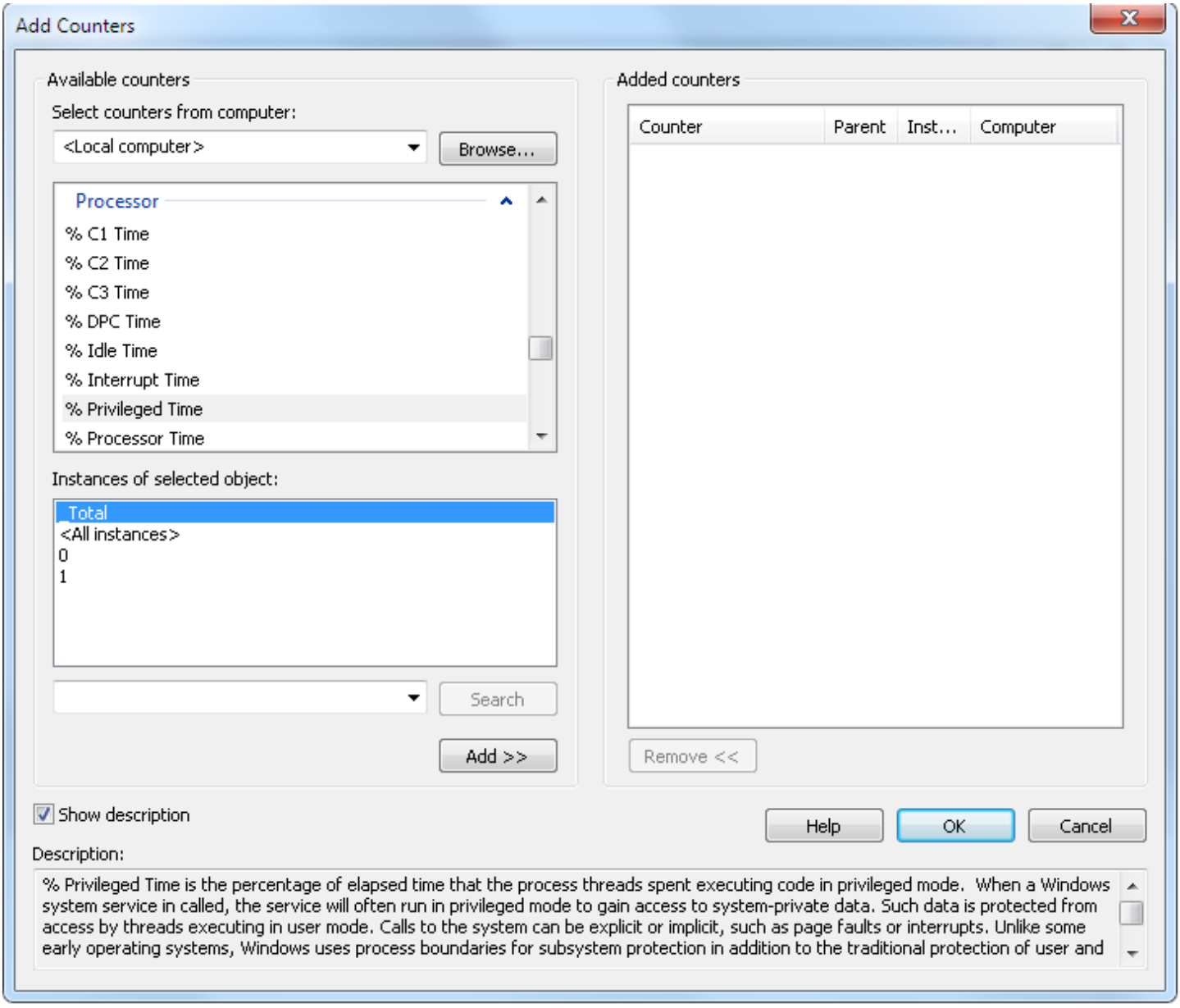
1. From the **Application** menu, select **Tools > Manage Plugins**.
2. Click the **Development** tab.
3. Click **New**.

Follow the wizard to create a Metric Group Plugin. Click **Browse** to select the **Windows Performance Monitor** . Configure the desired Metric Groups and Metrics. Configure each Metric to match one of the desired Windows Performance Counters. Finish the wizard, and build and deploy the Metric Group Plugin to the AppMon Server.

More information about the PDE, see Developing a User Plugin.

To find out which Performance Counters are available, run the command `perfmon` on the target machine. Right-click on the list of configured Performance Counters to open the** Add Counters** dialog. Use this dialog to browse the available performance objects, the available counters per object and their instances. Select **Explain** to show an additional dialog that contains information about the currently selected Performance Counter, if available.





Measures

The following measure can be subscribed:

- Windows CPU Usage
  - Processor Time - The percentage of elapsed time that the processor spends to execute a non-idle thread. It is calculated by measuring the duration of the idle thread that is active in the sample interval, and subtracting that time from interval duration. Each processor has an idle thread that consumes cycles when no other threads are ready to run. This counter is the primary indicator of processor activity, and displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time that the service is inactive, and subtracting that value from 100%.
  - User Time - The percentage of elapsed time the processor spends in the user mode. User mode is a restricted processing mode designed for applications, environment subsystems, and integral subsystems. The alternative, privileged mode, is designed for operating system components and allows direct access to hardware and all memory. The operating system switches application threads to privileged mode to access operating system services. It displays the average busy time as a percentage of the sample time.
  - Idle Time - The percentage of time the processor is idle during the sample interval.
- Windows Memory Usage
  - Free Memory - The amount of physical memory, in bytes, available to processes currently running. It is calculated by adding the amount of space on the Zeroed, Free, and Standby memory lists. Free memory is ready for use; Zeroed memory consists of pages of memory filled with zeros to prevent subsequent processes from seeing data used by a previous process; Standby memory is memory that has been removed from a process' working set (its physical memory) on route to disk, but is still available to be recalled. It displays the last observed value only; it is not an average.
- Windows Disk Usage
  - Custom Disk Free Space - The percentage of total usable space on the configured logical disk drive that is free. Configure this with a disk drive letter.
- Windows Network Usage
  - Custom Network Bytes Received/sec - The rate at which bytes are received over a single network adapter, including framing characters. Configure this with the exact interface name to monitor.
  - Custom Network Bytes Sent/sec - The rate at which bytes are sent over a single network adapter, including framing characters. Configure this with the exact interface name to monitor.
  - Custom Network Bytes Total/sec - The rate at which bytes are sent and received over a single network adapter, including framing characters. Configure this with the exact interface name to monitor.

- Windows Performance Monitor
  - Process Count - The number of processes at the time of data collection. This is an instantaneous count, not an average over the time interval.
  - Process Memory Usage - The current size, in bytes, of the working set of all processes. The working set is the set of memory pages touched recently by the threads in the process. If free memory is above a threshold, pages are left in the working set of a process even if they are not in use. When free memory falls below a threshold, pages are trimmed from working sets. If they are necessary, they are soft-faulted back into the working set before leaving the main memory.
  - Run Queue, current - The number of threads in the processor queue. It shows ready threads only, not running threads. There is a single queue for processor time, even on computers with multiple processors. Therefore, if a computer has multiple processors, divide this value by the number of processors servicing the workload.
  - Pages/Sec - The average number of pages faulted per second. It is measured by the number of pages faulted per second because only one page is faulted in each fault operation. This is also equal to the number of page fault operations. This counter includes both hard faults (those that require disk access) and soft faults (the faulted page is found elsewhere in physical memory.) Most processors can handle a large number of soft faults without significant consequence. However, hard faults, which require disk access, can cause significant delays.
  - Pages Input - The rate at which pages are read from disk to resolve hard page faults. Hard page faults occur when a process refers to a page in virtual memory that is not in its working set or elsewhere in physical memory, and must be retrieved from disk. When a page is faulted, the system tries to read multiple contiguous pages into memory to maximize the benefit of the read operation.
  - Pages Output - The rate at which pages are written to disk to free up space in physical memory. Pages are written back to disk only if they are changed in physical memory, so they are likely to hold data, not code. A high rate of pages output might indicate a memory shortage. Windows writes more pages back to disk to free up space when physical memory is in short supply.
  - Free Diskspace - The unallocated space, in megabytes, on the disk drive in megabytes. One megabyte is equal to 1,048,576 bytes.
  - Disk Reads/sec - The rate of read operations on the disk.
  - Disk Writes/sec - The rate of write operations on the disk.
  - PhysicalDisk Read - The rate at which bytes are transferred from the disk during read operations.
  - PhysicalDisk Write - The rate at which bytes are transferred to the disk during write operations.
  - LogicalDisk Free Percentage - The percentage of total usable space on all logical disc drives that is free.

Infrastructure monitoring walkthroughs

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

How to retrieve performance metrics with Windows Performance Monitor

Setting up application availability monitoring

Setting up infrastructure monitoring

Using custom JMX and PMI measures in charts

Using host and hostgroups for large system monitoring

Using host and process health monitoring

Walkthroughs

Infrastructure monitoring is the basis for all application performance monitoring. The underlying system must be available and healthy. To achieve this, monitor system metrics like CPU, memory, network and disk. If any of these is deficient, your response time will lag.

The infrastructure walkthroughs are designed to provide detailed instructions and examples for using AppMon to monitor and troubleshoot common system performance problems.

Setting up infrastructure monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Goal of this walkthrough

Scenario

Host monitoring of hosts - AppMon Agent installed

Host monitoring of hosts - AppMon Agent not installed

Monitor other system components

Use hosts and hostgroups

Analyze system monitoring data

Related pages

Infrastructure monitoring

Setting up application availability monitoring

Using host and process health monitoring

Using host and hostgroups for large system monitoring

Using custom JMX and PMI measures in charts

How to retrieve performance metrics with Windows Performance Monitor

Goal of this walkthrough

This walkthrough explains host and performance monitoring. AppMon uses AppMon Agents to query the data source and monitor your system health. Each AppMon Agent provides CPU, memory, network, and disk metrics for the host that the AppMon Agent runs on.

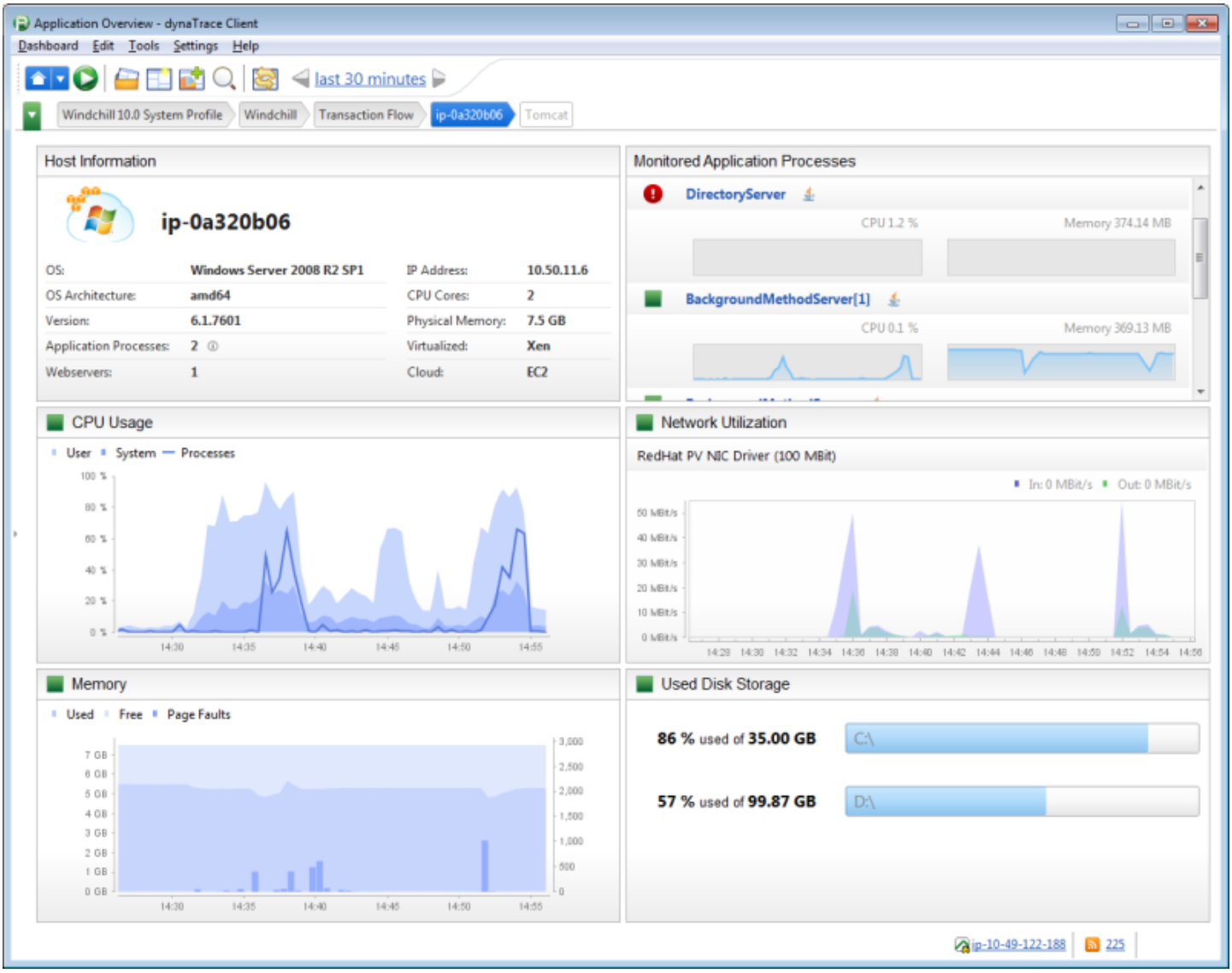
To monitor more hosts, install an AppMon Host Monitoring Agent on a machine; or simply configure a monitor that queries Windows Performance Counters, UNIX system Counters, SNMP, or any other data source.

Scenario

Install AppMon with several deployed AppMon Agents. Use the easyTravel demo application to learn about various AppMon features and functions.

Host monitoring of hosts - AppMon Agent installed

Built-in host monitoring provides all host metrics for you. You're all set.



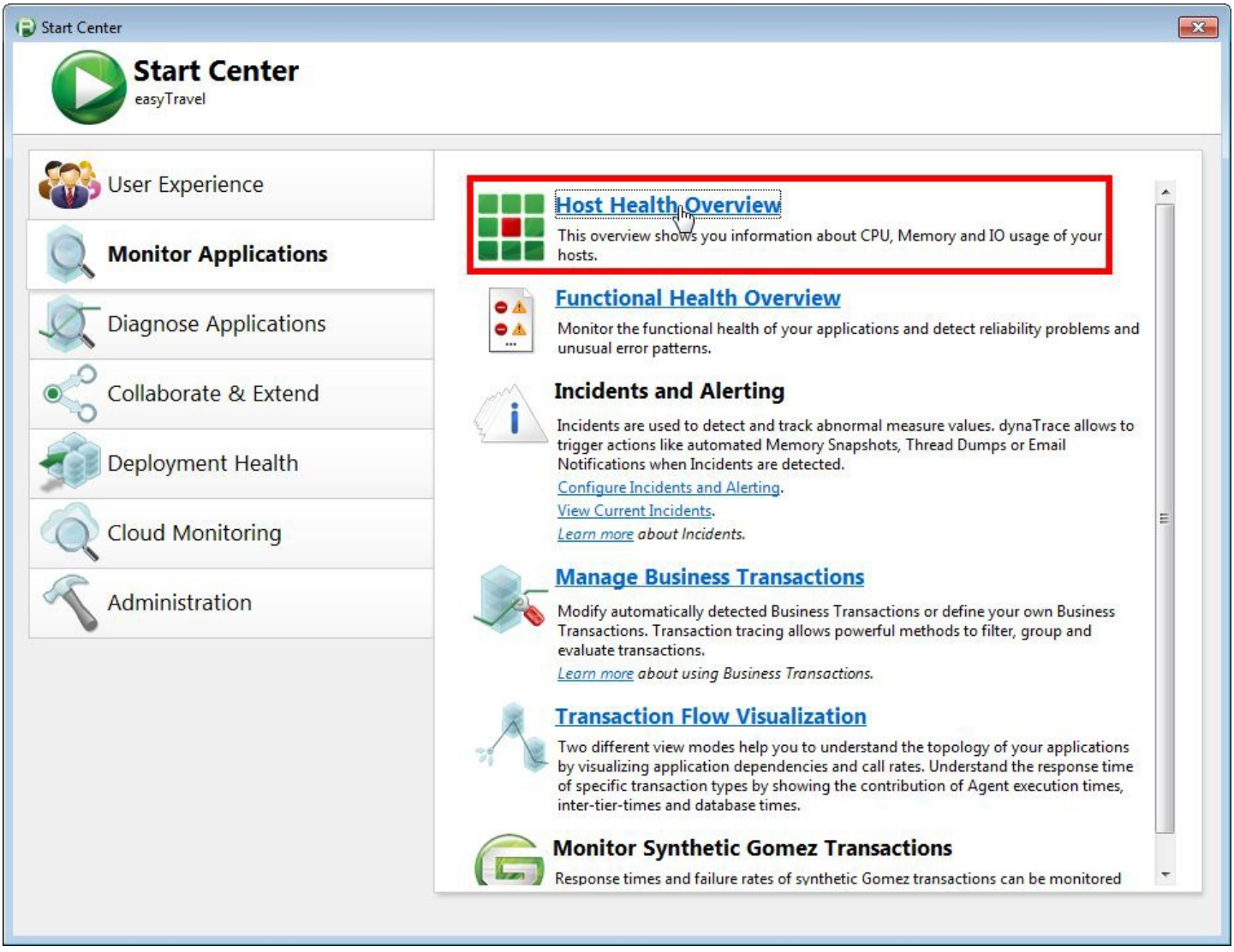
Host Monitor Main Window

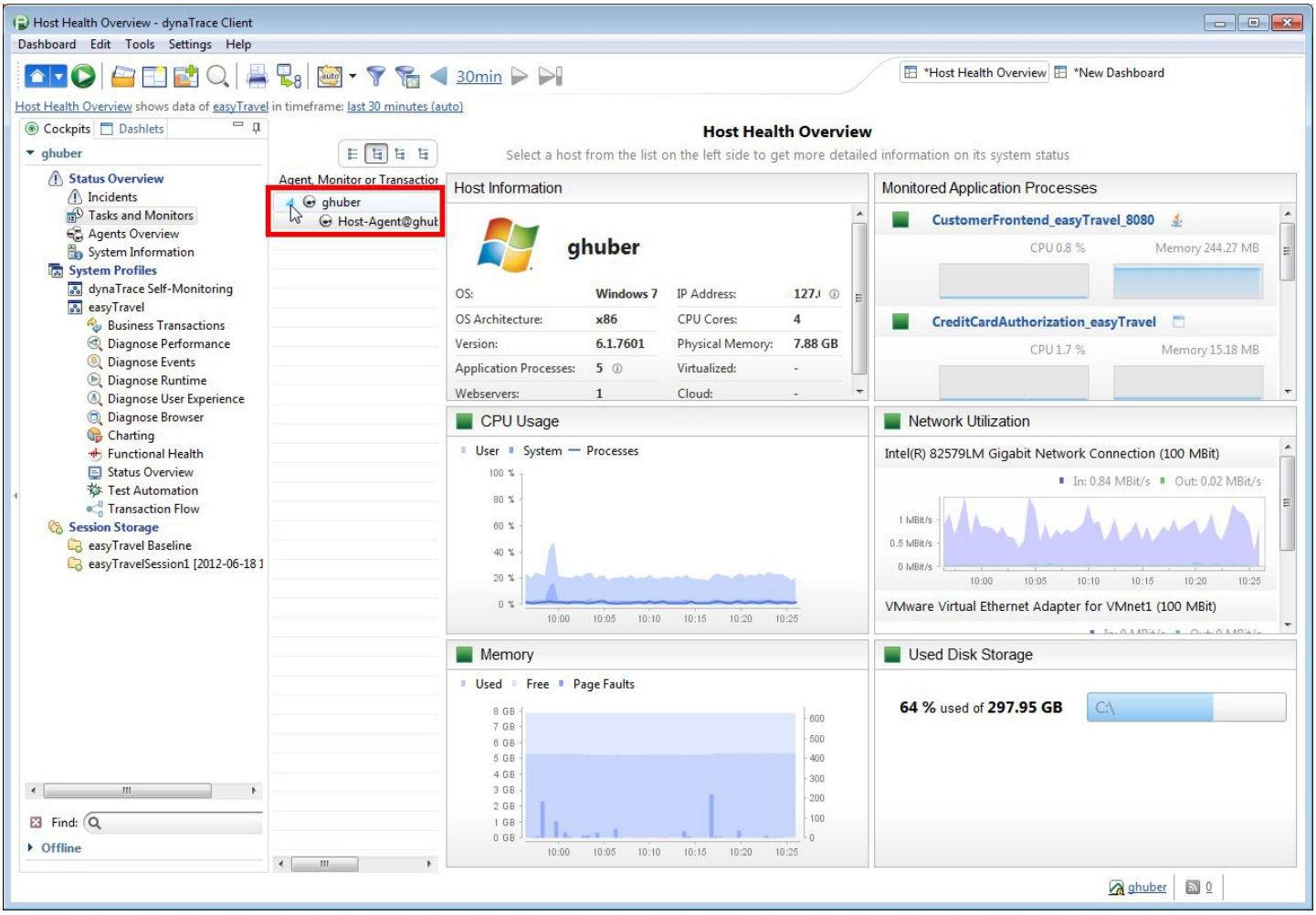
Host monitoring of hosts - AppMon Agent not installed

AppMon monitors your database server, backend server, and all other hosts. To configure an Agent on a machine, follow these steps:

1. Install an AppMon Host Monitoring Agent on a machine.
2. Configure this Agent to connect to your AppMon Server.
3. Map this to your System Profile.
4. After the configuration, you receive host Information.

This data is available in the **Host Health** dashboard. Access it through the **Start Center** or in the **Transaction Flow**, if the host is detected there.





Monitor other system components

Monitor more than just AppMon components. AppMon provides monitors which execute on a scheduled interval, and pull data from different data sources.

For a full list of out-of-the-box monitors, see Monitors.

To download additional monitors, see FastPacks and Plugins on the Dynatrace Community.

Monitor Windows systems

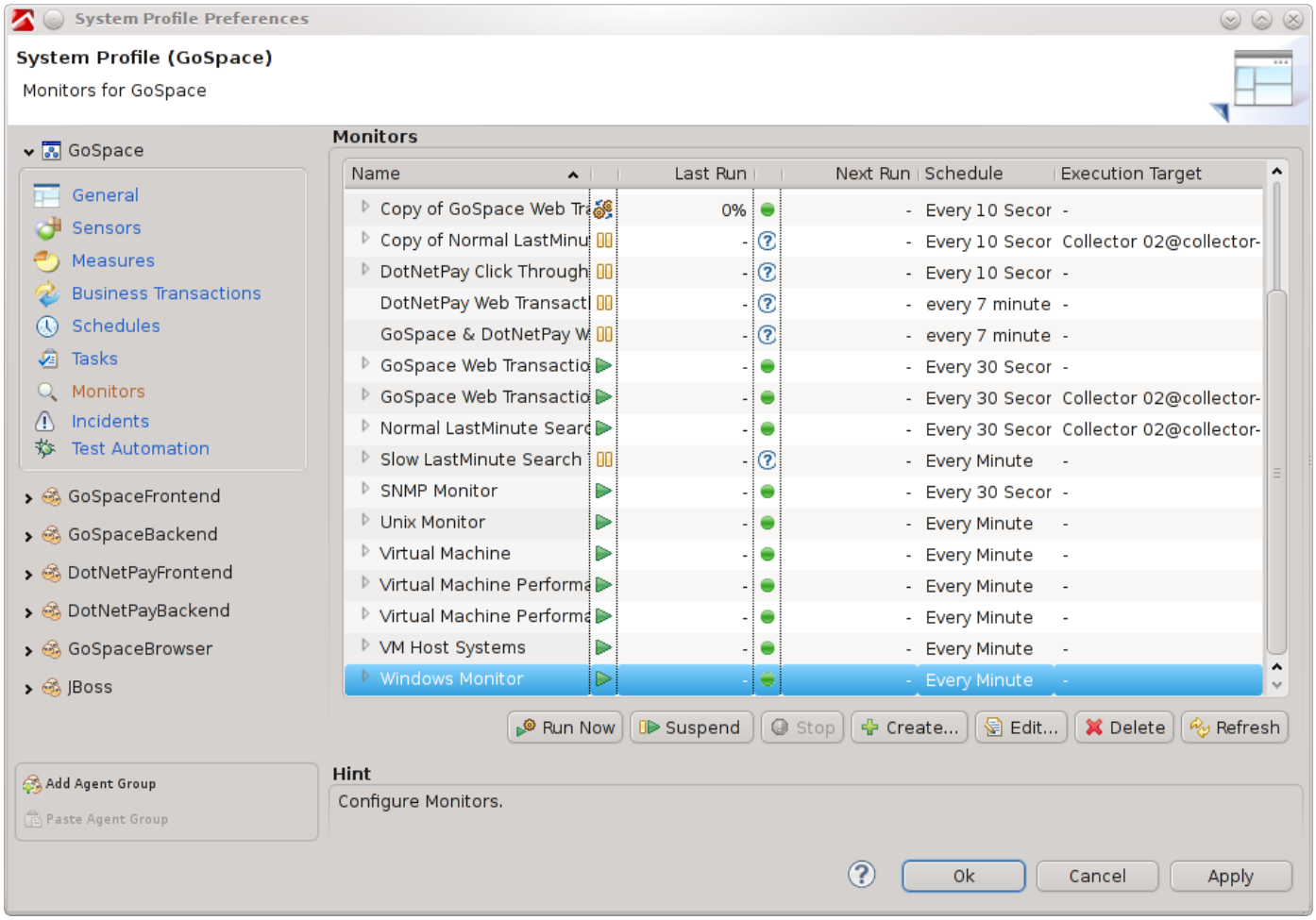
Check out the Windows Performance Monitor to query all Windows Performance Counters. To configure a Windows Performance Monitor, follow these steps:

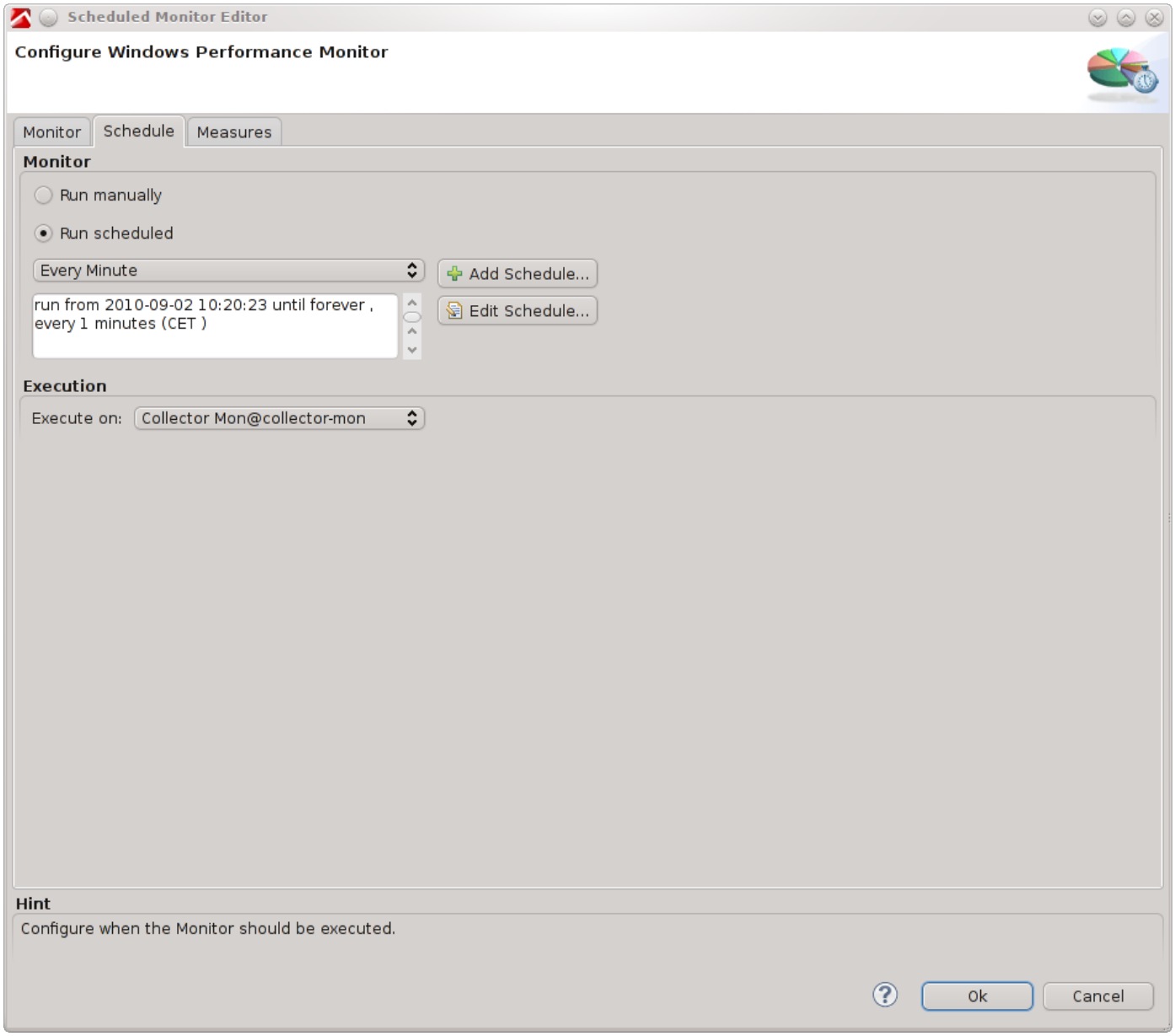
1. Run a Windows Collector service as a user with the necessary permissions. For more information, see How to Retrieve Performance Metrics with Windows Performance Monitor.

The Windows Collector service runs as LocalSystem user after the default installation. This service cannot open remote connections. Change this to a user with permission to open net use connections.

2. Schedule the Windows Performance Monitor to run on that Windows Collector service. Choose the default** Every Minute** schedule.

   Go to **System Profile > Monitor**. Edit the Windows Performance Monitor and choose your Windows Collector service in the **Schedule** dialog.





3. Save and enable the Windows Performance Monitor.

For more information, see How to Retrieve Performance Metrics with Windows Performance Monitor.

Monitor Unix systems

To configure a UNIX Monitor, follow these basic steps:

1. Make sure a collector service is running.
2. Schedule the UNIX Monitor to run on that collector. Choose the default **Every Minute** schedule. This works the same as the Windows Performance Monitor.
3. Run SSH on all of the machines that you monitor.

4. Configure the **User** and **Password** settings for SSH to a user with permission to connect to all machines that you will monitor.
5. Save and enable the UNIX Monitor. AppMon automatically monitors each host that contains an Agent.

Add other machines

1. Select the **Server** item in the **Settings** menu.
2. Go to **Infrastructure**.
3. Add machines underneath the **Host** tab.
   The host must be in the hostgroup for its operating system. For example, a Windows machine must be in the Windows hostgroup and a Linux machine must be in the Linux hostgroup.

For more information, see UNIX System Monitoring.

Use hosts and hostgroups

If you use Windows Performance Monitor, or something similar, see the walkthrough Using Host and Hostgroups for Large System Monitoring.

Analyze system monitoring data

The data that the AppMon Agents capture is available in the **Host Monitoring** dashboard, as shown above. These values are also available as regular Measures that you can add to any custom dashboard. You can also add Measures that are captured through Windows, UNIX, SNMP or Oracle monitors. After you configure a monitor, and it executes, use the Measures in a chart or to configure incidents.

Setting up application availability monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is availability monitoring?

Goal of this walkthrough

Detailed steps

Availability monitoring for multiple Agent Groups

Related pages

Setting up infrastructure monitoring

Using host and process health monitoring

Using host and hostgroups for large system monitoring

Using custom JMX and PMI measures in charts

How to retrieve performance metrics with Windows Performance Monitor

Synthetic Monitoring integration

What is availability monitoring?

Availability monitoring connects your customers to their business services 24/7.

Goal of this walkthrough

This walkthrough explains how to implement availability monitoring with AppMon. Availability monitoring uses a URL or web transaction monitor so that your applications and services perform without a hitch. It also detects components that are disconnected.

Scenario

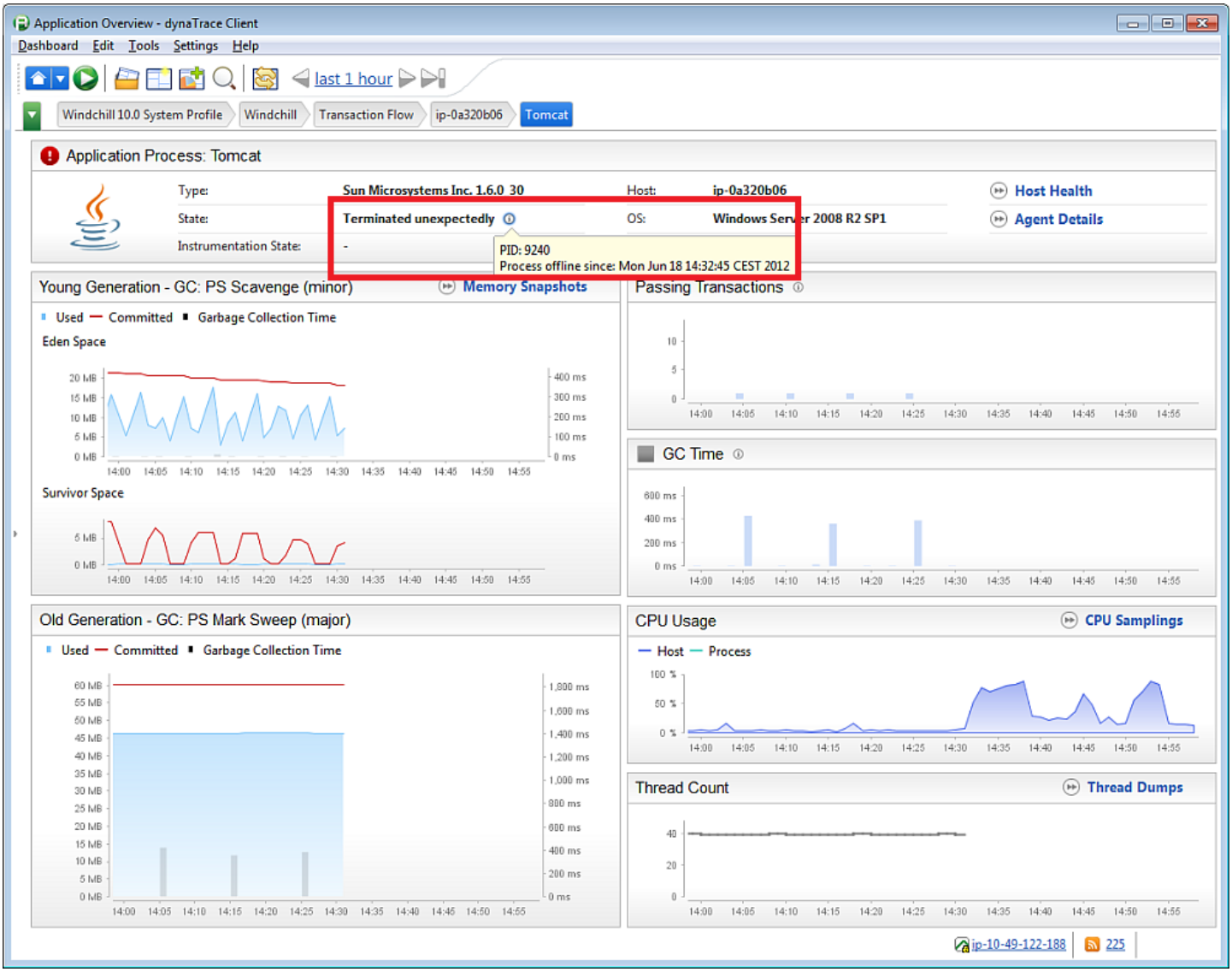
Use the easyTravel demo application to learn about various AppMon features and functions. If customers can't access the easyTravel web page, AppMon sends an email to the administrator.

Requirements

- dynaTrace Installation
- Standard Demo Applications - easyTravel installation

Detailed steps

AppMon automatically monitors all JVMs and CLRs that are connected through an Agent. It removes those that are unavailable for 72 hours.
Data appears in the** Transaction Flow ** dashlet, and in the **Process Health** dashboard. For more information, see Process Health Monitoring.



Process Health Dashboard

AppMon triggers an incident when an active application is not available. To see the incidents, look at the **Incidents dashlet**.

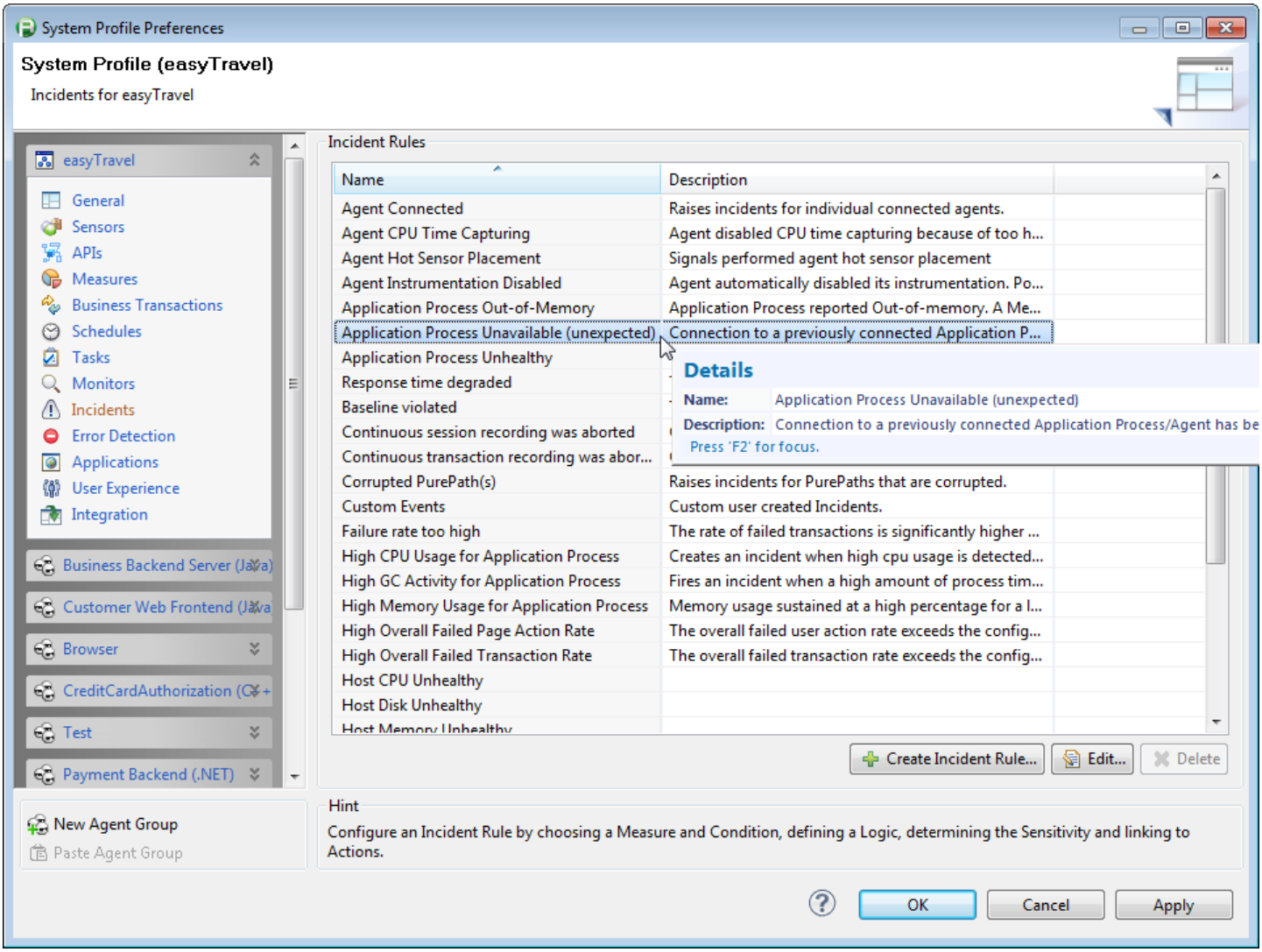
To see the running Agents, look at the **Agents Overview Dashlet** dashlet.

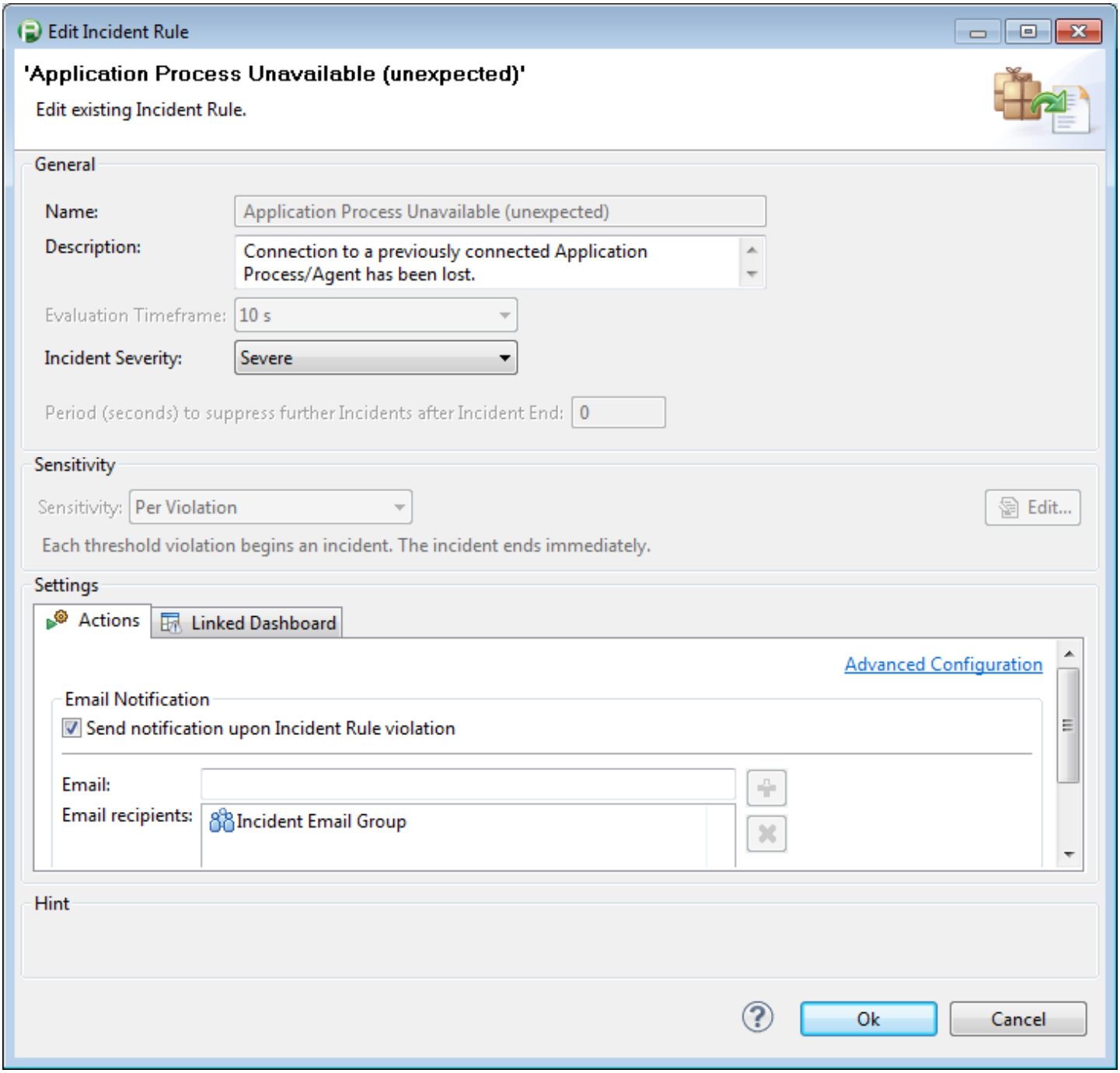
AppMon triggers an *Application Process Unavailable (unexpected) Severe* incident if a JVM or CLR fails. The **Agent Availability** icon then turns red.

To remove an Agent from your system, remove it from the **Offline Agents** section of the **Agents Overview Dashlet** dashlet so you don't trigger an incident.

Unavailability of agents email alert

If a JVM or CLR fails, a notification is sent to the Incident Email Group. To configure the email, go to **System Profile > Incidents** tab. Edit the **Application Process Unavailable (unexpected)** setting.





To test the notifications, stop easyTravel or one of its subsystems, and you will receive an email.

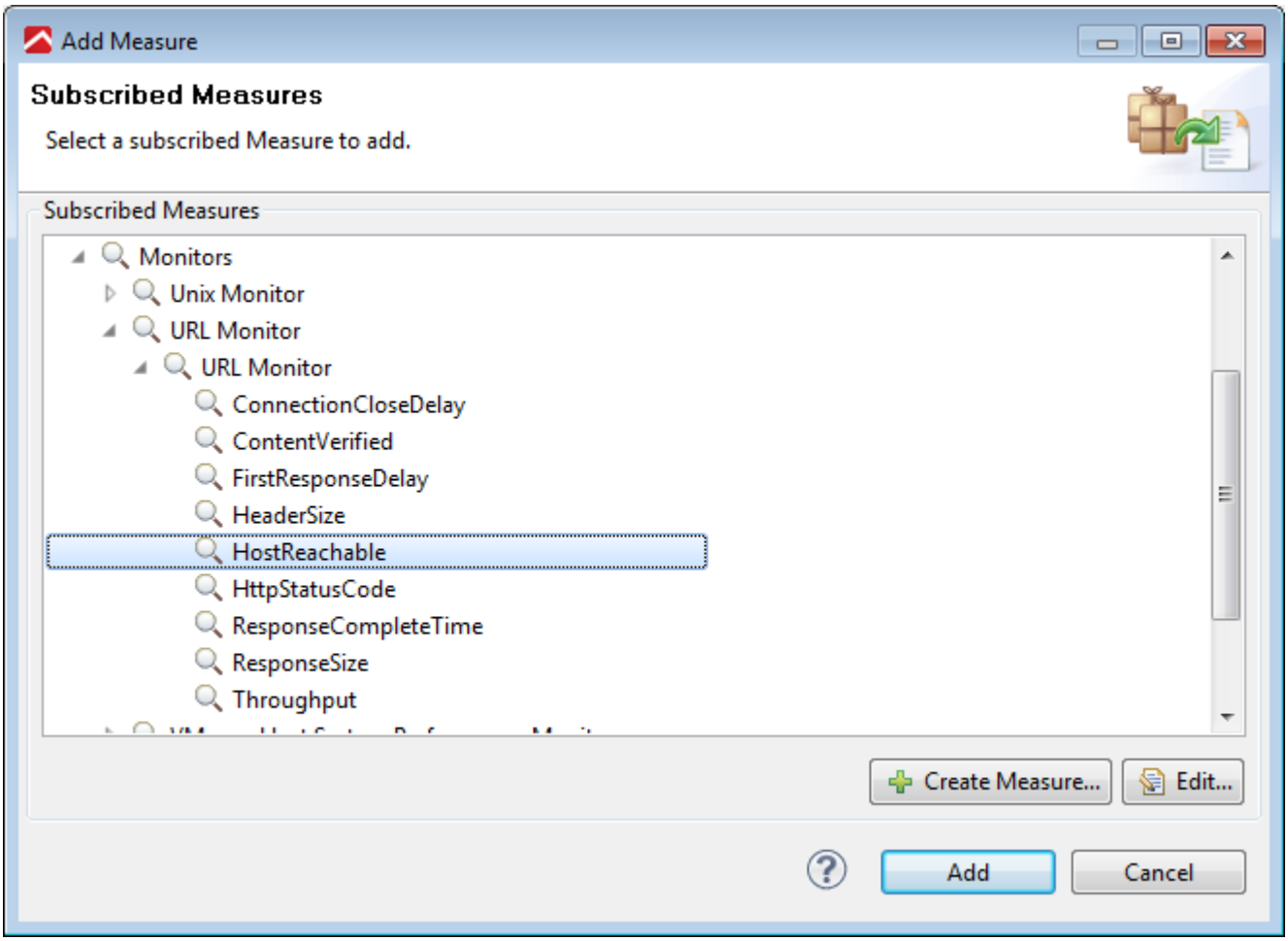
Dynatrace Synthetic Monitoring

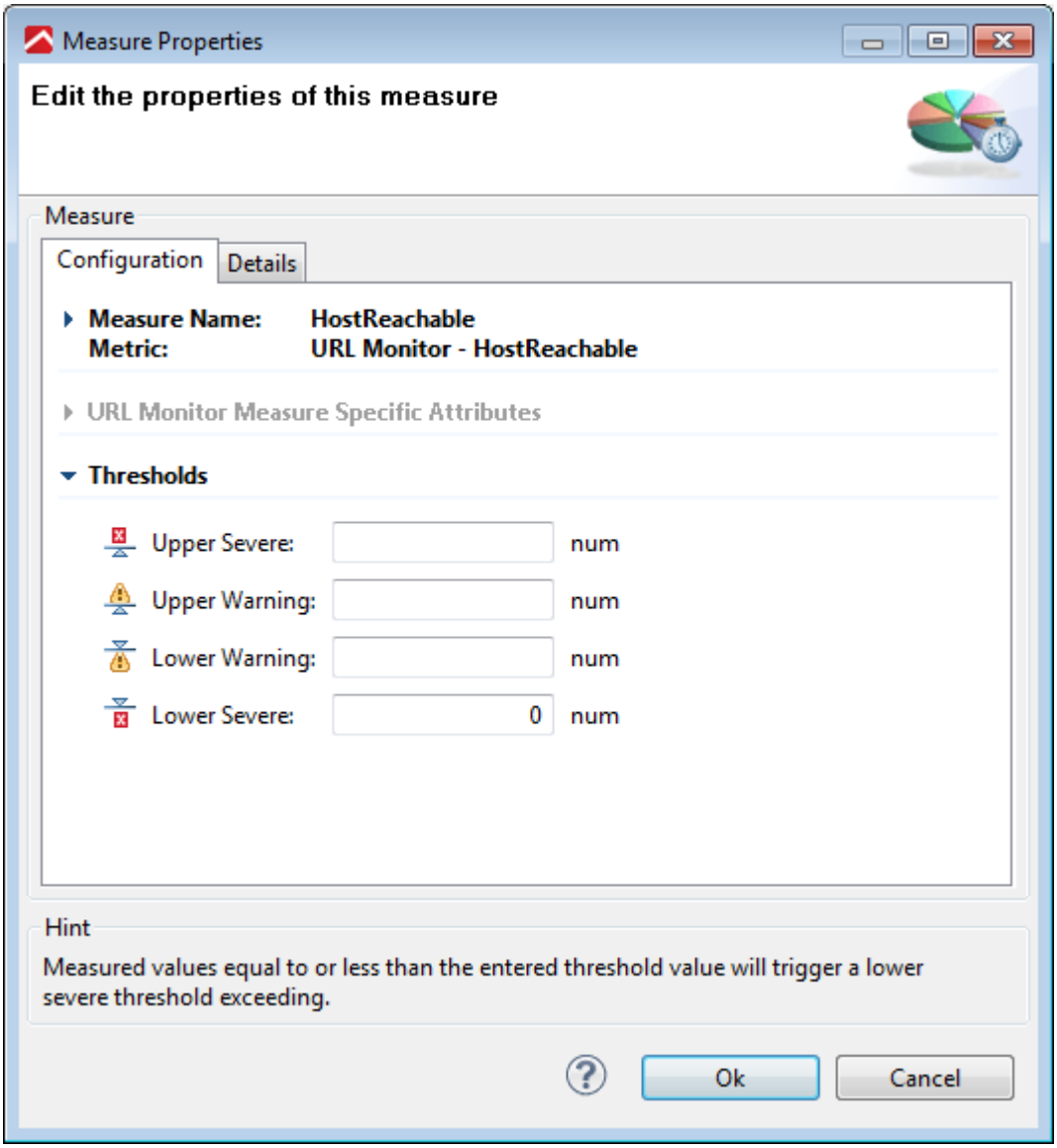
AppMon Synthetic Monitoring monitors the availability and performance of external websites. For more information, see Synthetic Monitoring Integration.

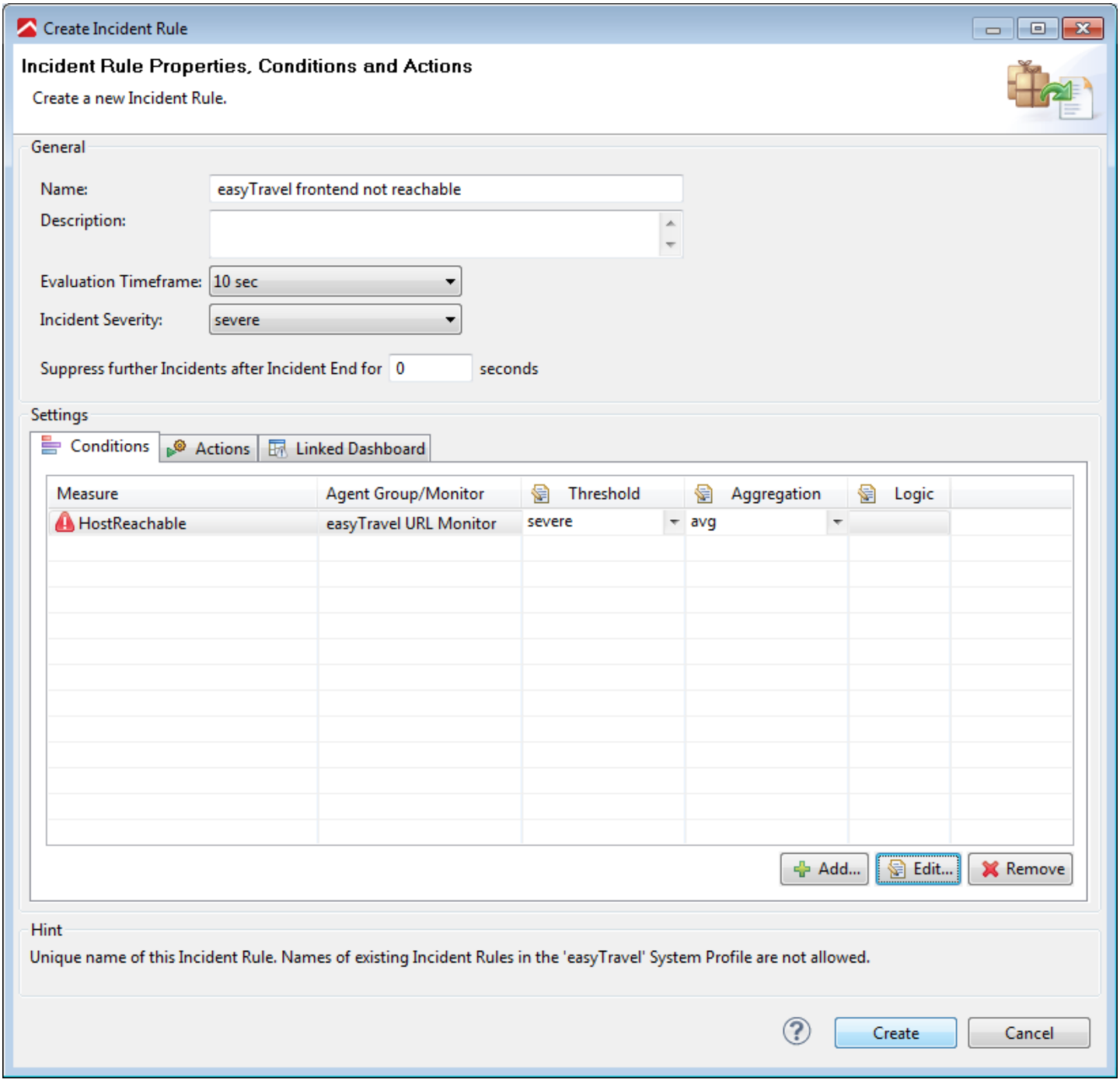
URL Monitor

Use the built-in URL Monitor to monitor the availability of the easyTravel URL. Create an incident based on the **HostReachable** Measure.

If this Measure is 0, you can't access the URL. Set the lower Measure threshold to 0 and create an incident. Then configure an email.




Measure Properties
Edit the properties of this measure
Measure
Configuration
Details
Measure Name: HostReachable
Metric: URL Monitor - HostReachable
URL Monitor Measure Specific Attributes
Thresholds
Upper Severe:
num
Upper Warning:
num
Lower Warning:
num
Lower Severe:
0
num
Hint
Measured values equal to or less than the entered threshold value will trigger a lower severe threshold exceeding.
Ok
Cancel



Availability monitoring for multiple Agent Groups

Scenario

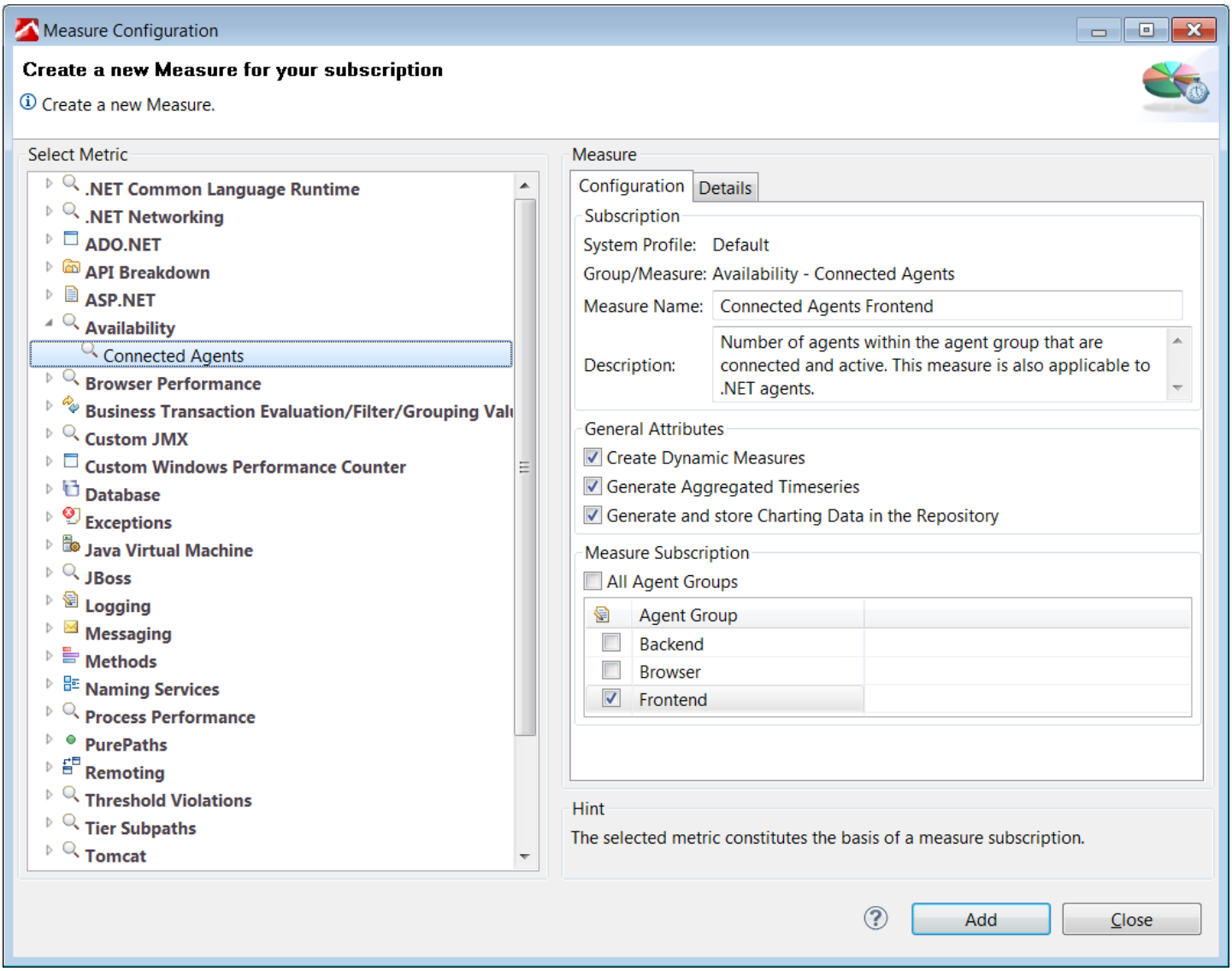
This scenario uses AppMon to monitor multiple groups of systems. Each group has unique requirements regarding who to inform if a system goes down.

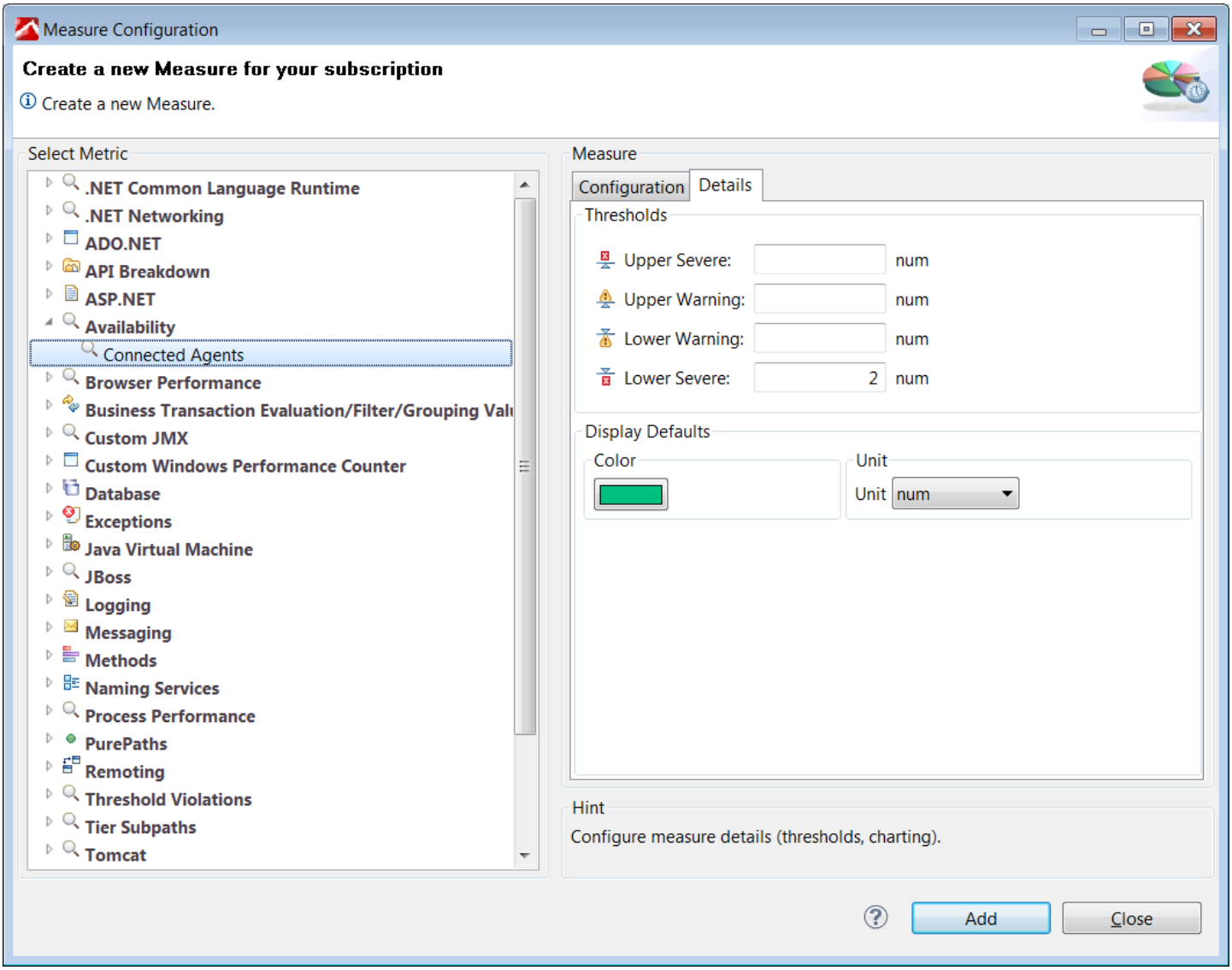
Requirements

- AppMon Installation
- Demo Applications - easyTravel installation

Detailed steps

Subscribe a Measure for the number of connected Agents for each group. To create different incidents for each agent group, subscribe to the **Availability - Connected Agents** Measure multiple times. Each Measure is subscribed to one agent group.





Define incidents for the two Measures from the previous step. These incidents have different actions. For example,different users are responsible for frontend and backend. They also have different thresholds. For example, when the backend is down to one server, this is a severe incident, yet when the frontend is down to *two*, this is severe.

Using host and process health monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is health monitoring?

Goal of this walkthrough

Detailed steps

Related pages

Infrastructure monitoring

What is health monitoring?

A healthy system performs better. AppMon uses defined parameters to monitor health. These parameters use metrics such as CPU, memory, network, and disk. JVMs and CLRs may use metrics such as GC utilization and CPU usage.

Goal of this walkthrough

This walkthrough explains health monitoring for hosts and processes.

Scenario

You need an AppMon installation with several deployed agents. Use your own application, or use the easyTravel demo application to learn about various AppMon features and functions.

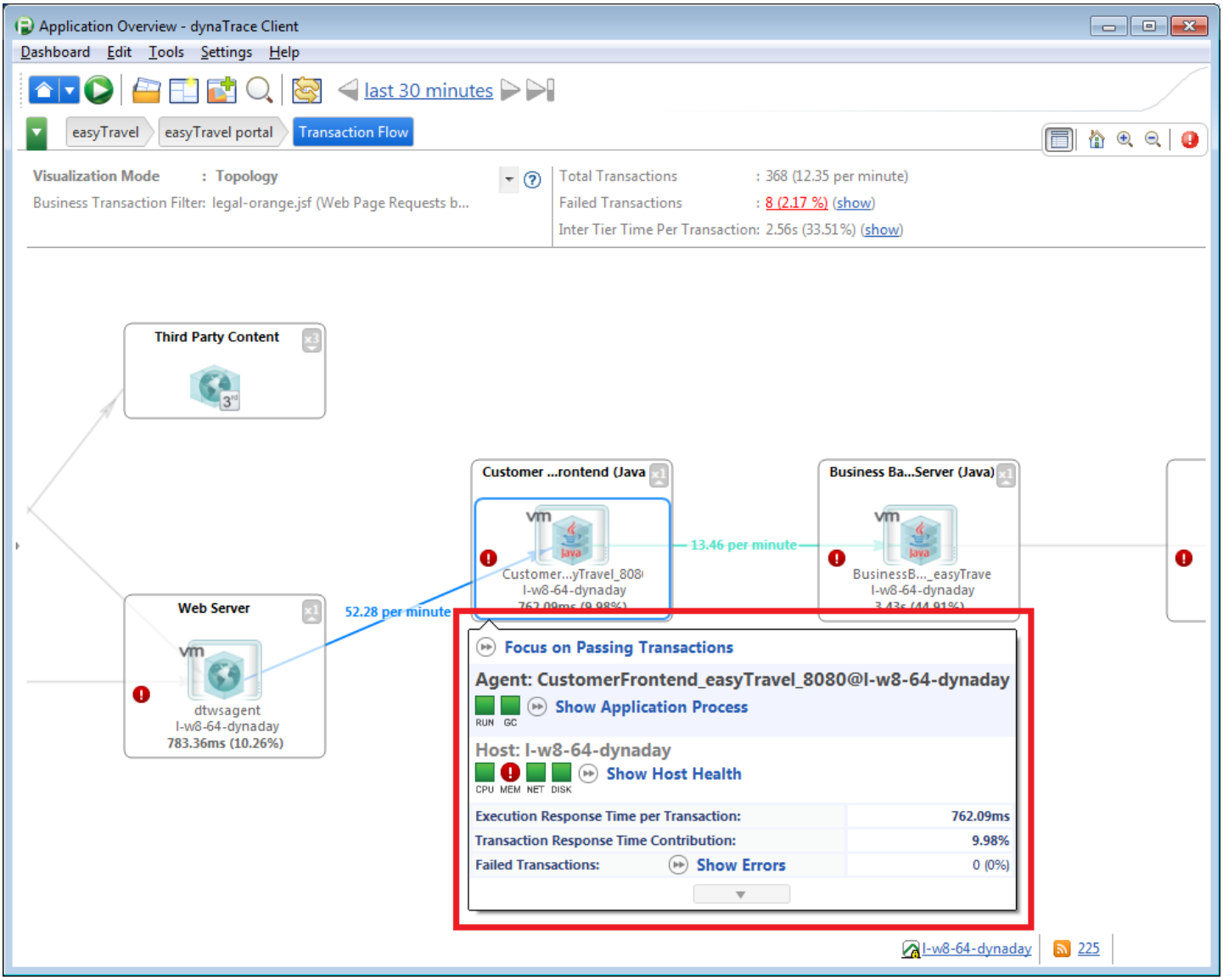
How does AppMon collect host information? For information, see Setting Up Infrastructure Monitoring.

Detailed steps

To calculate the host health, AppMon Agents capture CPU, memory, network, and disk metrics. To calculate process health, Java and .NET Agents capture JVM and CLR memory pool sizes and garbage collection activity.

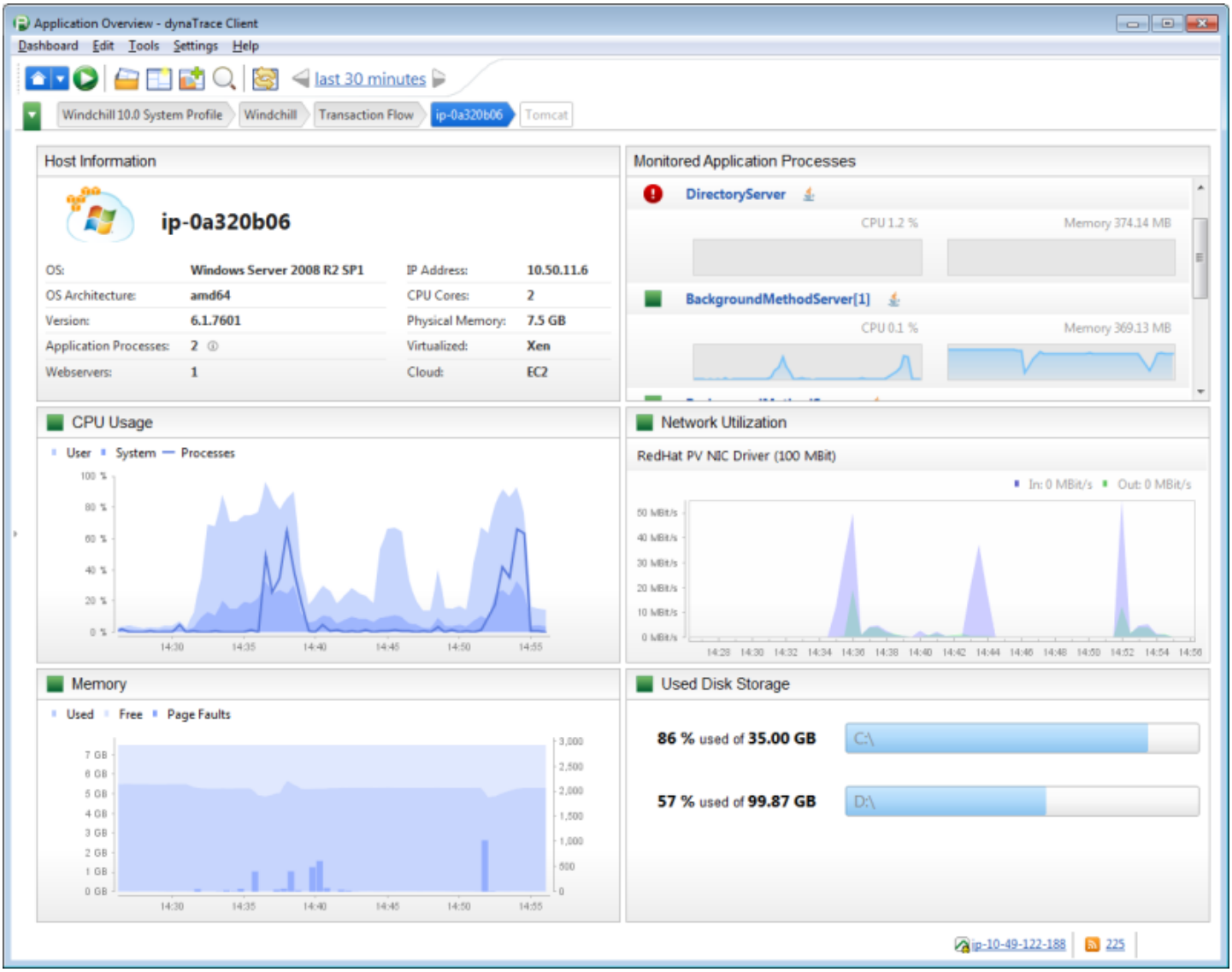
Host Health

Host health is automatically calculated based on CPU, memory, network, and disk metrics. You can see these values in the **Transaction Flow** on the respective agent node. Use this to identify the impact of an unhealthy host on your business transactions.



The Transaction Flow Dashlet

Click **Show Host Health** in the **Transaction Flow** dashlet, or open the **Host Health** dashboard through the **Start Center** for an overview of the host metrics and the health indicators.

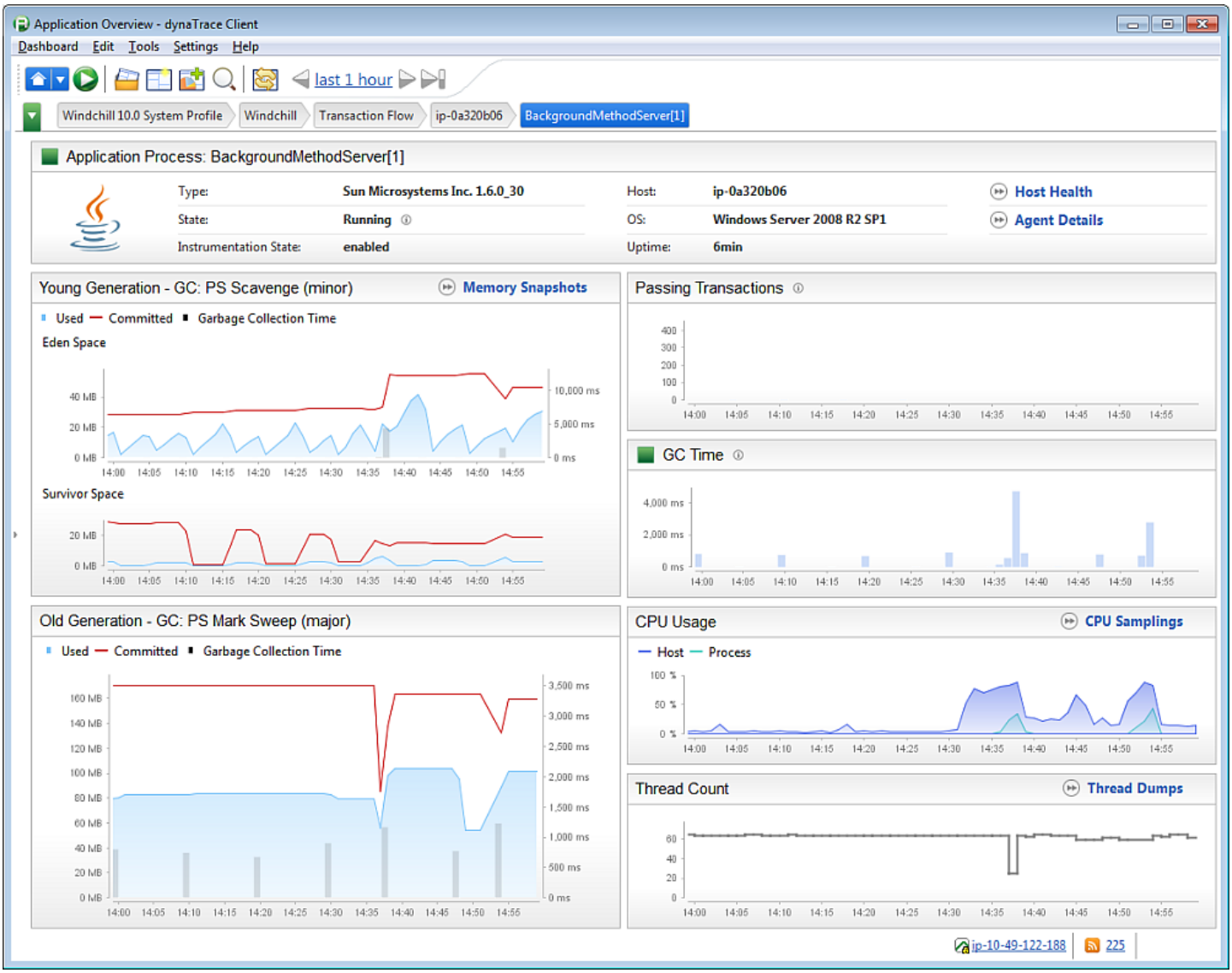


The Host Health Dashboard

AppMon also determines the impact of an unhealthy host on your application performance and business transactions. This lets you concentrate on the application and the underlying infrastructure, if you have application performance problems. If a host runs out of disk storage, we recommend you troubleshoot this rather than analyze application exceptions or log messages.

Process Health

AppMon Agents also collect detailed process health information that you can see in the **Transaction Flow** dashlet. From here, go to the **Process Health** dashboard for a detailed overview of CPU, memory, threads, and transaction throughput.



The Process Health Dashboard

If you suspect a memory leak because of increasing heap spaces, trigger a memory dump from here directly. This is also true for high CPU or thread numbers. Simply trigger a CPU sampling session or thread dump from this dashboard.

Using host and hostgroups for large system monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is multi-host monitoring?

Goal of this walkthrough

Detailed steps

Conclusion

Related pages

Infrastructure monitoring

Setting up infrastructure monitoring

Setting up application availability monitoring

Using host and process health monitoring

Using custom JMX and PMI measures in charts

How to retrieve performance metrics with Windows Performance Monitor

This section applies to monitors such as the Windows or UNIX Performance Monitors. This includes AppMon Agents with built-in host monitoring capabilities.

To monitor the CPU, memory, network and disk of each host, see Host and System Monitoring..

What is multi-host monitoring?

Use multi-host monitoring to monitor many hosts, and define them all only once. As new hosts are automatically added, they are automatically monitored. Each monitor provides dynamic measures that you can see when you drill down to a specific dashboard.

Monitor many and define once

Each configured monitor needs a host as a parameter. For example, the Windows Performance Monitor must identify which machine it should monitor. Since it can monitor multiple hosts, you don't have to repeatedly configure a monitor. Configure it once in the predefined monitors when you set up system monitoring.

Automatically add and monitor hosts

Each monitor delivers data in dynamic measures, so you can manage the data and create charts easily. For example, if you use the Windows Performance Monitor to monitor the memory usage of five machines, you get one dynamic measure. Add this to a chart that contains one dynamic measure for each host you monitor.

Get dynamic measures

New hosts appear in the predefined monitor when an agent starts on a host. AppMon adds them automatically because they belong to a specific hostgroup.

Drill down for data

As you diagnose a transaction, observe the machine health. Drill down to a predefined dashboard that contains the host measures. AppMon shows you the relevant measures for the transaction you're analyzing.

Goal of this walkthrough

In this walkthrough, use multi-hosting to set up a VM monitor that monitors multiple VMs. Then, chart the CPU usage of these VMs in a split chart, and save the dashboard. Finally, use the information to perform a drill-down.
Alternately, go to **Start Center > System Health and Host Health Monitoring** and look at the predefined **System Monitoring** dashboards. The dashboards use the same functionality to monitor Windows and UNIX machines.

Scenario

In this scenario, VMWare runs on a host that includes three VMs. Two of them host the easyTravel Demo Application. The third hosts a collector. Set up the VM monitoring process and correlate the transactions with the VM Chart.

Requirements

- ESX, ESXi or vCenter (Version 4+)
- Multiple VMWare virtual machines on a VM Host
- VMWare tools on guests
- Demo application on some of the VMs

Detailed steps

Set up the monitor

After an Agent starts and establishes its connection to AppMon, you should recognize the host. If you don't, add it to your CMDB and to a specific hostgroup which represents the operating system:

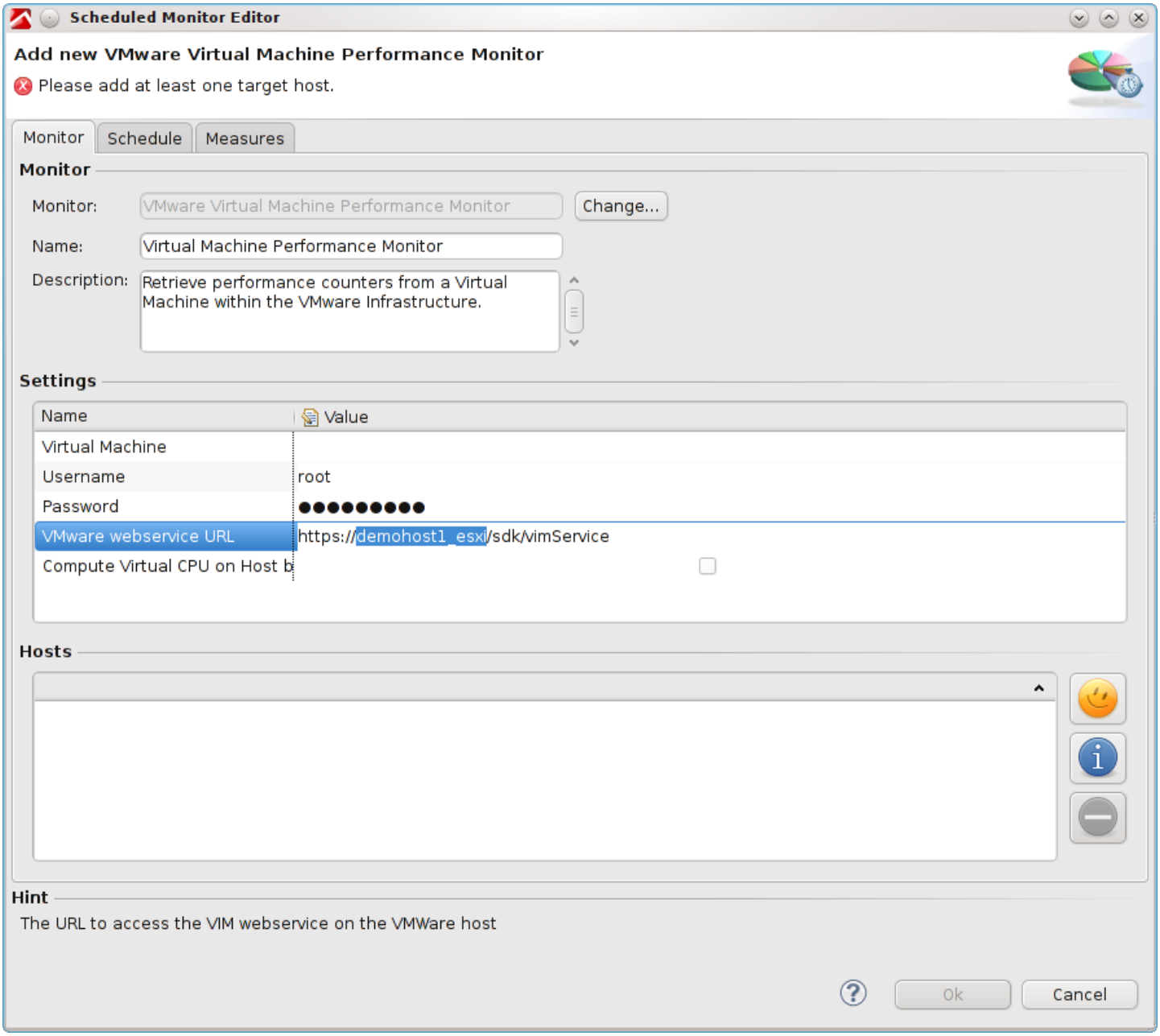
1. Open the System Profile and go the **Monitor** section to create a new VM monitor.
2. Enter the name of your vHost into the service URL.
3. Enter the appropriate user and password necessary to use the VMWare web service API.
4. Select the correct host that represent your VMs and choose the default** Every Minute** schedule.
5. To verify it works, click **System Profile > Apply**.
6. Click **Run Now** on the new monitor. All your hosts should appear green.

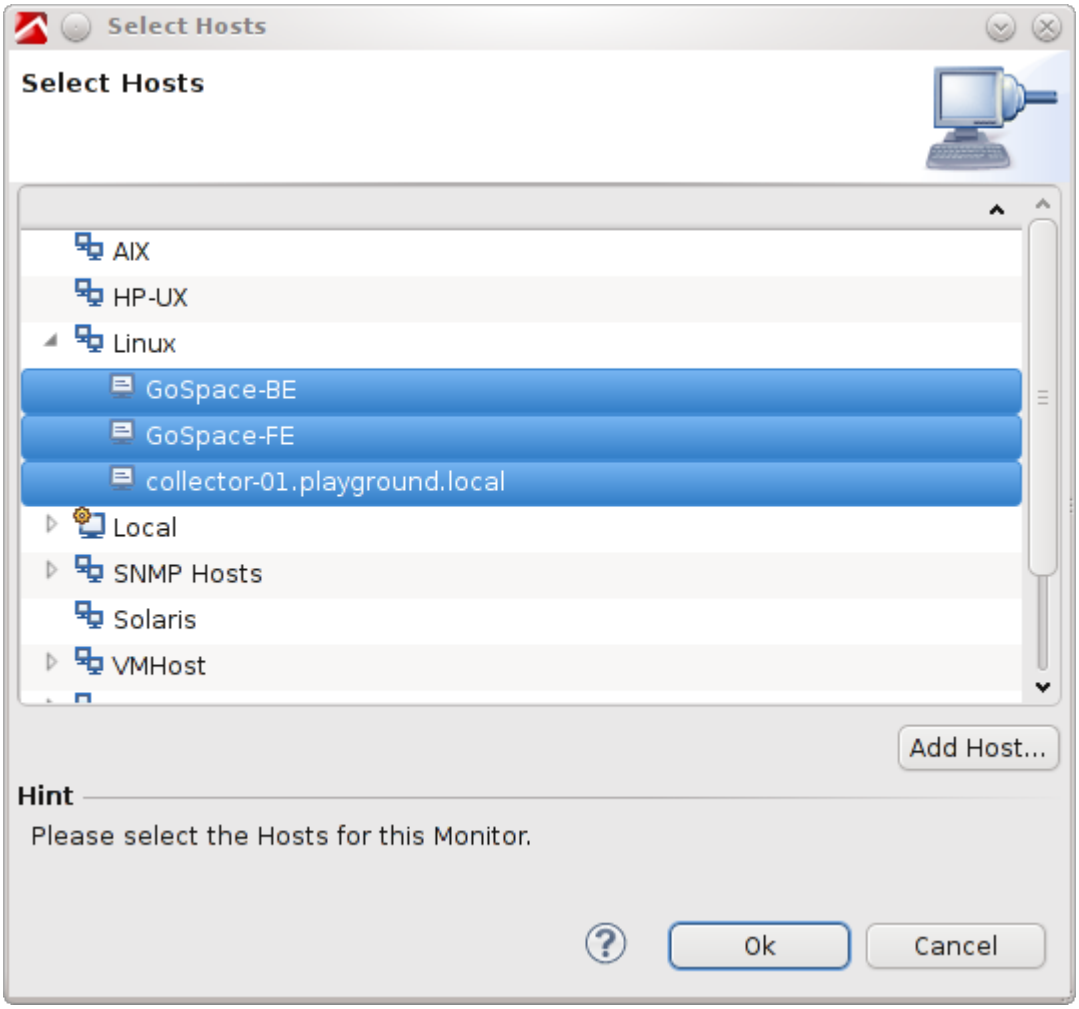
Note

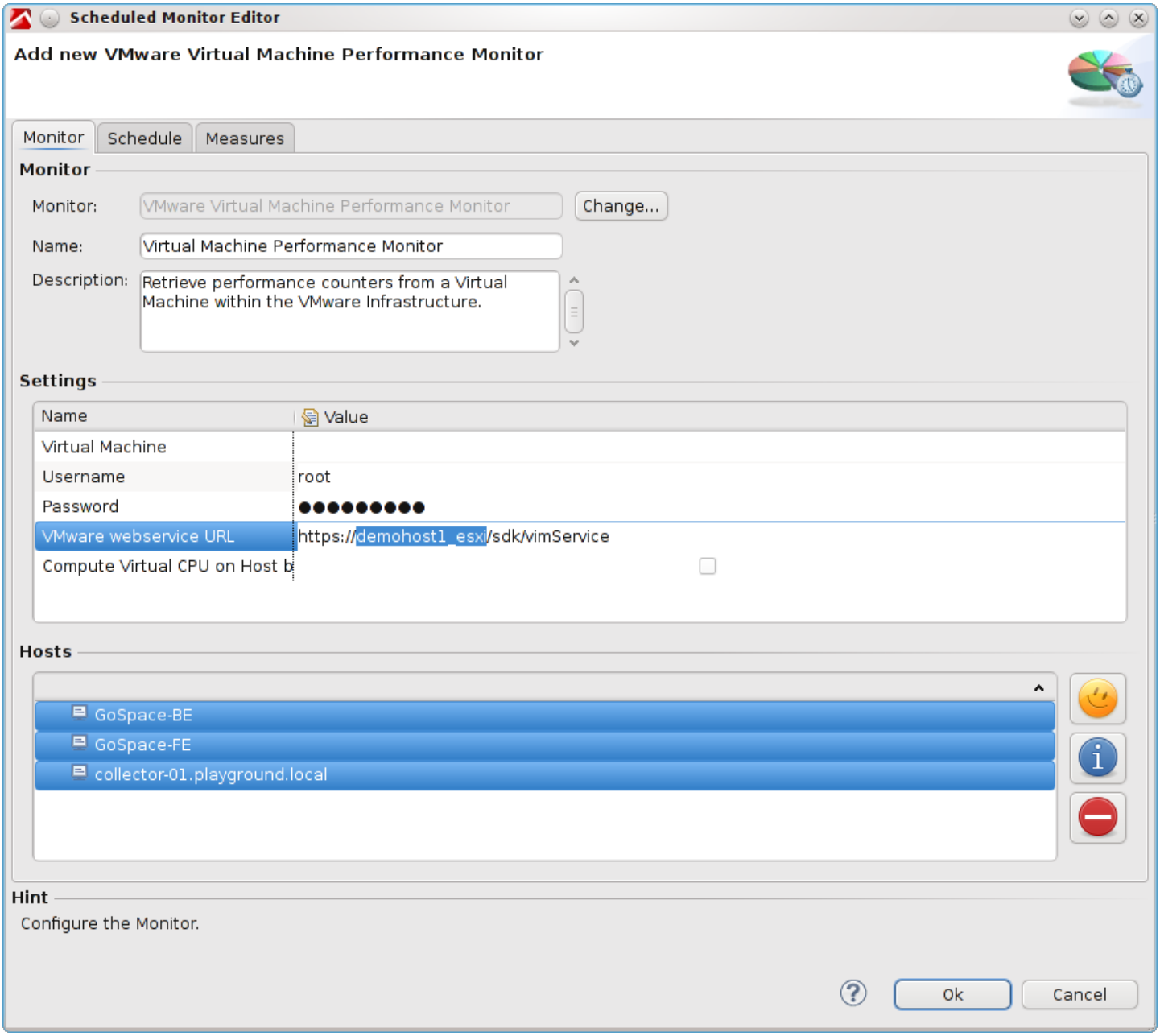
VMWare guest tools must be installed so the VMWare monitor can establish the connection between a host and a VM.

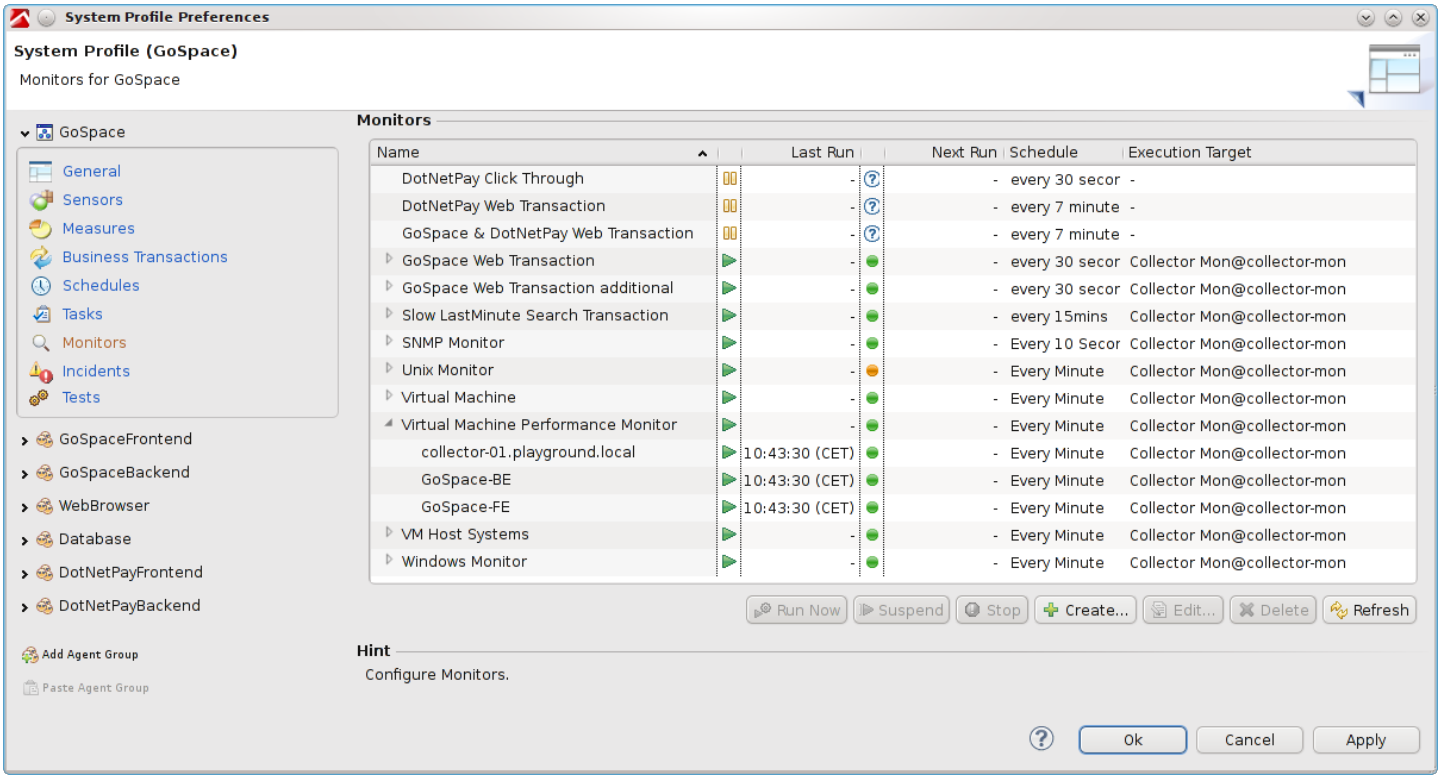
Add measures to charts

After the monitor is set up successfully, add the measures to charts. Add the CPU that the VM uses and change the chart to a stack area chart. You can see which VM consumes how much real CPU based on the host. When you look at the table under the chart, notice that all of the configured hosts are available as a dynamic measure. You can also define one incident on the dynamic measure and it applies to all incidents.


Scheduled Monitor Editor
Add new VMware Virtual Machine Performance Monitor
Please add at least one target host.
Monitor
Schedule
Measures
Monitor
Monitor:
VMware Virtual Machine Performance Monitor
Change...
Name:
Virtual Machine Performance Monitor
Description:
Retrieve performance counters from a Virtual Machine within the VMware Infrastructure.
Settings
Name
Value
Virtual Machine
Username
root
Password
VMware webservice URL
https://demohost1_esxi/sdk/vimService
Compute Virtual CPU on Host b
Hosts
Hint
The URL to access the VIM webservice on the VMWare host
Ok
Cancel


Select Hosts
Select Hosts
AIX
HP-UX
Linux
GoSpace-BE
GoSpace-FE
collector-01.playground.local
Local
SNMP Hosts
Solaris
VMHost
Add Host...
Hint
Please select the Hosts for this Monitor.
Ok
Cancel


Scheduled Monitor Editor
Add new VMware Virtual Machine Performance Monitor
Monitor
Schedule
Measures
Monitor
Monitor:
VMware Virtual Machine Performance Monitor
Change...
Name:
Virtual Machine Performance Monitor
Description:
Retrieve performance counters from a Virtual Machine within the VMware Infrastructure.
Settings
Name
Value
Virtual Machine
Username
root
Password
VMware webservice URL
https://demohost1_esxi/sdk/vimService
Compute Virtual CPU on Host b
Hosts
GoSpace-BE
GoSpace-FE
collector-01.playground.local
Hint
Configure the Monitor.
Ok
Cancel

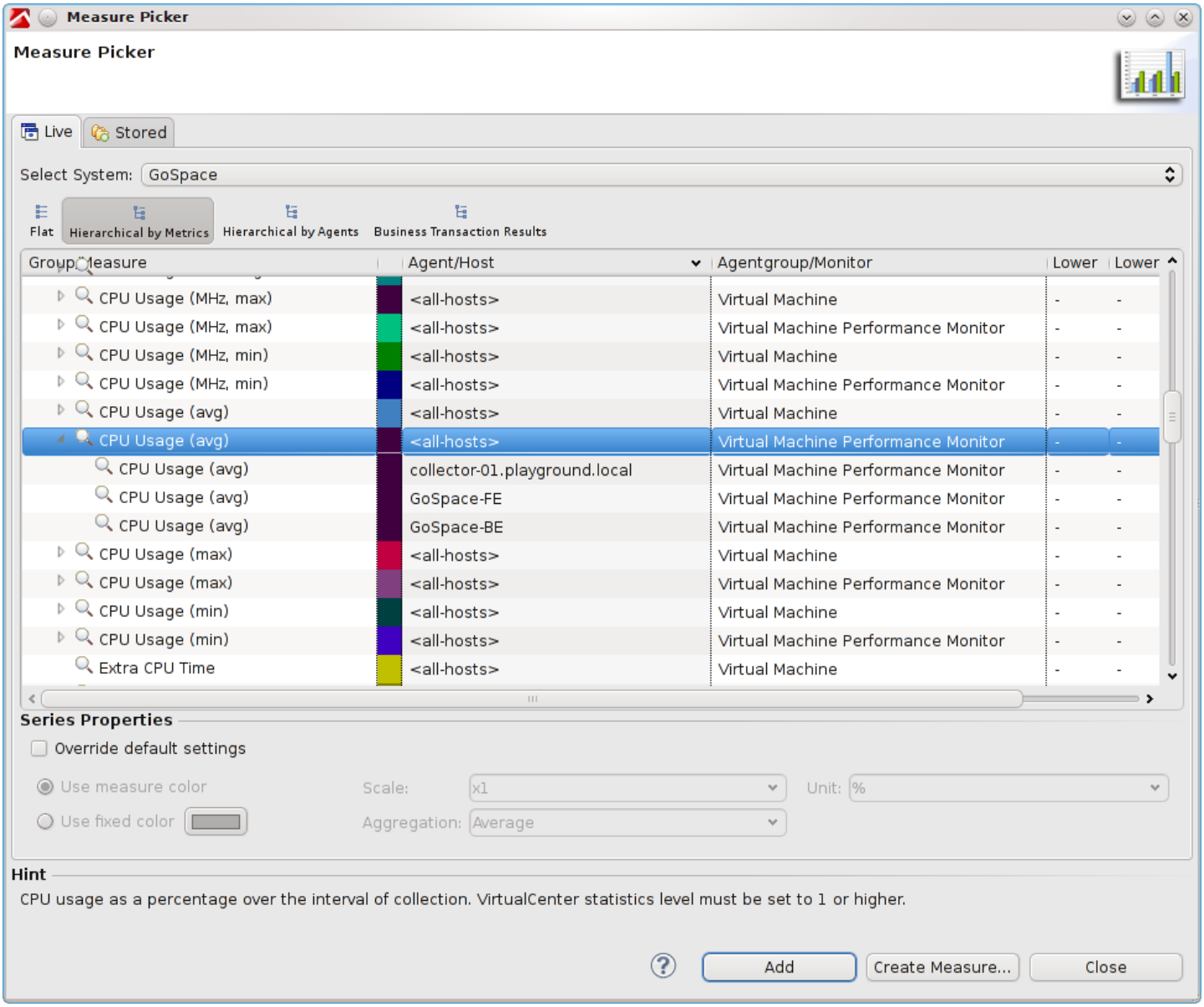


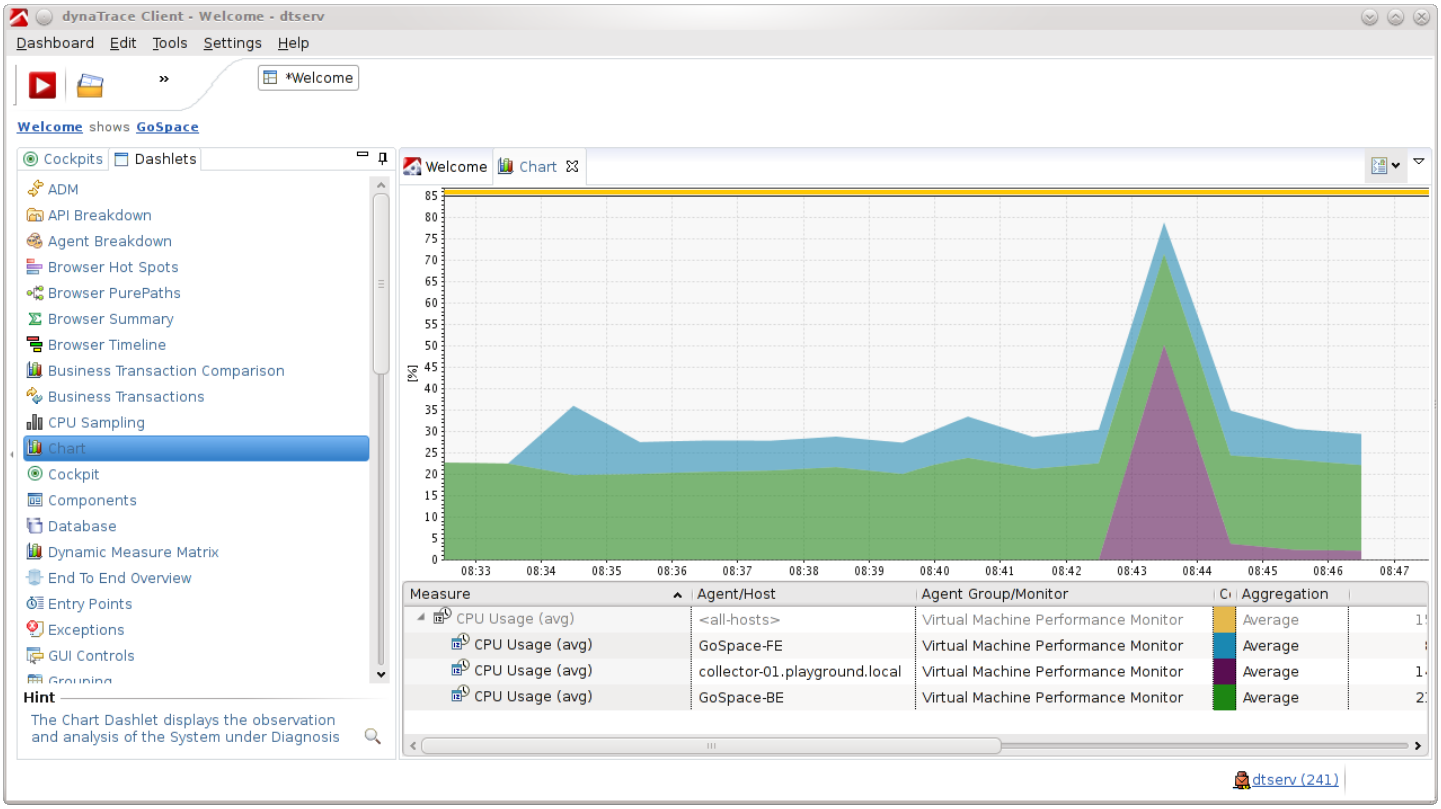
If you set up a UNIX or Windows Performance Monitor for the same host, you can chart the CPU as seen by a guest, side-by-side, in the VMs view. Store the dashboard for later use, but leave it open.

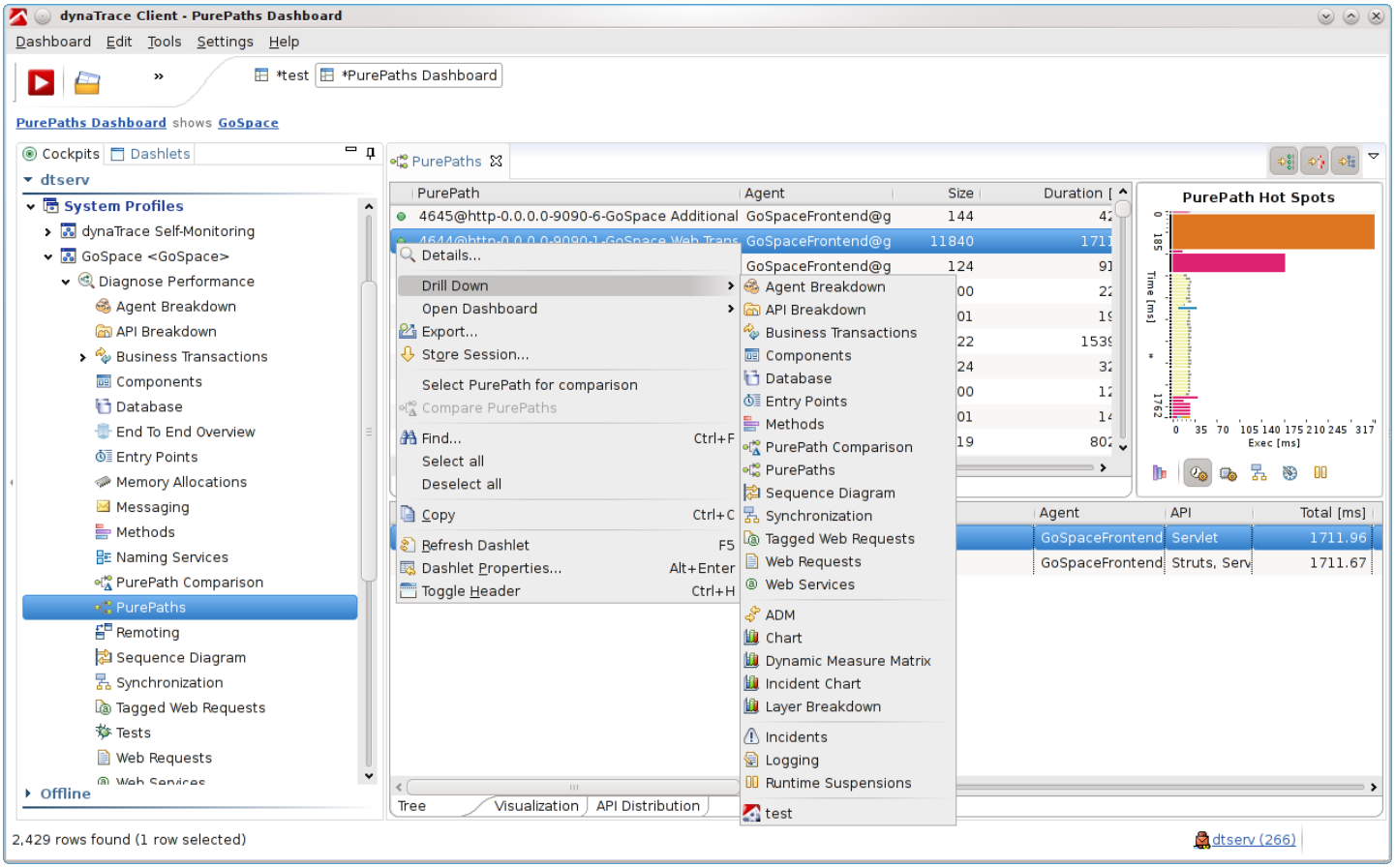
Use monitor measures for diagnostics

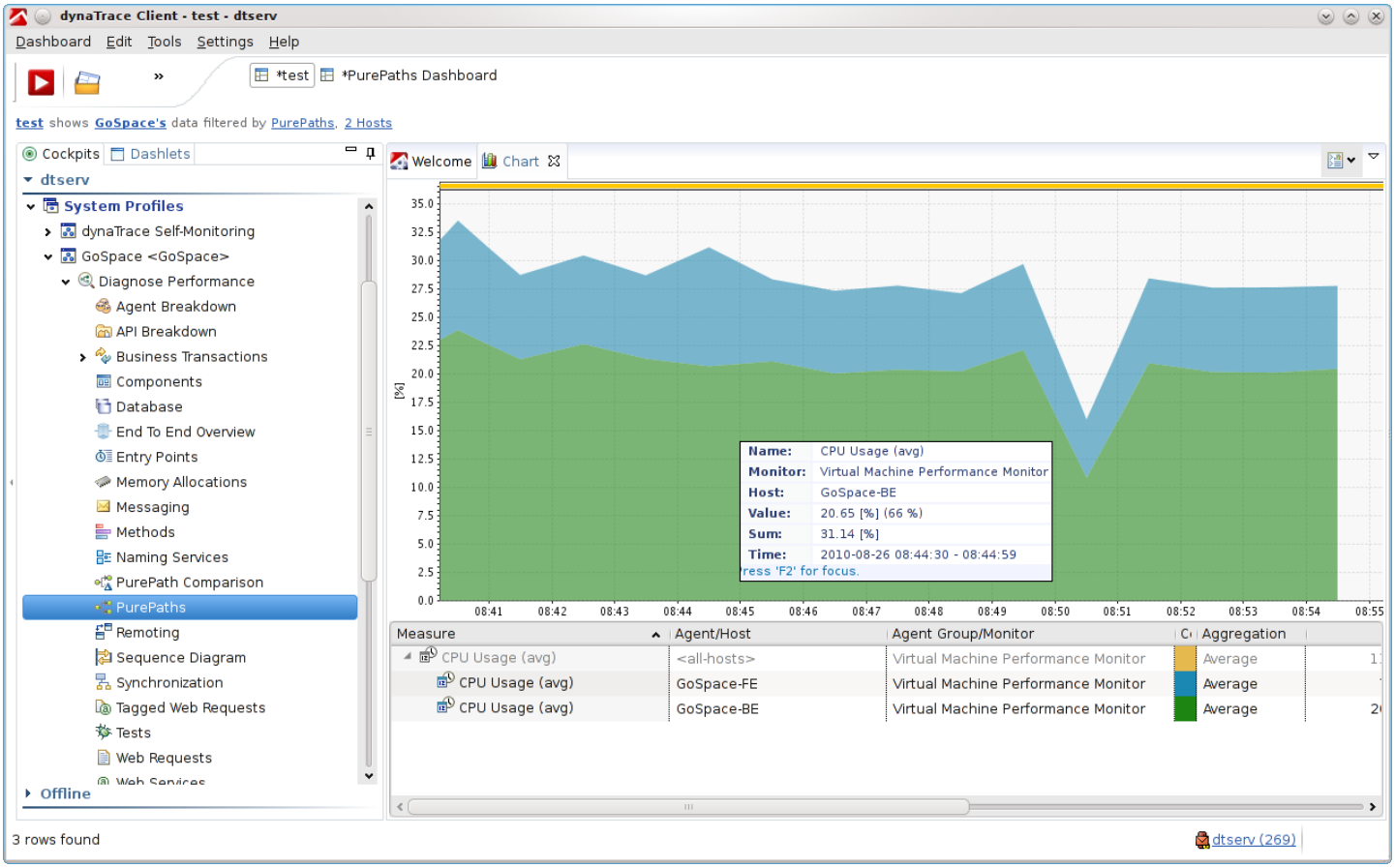
Now, use this dashboard for diagnostics.

1. Open the **PurePath** dashlet.
2. Run the **Last Minute Search** on the sample application so you have some data.
3. Open the PurePath and drill down to the new dashboard. The dashboard filters all hosts crossed by the PurePath.


Measure Picker
Measure Picker
Live
Stored
Select System: GoSpace
Flat
Hierarchical by Metrics
Hierarchical by Agents
Business Transaction Results
Group/Measure
Agent/Host
Agentgroup/Monitor
Lower
Lower
CPU Usage (MHz, max) <all-hosts> Virtual Machine - -
CPU Usage (MHz, max) <all-hosts> Virtual Machine Performance Monitor - -
CPU Usage (MHz, min) <all-hosts> Virtual Machine - -
CPU Usage (MHz, min) <all-hosts> Virtual Machine Performance Monitor - -
CPU Usage (avg) <all-hosts> Virtual Machine - -
CPU Usage (avg) <all-hosts> Virtual Machine Performance Monitor - -
CPU Usage (avg) collector-01.playground.local Virtual Machine Performance Monitor - -
CPU Usage (avg) GoSpace-FE Virtual Machine Performance Monitor - -
CPU Usage (avg) GoSpace-BE Virtual Machine Performance Monitor - -
CPU Usage (max) <all-hosts> Virtual Machine - -
CPU Usage (max) <all-hosts> Virtual Machine Performance Monitor - -
CPU Usage (min) <all-hosts> Virtual Machine - -
CPU Usage (min) <all-hosts> Virtual Machine Performance Monitor - -
Extra CPU Time <all-hosts> Virtual Machine - -
Series Properties
Override default settings
Use measure color
Use fixed color
Scale: x1
Unit: %
Aggregation: Average
Hint
CPU usage as a percentage over the interval of collection. VirtualCenter statistics level must be set to 1 or higher.
Add
Create Measure...
Close


dynaTrace Client - Welcome - dtserv
Dashboard Edit Tools Settings Help
*Welcome
Welcome shows GoSpace
Cockpits Dashlets
ADM
API Breakdown
Agent Breakdown
Browser Hot Spots
Browser PurePaths
Browser Summary
Browser Timeline
Business Transaction Comparison
Business Transactions
CPU Sampling
Chart
Cockpit
Components
Database
Dynamic Measure Matrix
End To End Overview
Entry Points
Exceptions
GUI Controls
Hint
The Chart Dashlet displays the observation and analysis of the System under Diagnosis
Welcome Chart
[%]
08:33 08:34 08:35 08:36 08:37 08:38 08:39 08:40 08:41 08:42 08:43 08:44 08:45 08:46 08:47
Measure Agent/Host Agent Group/Monitor Aggregation
CPU Usage (avg) <all-hosts> Virtual Machine Performance Monitor Average
CPU Usage (avg) GoSpace-FE Virtual Machine Performance Monitor Average
CPU Usage (avg) collector-01.playground.local Virtual Machine Performance Monitor Average
CPU Usage (avg) GoSpace-BE Virtual Machine Performance Monitor Average
dtserv (241)


dynaTrace Client - PurePaths Dashboard
Dashboard Edit Tools Settings Help
*test *PurePaths Dashboard
PurePaths Dashboard shows GoSpace
Cockpits Dashlets
dtserv
System Profiles
dynaTrace Self-Monitoring
GoSpace <GoSpace>
Diagnose Performance
Agent Breakdown
API Breakdown
Business Transactions
Components
Database
End To End Overview
Entry Points
Memory Allocations
Messaging
Methods
Naming Services
PurePath Comparison
PurePaths
Remoting
Sequence Diagram
Synchronization
Tagged Web Requests
Tests
Web Requests
Offline
PurePaths
PurePath Agent Size Duration
4645@http-0.0.0.0-9090-6-GoSpace Additional GoSpaceFrontend@g 144
GoSpaceFrontend@g 11840
GoSpaceFrontend@g 124
Details...
Drill Down
Open Dashboard
Export...
Store Session...
Select PurePath for comparison
Compare PurePaths
Find... Ctrl+F
Select all
Deselect all
Copy Ctrl+C
Refresh Dashlet F5
Dashlet Properties... Alt+Enter
Toggle Header Ctrl+H
Agent Breakdown
API Breakdown
Business Transactions
Components
Database
Entry Points
Methods
PurePath Comparison
PurePaths
Sequence Diagram
Synchronization
Tagged Web Requests
Web Requests
Web Services
ADM
Chart
Dynamic Measure Matrix
Incident Chart
Layer Breakdown
Incidents
Logging
Runtime Suspensions
test
PurePath Hot Spots
Time [ms]
Exec [ms]
Agent API Total [ms]
GoSpaceFrontend Servlet 1711.96
GoSpaceFrontend Struts, Serv 1711.67
Tree Visualization API Distribution
2,429 rows found (1 row selected)
dtserv (266)



The same process works for all monitors. System Monitoring works the same with Windows and UNIX-like hosts.

Conclusion

Multi-host monitoring successfully monitors, charts, and produces alerts for a number of hosts. You can analyze diagnostics easily because the charts filter to crucial data.

Using custom JMX and PMI measures in charts

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What are JMX PMI measures?

Goal of this walkthrough

Detailed steps

What are JMX PMI measures?

Use the Java Management Extension (JMX) to manage and monitor Java-based applications. The JMX interface invokes methods on MBeans to retrieve Performance Monitoring Infrastructure (PMI) data. AppMon supports JMX and provides an interface for third-party applications integration.

Goal of this walkthrough

This walkthrough explains how to do the following:

- Browse the MBeans and PMI groups on an application server.
- Add these attributes as measures.
- Add these measures to charts.

Scenario

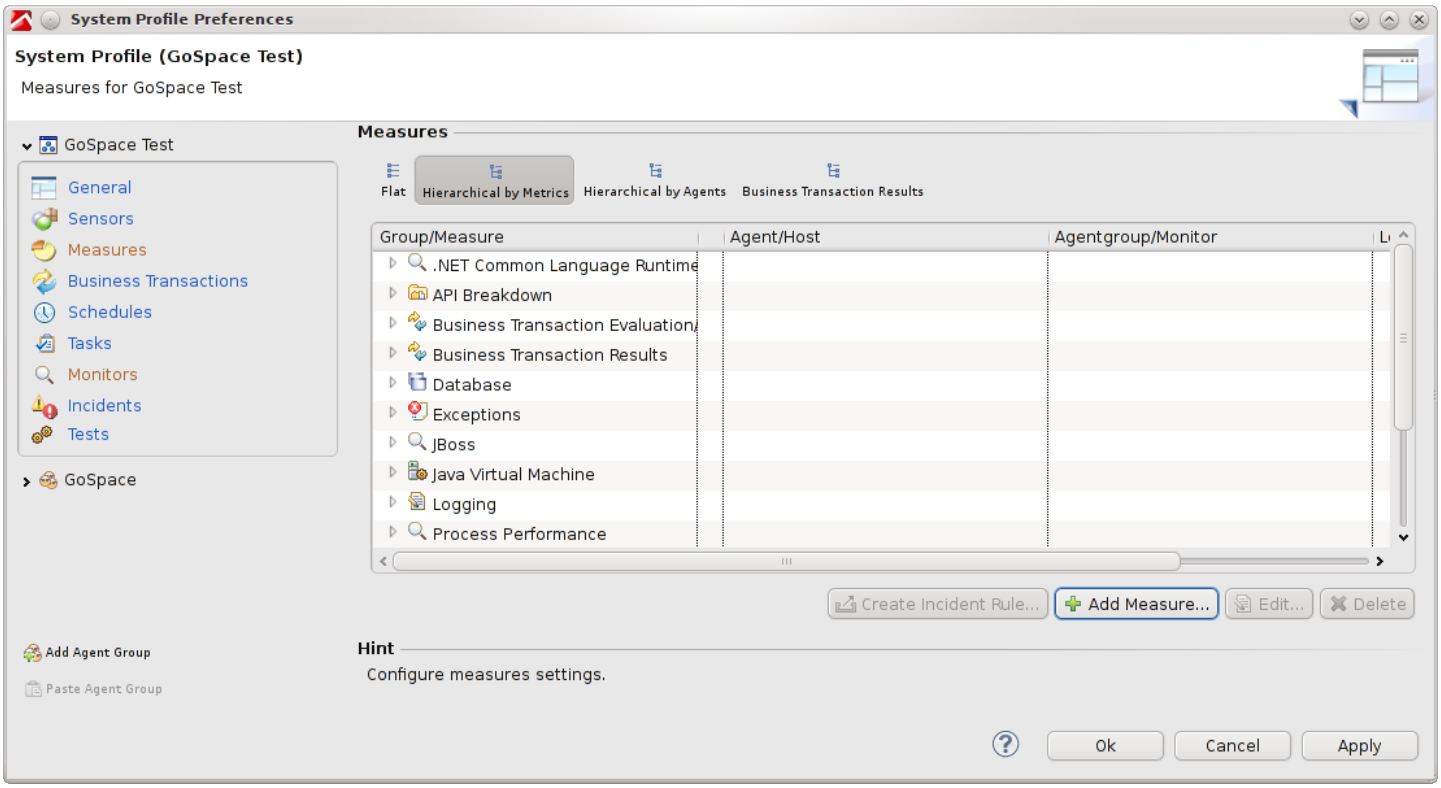
Use GoSpace for this demonstration. Simply add the InUseConnectionCount of the used DataSource as a measure and chart it. This will tell you how many connections GoSpace uses at any given time. PMI works exactly the same way.

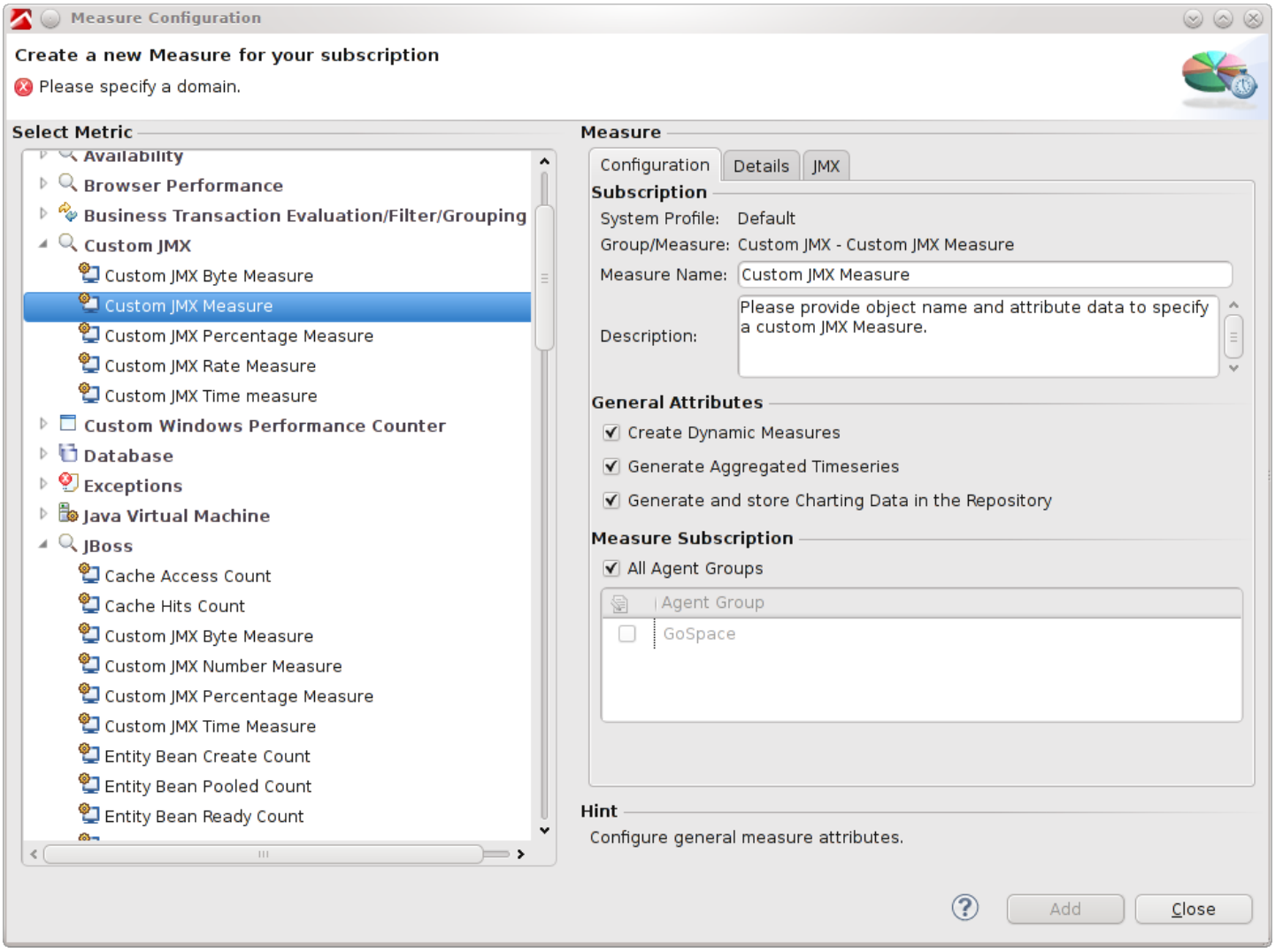
Also, use the easyTravel demo application to learn about various AppMon features and functions.

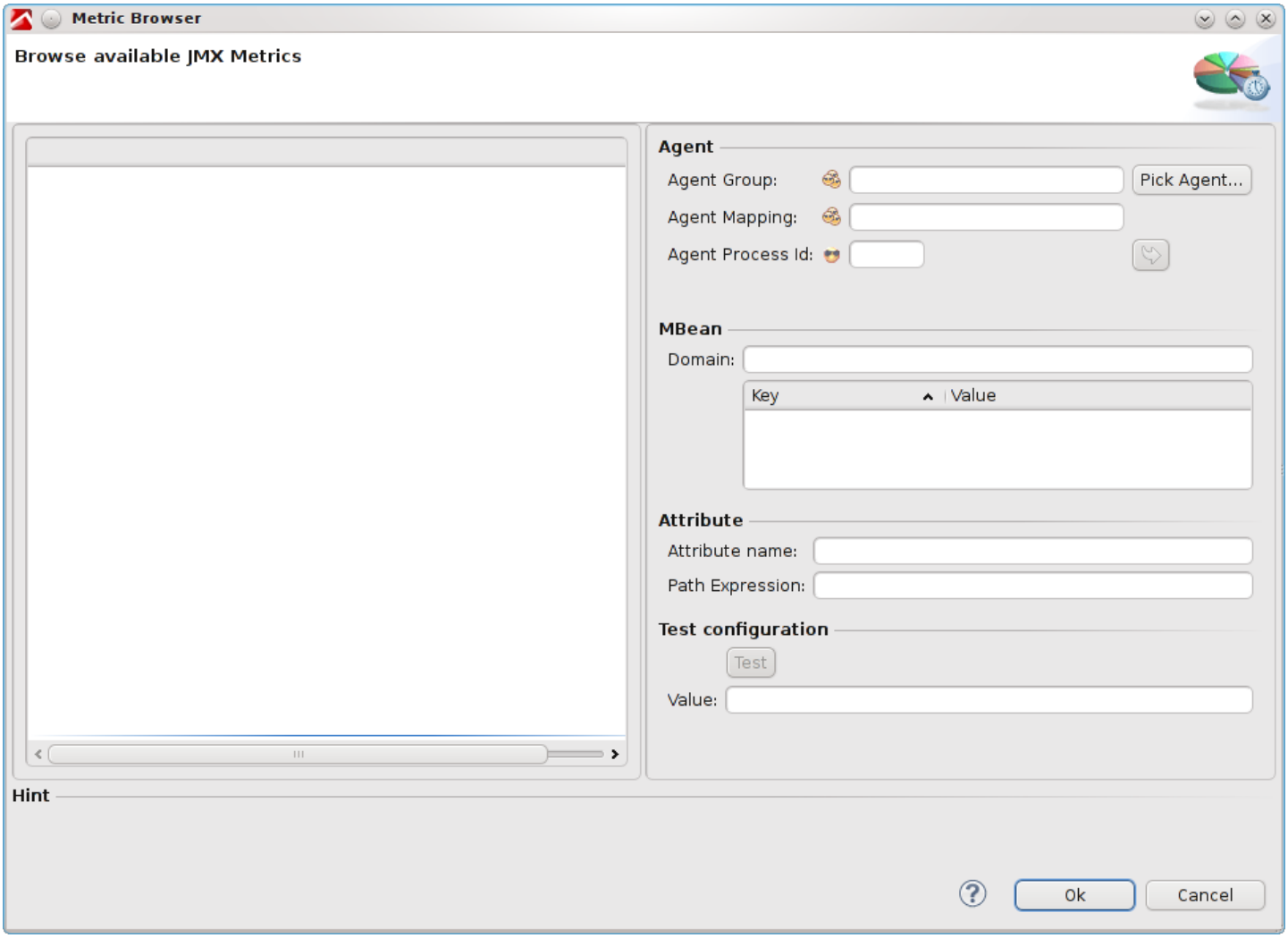
Detailed steps

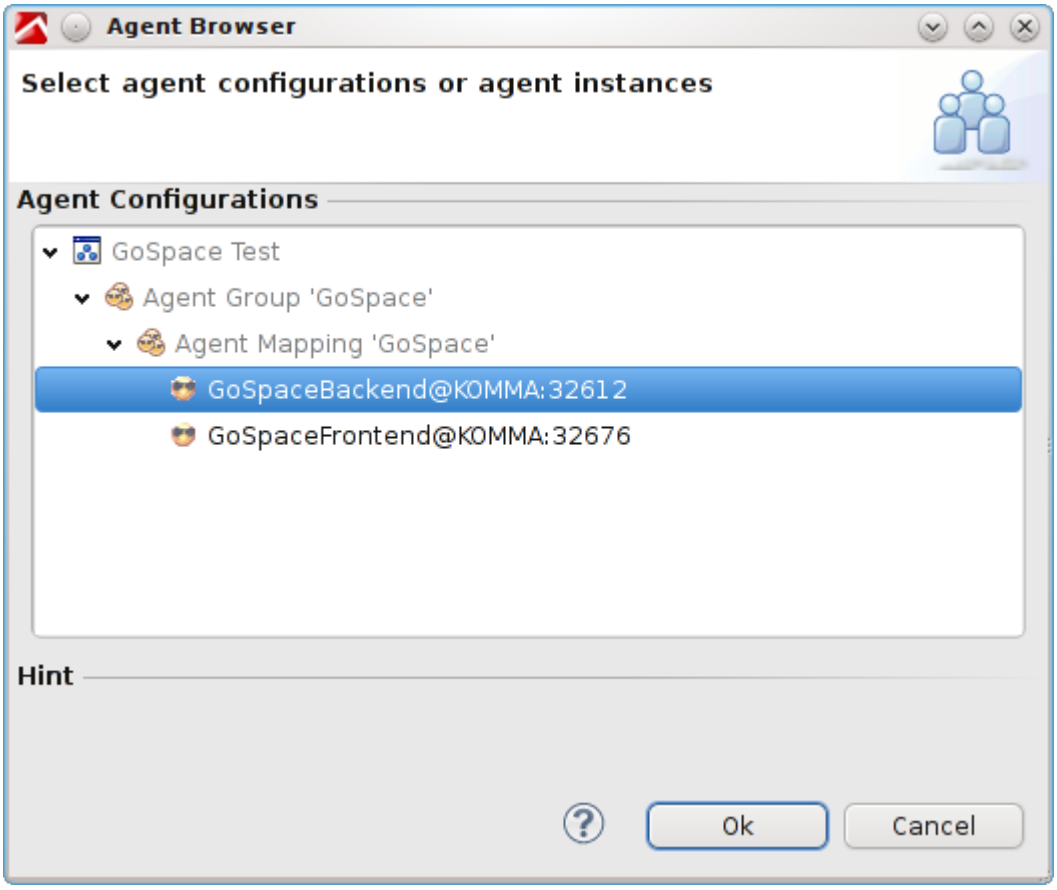
Pick the measure

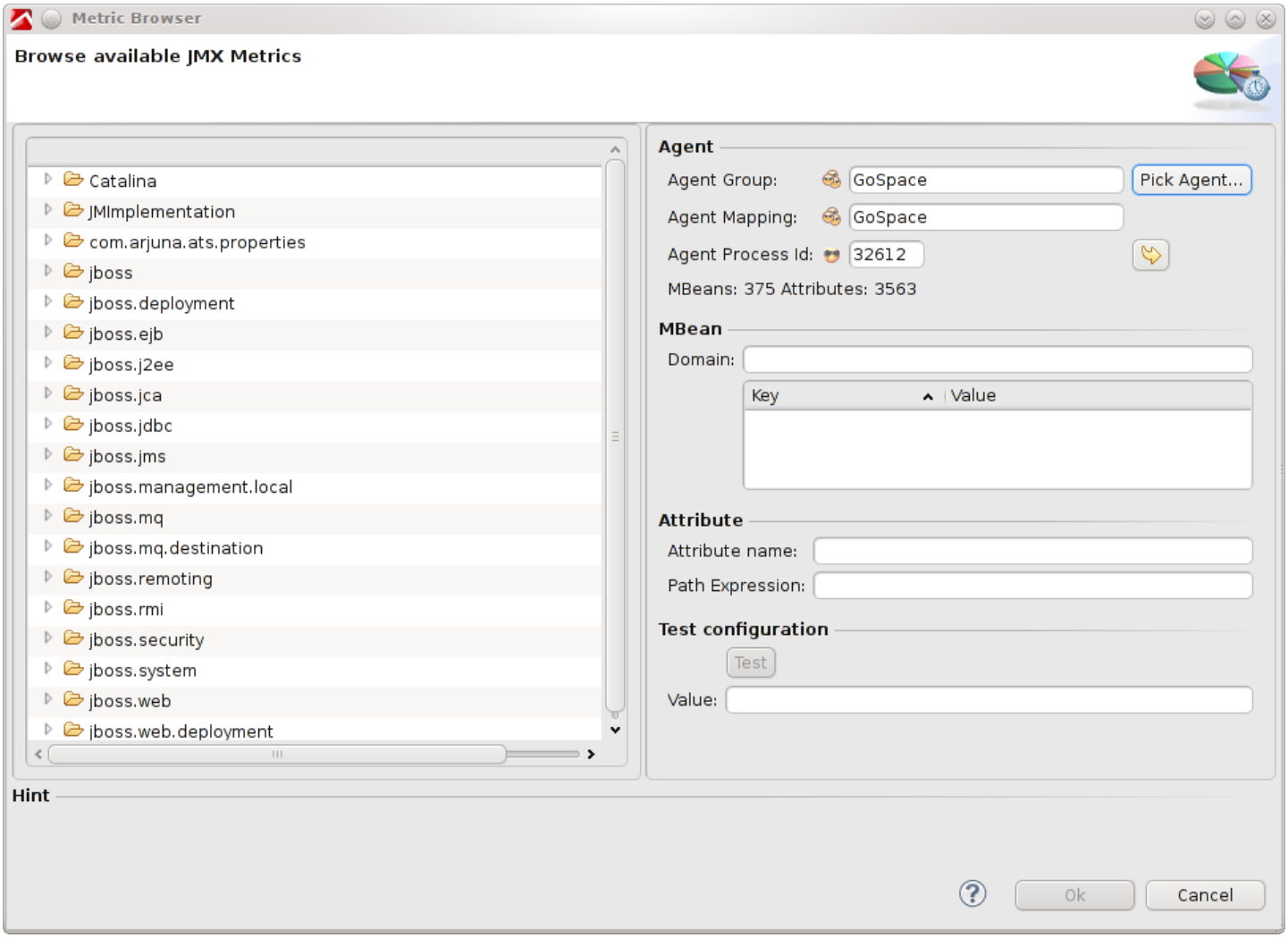
1. Go to **System Profile > Measure** and add a measure to monitor JMX attributes.
2. Pick a custom **JMX Measure** type. A JMX browser appears.
3. Choose an agent. A list of all MBeans for that agent appears.
4. Choose the `InUseCount` of a `ManagedConnectionPool` attributes.

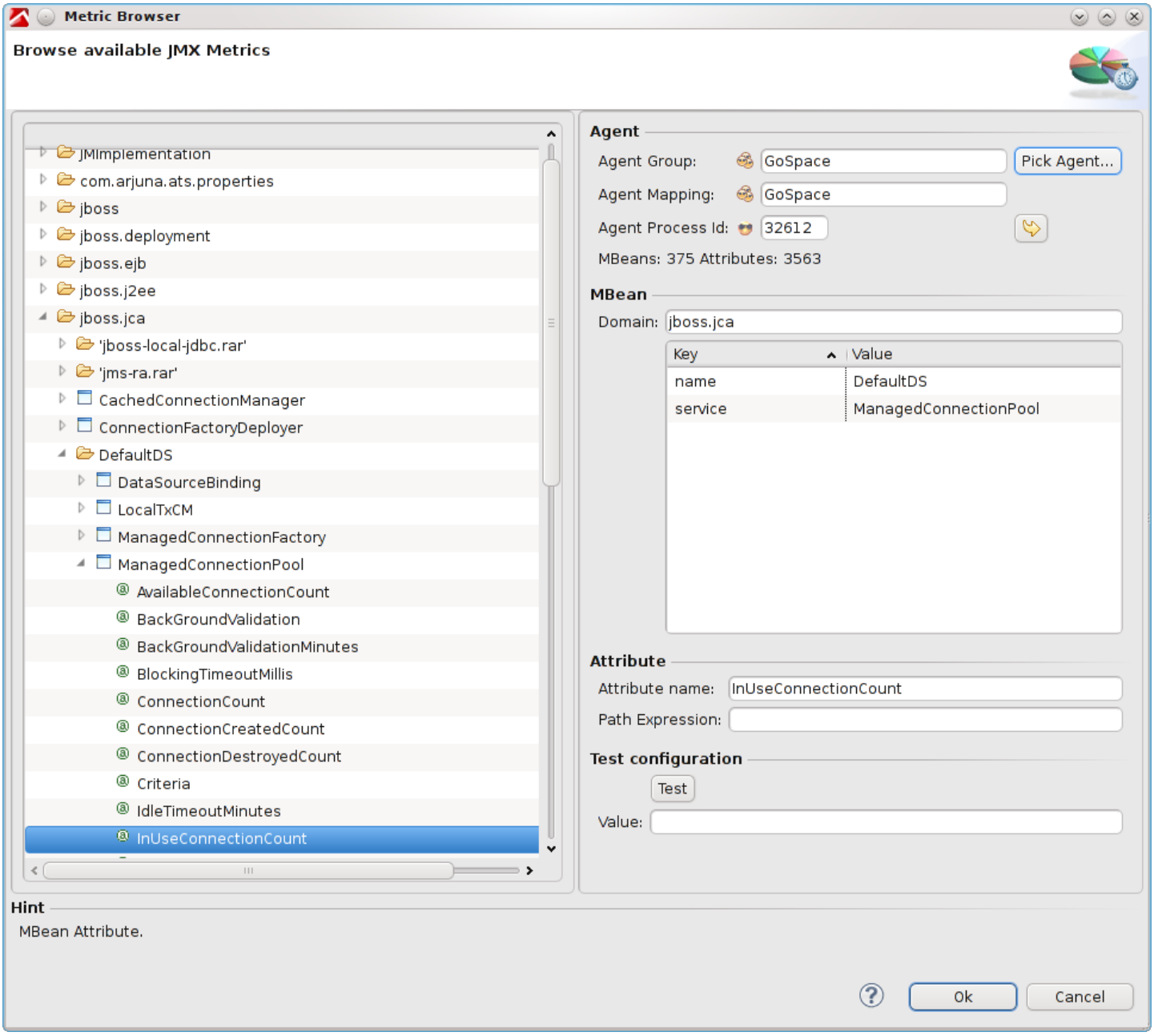


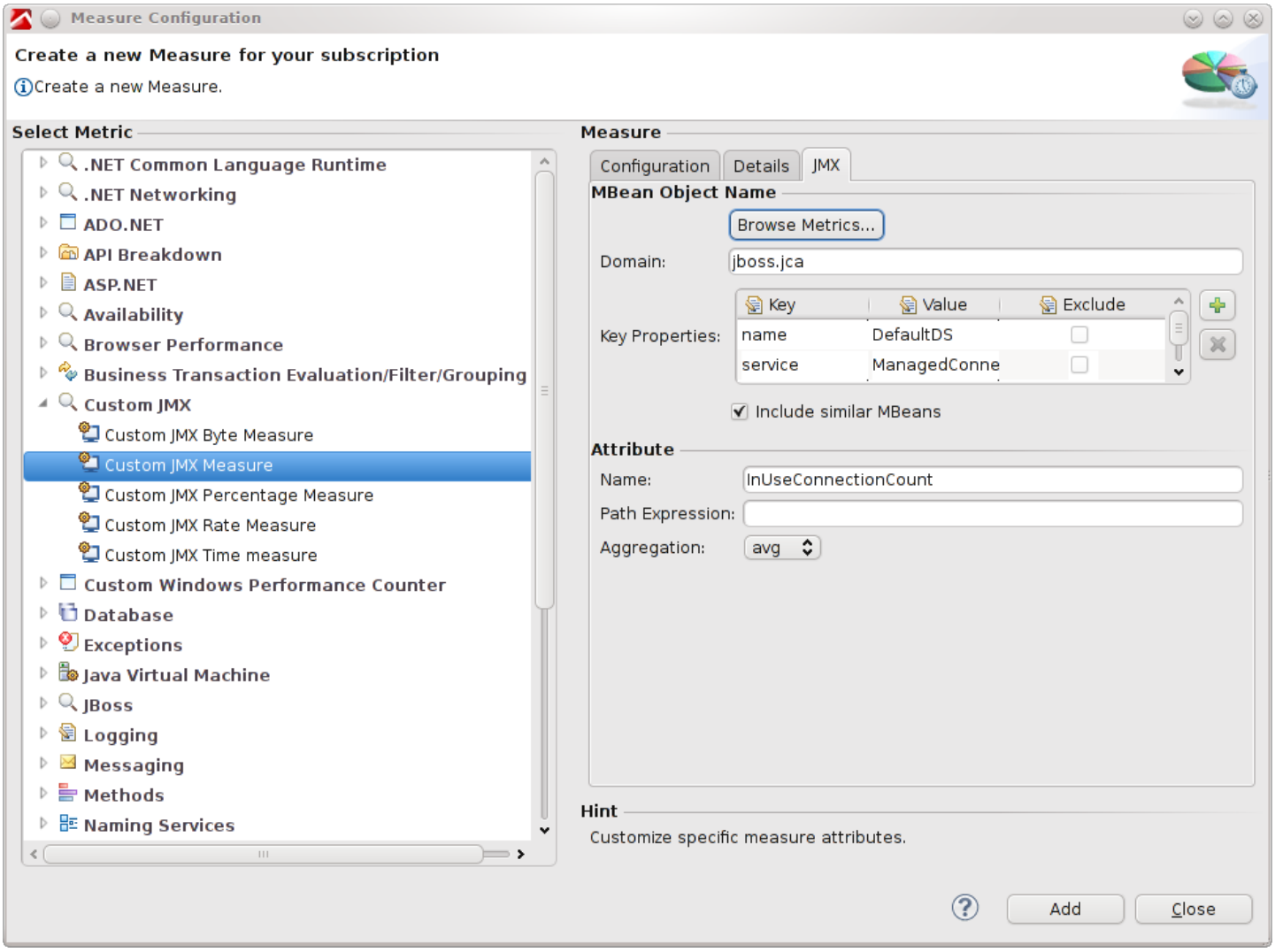

Measure Configuration
Create a new Measure for your subscription
Please specify a domain.
Select Metric
Availability
Browser Performance
Business Transaction Evaluation/Filter/Grouping
Custom JMX
Custom JMX Byte Measure
Custom JMX Measure
Custom JMX Percentage Measure
Custom JMX Rate Measure
Custom JMX Time measure
Custom Windows Performance Counter
Database
Exceptions
Java Virtual Machine
JBoss
Cache Access Count
Cache Hits Count
Custom JMX Byte Measure
Custom JMX Number Measure
Custom JMX Percentage Measure
Custom JMX Time Measure
Entity Bean Create Count
Entity Bean Pooled Count
Entity Bean Ready Count
Measure
Configuration
Details
JMX
Subscription
System Profile: Default
Group/Measure: Custom JMX - Custom JMX Measure
Measure Name: Custom JMX Measure
Description: Please provide object name and attribute data to specify a custom JMX Measure.
General Attributes
Create Dynamic Measures
Generate Aggregated Timeseries
Generate and store Charting Data in the Repository
Measure Subscription
All Agent Groups
Agent Group
GoSpace
Hint
Configure general measure attributes.
Add
Close

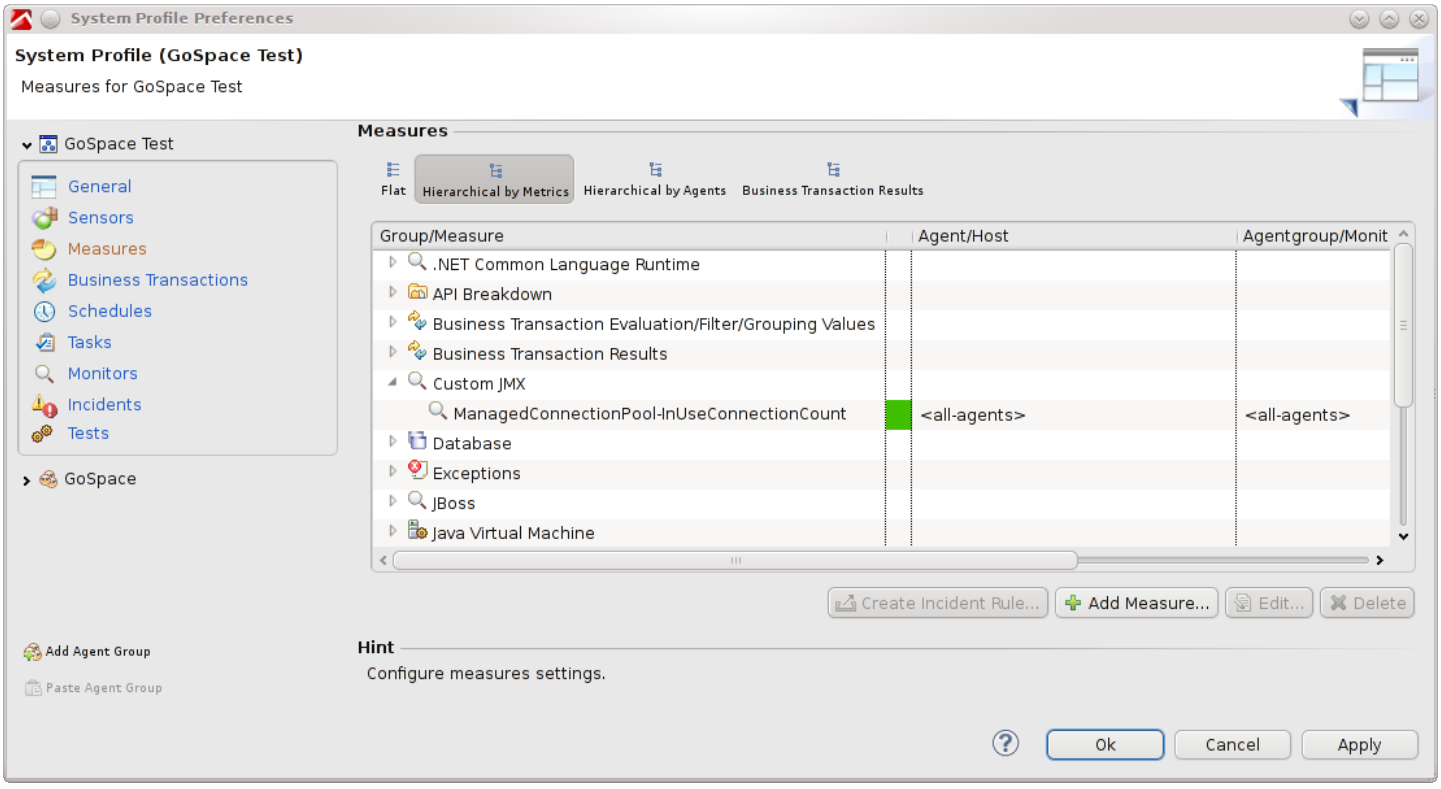

Metric Browser
Browse available JMX Metrics
Agent
Agent Group:
Pick Agent...
Agent Mapping:
Agent Process Id:
MBean
Domain:
Key
Value
Attribute
Attribute name:
Path Expression:
Test configuration
Test
Value:
Hint
Ok
Cancel


Agent Browser
Select agent configurations or agent instances
Agent Configurations
GoSpace Test
Agent Group 'GoSpace'
Agent Mapping 'GoSpace'
GoSpaceBackend@KOMMA:32612
GoSpaceFrontend@KOMMA:32676
Hint
Ok
Cancel


Metric Browser
Browse available JMX Metrics
Catalina
JMImplementation
com.arjuna.ats.properties
jboss
jboss.deployment
jboss.ejb
jboss.j2ee
jboss.jca
jboss.jdbc
jboss.jms
jboss.management.local
jboss.mq
jboss.mq.destination
jboss.remoting
jboss.rmi
jboss.security
jboss.system
jboss.web
jboss.web.deployment
Agent
Agent Group:
GoSpace
Pick Agent...
Agent Mapping:
GoSpace
Agent Process Id:
32612
MBeans: 375 Attributes: 3563
MBean
Domain:
Key
Value
Attribute
Attribute name:
Path Expression:
Test configuration
Test
Value:
Hint
Ok
Cancel


Metric Browser
Browse available JMX Metrics
JMImplementation
com.arjuna.ats.properties
jboss
jboss.deployment
jboss.ejb
jboss.j2ee
jboss.jca
'jboss-local-jdbc.rar'
'jms-ra.rar'
CachedConnectionManager
ConnectionFactoryDeployer
DefaultDS
DataSourceBinding
LocalTxCM
ManagedConnectionFactory
ManagedConnectionPool
AvailableConnectionCount
BackGroundValidation
BackGroundValidationMinutes
BlockingTimeoutMillis
ConnectionCount
ConnectionCreatedCount
ConnectionDestroyedCount
Criteria
IdleTimeoutMinutes
InUseConnectionCount
Agent
Agent Group: GoSpace
Pick Agent...
Agent Mapping: GoSpace
Agent Process Id: 32612
MBeans: 375 Attributes: 3563
MBean
Domain: jboss.jca
Key
Value
name
DefaultDS
service
ManagedConnectionPool
Attribute
Attribute name: InUseConnectionCount
Path Expression:
Test configuration
Test
Value:
Hint
MBean Attribute.
Ok
Cancel


Measure Configuration
Create a new Measure for your subscription
Create a new Measure.
Select Metric
.NET Common Language Runtime
.NET Networking
ADO.NET
API Breakdown
ASP.NET
Availability
Browser Performance
Business Transaction Evaluation/Filter/Grouping
Custom JMX
Custom JMX Byte Measure
Custom JMX Measure
Custom JMX Percentage Measure
Custom JMX Rate Measure
Custom JMX Time measure
Custom Windows Performance Counter
Database
Exceptions
Java Virtual Machine
JBoss
Logging
Messaging
Methods
Naming Services
Measure
Configuration
Details
JMX
MBean Object Name
Browse Metrics...
Domain:
jboss.jca
Key Properties:
Key
Value
Exclude
name
DefaultDS
service
ManagedConne
Include similar MBeans
Attribute
Name:
InUseConnectionCount
Path Expression:
Aggregation:
avg
Hint
Customize specific measure attributes.
Add
Close


System Profile Preferences
System Profile (GoSpace Test)
Measures for GoSpace Test
GoSpace Test
General
Sensors
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Tests
GoSpace
Add Agent Group
Paste Agent Group
Measures
Flat
Hierarchical by Metrics
Hierarchical by Agents
Business Transaction Results
Group/Measure
Agent/Host
Agentgroup/Monit
.NET Common Language Runtime
API Breakdown
Business Transaction Evaluation/Filter/Grouping Values
Business Transaction Results
Custom JMX
ManagedConnectionPool-InUseConnectionCount
<all-agents>
<all-agents>
Database
Exceptions
JBoss
Java Virtual Machine
Create Incident Rule...
Add Measure...
Edit...
Delete
Hint
Configure measures settings.
Ok
Cancel
Apply

JMX attribute note

If the JMX attribute type is a bean that provides getter methods to access measurement values, use the path expression to specify how to access the measurement values. If the bean provides the method getValue(), which returns the desired measurement, access this value with a JMX subscription and set the path expression to getValue().

The following table provides values, path expressions and explanations for attribute types:

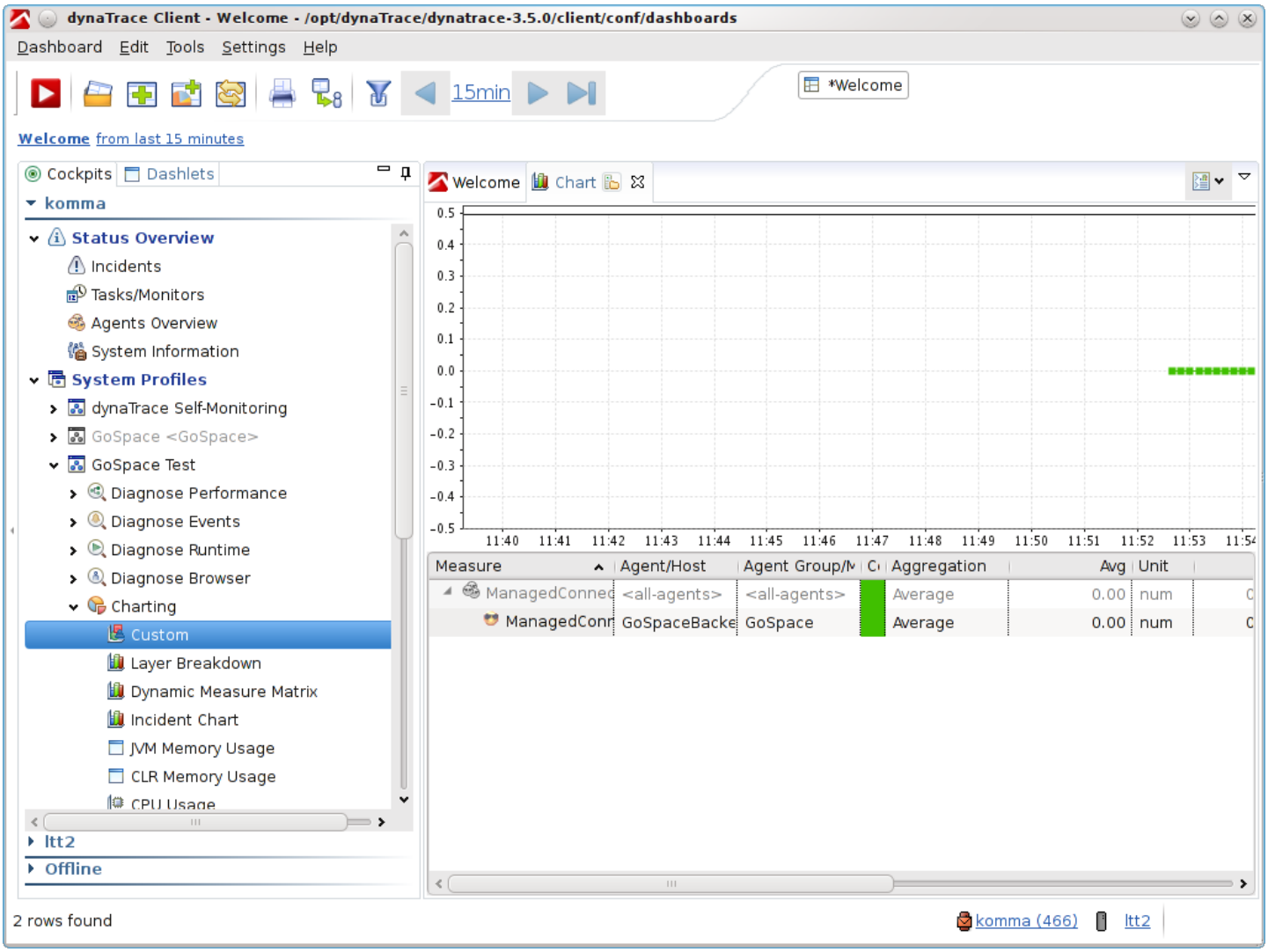
Attribute Type	Desired Value	Path Expression	Explanation
getter	Return value	`getValue()`	Calls method `getValue()` on the JMX attribute.
nested getter	Return value of nested getter	`getValue().getNestedValue()`	Calls method `getValue()` on the JMX attribute. Calls method `getNestedValue` on the object returned by `getValue()`.
getter with parameter	Return value for specific parameter	`getValue(param)`	Calls method `getValue()` with one string parameter, with parameter value `param`.
getter returning collection or array	Aggregate of values in collection or array	`getValue().size()`	Iterates over elements of collection or array. For a map, it iterates over the values of the map. Calls method `size()` on each element and calculates an aggregate value. For example, the sum average according to the aggregation configuration of the JMX subscription.
getter returning collection or array	Value of a specific element of the collection or array	`getValue()[key].size()`	Fetches the element identified by `?key` from the collection or array. Calls method `size()` on this element.

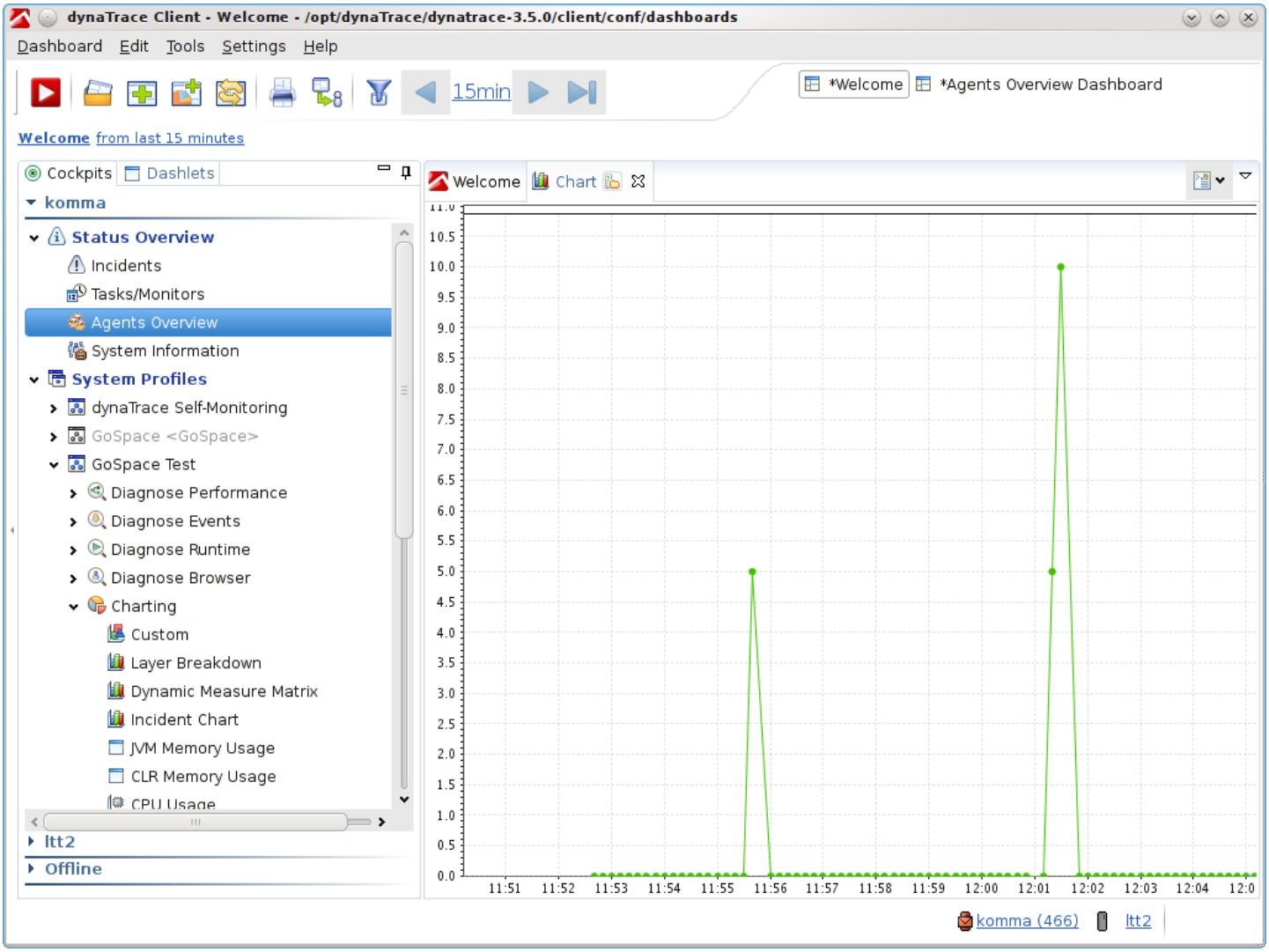
Chart the measure

Now chart the new measure.

1. Open a new custom chart.
2. Right-click and select **Add Series**.

3. Select the new measure in the new dialog, and add it. If you execute a load, the connection count should increase.





Use this measure to define incidents just like you define measures. If you want to do the same on WebSphere, use the WebSphere Custom PMI metric.

How to retrieve performance metrics with Windows Performance Monitor

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Goal of this walkthrough

Prerequisites

User authentication and permissions

Hardening the monitored systems

Testing performance monitor query

Configuring the AppMon Collector

Related pages

Setting up infrastructure monitoring

Setting up application availability monitoring

Using host and process health monitoring

Using host and hostgroups for large system monitoring

Using custom JMX and PMI measures in charts

Set up collectors

External links

Microsoft typeperf command

Windows RPC dynamic ports configuration

Goal of this walkthrough

The Microsoft Windows Performance Monitor (PerfMon) moves performance data from Windows based machines into a monitoring environment. To collect this data in AppMon, you'll do some configuration that affects your Windows environment and the AppMon Collector.

This walkthrough describes how to set up and configure users, permissions, services, and the AppMon Collector. It also covers security on hardening the monitoring solution.

Scenario

The following information has been tested on Windows Server 2003 and 2008 (R2), but should work for other versions. It covers configuring the monitored machine and the AppMon collector. You'll configure the operation system, so you should have administrative rights and some operation knowledge.

> **Note**
>
> Execute the Windows Performance Monitor on a Windows machine. If it is configured to monitor a remote machine, it still depends on the Windows API to query these metrics.

Prerequisites

Required services

Run the following services on the monitored machine:

Service	Service Name
Remote Registry Service	`RemoteRegistry`
Remote Procedure Call (RPC)	`RpcSs`

Do a local check to make sure those services are running and accessible. To validate, execute the following commands:

RemoteRegistry

```
C:\>sc query RemoteRegistry

SERVICE_NAME: RemoteRegistry
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE              : 4  RUNNING
                                (STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```

Remote Procedure Call (RPC)

```
c:\>sc query RpcSs

SERVICE_NAME: RpcSs
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE              : 4  RUNNING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```

Both services should return the `RUNNING` state.

Firewall settings

To access the above services via a network, make sure that a local firewall does not block traffic. In Windows 2008 the Advanced Firewall does not allow general access, by default. If the monitored machine is a member of an active directory domain, the firewall may be enforced through group policies. Also if the monitoring machine (AppMon Collector) is in a different domain, no domain or subnet, firewall rules may block the access.

In Windows 2008, enable the predefined Remote Service Management (RPC) rule. To do this, use the `netsh` command. The name may be different depending on the OS language.

```
c:\> netsh advFirewall firewall show rule name="Remote Service Management (RPC)"

Rule Name:                            Remote Service Management (RPC)
----------------------------------------------------------------------
Enabled:                              No
Direction:                            In
Profiles:                             Domain,Private,Public
Grouping:                             Remote Service Management
LocalIP:                              Any
RemoteIP:                             Any
Protocol:                             TCP
LocalPort:                            RPC
RemotePort:                           Any
Edge traversal:                       No
Action:                               Allow
Ok.
```

This output means the rule is disabled, and the RPC is not available via remote. Note the profiles to which this rule applies. Make sure you use the correct firewall profile, especially if it is customized in an Active Directory domain environment. Enable this rule for `RemoteIP *Any*`, or your desired source `subnet/ip`, if your monitoring station is located in another network segment.

To enable it, see the following codeblock:

```
c:\> netsh advFirewall firewall set rule name="Remote Service Management (RPC)" new enabl

Updated 1 rule(s).
Ok.
```

Windows Server 2003 does not have predefined rules like Windows Server 2008. It defines a few services that contain the necessary settings. Make sure that the following ports are open, so they can be accessed from the source machine:

Port	Service
445	MS RPC

The default configuration contains the file and print services that enables port 445. If you don't need the other services, disable them. Again, use the `netsh` command to enable this service:

```
c:\>netsh firewall set service type=FILEANDPRINT mode=ENABLE
Ok.
```

```
c:\>netsh firewall show portopening

Port configuration for Domain profile:
Port   Protocol  Mode     Name
-------------------------------------------------------------------
139    TCP       Enable   NetBIOS Session Service
445    TCP       Enable   SMB over TCP
137    UDP       Enable   NetBIOS Name Service
138    UDP       Enable   NetBIOS Datagram Service

Port configuration for Standard profile:
Port   Protocol  Mode     Name
-------------------------------------------------------------------
139    TCP       Enable   NetBIOS Session Service
445    TCP       Enable   SMB over TCP
137    UDP       Enable   NetBIOS Name Service
138    UDP       Enable   NetBIOS Datagram Service
3389   TCP       Enable   Remote Desktop
```

See the notes about dynamically assigned ports for DCOM RPC at the end of this page.

Remote accessibility of services

Test the availability of the necessary services on the monitored machine. Make sure to run these commands as the monitoring user:

Remote Registry

```
C:\>sc \\<server> query RemoteRegistry

SERVICE_NAME: RemoteRegistry
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE              : 4  RUNNING
                                (STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```

Remote Procedure Call (RPC)

```
C:\>sc \\<server> query RpcSs

SERVICE_NAME: RpcSs
        TYPE               : 20  WIN32_SHARE_PROCESS
        STATE              : 4  RUNNING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```

DCOM

If the previous checks were successful, but slow (1s), verify that firewall profile allows DCOM (Port 135). Make sure the rule is valid for your source network and the current active firewall profile.

Check the current active profile for the Advanced Windows Firewall:

```
c:\>netsh advFirewall show currentprofile

Public Profile Settings:
----------------------------------------------------------------------
State                                 ON
Firewall Policy                       BlockInbound,AllowOutbound
LocalFirewallRules                    N/A (GPO-store only)
LocalConSecRules                      N/A (GPO-store only)
InboundUserNotification               Disable
RemoteManagement                      Disable
UnicastResponseToMulticast            Enable

Logging:
LogAllowedConnections                 Disable
LogDroppedConnections                 Disable
FileName                              %systemroot%\system32\LogFiles\Firewall\pfirewall.l
MaxFileSize                           4096

Ok.
```

The predefined `Performance Logs and Alerts (DCOM-In)` rule enables DCOM, depending on the profile subnet:

```
c:\>netsh advFirewall firewall show rule name="Performance Logs and Alerts (DCOM-In)"

Rule Name:                            Performance Logs and Alerts (DCOM-In)
----------------------------------------------------------------------
Enabled:                              No
Direction:                            In
Profiles:                             Domain
Grouping:                             Performance Logs and Alerts
LocalIP:                              Any
RemoteIP:                             Any
Protocol:                             TCP
LocalPort:                            135
RemotePort:                           Any
Edge traversal:                       No
Action:                               Allow

Rule Name:                            Performance Logs and Alerts (DCOM-In)
----------------------------------------------------------------------
Enabled:                              Yes
Direction:                            In
Profiles:                             Private,Public
Grouping:                             Performance Logs and Alerts
LocalIP:                              Any
RemoteIP:                             LocalSubnet
Protocol:                             TCP
LocalPort:                            135
RemotePort:                           Any
Edge traversal:                       No
Action:                               Allow
Ok.
```

Make sure the rule matches your setup, and enable or modify the rule. Another default rule enables DCOM for any subnet. Check your Policy before you use this.

```
c:\>netsh advFirewall firewall show rule name="DFS Management (DCOM-In)"

Rule Name:                            DFS Management (DCOM-In)
----------------------------------------------------------------------
Enabled:                              No
Direction:                            In
Profiles:                             Domain,Private,Public
Grouping:                             DFS Management
LocalIP:                              Any
RemoteIP:                             Any
Protocol:                             TCP
LocalPort:                            135
RemotePort:                           Any
Edge traversal:                       No
Action:                               Allow
Ok.
```

User authentication and permissions

To access the performance data of a machine remotely, the querying user must have access. For that purpose, Windows 2003/2008 provides the built-in group Performance Monitor Users. The monitoring user must be a member of this group and be known on both machines. On the source machine this user must be a member of the Performance Monitor Users for access rights to certain registry paths.

Scenario: source and target machines are members of a domain

If the source and target monitoring machines are members of the same Active Directory domain, follow these setup steps:

1. Create a Domain Monitoring User (ID monitor) in Active Directory.
2. Add this user to the local group **Performance Monitor Users** on each machine you query, and on the machine that executes the AppMon Collector. If this is a large number of machines, use the central group policy.
3. Configure the AppMon Service to run as the Monitor user.

Scenario: source and target machines are not members of a domain

If the machines are single hosts with independent user accounts and security policies, follow these setup steps:

1. Create the monitor user on all machines, with the same ID and password.
2. Add the monitor user to the local group Performance Monitor Users on all machines.
3. Configure the AppMon Service to run as the monitor user.

Scenario: source machine is not in a domain, target machine is

If the monitoring machine is not a member of a domain but the monitored machine is, you can't use a domain user to execute the performance metric queries. Use the same setup that you would for two independent machines.

Scenario: machines in different domains or forests

If the machines are in different domains, you need appropriate trusts and delegations established in the forest. This is similar to the first scenario. You may want to use global and domain local groups, instead of single users for granting rights.

Hardening the monitored systems

User permissions and groups

The monitor user needs no additional rights. In production systems, limit permissions to those who are necessary and adapt your security policy. By default, a newly created user belongs to the Users or Domain Users group, which isn't required for the monitoring tasks. The users should be removed from those groups. They should be members of the Performance Monitor Users.

File permissions

On the AppMon Collector machine, the monitor user running with the Collector service requires full access to the AppMon Collector installation directory. If you have installed the collector with another user, like an Administrative user, change the file permissions accordingly.

Local and global security policy

Use the monitor for monitoring only. Deny the interactive login locally and remotely for this user. The user must log in as a service. To do this, edit your applicable local or global (in an Active Directory domain) security policy to deny login rights to the Monitor user.

For single hosts, follow these steps:

1. Edit the **Local Computer Policy** via MMC on each monitored machine.
2. Go to `Computer Configuration\Windows Settings\Security Settings\Local Policies\User Rights Assignment`.
3. Add the user or group to the **Deny Login Local** policy. If you are in a domain, do this centrally.

Testing performance monitor query

To test the setup, follow these steps and do not restart the AppMon Collector service. If the setup is successful, the AppMon Collector should query Windows performance data.

1. Execute the commands on the monitoring machine (the machine on which the AppMon collector is running).

2. Log in with the monitor user (recommended) or open a command shell as the monitor user. Use the `*runas*` command.

```
C:\>runas /user:monitor@yourdomain.local cmd
Enter the password for monitor@yourdomain.local:
Attempting to start cmd as user "monitor@yourdomain.local" ...
```

3. In the newly opened command shell use `typeperf` to test if you can query the remote machine performance counters, such as one memory counter:

```
C:\>typeperf -sc 2 "\\<monitored machine name>\Memory\Available MBytes"

"(PDH-CSV 4.0)","\\2008VMX64\Memory\Available MBytes"
"10/01/2010 15:22:09.602","2220.000000"
"10/01/2010 15:22:10.605","2220.000000"

The command completed successfully.
```

If you can successfully query the performance counters, you should be able to set up the AppMon Windows Performance Monitor**.** If you do not get any data, check your firewall and permission settings. Also, make sure you use the correct user to execute the query.

Use the `typeperf` command to check which performance counters and objects are available on your monitored machines. To add custom counters that are not predefined in AppMon, use this command to identify them or export them to a text file, rather than using `*perfmon.exe*`. To query all counters for a specific object use the following code:

```
c:\> typeperf \-q Processor
\Processor\(*)\% Processor Time
\Processor\(*)\% User Time
\Processor\(*)\% Privileged Time
\Processor\(*)\Interrupts/sec
\Processor\(*)\% DPC Time
\Processor\(*)\% Interrupt Time
\Processor\(*)\DPCs Queued/sec
\Processor\(*)\DPC Rate
\Processor\(*)\% Idle Time
\Processor\(*)\% C1 Time
\Processor\(*)\% C2 Time
\Processor\(*)\% C3 Time
\Processor\(*)\C1 Transitions/sec
\Processor\(*)\C2 Transitions/sec
\Processor\(*)\C3 Transitions/sec
```

Configuring the AppMon Collector

Configure the AppMon Collector Service to run as the monitor user. The Windows plugin executes as this user via the Collector and derives its permissions. Make sure that the user that runs with the collector service has full access rights on the AppMon Collector install directory.

For more information, see Collector Configuration..

Database monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Database Agent

Database Agent configuration

Auto-mapping of Database Agents into System Profiles

Access database monitoring data

Related pages

Database health monitoring

Database live view

Database monitoring measures

SQL execution plan

Database Agent configuration

Database dashlet

Database Hotspots dashlet

JDBC sensor

ADO.NET sensor

PHP Agent configuration

Watch and learn

Database diagnostics use cases with Dynatrace

Analyzing databases

Blog posts

Using bind variables to improve Oracle database performance

Fixing SQL Server plan cache bloat with parameterized queries

For AppMon 6.3 and later, database monitoring is available on the DB instance level, giving insights into DBMS health, request frequency, execution plans and CPU and disk, comparable to the infrastructure overview of hosts and processes. In AppMon 2017 May, database monitoring is available for **Oracle Database** (including Oracle **RAC**), **Microsoft SQL Server**, **MySQL** and **IBM DB2**.

This complements the fully integrated, application-related capturing of connection pool usage, SQL statements and transaction response time contribution, that has been available for each and every DB call for a long time via JDBC, ADO.NET and PHP DB sensors in Java, .NET and PHP Agents respectively.

Data for any monitored database appearing in a PurePath automatically maps into the System Profile.

Database Agent

The **Database Agent** directly runs on the AppMon Collector and as such is delivered out-of-the-box and no other components are needed. It connects to a database instance and retrieves data every 30 seconds (measures) or on-demand (live view, execution plan).



Database Agent configuration

Database Agent configuration

The only configuration necessary is to enter the connection credentials for the DBMS instances to be monitored and to select the Collector the Agent is part of.
See **Settings > Dynatrace Server > Databases** or Database Agent Configuration if you need instructions for the dialog box.

Auto-mapping of Database Agents into System Profiles

To merge global DB-monitoring data with application-specific ones (SQL statements, etc.), Database Agents are dynamically mapped into the System Profiles – if a database tier is detected on a PurePath and that same database (instance) is monitored. This mapping can be checked in the Agent Overview of a specific System Profile.

A database mapping into a specific System Profile is a prerequisite for viewing database monitoring data in the WebUI, as WebUI views are always filtered by the selected System Profile.

Access database monitoring data

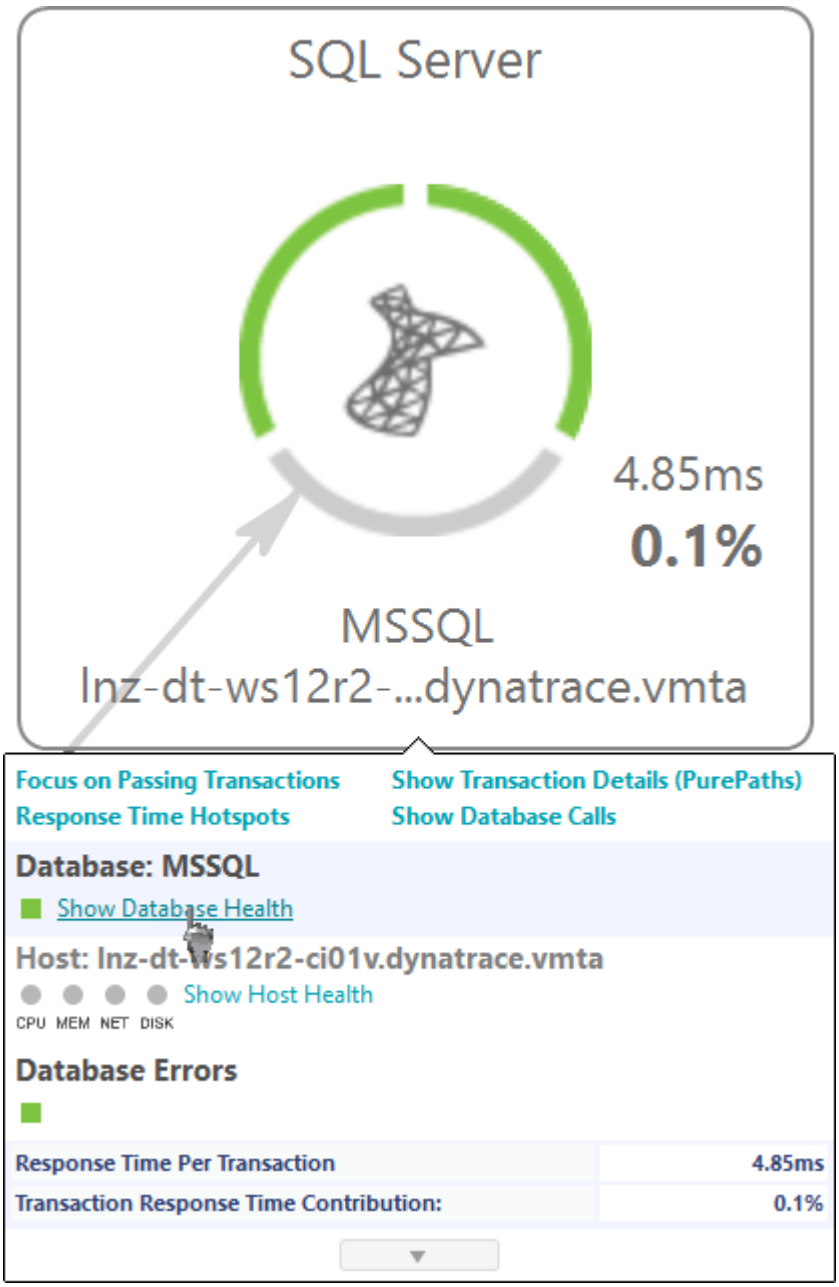
Access dashboards from the Monitoring Overview breadcrumbs or by a drill-down from a specific database node in the Transaction Flow.

Monitored database instances overview

The list of all monitored database instances is available in the Databases item of the cockpit. Double click it to open the list of monitored database instances.

Drill down from Transaction Flow to Database Monitoring dashboards

There are several drill-downs from a database-related Transaction Flow node to specific dashboards, such as Database Health.



Permissions

The following permissions in AppMon control access to database monitoring:

- Show Database Monitoring
- Manage Database Monitoring

Database health monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Oracle

SQL Server

MySQL

DB2

Related pages

Database Agent configuration

Database Agent troubleshooting

Database monitoring

SQL execution plan

Watch and learn

Database diagnostics use cases with Dynatrace

Blog posts

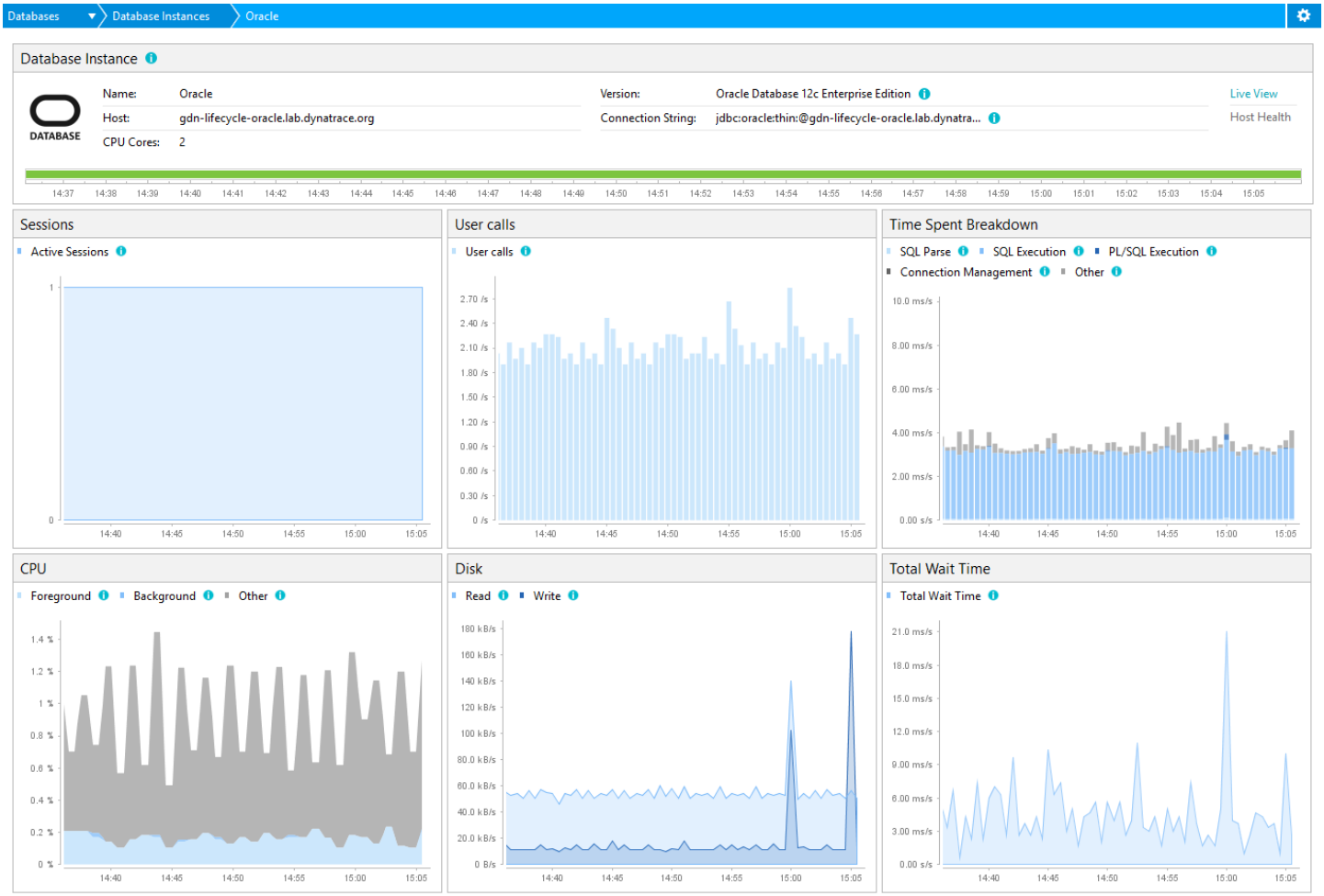
Using bind variables to improve Oracle database performance

Fixing SQL Server plan cache bloat with parameterized queries

The Database Instance Dashboard can be accessed by selecting a database instance either from the

breadcrumb navigation or from the transaction flow. The data is queried every 30 seconds by the Database Agent.

Oracle



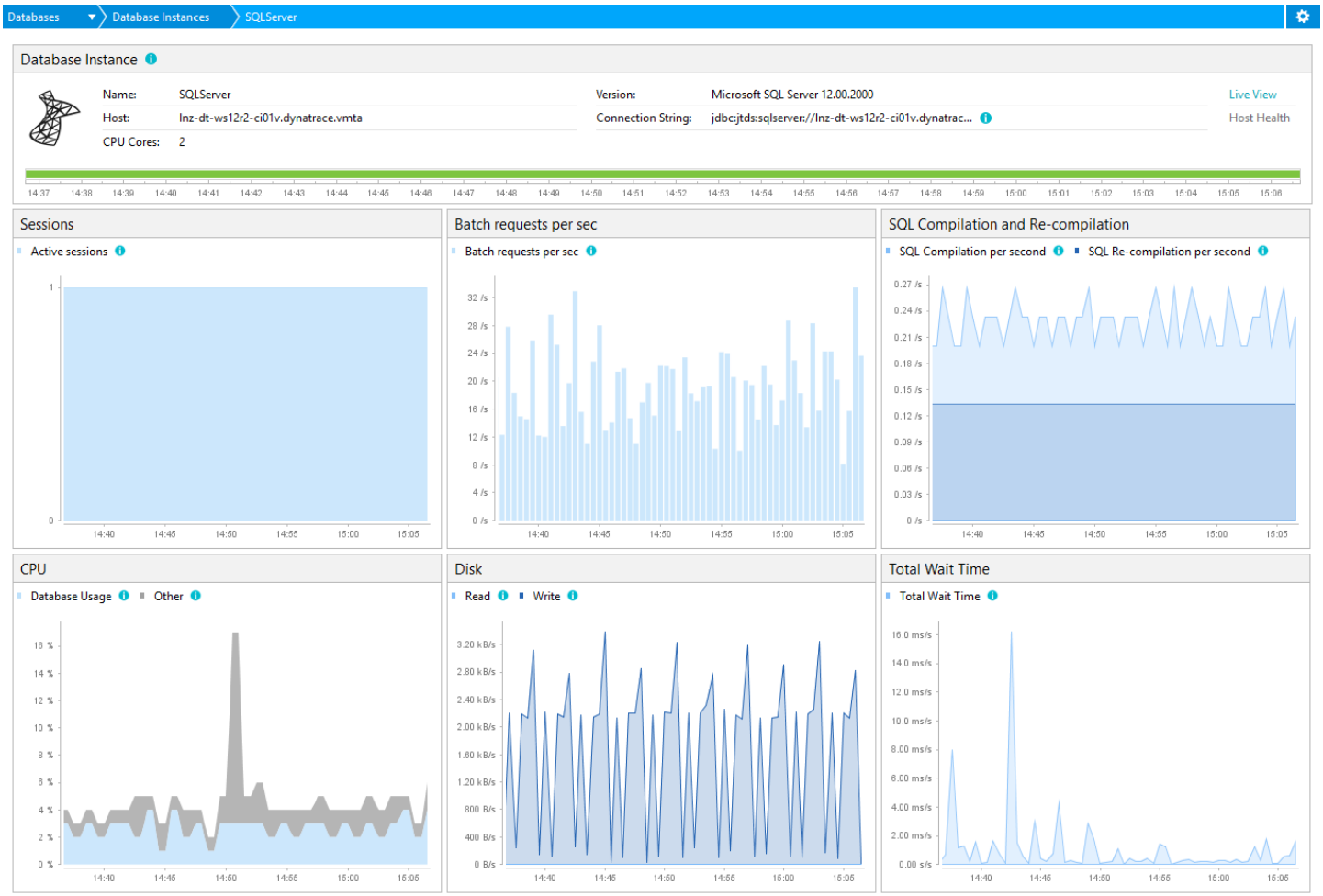
The following graphs are displayed for Oracle:

- **Sessions**: number of total and active sessions on the database instance
- **User calls**: number of log ins, parses and execute calls
- **Time Spent Breakdown**: show where the Oracle database uses most of its time (connection management, parsing or executing SQL statements or executing stored procedures).
- **CPU Usage**: CPU usage of the foreground & background processes of the database as well as other non-database related processes on the host
- **Disk I/O**: physical read & write from Oracle
- **Total Wait Time**: When an Oracle process can not get CPU time, it will pause. The time spent waiting on resources is called “wait time”.

See also:

- Blog: Using Bind Variables to Improve Oracle Database Performance
- Blog: Fixing SQL Server Plan Cache Bloat with Parameterized Queries

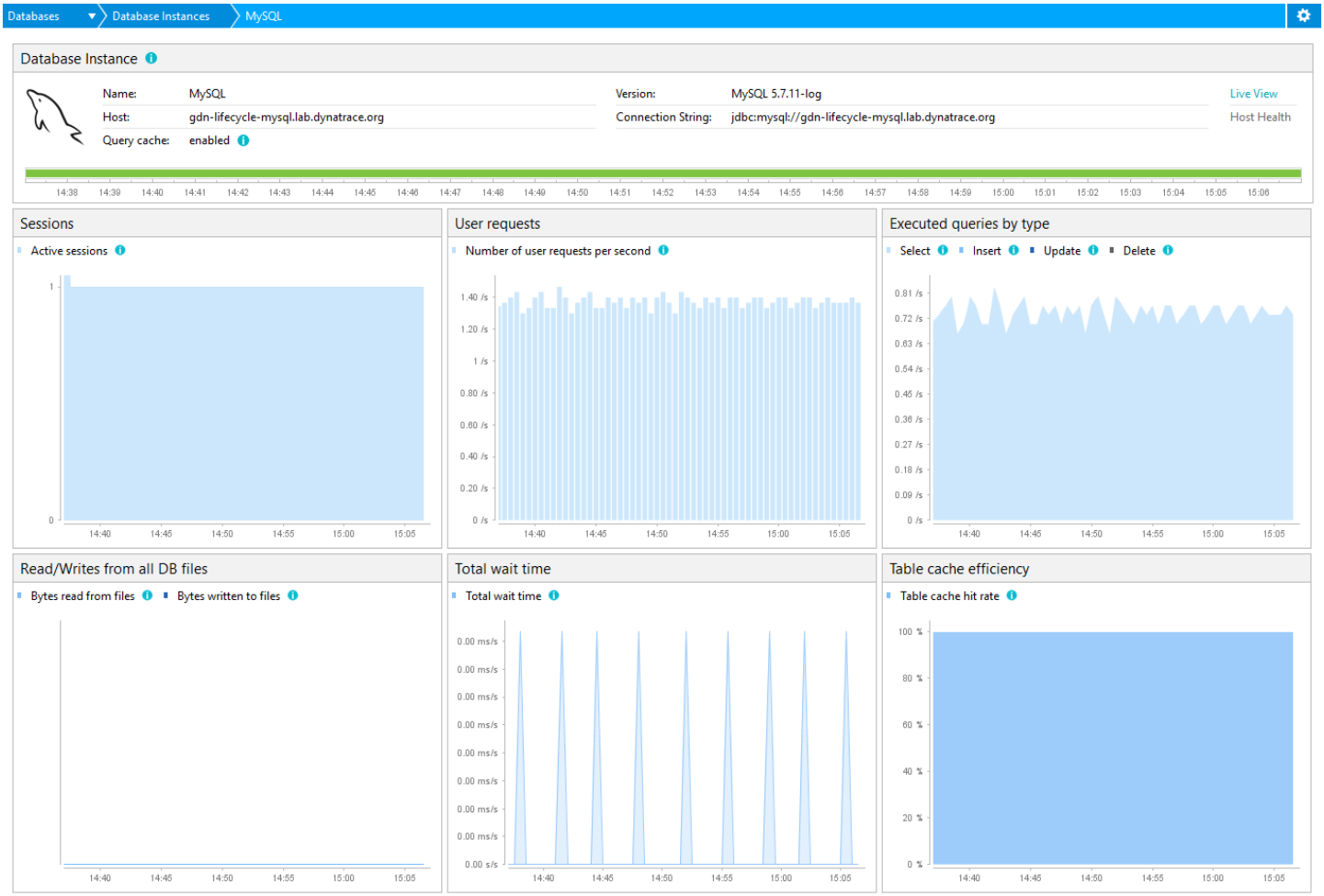
SQL Server



The following graphs are displayed for SQL Server:

- **Sessions**: number of total and active (running) sessions on the database instance
- **Batch requests per sec**: number of queries that are being executed per second
- **SQL Compilation & Re-Compilation**:
  - number of SQL compilations per second. Indicates the number of times the compile code path is entered. After SQL Server user activity is stable, this value reaches a steady state
  - number of statement recompiles per second. Counts the number of times statement recompiles are triggered. Generally, you want the number of recompilations to be low.
- **CPU**: CPU usage of the database instance and other processes on the host
- **Disk**: physical read & write by SQL Server
- **Total Wait Time**: total time SQL Server waits for resources.

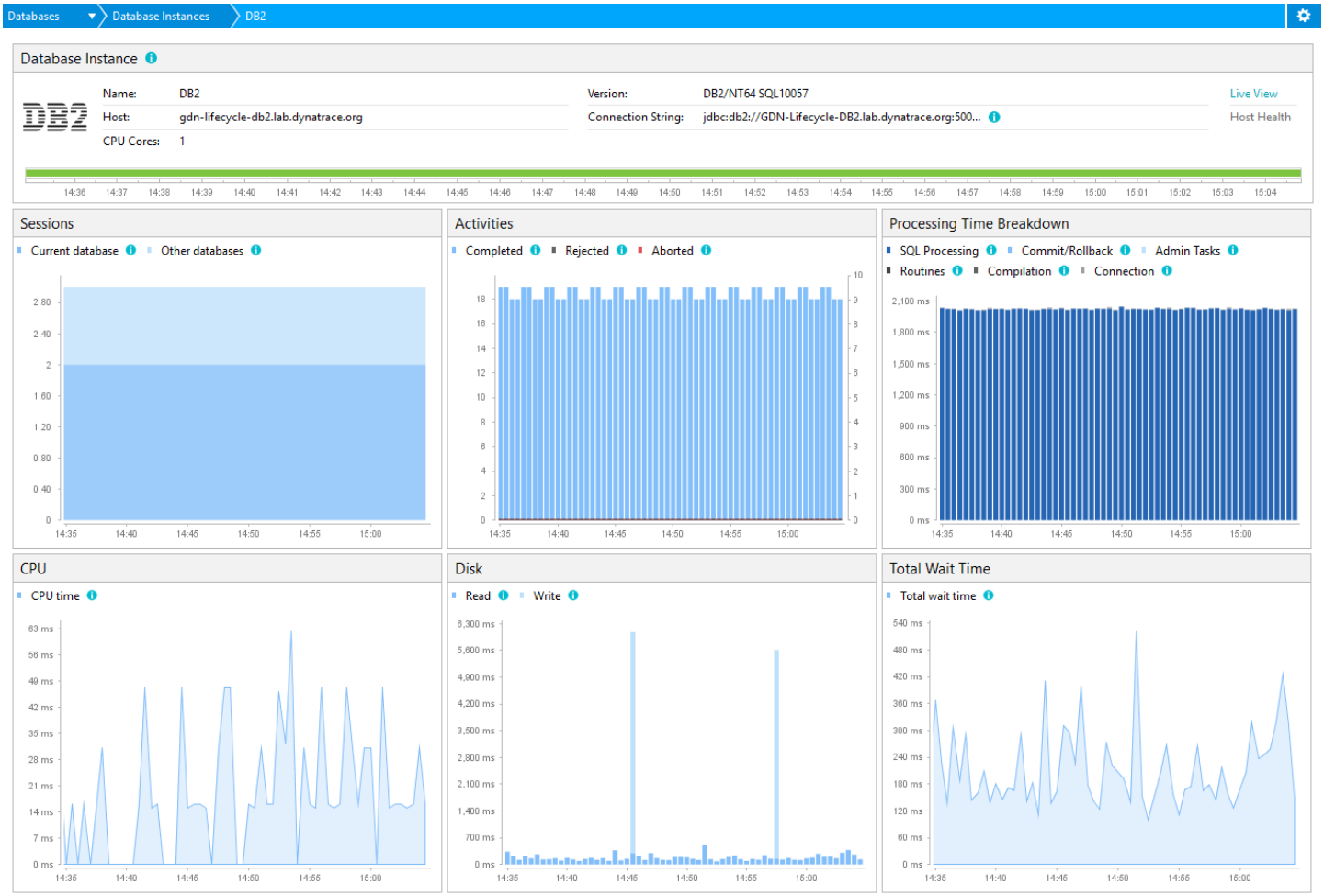
MySQL



The following graphs are displayed for MySQL:

- **Sessions**: number of active sessions on the database instance
- **User requests**: number of executed user queries per second
- **Executed queries by type**: division of queries by type
- **Read/Write form all DB files**: number of bytes read from and written to all database files
- **Total wait time**: total time spent waiting per second
- **Table cache efficiency**: table cache hit rate

DB2



- **Sessions**: number of sessions on the monitored and non-monitored database instances.
- **Activities**: number of successfully completed, rejected, and aborted activies.
- **Processing Time Breakdown**: processing time for database activities
- **CPU**: CPU usage of the database instance and other processes on the host
- **Disk**: physical read & write times
- **Total Wait Time**: total time spent waiting in milliseconds

Database live view

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Database monitoring

Database Agent configuration

Database Agent troubleshooting

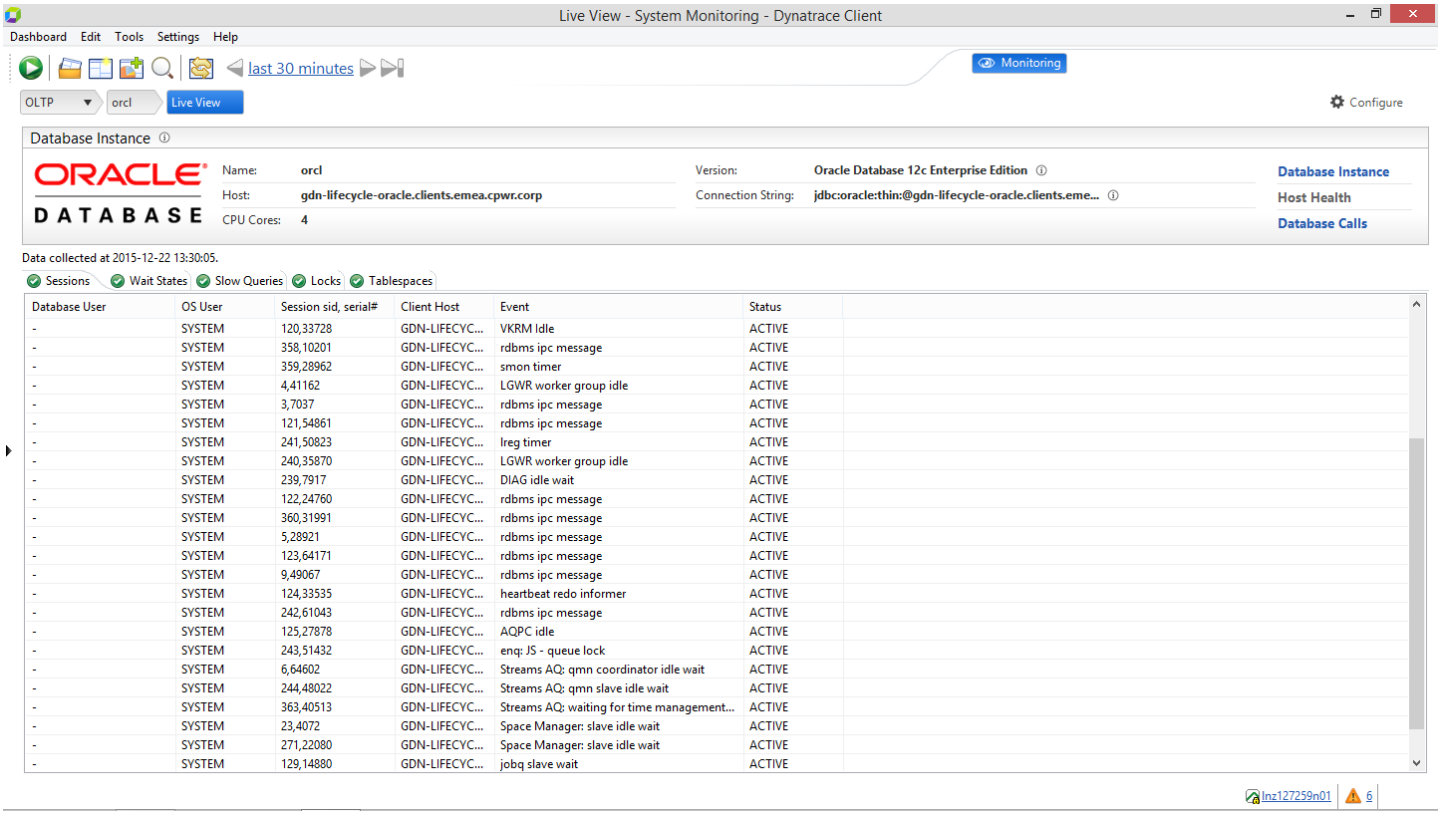
Watch and learn

Database diagnostics use cases with Dynatrace

The Live View gives you a quick overview on what is currently running on the database instance. All the data available in the Database Live View is retrieved in real time.

To access the Database Live View:

1. In the AppMon Client cockpit, expand **Server** and double-click **Databases** to open the **Database Instance** Dashboard.
2. In the database list, double-click the desired database entry. This opens a Database Instance dashboard for the selected database.
3. Click the Live View link in the upper right corner to open the Database Live view.



- **Sessions** / users connected to the database instance
- **Wait States**
- **Slow Queries** currently active
- **Locks** currently active
- **Tablespace** – total and free size; available for Oracle only

Sessions, Locks and Tablespace are queried in real-time. For wait states & slow queries, the Database Agent takes two snapshots with 15 seconds interval to determine to currently most active SQL statements and the most active wait states.

SQL execution plan

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Database monitoring

Database Agent configuration

Database Agent troubleshooting

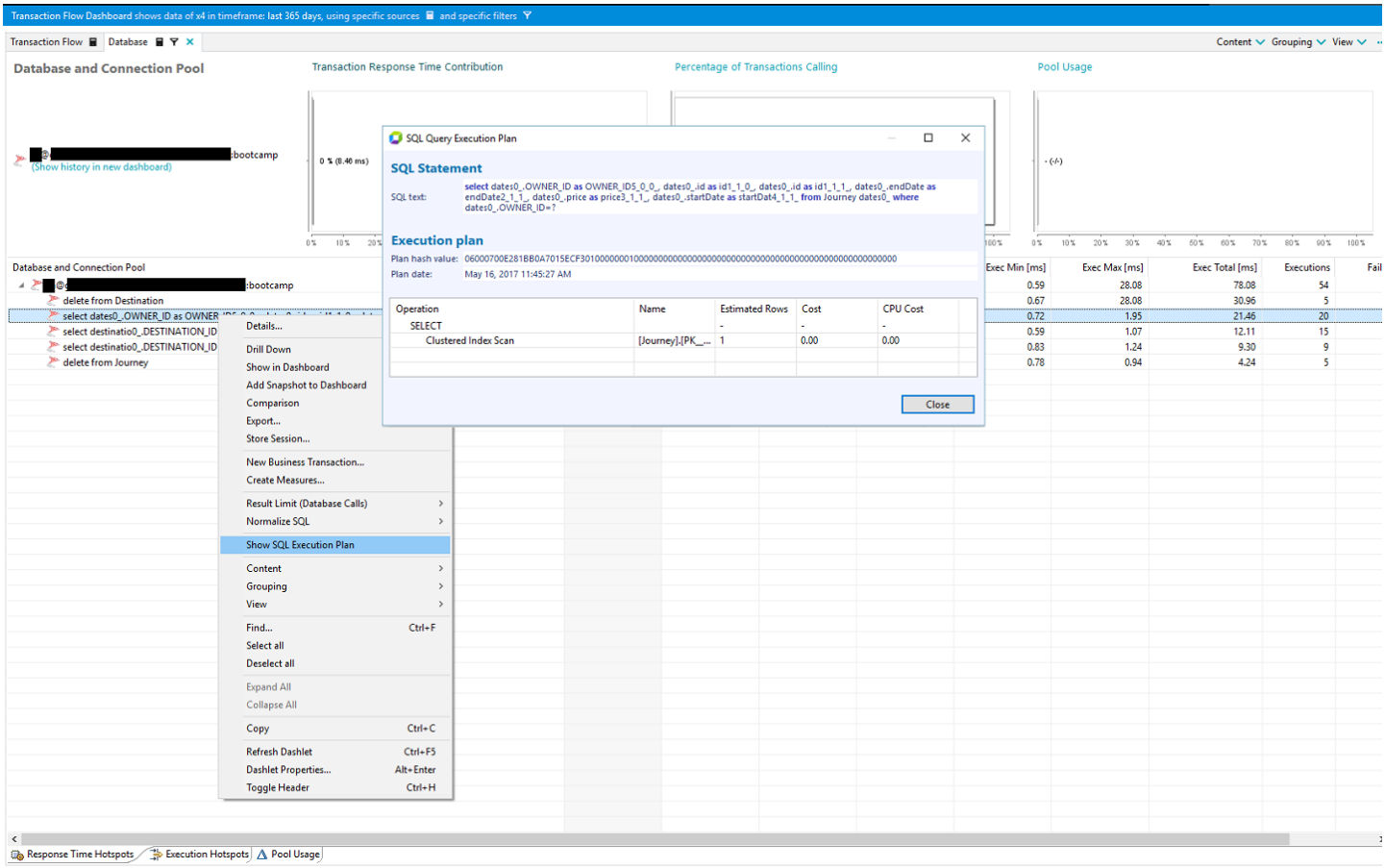
Watch and learn

Database diagnostics use cases with Dynatrace

A query execution plan is the optimized set of steps used to access data in a SQL relational database management system.
It is retrieved from the database cache on demand and is not stored in AppMon.

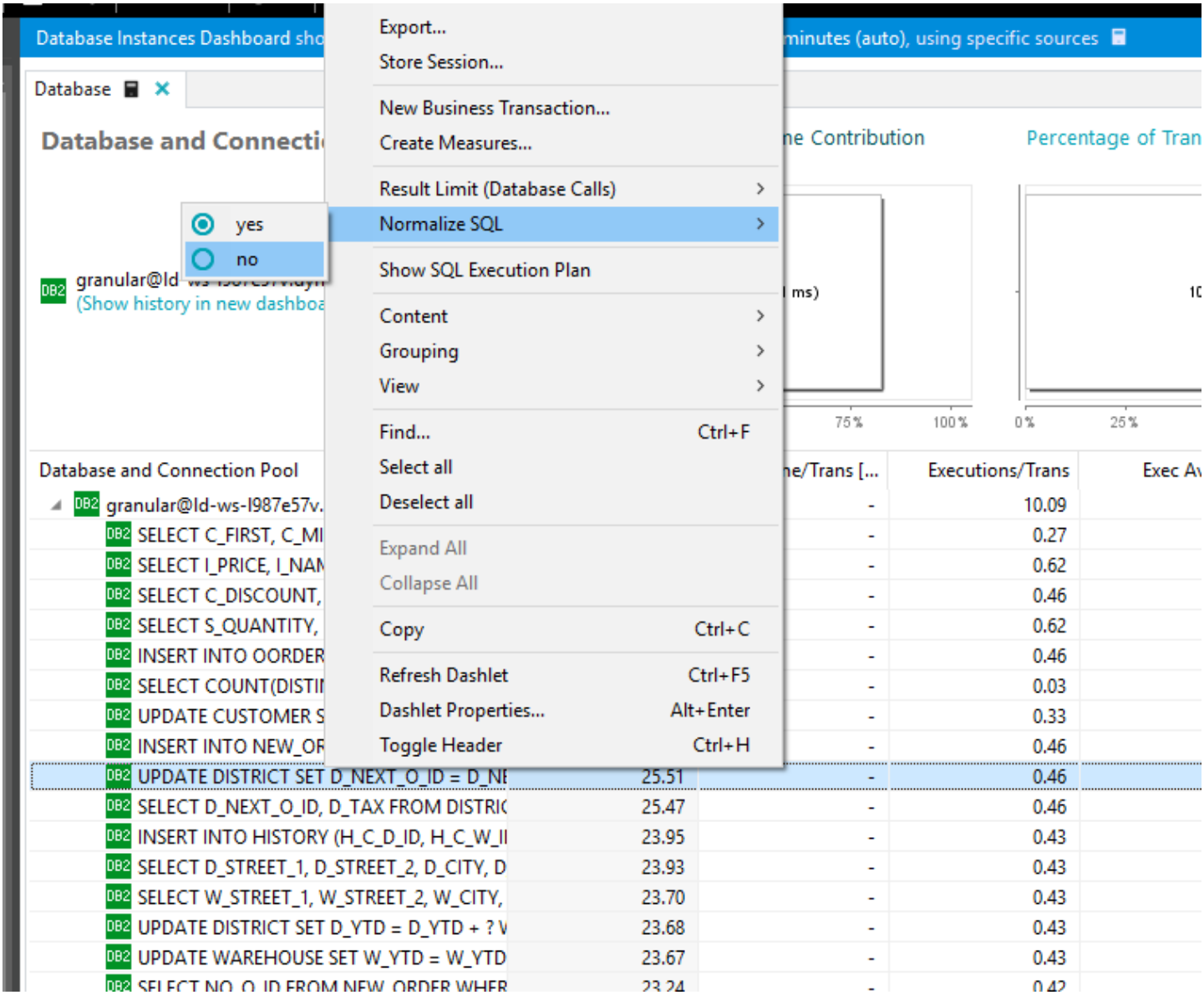
By examining it in AppMon, find answers if, why and where a query is slow and indices are used or not – or, setting timings in relation, a different component is the culprit.

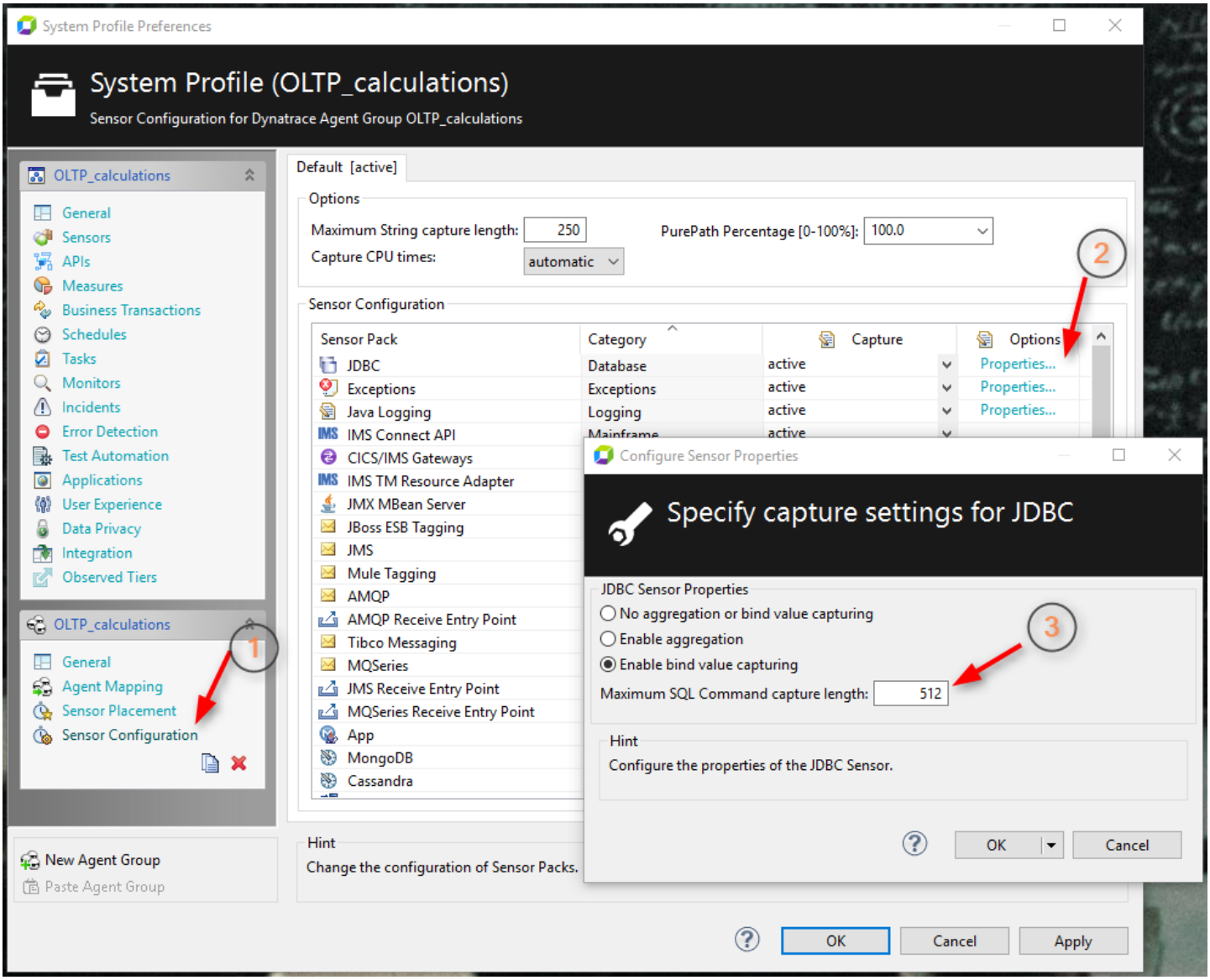
You can access execution plans from the context menu **Show SQL Execution Plan** ofa SQL statement on the Database Dashlet or a PurePath.



The following limitations apply:

- Stored procedures and functions are supported only for MSSQL. For other databases, statements that start with call, exec or {call should have the **Show SQL Execution Plan** option disabled.
- If the DB cache has been cleared, prior execution plans are not available.
- Not supported for the Oracle Database setting **cursor_sharing set to force**
- In order to request an execution plan the user must have the DB permission to view SQL statements and the AppMon one (**Settings menu > Dynatrace Server > Settings item > Confidential Strings horizontal tab**) to view confidential strings.
- Using the jTDS driver for MSSQL enabling query preprocessing (`JtdsStatement.setEscapeProcessing()` method) may cause problems with looking up query execution plans as the query text will get modified.
- On the Database dashlet the presented SQLs have their whitespace normalized. Disable whitespace normalization from the SQL query's context menu (see screenshot below)
- By default, for performance reasons, db-related sensors (JDBC, ADO.NET, PHP …) store only the first 512 characters of a query. Truncated queries will not match explain plans. This value is modified in the properties of the JDBC and ADO.NET sensor configurations, in the agent section of a system profile. (see screenshot below)


Database Instances Dashboard sho
minutes (auto), using specific sources
Database
Database and Connecti
ne Contribution
Percentage of Tran
yes
no
granular@ld
(Show history in new dashboa
Export...
Store Session...
New Business Transaction...
Create Measures...
Result Limit (Database Calls)
Normalize SQL
Show SQL Execution Plan
Content
Grouping
View
Find... Ctrl+F
Select all
Deselect all
Expand All
Collapse All
Copy Ctrl+C
Refresh Dashlet Ctrl+F5
Dashlet Properties... Alt+Enter
Toggle Header Ctrl+H
ms)
75 %
100 %
0 %
25 %
Database and Connection Pool
ne/Trans [...
Executions/Trans
Exec Av
granular@ld-ws-l987e57v.
SELECT C_FIRST, C_MI
SELECT I_PRICE, I_NAM
SELECT C_DISCOUNT,
SELECT S_QUANTITY,
INSERT INTO OORDER
SELECT COUNT(DISTI
UPDATE CUSTOMER S
INSERT INTO NEW_OR
UPDATE DISTRICT SET D_NEXT_O_ID = D_NE 25.51 - 0.46
SELECT D_NEXT_O_ID, D_TAX FROM DISTRIC 25.47 - 0.46
INSERT INTO HISTORY (H_C_D_ID, H_C_W_I 23.95 - 0.43
SELECT D_STREET_1, D_STREET_2, D_CITY, D 23.93 - 0.43
SELECT W_STREET_1, W_STREET_2, W_CITY, 23.70 - 0.43
UPDATE DISTRICT SET D_YTD = D_YTD + ? V 23.68 - 0.43
UPDATE WAREHOUSE SET W_YTD = W_YTD 23.67 - 0.43
SELECT NO_O_ID FROM NEW_ORDER WHER 23.24 - 0.42


System Profile Preferences
System Profile (OLTP_calculations)
Sensor Configuration for Dynatrace Agent Group OLTP_calculations
OLTP_calculations
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Test Automation
Applications
User Experience
Data Privacy
Integration
Observed Tiers
OLTP_calculations
General
Agent Mapping
Sensor Placement
Sensor Configuration
1
New Agent Group
Paste Agent Group
Default [active]
Options
Maximum String capture length: 250
PurePath Percentage [0-100%]: 100.0
Capture CPU times: automatic
2
Sensor Configuration
Sensor Pack
Category
Capture
Options
JDBC
Database
active
Properties...
Exceptions
Exceptions
active
Properties...
Java Logging
Logging
active
Properties...
IMS Connect API
Mainframe
active
CICS/IMS Gateways
IMS TM Resource Adapter
JMX MBean Server
JBoss ESB Tagging
JMS
Mule Tagging
AMQP
AMQP Receive Entry Point
Tibco Messaging
MQSeries
JMS Receive Entry Point
MQSeries Receive Entry Point
App
MongoDB
Cassandra
Hint
Change the configuration of Sensor Packs.
OK
Cancel
Apply
Configure Sensor Properties
Specify capture settings for JDBC
JDBC Sensor Properties
No aggregation or bind value capturing
Enable aggregation
Enable bind value capturing
3
Maximum SQL Command capture length: 512
Hint
Configure the properties of the JDBC Sensor.
OK
Cancel

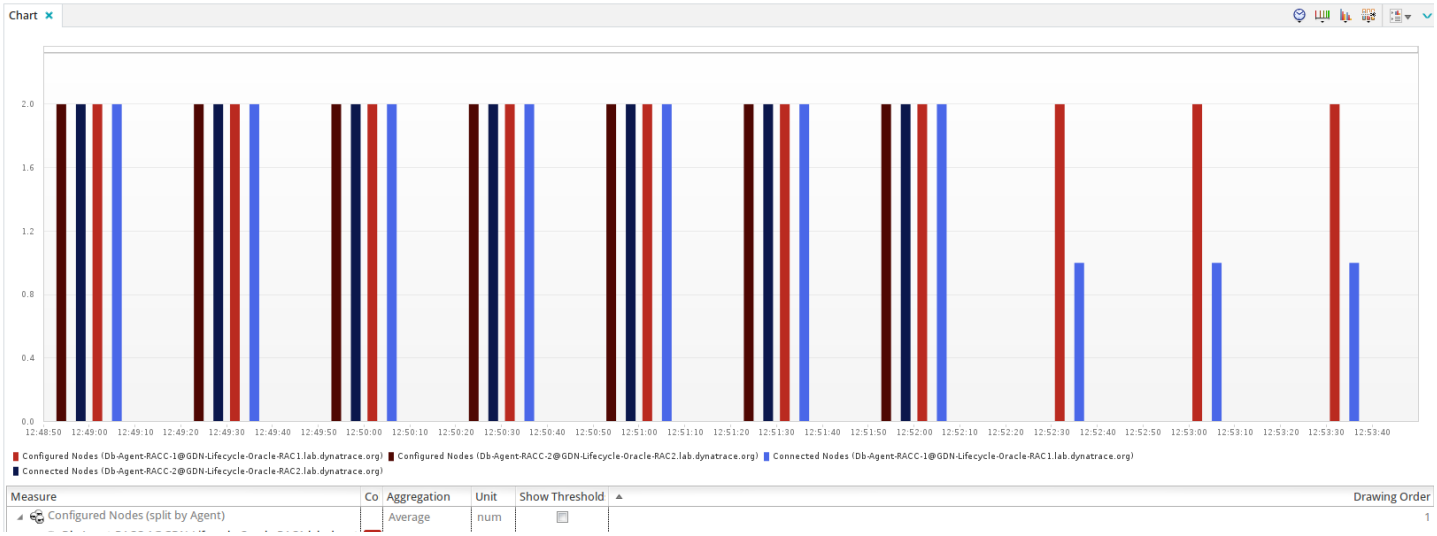
Database monitoring measures

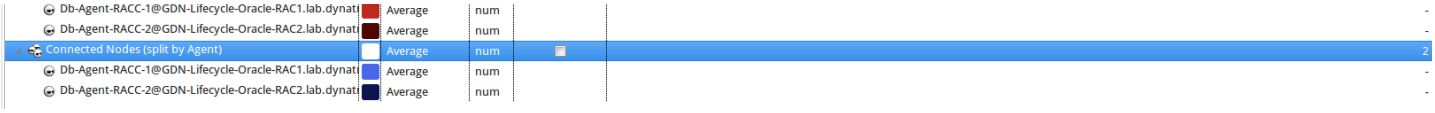
- **All sessions count**: Number of all authenticated sessions (including running, sleeping, dormant, and preconnected).
- **Running Session Count**: Number of authenticated sessions currently running one or more requests.
- **Batch requests per second**: Number of T-SQL command batches received by the server per second.
- **Database CPU idle**: Current CPU idle cycles, in percent. CPU is described as idle when it is not being used by any process.
- **Database CPU usage**: Current CPU load caused by the database processes, in percent. A change in the metric value may occur because of a change in either workload mix or workload throughput being performed by the database. Although there is no 'correct' value for this metric, it can be used to detect a change in the operation of a system.
- **Non Database CPU usage**: Current CPU load caused by non-database operating system processes, in percent. A change in the metric value may occur because of other processes executed on same host, including OS-specific processes. Although there is no 'correct' value for this metric, it can be used to detect a change in the operation of a system.
- **Number of bytes read per second issued on the files**: Number of bytes read per second, for all files in all databases in the SQL Server instance.
- **Number of bytes written per second issued on the files**: Number of bytes written per second, for all files in all databases in the SQL Server instance.
- **SQL Compilations per sec**: Number of query and procedure compilations per second.
- **SQL Re-Compilations per sec**: Number of query and procedure recompilations per second.
- **Total wait time per second**: Average time spent by all processes waiting on events per second. This includes waits encountered by threads that executed.

SQL Server

- **SQL parse time**: Average time spent on parsing SQL statements per second.
- **SQL execute time**: Average time spent on executing SQL statements per second. Note that this also includes the amount of time spent performing fetches of query results, but does not include network latency outside the instance, which is part of actual end-to-end response time for a SQL statement.

- **PL/SQL execute elapsed time**: Average time spent running the PL/SQL interpreter per second. This does not include time spent recursively executing/parsing SQL statements or time spent recursively executing the Java VM.
- **Connection management**: Average number of seconds spent on connection management per second, including time spent performing session connect and disconnect calls.
- **Time spent on other activities**: Average number of seconds spent on other activities per second. This includes time spent performing Database user-level calls, excluding instance background processes and four other activities measured by other metrics (SQL parse time, SQL execute time, PL/SQL execute elapsed time, Connection management).
- **Total wait time per second**: Average time spent by all processes waiting for events per second. This value is regarding instance-wide time for all non-idle wait classes (which is Application, Configuration, Administrative, Concurrency, Commit, Network, User I/O, System I/O, Scheduler and Other).
- **User calls count**: Average number of user calls (such as login, parse, fetch, or execute) per second.
- **CPU idle**: Current CPU idle cycles, in percent. CPU is described as idle when it is not being used by any process.
- **Foreground CPU usage**: Current CPU load caused by the foreground database processes, in percent. A change in the metric value may occur because of a change in either workload mix or workload throughput being performed by the database. Although there is no ‘correct’ value for this metric, it can be used to detect a change in the operation of a system.
- **Background CPU usage**: Current CPU load caused by the background database processes, in percent. A change in the metric value may occur because of a change in either workload mix or workload throughput being performed by the database. Although there is no ‘correct’ value for this metric, it can be used to detect a change in the operation of a system.
- **Non Database CPU usage**: Current CPU load caused by non-database operating system processes, in percent. A change in the metric value may occur because of other processes executed on same host, including OS-specific processes. Although there is no ‘correct’ value for this metric, it can be used to detect a change in the operation of a system.





This scenario is also presented on measure chart below:

- Offline node will stop reporting measures (because machine or database if offline)
- Another node will change “Connected nodes” measure value from 2.0 to 1.0 (since number of active nodes was reduced)
- Another node won’t change “Configured nodes” measure value, it will be 2.0 (since two nodes are configured as Oracle Database RAC)

Consider a scenario, when you have RAC cluster with two nodes, both monitored by AppMon Database Agent. In case you turn off one of the nodes, following things will happen:

Example

- **Active session count**: Number of active, non-idle sessions currently executing SQLs.
- **All session count**: Number of all sessions, including active, inactive, idle, and others.
- **Average physical read bytes**: Average number (in bytes per second) of disk reads by all database instance activity, including application reads, backup and recovery, and other utilities.
- **Average physical write bytes**: Average number (in bytes per second) of disk writes for all database instance, including application activity, backup and recovery, and other utilities.
- **Configured nodes** (RAC only): Number of nodes configured in cluster, reported by each database instance (node), including offline or currently not responding nodes. Configured nodes mean nodes, that are part of this cluster database, even if they are offline or malfunctioning at the moment.
- **Connected nodes** (RAC only): number of connected (online, active and working) cluster nodes, reported by each database instance (node). Connected nodes mean nodes, that are currently online, active and working.

Oracle and Oracle RAC

- **Active session count**: Number of currently active user sessions, excluding sleeping sessions and background processes.
- **All session count**: Number of currently established user sessions, including sleeping ones, but excluding background processes.
- **Delete statements per second**: Average number of DELETE statements requested per second. Note that this value correspond to the number of request issued and might not be corresponding to number of requests successfully completed.

- **Insert statements per second**: Average number of INSERT statements requested per second. Note that this value correspond to the number of request issued and might not be corresponding to number of requests successfully completed.
- **Select statements per second**: Average number of SELECT statements requested per second. Note that this value correspond to the number of request issued and might not be corresponding to number of requests successfully completed.
- **Update statements per second**: Average number of UPDATE statements requested per second. Note that this value correspond to the number of request issued and might not be corresponding to number of requests successfully completed.
- **Executed queries per second**: Average number of statements executed per second by the database server. This includes only statements sent to the server by clients and not statements executed within stored programs. This variable does not count COM_PING, COM_STATISTICS, COM_STMT_PREPARE, COM_STMT_CLOSE, or COM_STMT_RESET commands.
- **Number of bytes read from files**: Average number of bytes read per second from all files.
- **Number of bytes written to files**: Average number of bytes written per second to all files.
- **Table cache hit rate**: Average cache hit ratio, in percent.
- **Total wait time per second**: Average time spent waiting on events per second. Idle events are ignored.

MySQL

- **Session count - all databases**: Overall number of sessions for all databases.
- **Sessions connected to the monitored database**: Number of sessions for monitored databases.
- **Sessions connected to non-monitored databases**: Number of sessions for non-monitored databases.
- **Activities - Completed**: Average number of successfully completed activities per second. Use this element to determine the throughput of activities in the system. Refer to the DB2 documentation for the following monitor elements:
  - ACT_COMPLETED_TOTAL
- **Activities - Rejected**: Average number of rejected activities per second. Use this element to determine whether predictive thresholds and work actions that prevent execution are effective, and whether they are too restrictive. Refer to the DB2 documentation for the following monitor elements:
  - ACT_REJECTED_TOTAL
- **Activities - Aborted**: Average number of aborted activities per second, or activities completed with errors. Activities may be aborted due to cancellation, errors, or reactive thresholds. Refer to the DB2 documentation for the following monitor elements:
  - ACT_ABORTED_TOTAL
- **SQL processing time**: Average number of seconds spent on processing SQL per second

- **SQL processing time**: Average number of seconds spent on processing SQL per second, including time spent to perform section execution. Refer to the DB2 documentation for the following monitor elements:
  - TOTAL_SECTION_PROC_TIME
- **Commit and rollback processing time**: Average time spent on commits and rollbacks per second, including time spent to commit processing and rollback operations on the database server. Refer to the DB2 documentation for the following monitor elements:
  - TOTAL_COMMIT_PROC_TIME
  - TOTAL_ROLLBACK_PROC_TIME
- **Administration tasks processing time**: Average time spent on administration tasks per second, including time spent to perform runstats, reorg, load processing, online backups, and index builds on database server. Refer to the DB2 documentation for the following monitor elements:
  - TOTAL_RUNSTATS_PROC_TIME
  - TOTAL_REORG_PROC_TIME
  - TOTAL_LOAD_PROC_TIME
  - TOTAL_BACKUP_PROC_TIME
  - TOTAL_INDEX_BUILD_PROC_TIME
- **User routines processing time**, Average number time spent on executing routines outside of known DB2 per second. Refer to the DB2 documentation for the following monitor elements:
  - TOTAL_ROUTINE_USER_CODE_PROC_TIME
- **Compilation time**: Average time spent on compilation per second, including time spent performing explicit and implicit compiles on the database server. Refer to the DB2 documentation for the following monitor elements:
  - TOTAL_COMPILE_PROC_TIME
  - TOTAL_IMPLICIT_COMPILE_PROC_TIME

    Explicit compiles are compilations directly initiated by a user request such as a bind, rebind, prepare or execute immediate. Implicit compiles are compilations that are not directly requested by the user. That is, they are not a result of a bind, rebind, prepare or execute immediate request. For example, an implicit compilation may occur when executing a statement that was bound using the VALIDATE RUN option if the statement needs to be compiled at execution time.
- **Connection management time**: Average time spent on connection management per second, including time spent processing a connection or switch user request. Refer to the DB2 documentation for the following monitor elements:
  - TOTAL_CONNECT_REQUEST_PROC_TIME
- **CPU time**: Average amount of CPU time used within DB2 per second. It includes both user and system CPU time. Refer to the DB2 documentation for the following monitor elements:
  - TOTAL_CPU_TIME

- **Read time**: Average time per second spent to:
  - Read in data and index pages from the table space containers (physical) for all types of table spaces.
  - Perform the direct reads.
  - Read data and index pages from the table space containers (physical) by asynchronous engine dispatchable units (EDUs) for all types of table spaces. Refer to the DB2 documentation for the following monitor elements:
    - POOL_READ_TIME
    - DIRECT_READ_TIME
    - POOL_ASYNC_READ_TIME
- **Write time**: Average time per second spent to:
  - Perform the direct writes,
  - Complete each asynchronous write. Refer to the DB2 documentation for the following monitor elements:
    - TOTAL_CPU_TIME
    - POOL_WRITE_TIME
    - DIRECT_WRITE_TIME
    - POOL_ASYNC_WRITE_TIME
- **Total wait time**: Average time spent to waiting within DB2 per second. Refer to the DB2 documentation for the following monitor elements:
  - TOTAL_WAIT_TIME

DB2

This page lists available measures for database monitoring.

User experience management

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Getting started with UEM

Read more about UEM

Related pages

User experience use cases and tasks

Monitor user visits by geographic region

Monitor user satisfaction

Trace user activity through all channels

Monitor top landing and exit pages

Find specific user visits

Drilldown to the AppMon Client

User experience management - FAQ

Watch and learn

UEM geographic configuration

End-user perspective

Web performance optimization with Dynatrace UEM

Overview

AppMon User Experience Management (UEM) lets you track user behavior on web pages. It helps you identify user problems before your user complains. It also lets you receive business analytics data like the geographic distribution of your web page visitors and average session duration.

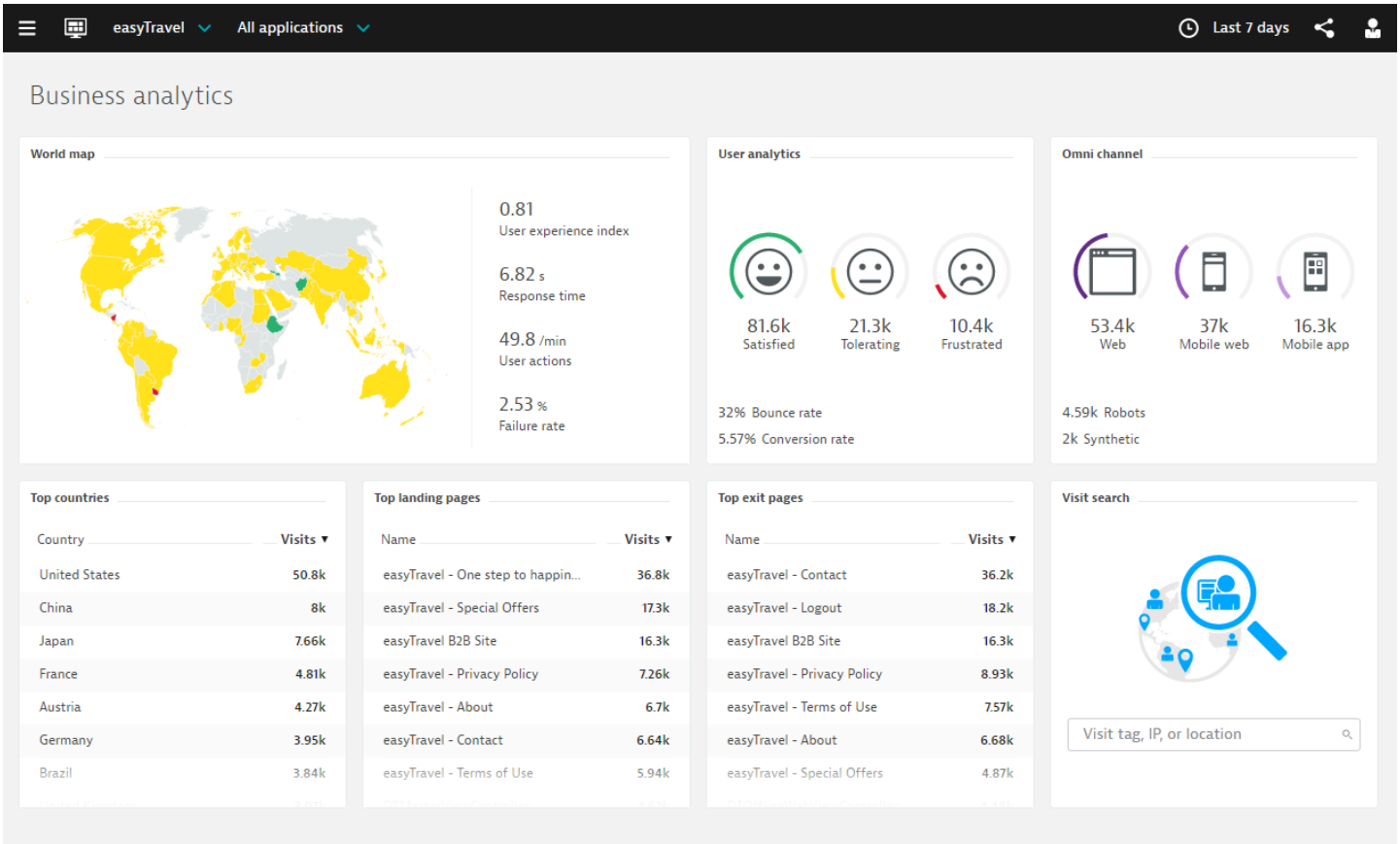
Getting started with UEM

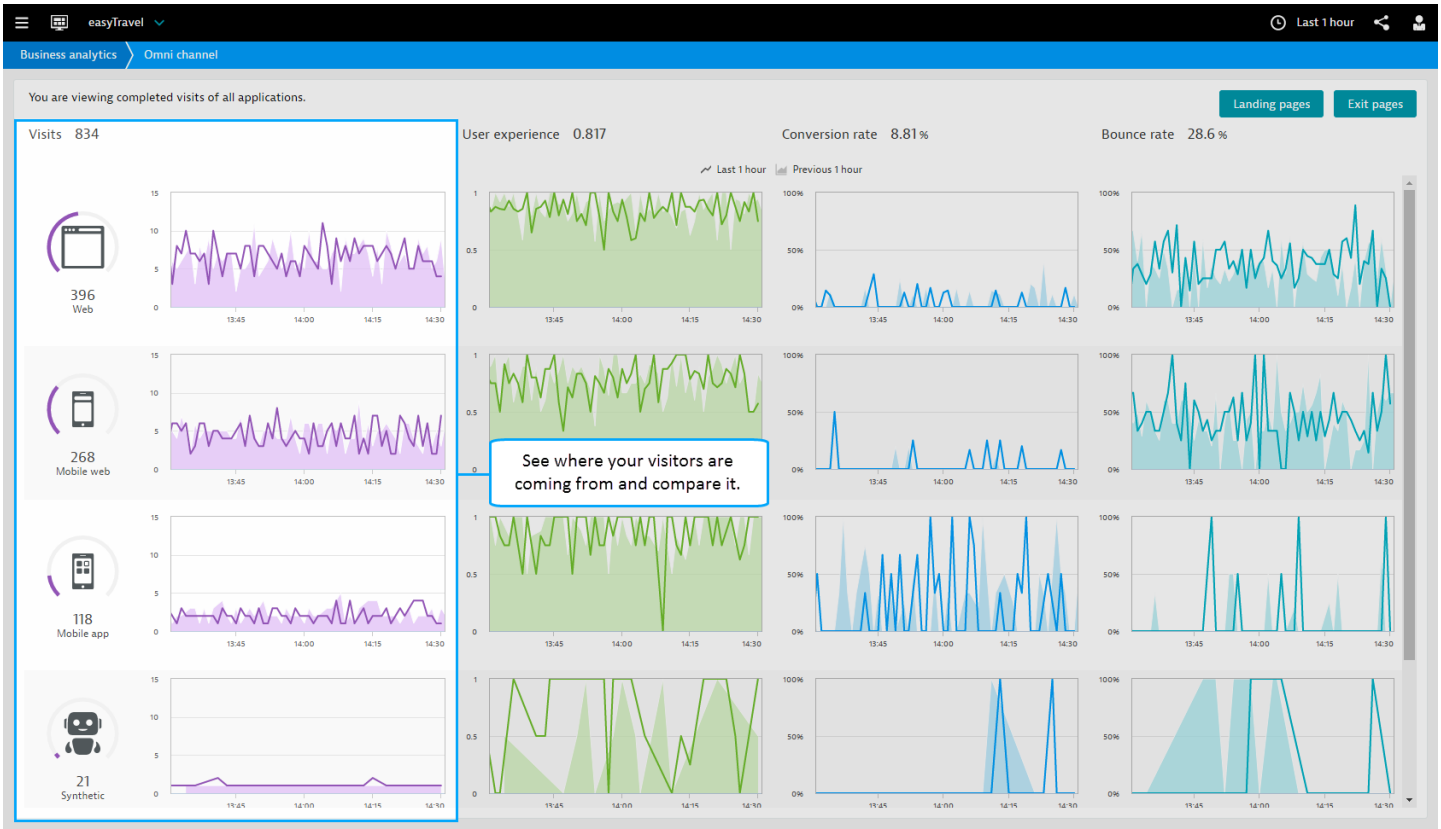
The Business analytics dashboard gives a central point to gauge application performance from the user's point of view. Efficiently filter through millions of user visits, search for specific visits and pass visit details along to support or engineering to accelerate problem resolution. Drill down from visits displayed in AppMon Web to see the full visit PurePath data in the AppMon Client.

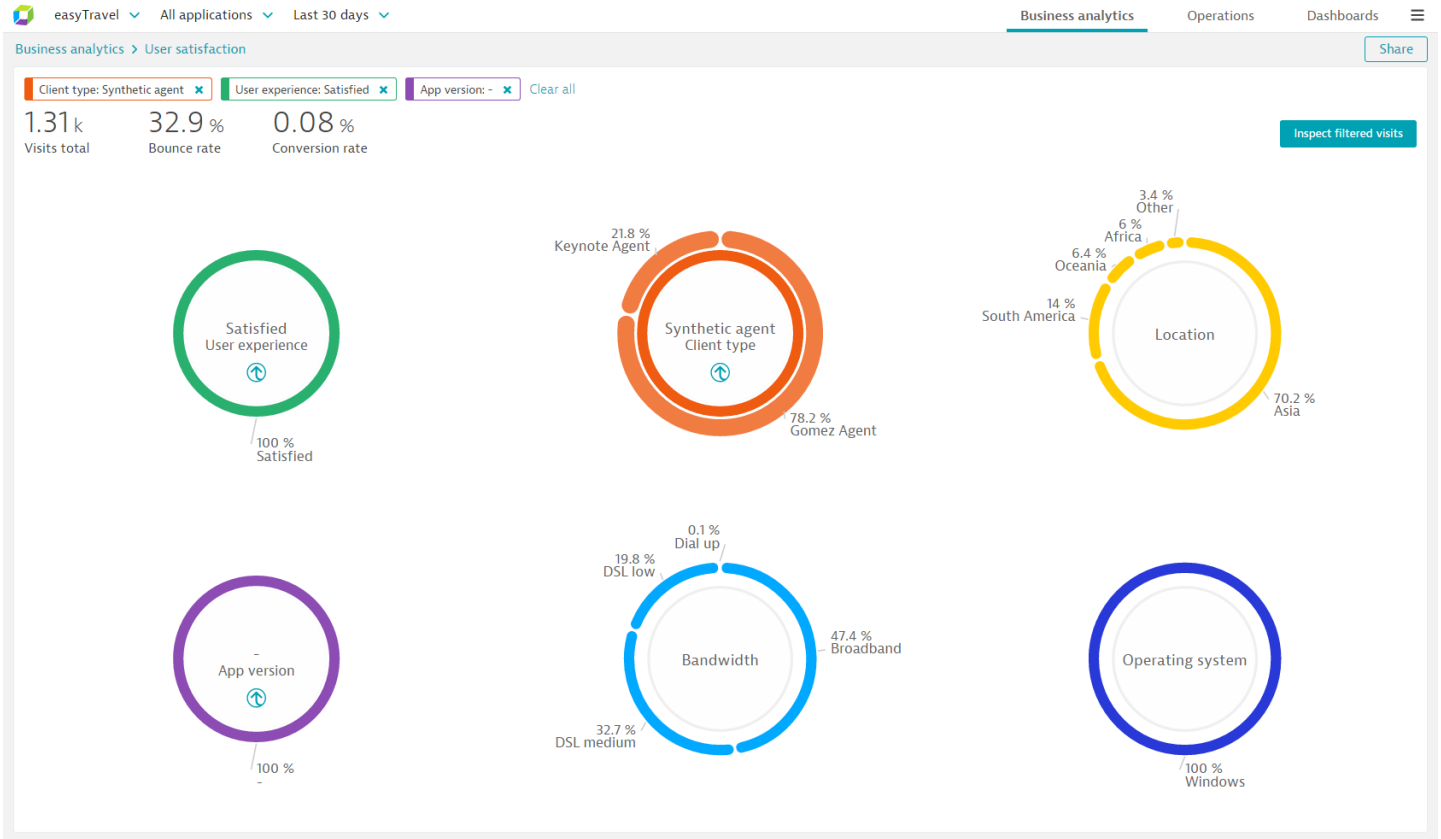
Use tiles on this dashboard to:

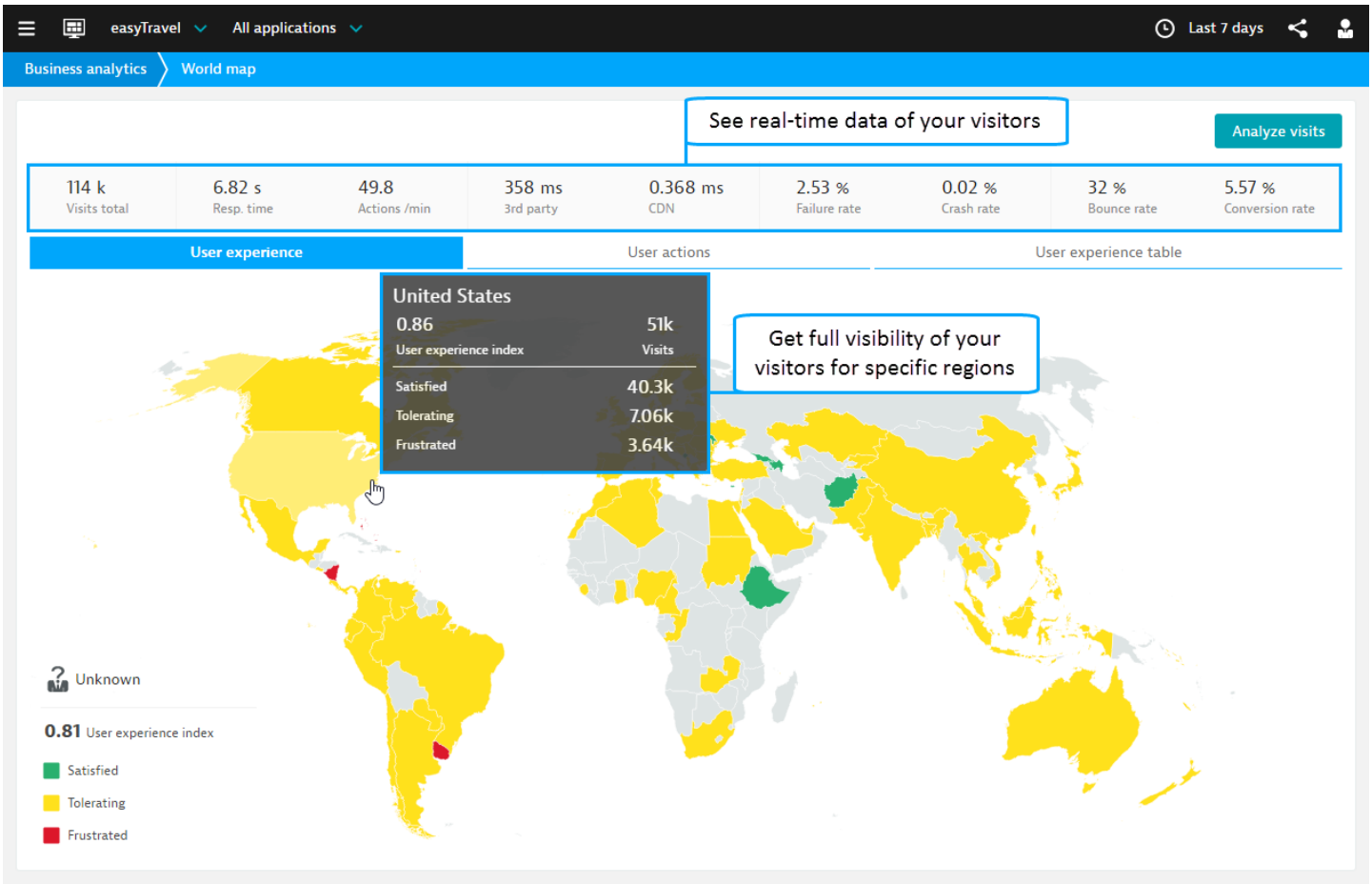
- Analyze site visits to your application or user actions that come from anywhere on the globe based on geographic region.
- See details of your application performance from the point of view of actual users of the application.
- Trace a user's activity through all the channels they access.
- See details of the top countries where your users are accessing the application.
- Analyze user traffic in terms of first and last pages accessed in a visit.
- Quickly find details on any particular user visit of groups of user visits.

Click or tap a tile to open a full screen view for a deeper-dive into related metrics and analytics data.


easyTravel
All applications
Last 7 days
Business analytics
World map
0.81
User experience index
6.82 s
Response time
49.8 /min
User actions
2.53 %
Failure rate
User analytics
81.6k
Satisfied
21.3k
Tolerating
10.4k
Frustrated
32% Bounce rate
5.57% Conversion rate
Omni channel
53.4k
Web
37k
Mobile web
16.3k
Mobile app
4.59k Robots
2k Synthetic
Top countries
Country
Visits
United States 50.8k
China 8k
Japan 7.66k
France 4.81k
Austria 4.27k
Germany 3.95k
Brazil 3.84k
Top landing pages
Name
Visits
easyTravel - One step to happin... 36.8k
easyTravel - Special Offers 17.3k
easyTravel B2B Site 16.3k
easyTravel - Privacy Policy 7.26k
easyTravel - About 6.7k
easyTravel - Contact 6.64k
easyTravel - Terms of Use 5.94k
Top exit pages
Name
Visits
easyTravel - Contact 36.2k
easyTravel - Logout 18.2k
easyTravel B2B Site 16.3k
easyTravel - Privacy Policy 8.93k
easyTravel - Terms of Use 7.57k
easyTravel - About 6.68k
easyTravel - Special Offers 4.87k
Visit search
Visit tag, IP, or location


easyTravel
Last 1 hour
Business analytics
Omni channel
You are viewing completed visits of all applications.
Landing pages
Exit pages
Visits 834
User experience 0.817
Conversion rate 8.81 %
Bounce rate 28.6 %
Last 1 hour
Previous 1 hour
396
Web
268
Mobile web
118
Mobile app
21
Synthetic
See where your visitors are coming from and compare it.


easyTravel
All applications
Last 30 days
Business analytics
Operations
Dashboards
Business analytics > User satisfaction
Share
Client type: Synthetic agent
User experience: Satisfied
App version: -
Clear all
1.31 k
Visits total
32.9 %
Bounce rate
0.08 %
Conversion rate
Inspect filtered visits
Satisfied
User experience
100 %
Satisfied
21.8 %
Keynote Agent
Synthetic agent
Client type
78.2 %
Gomez Agent
3.4 %
Other
6 %
Africa
6.4 %
Oceania
14 %
South America
Location
70.2 %
Asia
App version
100 %
0.1 %
Dial up
19.8 %
DSL low
Bandwidth
47.4 %
Broadband
32.7 %
DSL medium
Operating system
100 %
Windows


easyTravel
All applications
Last 7 days
Business analytics
World map
See real-time data of your visitors
Analyze visits
114 k
Visits total
6.82 s
Resp. time
49.8
Actions /min
358 ms
3rd party
0.368 ms
CDN
2.53 %
Failure rate
0.02 %
Crash rate
32 %
Bounce rate
5.57 %
Conversion rate
User experience
User actions
User experience table
United States
0.86
User experience index
51k
Visits
Satisfied
40.3k
Tolerating
7.06k
Frustrated
3.64k
Get full visibility of your visitors for specific regions
Unknown
0.81 User experience index
Satisfied
Tolerating
Frustrated

Read more about UEM

How Does UEM Work?

How to configure UEM?

How to analyze real time UEM data using the PureLytics stream?

Frequently asked questions

User experience use cases and tasks

Advanced UEM

- Fixing Real Problems of Real Users
- UEM Tips and Tricks
- Third party content analysis
- Third Party Content Analysis
- UEM Health Check
- UEM and streaming media

Getting started with UEM

- User Experience Across Web & Mobile
- Checklist for UEM Visits
- User experience management
- User experience management overview
- First Steps with UEM
- User Experience Management & Administration

UEM Dashlets Explained

Following a single User’s Click

Mobile UEM

What is Mobile UEM?

Instrumenting Android Apps

Android Agent troubleshooting and FAQ

Instrumenting iOS Apps

iOS Agent troubleshooting and FAQ

How does UEM work?

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

What is a User Action?

What is a visit?

What is Apdex?

What is the User Experience Index?

Geographical location

Specific concepts for Web Applications

Specific concepts for mobile apps

Related pages

Manual XHR framework instrumentation

Perceived render time

Third-party content analysis

User Action timings

Visually complete metric

W3C resource timing metrics

Watch and learn

Introduction to user experience management

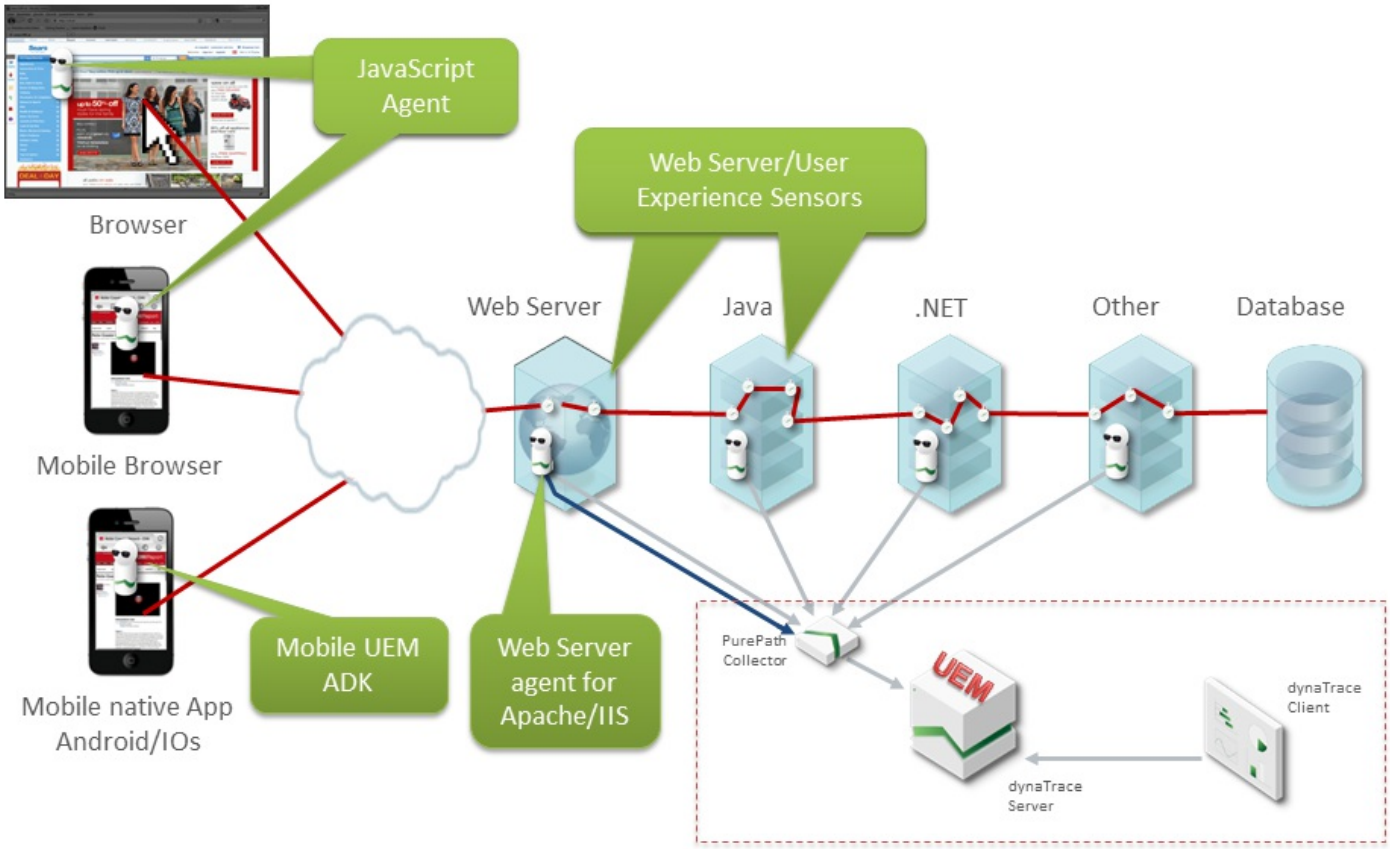
Blog posts

User experience across web & mobile

Fixing real problems of real users

What is mobile UEM?

AppMon User Experience Management (UEM) captures transactions triggered within browsers (desktop or mobile) or mobile apps from end to end: browser or app, over web server and application server, into the database.



UEM for Web Applications

AppMon uses JavaScript code, called the JavaScript Agent, to measure user interactions on your web site. The Web Server or Java Agent (Server-side Agent) automatically injects the JavaScript Agent into your HTML code. After executing in the user's browser, it continuously sends performance and other data to the web site where the Server-side Agent runs. The Server-side Agent intercepts these requests and sends them to the AppMon Collector, and to the Server where they are evaluated and correlated to Server-side PurePaths.

UEM for mobile apps

The Mobile App ADK for Android and iOS lets you capture transactions within mobile apps. A library with a simple API must be called in the mobile app code to indicate the start and stop of a user action. The captured data is sent to a server-side Agent, where it's handled analogous to the data sent by the JavaScript Agent as described above. See the Mobile Agent for more information.

UEM only vs. Full-stack

AppMon allows you to monitor user experience only, without capturing any server-side data. With UEM-only solution you don't have to utilize technology capturing Agents like Java or .NET. You still need the Web Server Agent to collect data from the JavaScript Agent. This Agent can be installed at any server machine, not necessary the one, producing monitored pages. However, you are only able to inject JavaScript Agent automatically if the Web Server Agent is installed at the same machine. Otherwise you have to inject JavaScript Agent manually. The table below lists which information and features are available for UEM-only solution, comparing to the full stack solution.

Feature	UEM only	Full stack
Automatic JS Agent injection	✓ 1	✓
Analyze PurePath	✗	✓
HTML Error codes capture	✗	✓
Frontend data capture	✗	✓
Backend data capture	✗	✓
Business Analytics	✓	✓
World Map	✓	✓
User analytics	✓	✓

Omni channel	✔	✔
Visits	✔	✔
User actions	✔	✔

[1] Only if Web Server Agent, collecting data from the JavaScript Agent, is installed at the server machine which generates the web page.

What is a User Action?

A user action contains all client activity during the transition from one state to another. Every user action contains at least one web request. For example, when a user navigates to a page, the browser downloads many resources and calls the JavaScript `onload` handler, which in turn might perform XHR requests, or other actions.

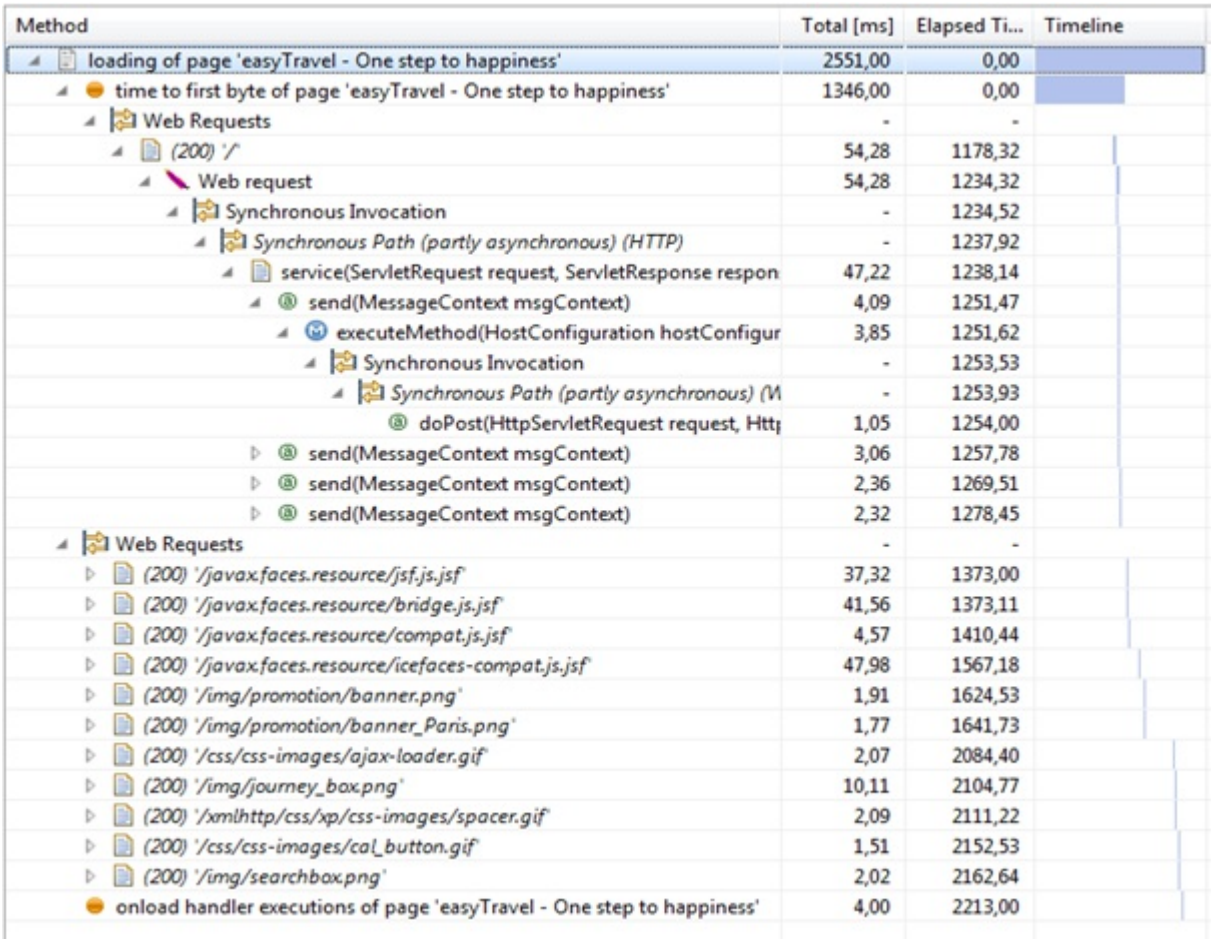
A user action's duration is called *response time*. It represents the time the user waits until it is possible to continue with the use case. A low response time indicates better performance than a high one.

Read more about User Actions

The web has the following different user actions:

Initial page load

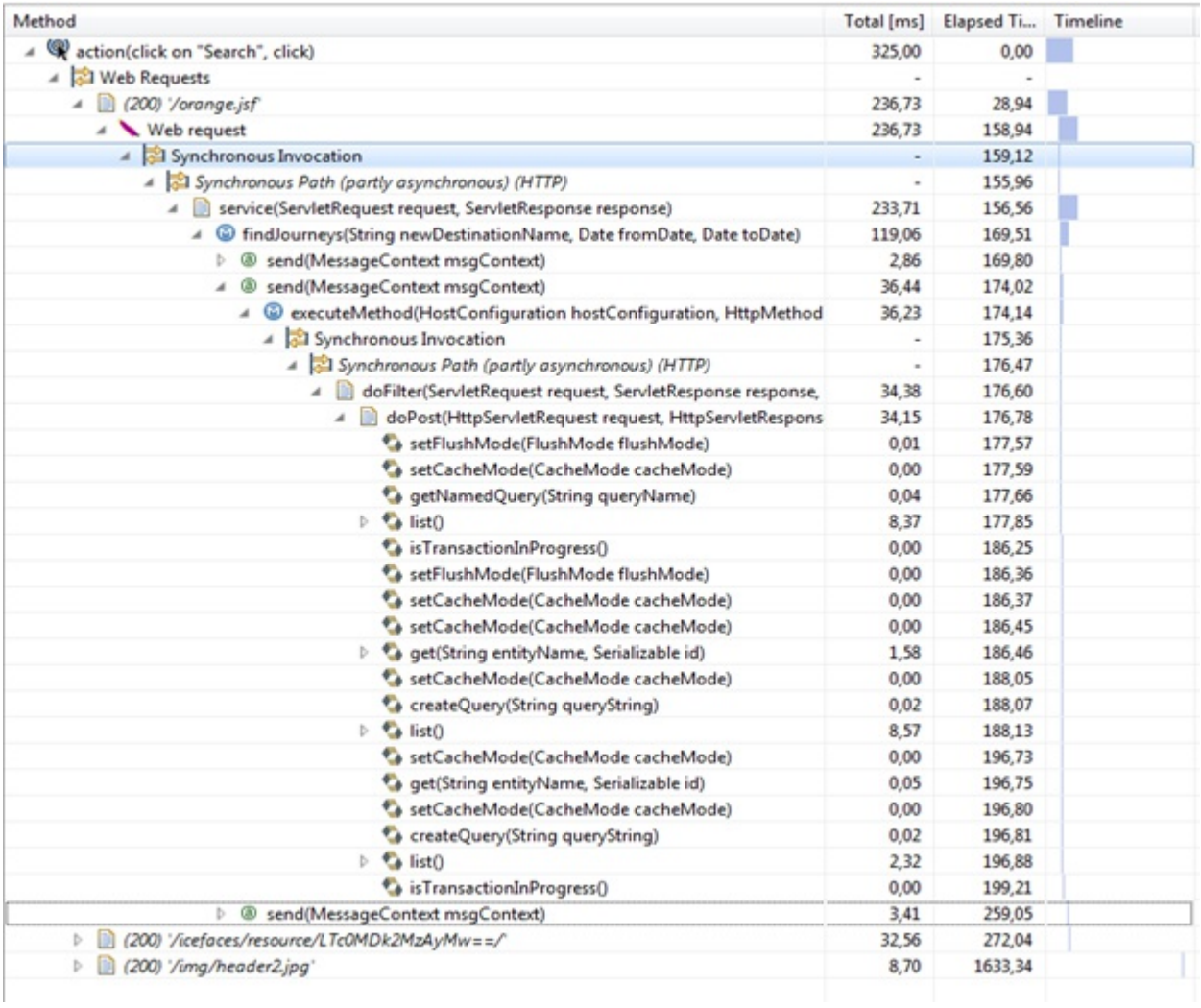
A page load is represented as a single user action such as a user entering a URL in the browser's address bar. The start time of the user action is set to the W3C navigationStart time if available. If not available, it is set to the time JavaScript agent initializes in the browser. The end time is set to the end of the last onload handler. If the onload handler starts any XMLHttpRequests, the user action ends after the XMLHttpRequest finishes.



Method	Total [ms]	Elapsed Ti...	Timeline
loading of page 'easyTravel - One step to happiness'	2551,00	0,00	
time to first byte of page 'easyTravel - One step to happiness'	1346,00	0,00	
Web Requests	-	-	
(200) '/'	54,28	1178,32	
Web request	54,28	1234,32	
Synchronous Invocation	-	1234,52	
Synchronous Path (partly asynchronous) (HTTP)	-	1237,92	
service(ServletRequest request, ServletResponse respon	47,22	1238,14	
send(MessageContext msgContext)	4,09	1251,47	
executeMethod(HostConfiguration hostConfigur	3,85	1251,62	
Synchronous Invocation	-	1253,53	
Synchronous Path (partly asynchronous) (W	-	1253,93	
doPost(HttpServletRequest request, Htt	1,05	1254,00	
send(MessageContext msgContext)	3,06	1257,78	
send(MessageContext msgContext)	2,36	1269,51	
send(MessageContext msgContext)	2,32	1278,45	
Web Requests	-	-	
(200) '/javax.faces.resource/jsf.js.jsf'	37,32	1373,00	
(200) '/javax.faces.resource/bridge.js.jsf'	41,56	1373,11	
(200) '/javax.faces.resource/compat.js.jsf'	4,57	1410,44	
(200) '/javax.faces.resource/icefaces-compat.js.jsf'	47,98	1567,18	
(200) '/img/promotion/banner.png'	1,91	1624,53	
(200) '/img/promotion/banner_Paris.png'	1,77	1641,73	
(200) '/css/css-images/ajax-loader.gif'	2,07	2084,40	
(200) '/img/journey_box.png'	10,11	2104,77	
(200) '/xmlhttp/css/xp/css-images/spacer.gif'	2,09	2111,22	
(200) '/css/css-images/cal_button.gif'	1,51	2152,53	
(200) '/img/searchbox.png'	2,02	2162,64	
onload handler executions of page 'easyTravel - One step to happiness'	4,00	2213,00	

Web 2.0 action or XHR call

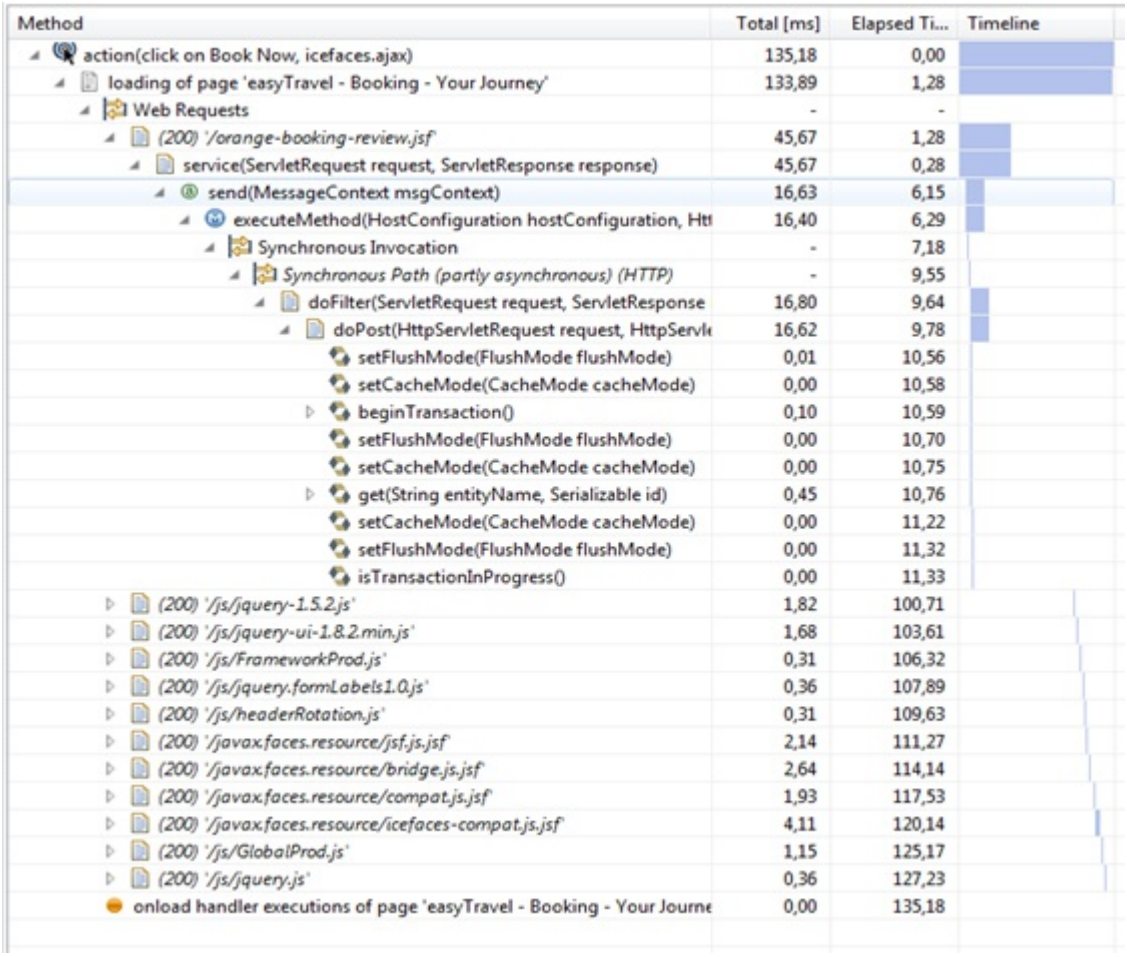
In Web 2.0 applications, when a user interacts with the web site, some actions may cause an incomplete page reload. For example, a user might click a **Search** button that triggers an XMLHttpRequest and alters the current page's DOM tree. Each JavaScript execution with at least one XHR represents as a user action. JavaScript executions without an associated XMLHttpRequest are pure client-side actions, and do not count as user actions. The user action start time is set to the time of the event. The end time is set to the end of the last onreadystatechange event triggered by the XMLHttpRequest. Any XMLHttpRequest triggering additional XMLHttpRequests are also added to the same user action.



Method	Total [ms]	Elapsed Ti...	Timeline
action(click on "Search", click)	325,00	0,00	
Web Requests	-	-	
(200) '/orange.jsf'	236,73	28,94	
Web request	236,73	158,94	
Synchronous Invocation	-	159,12	
Synchronous Path (partly asynchronous) (HTTP)	-	155,96	
service(ServletRequest request, ServletResponse response)	233,71	156,56	
findJourneys(String newDestinationName, Date fromDate, Date toDate)	119,06	169,51	
send(MessageContext msgContext)	2,86	169,80	
send(MessageContext msgContext)	36,44	174,02	
executeMethod(HostConfiguration hostConfiguration, HttpMethod	36,23	174,14	
Synchronous Invocation	-	175,36	
Synchronous Path (partly asynchronous) (HTTP)	-	176,47	
doFilter(ServletRequest request, ServletResponse response,	34,38	176,60	
doPost(HttpServletRequest request, HttpServletRespons	34,15	176,78	
setFlushMode(FlushMode flushMode)	0,01	177,57	
setCacheMode(CacheMode cacheMode)	0,00	177,59	
getNamedQuery(String queryName)	0,04	177,66	
list()	8,37	177,85	
isTransactionInProgress()	0,00	186,25	
setFlushMode(FlushMode flushMode)	0,00	186,36	
setCacheMode(CacheMode cacheMode)	0,00	186,37	
setCacheMode(CacheMode cacheMode)	0,00	186,45	
get(String entityName, Serializable id)	1,58	186,46	
setCacheMode(CacheMode cacheMode)	0,00	188,05	
createQuery(String queryString)	0,02	188,07	
list()	8,57	188,13	
setCacheMode(CacheMode cacheMode)	0,00	196,73	
get(String entityName, Serializable id)	0,05	196,75	
setCacheMode(CacheMode cacheMode)	0,00	196,80	
createQuery(String queryString)	0,02	196,81	
list()	2,32	196,88	
isTransactionInProgress()	0,00	199,21	
send(MessageContext msgContext)	3,41	259,05	
(200) '/icefaces/resource/LTc0MDk2MzAyMw==/'	32,56	272,04	
(200) '/img/header2.jpg'	8,70	1633,34	

New page navigation

A user navigating to a new page via a link or a JavaScript execution is also a user action. In this case, the start time of the user action is set to the time the user clicked the link. The end time is calculated the same for the initial page.



Streaming media playback

You can track HTML 5 video and audio stream playbacks, which is a user action. See Monitor Streaming Media Activity for more information.

What is a visit?

A visit is a group of user actions performed by the same user within a certain time period. It is created with the first user action and ends after the user is inactive for a specified period of time. See System Profile > User Experience for more information. This user inactivity time setting defaults to 30 minutes. You can define it on an application-level (System Profile - Applications). A visit represents the user's click-path and helps to analyze problems. A visit duration is calculated from the start of the first user action to the end of the last user action.

What is Apdex?

Apdex (Application Performance Index as defined by the Apdex Alliance) is an open, standard way to depict the performance of User Actions. Apdex is a ratio of slow and fast requests. It is application independent and represents performance as a single value between 0 and 1. Everything below .5 is unacceptable everything above .93 is excellent. Use this value to compare the performance of different applications and to understand long-term performance trends.

Read more about Apdex

The Apdex is calculated by categorizing user actions by their response times into Apdex Performance Zones:

- Satisfied — Comprised of a chosen threshold (T), and the following formula:

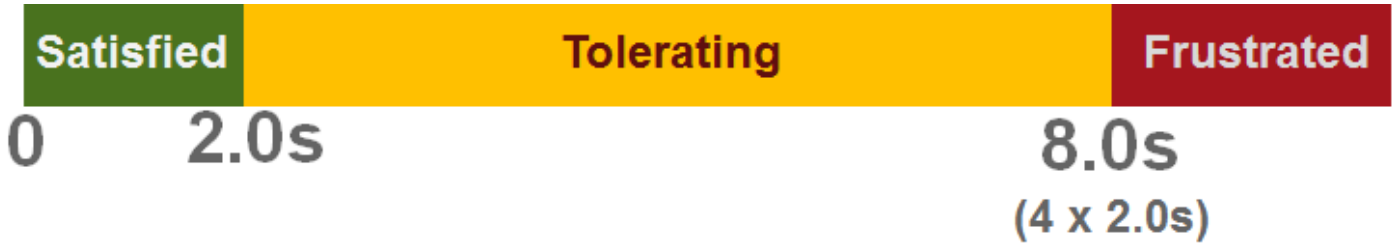
$$Apdex = \frac{Satisfied\ actions + \frac{Tolerated\ actions}{2}}{Total\ actions}$$

  Everything faster than T is considered satisfied.
- Tolerating — slower than T, but faster than 4T.
- Frustrated — slower than 4T

For example, assume you have an application-wide satisfied-threshold called page load baseline.

To set this, click **Client - System Profile > User Experience vertical tab > Default or Application Specific Settings horizontal tab > General > Action groups**.

The threshold for page load baseline is defined as two seconds. User actions map to the following zones based on their response time:



If there are 10 satisfied actions, five tolerating actions and two frustrated actions, the result is an Apdex of (10 + 5 / 2) / 17 = 0.73.

Additional Bandwidth Factor in the Apdex Calculation
AppMon UEM assumes that users with a slow network connection (such as from a mobile phone) accept slower user actions.

Click **System Profile > User Experience > Default or Application Specific Settings horizontal tab > Web Applications sub-tree** to switch the bandwidth calculation to on. This evaluates the performance of a user action, where 1,500 Kbits/sec is the norm (factor 1 for the Apdex performance zone thresholds) and 56 Kbits/sec is slow (factor 2).

For example, a bandwidth of 630 Kbits/sec results in a factor of ~1.6. The satisfied-tolerating threshold for the above example changes to 2.0 x 1.6 = 3.2s. The tolerating-frustrated threshold changes to 4 x 2.0 x 1.6 = 12.6s.

What is the User Experience Index?

AppMon uses the term User Experience rather than Apdex. User experience analyzes how the performance is perceived by the application's users, and categorizes visits into satisfying, tolerating and frustrating, depending on the performance of user actions and errors.

- Users have a frustrating experience if:
  - Their last action failed ("The Web site does not work - I'm leaving!")
  - Their last action was frustrated ("The Web site is too slow - I'm leaving!")
  - More than 50% of all actions were frustrating
- Users have a satisfying experience if:
  - No action failed
  - More than 50% of all actions were satisfying

- Users have a tolerating experience if their experience is neither frustrating or satisfying, which means that:
  - Their last action was not frustrating or failed
  - Less than 50% of all actions were satisfying
  - More than 50% of all actions were at least tolerating

Read more

The percentages in the above calculations can be adjusted. Click **System Profile > User Experience > Default or Application Specific Settings horizontal tab > General > Visit Experience Zone** to modify these thresholds.

A visit's user experience zone is determined by a well-ordered series of calculations, each representing a possible reason for a particular experience zone. The reason with the highest precedence is recorded in the visit, and determines the visit's experience zone. The ordered list of reasons is given in the following table:

Precedence	Visit Experience Zone Reason
Evaluated first	Frustrated: There was an error in the last action of this visit.
↓	Frustrated: There was a crash in the last action of this visit.
↓	Frustrated: The last action of this visit was Frustrated.
↓	Frustrated: Too many actions in this visit were Frustrated.
↓	Tolerating: There are failed actions in this visit.
↓	Tolerating: Too many actions in this visit were Tolerating or Frustrated.
Evaluated last	Satisfied

Geographical location

AppMon captures the client IP address. To do this, it watches for appropriate HTTP headers and converts this address to a geographical location with a built-in database. See Configure Geographic Location for more information.

Specific concepts for Web Applications

Read more

Bandwidth

To calculate the time it takes to download resources, the JavaScript Agent must measure the actual bandwidth. To do this, it requests a download of up to seven dummy images of increasing size until an image download takes at least 100 milliseconds. The first two download as GIFs. The third to seventh are BMPs. They have the following approximate sizes: 0 KB, 3 KB, 10 KB, 30 KB, 100 KB, 300 KB, 1 MB. The sizes are approximate because there is a 35-bit overhead to the GIF and 66-bit overhead to the BMP.

Bandwidth Calculation Example

size of requested image	download time	bandwidth
3 kbyte	23 ms	-
10 kbyte	87 ms	-
30 kbyte	123 ms	2 Mbit/sec

$$Bandwidth = \frac{Size\ of\ requested\ image}{Download\ time} = \frac{30 \cdot 1024 \cdot 8}{\frac{0.123}{10^6}} = 1.988$$

The bandwidth displays in several UEM dashlets (for example, Visits Dashlet) and are used as a measure for charting and alerting. Since this measuring might cause significant network overhead, the bandwidth calculation is disabled by default. You can enable it globally or per application, and separately for mobile. See System Profile - User Experience for more information. You can configure the calculation interval. It defaults to five minutes.

Latency

The JavaScript Agent calculates the network latency between browser and web or application server (JavaScript Agent signal round-trip). Use the network latency can be as a Measure for charting and alerting.

Third party content

The JavaScript Agent can detect third party images and scripts and calculates load times and count – used for different kinds of third-party measures. See Third-Party Content Analysis for more information.

W3C resource timing metrics

Most modern browsers now provide performance metrics for important page loading activity, such as domain look-up (DNS), establishing connections to HTTP servers (TCP), and securing connections (SSL). AppMon automatically collects those metrics with the JavaScript Agent. See W3C Resource Timing Metrics for more information.

Perceived render time

The perceived render time is the duration it takes to render the visible part of a web page. This time can be an indicator for the time users have to wait until a web page is operable. AppMon can calculate this metric with the JavaScript Agent. See Perceived Render Time for more information.

Auto detection of XHR (Ajax) calls

AppMon UEM supports basic XHR call detection and capturing of all XHR calls via XMLHttpRequest and ActiveXObject on Internet Explorer, such as Google Web Toolkit, zKoss, or custom XHR calls. AppMon UEM also captures XHR (AJAX) actions and creates user actions for the following JavaScript frameworks:

UEM is compatible with all JavaScript frameworks. Automatic click detection is available for:

Technology	Versions	Support level
dojo	1.6.1 – 1.12.2[1]	Supported
extjs	3.4.0 – 6.2.0[1] [2]	Supported
icefaces	2 & 3[1]	Supported
jquery	1.3 – 1.12.4, 2.0 - 2.4 & 3.0-3.2[1]	Supported
prototype	1.6.0 – 1.7.3[1]	Supported
angular	1.0 – 1.5.8, 2.0.0[1]	Supported
mootools	1.4.5 – 1.6.0[1]	Supported

[1] Other versions of these frameworks still work, but certain measures might be missing. The JavaScript ADK can be used to capture the missing data.

[2] In certain cases the JS agent does not capture the correct end time for extjs 6 user actions if the promise API is used.

The following frameworks are covered with the basic XHR detection:

Technology	Versions	Support level
GWT		Supported
zk		Supported
RichFaces		Supported
ASP.net AJAX		Supported
YUI		Supported

Ember.js	Supported

The following frameworks are covered with the jQuery sensor:

Technology	Versions	Support level
backbone.js		Supported

Note

Since Dojo 1.7, modules are loaded via *require()*. This is not fully supported and can lead to some user action loss depending on the loaded module. This may affect the capture of event listeners added via module *dojo/on* and the capture of XHR requests executed via module *dojo/request/xhr*.

Also, manual bootstrapping is not supported in this release. AppMon only captures AngularJs requests if the web app is automatically initialized with Angular using “ng-app” directives.

Streaming media

The JavaScript Agent detects embedded streaming media items with the *HTML5 <media>* tags. It collects playback information and uses them for some Measures. Use the JavaScript ADK to track embedded media streams that use different technologies.

See Monitor Streaming Media Activity for more information.

Specific concepts for mobile apps

See System Profile - User Experience for more information.

User Action timings

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Third-party content analysis

W3C resource timing metrics

Monitor streaming media activity

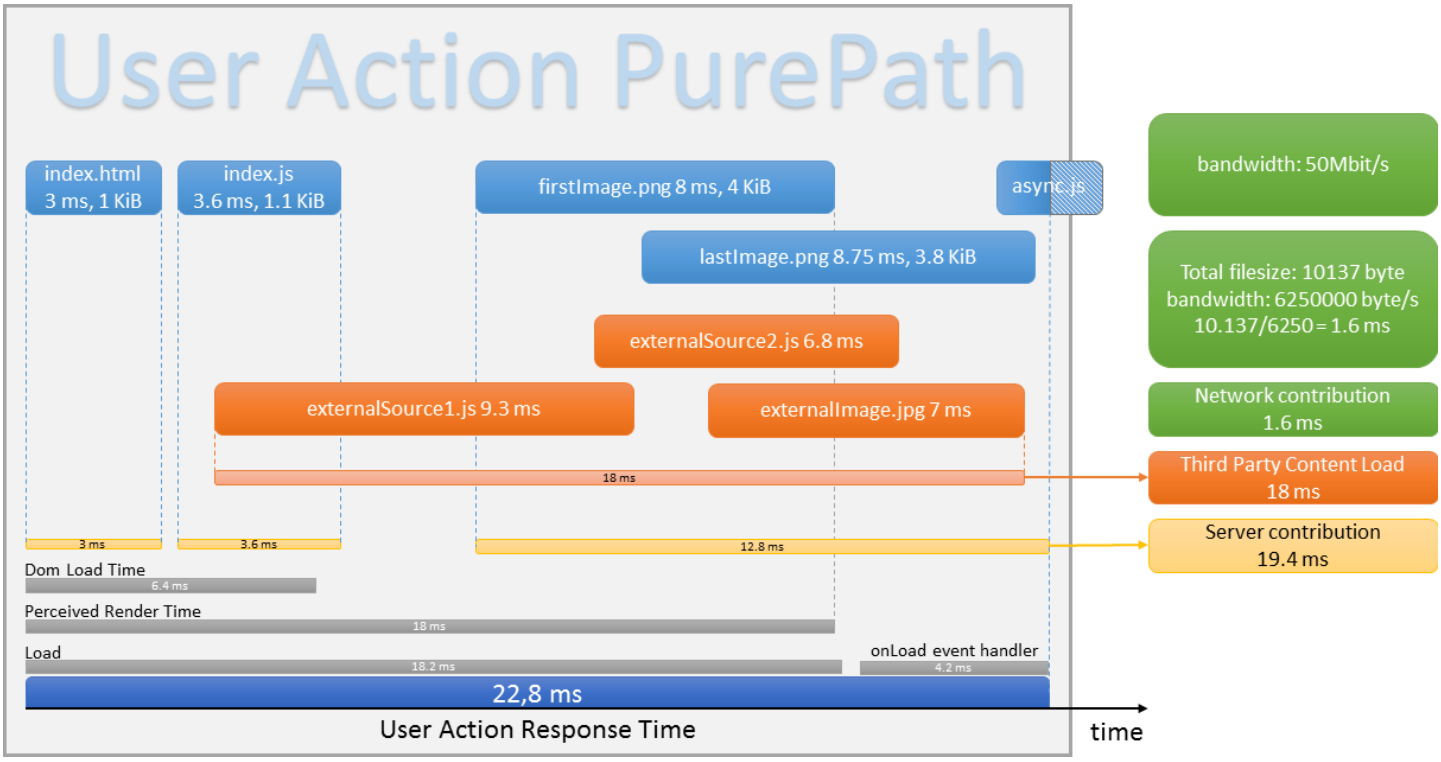
Configure geographic location

Watch and learn

Creating User Action Business Transactions

Load actions

UEM provides several timing measures that can be used in charting, Business Transactions, and more. This page describes the timings a User Action PurePath provides.



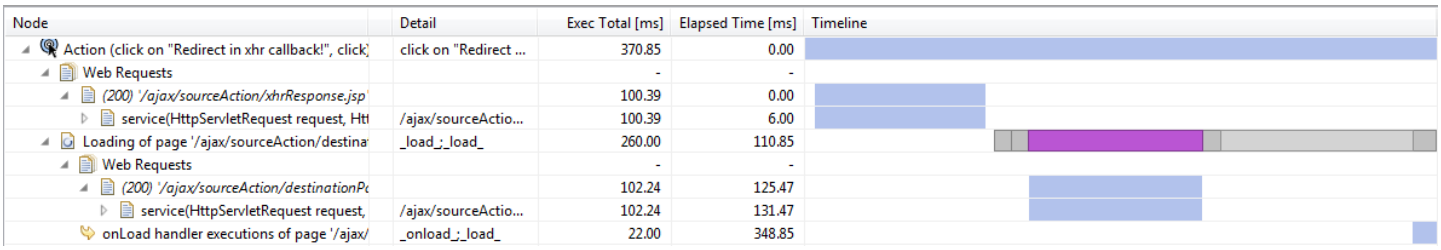
This example shows the load action of a page `index.html`, which loads several synchronous and asynchronous scripts, partially from third party providers.

- The script `async.js` is detected on the server side and has started during the onload action, but ended afterwards. Therefore it only counts partially to the action's server contribution.
- Third party resource timings are never part of server or network contribution, but their load times can be used via the measure "Third Party Content Load".
- Overlapping requests like `firstImage.png`, `lastImage.png`, and `async.js` do not increase server contribution, but are addressed by network contribution calculation.
- Network contribution is only available if the PurePath visit possesses a bandwidth value.
- The network contribution is the sum of the file size of all requests divided by the bandwidth value (bytes per second).
- Dom Load Time is the time from page initialization to the DomContentLoaded event or, for older Internet Explorer versions, to the time the DOM is "interactive".
- See Perceived Render Time for detailed information about this measure. The first image contributes, the last one is not visible from the beginning on and therefore does not contribute to the perceived render time.
- `LastImage.png` loads asynchronously, thus does not lock the load handler.
- The onload event handler completion ends the action.

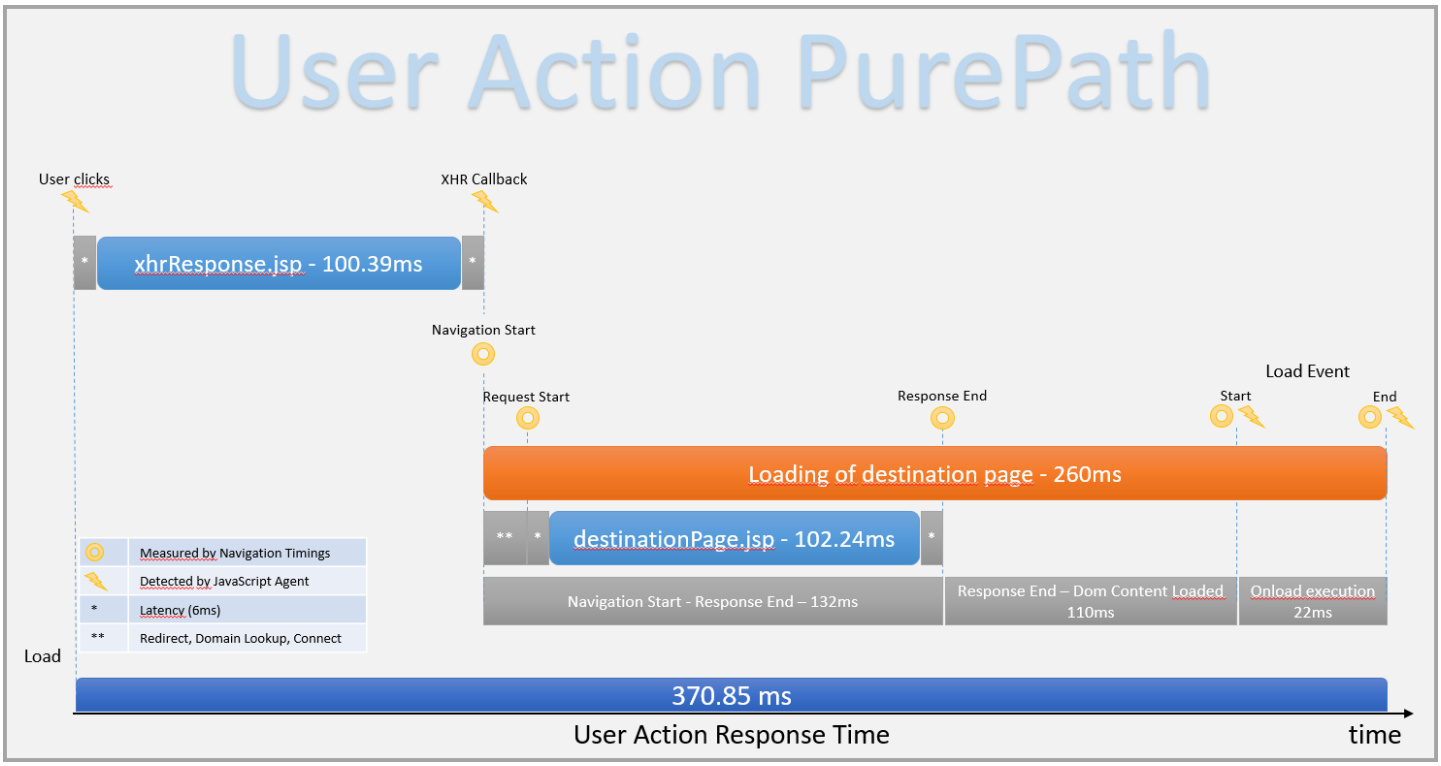
Source actions

There are several ways to trigger a redirect to another page. For example, clicking an anchor tag, or an XMLHttpRequest callback could rewrite the `location.href` variable.

The JavaScript Agent can track these user-triggered redirects and correlate the corresponding requests to user actions. For XMLHttpRequest callbacks, a user action PurePath would look like this:



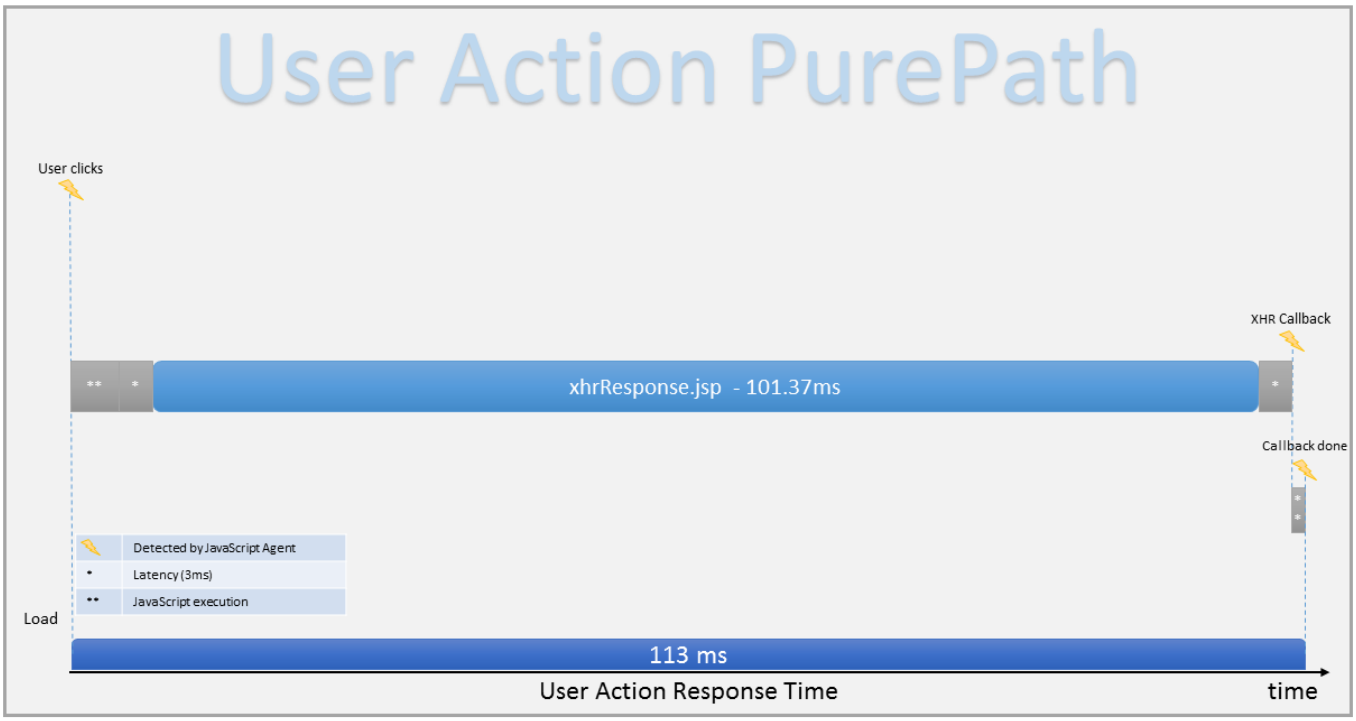
The following explanation shows why the timeline looks the way it is shown above, and what parameters are used for timing calculations. Not all browsers support the Performance Timing API (which includes Navigation Timings). If it's not available, most timings are calculated using server-side records of web requests in combination with the information the JavaScript agent provides.



XMLHttpRequest actions

Several frameworks and services provide support for loading resources asynchronously. Some of these resources are of special interest for the whole experience of a website. If the corresponding modules are enabled, the JavaScript Agent is capable of capturing these requests and provide response times from the very first click until the end user receives the final result. A user action PurePath is generated.

Node	Detail	Exec Total [ms]	Elapsed Time [ms]	Timeline
Action (click on "xhr after timeout!", click)	click on "xhr after ...	113.00	0.00	
Web Requests		-	-	
(200) '/ajax/timedAction/xhrResponse.jsp'		101.37	7.46	
service(HttpServletRequest request, Htt	/ajax/timedAction...	101.37	7.46	



Perceived render time

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Setup

Calculation of perceived render time

User Action PurePath display

Useable measures

Limitations

Related pages

User Action timings

Third-party content analysis

User Action timings

Monitor streaming media activity

Configure geographic location

W3C resource timing metrics

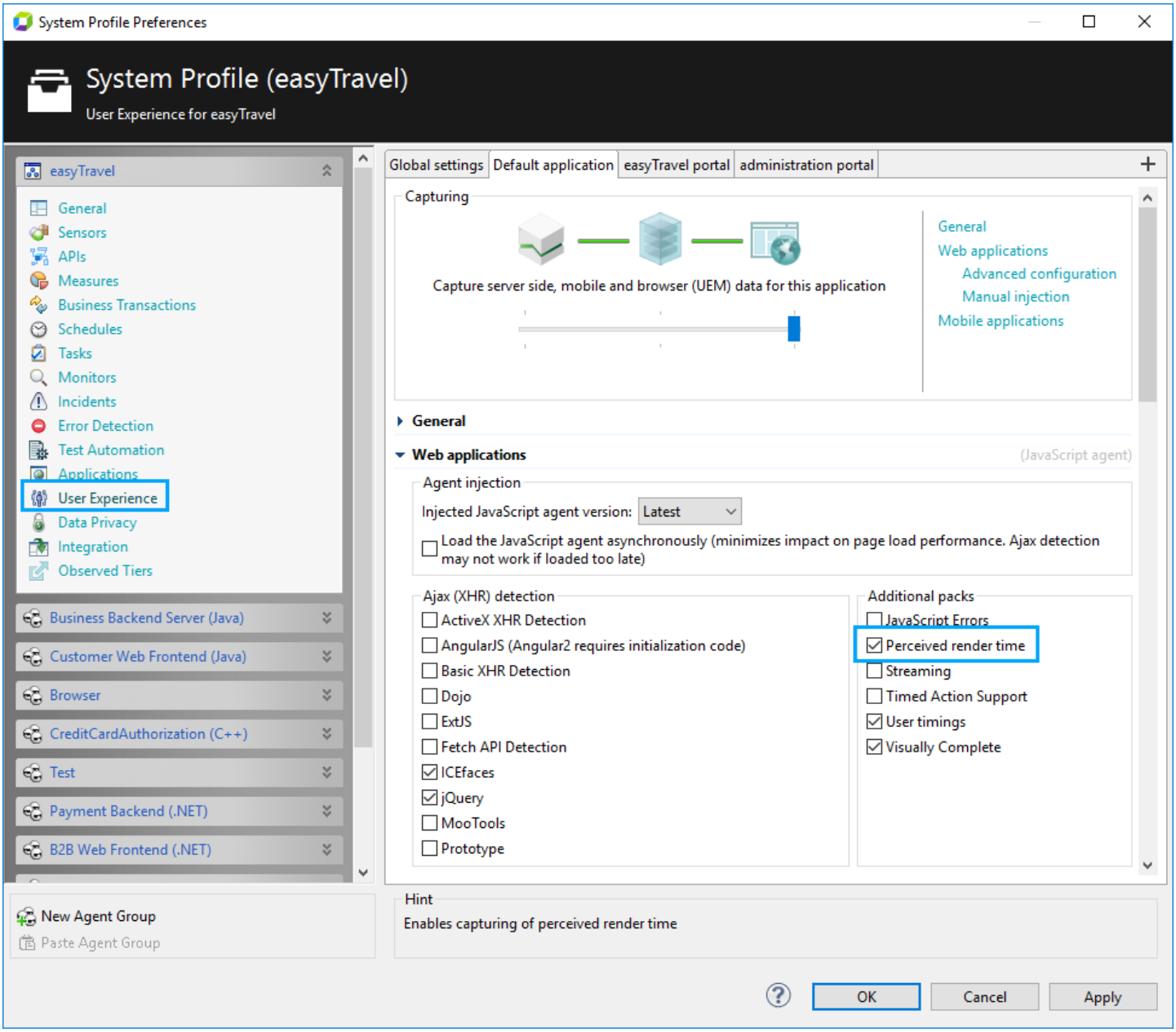
Overview

The perceived render time represents how long the initial visible page takes to load.
With usual text / image size ratios, this largely depends on the load time of the slowest image within the page's visible range.

Setup

The perceived render time plugin is supported for all major browsers. To capture the perceived render time data, check the proper option in your System Profile's user experience section. For more information, see System Profile - User Experience.

System Profile Preferences



Calculation of perceived render time

The perceived render time value is determined by the loading time of <img/> - elements on the page. When you enable the plugin, all image elements on the page are instrumented with a load-success, load-fail and load-abort handler. After the image loading process completes (successful or not), the image's load time is considered perceived render time. The load time becomes the new perceived render time if the following is true:

- The image's load time is the slowest determined load time until now.
- The image is within the initial browser window size.

Browser window size and images relevant for the calculation

As mentioned above, only images within the initial browser window size can affect the calculated value.



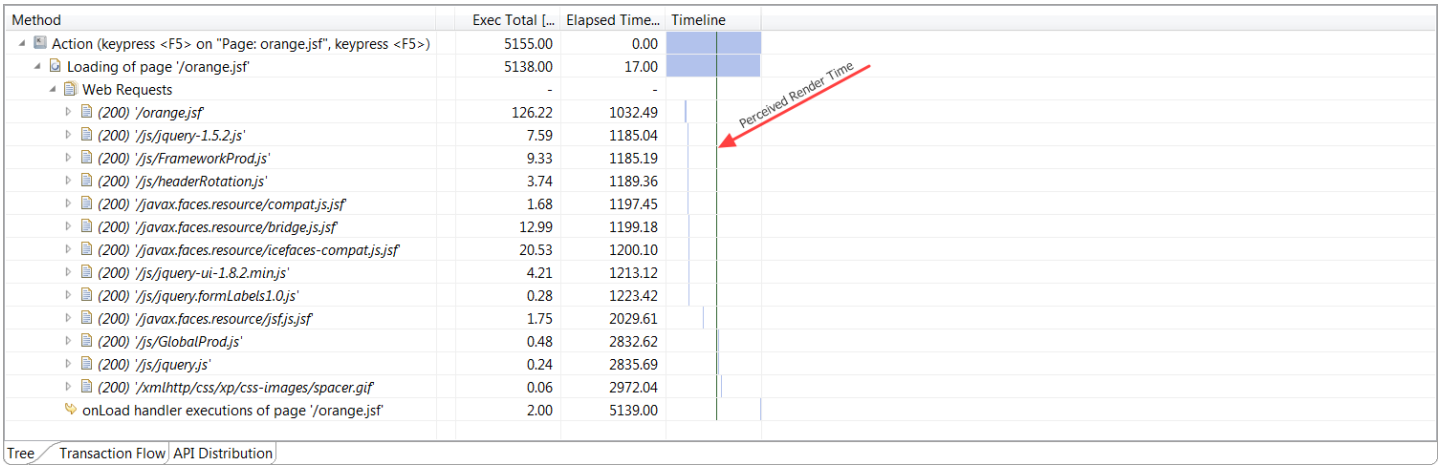
The blue rectangle represents the initially visible part of the page. All image elements within this rectangle affect the perceived render time.

All images within a green rectangle are initially visible and can affect the perceived render time. Images within a red rectangle are outside the initial visible range and are irrelevant for the calculation.

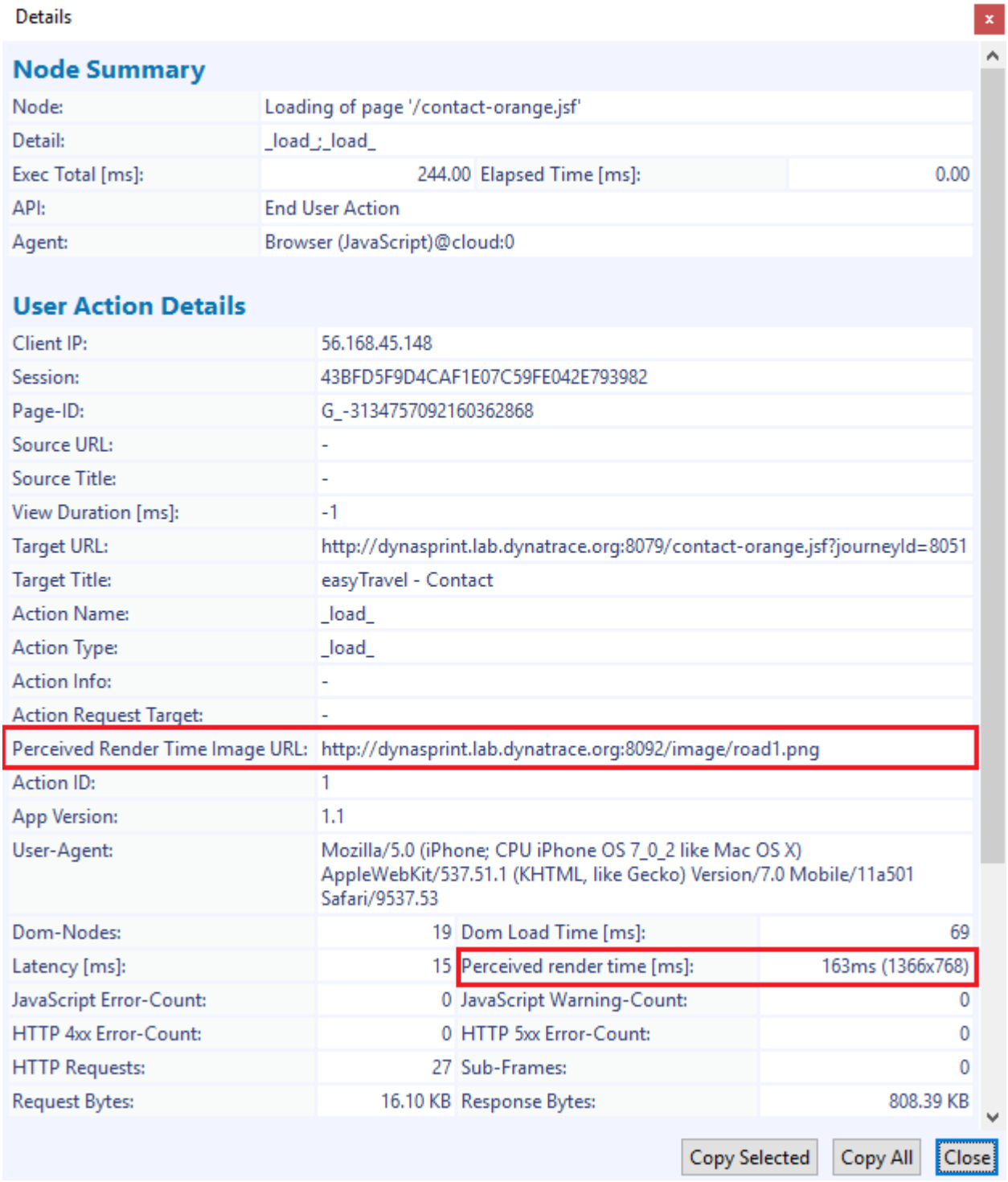
Since there can always be more images affecting the shown screen, the final perceived render time can only be determined once the browser reaches the loadEventEnd.

User Action PurePath display

The perceived render time value displays as a thin line in the User Action PurePath. The left side of this line lists all elements that can affect the perceived render time calculation. All elements with terminated loading processes on the right side of the line do not affect the calculation.



To determine the exact value of the perceived render time and which image was used to calculate it, right-click on a PurePath's loading action to display the details. The browser window size displays as a reference.



Useable measures

The following Measures are available when using the perceived render time plugin.

Name	Location	Description
Browser window size	UEM - Business Transaction Evaluation/Filter/Splitting values - User actions - Browser window size	The browser window size (categorized)

Perceived render time	UEM - User Actions - Perceived render time	Measures the perceived render time in milliseconds.

Limitations

Using the perceived render time plugin has these limitations.

- Only image files directly embedded using the <img>-tag are calculation-relevant. Any images embedded via CSS (stylesheet rule or inline) do not affect the calculation.
- Entirely covered images (by elements with a higher z-index) are used for calculation, although they are not visible.
- Invisible images (by CSS-rule “visibility:hidden”) also affect perceived render time.
- If no images are detected within the visible screen area, no perceived render time value is calculated.

Third-party content analysis

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Third-party content analysis measures

Third Party Content dashboard

Impact vs W3C Resource Timing

Related pages

User Action timings

W3C resource timing metrics

User Action timings

Monitor streaming media activity

Configure geographic location

Overview

The JavaScript Agent detects third-party content. AppMon analyzes the page content and checks for scripts and images from different hosts. It detects when these resources are loaded and sends this information to the agent through the monitor signal.

Because pages potentially contain a large number of third-party resources, only third-party resource summary information transfers. Use configuration settings to specify the third-party resource detail. At the lowest detail level, only summary information such as min, max, average, busy times, and count for each host transfer. With detailed capturing enabled, AppMon transfers the slowest resources over a configured threshold with the complete URL to find the slowest third-party resources on a page.

You can capture all images and scripts on a page from your own host if the entire website is hosted on a CDN. AppMon can monitor the performance of CDN hosted resources. See CDN Content Analysis for more information.

Third-party content analysis measures

Measure third-party content count

Use this Measure to measure the number of third-party resources per user action.

Configuration options include:

- **Host**: String match that filters specific hosts or content providers, for example a set of several domains.
- **Timing type**:
  - Impact: Measured by the JavaScript Agent.
  - W3C Resource Timings: Measured by the browser and provided to the JavaScript Agent through the W3C Resource Timing API.
- **Resource type**: To measure only specific resource types, including:
  - All
  - Images
  - Scripts
  - Style sheets (only for W3C Resource Timings)
  - Other resources (only for W3C Resource Timings)
  - Custom resources (resources added via the ADK)
- **Resource state**: Includes the following:
  - All
  - Successful
  - Interrupted
  - Failed
  - Cached

Measure third party content load time

Use this Measure to measure content load times.

There are several specific configuration options:

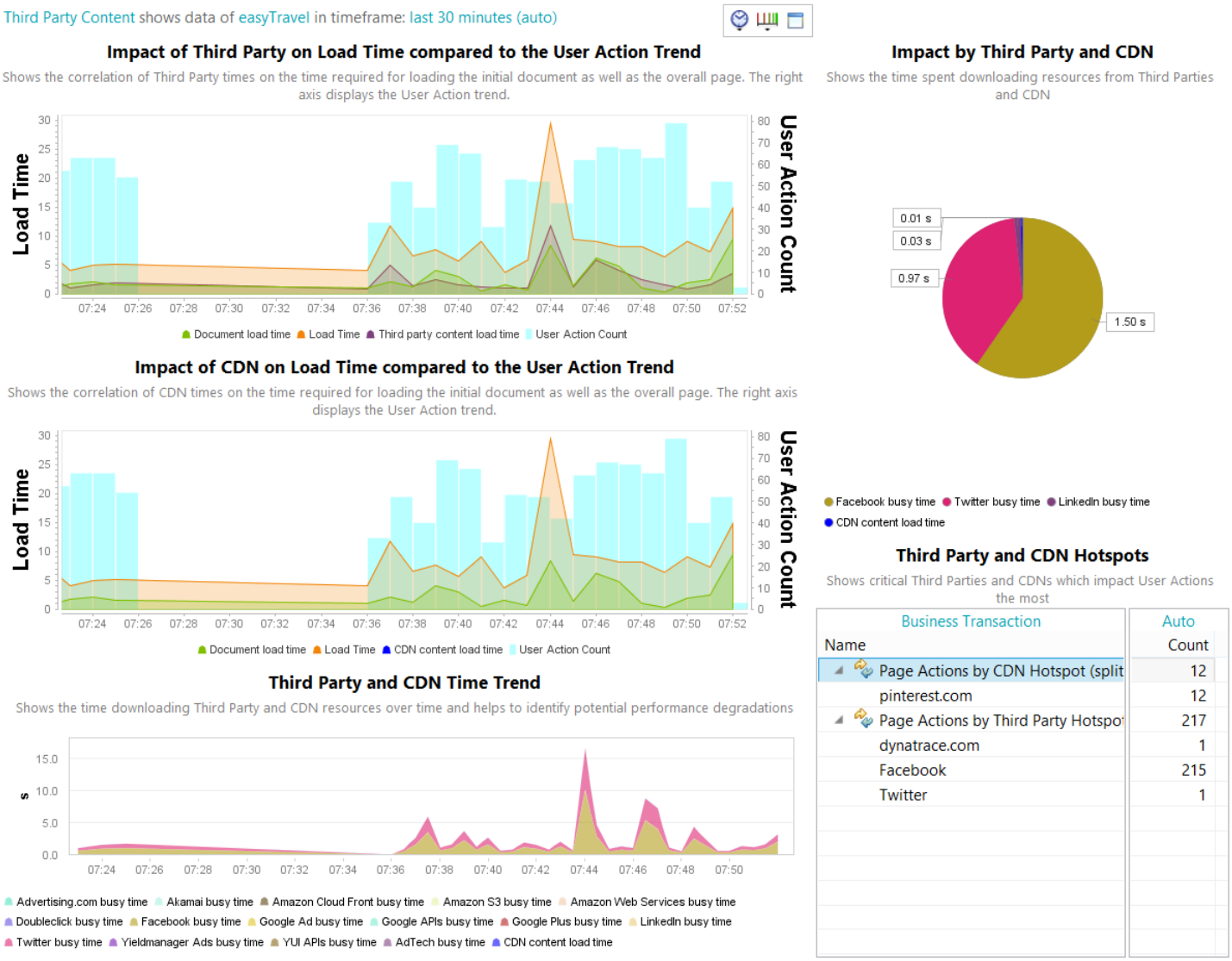
- Host: Filter specific hosts. This is the same as the Third Party Content Count measure.
- Duration type: Several different types of load times are available
  - Busy time: Overall Time when at least one resource was loading from a certain domain and/or a certain resource type. In case resources get downloaded in parallel overlapping times are considered and not counted multiple times.
  - Average duration: average duration of all resources.
  - Shortest duration: duration of fastest resource.
  - Longest duration: duration of slowest resource.
- Resource type:
  - All
  - Images
  - Scripts
  - Stylesheets (only for W3C Resource Timings)
  - Other resources (only for W3C Resource Timings)
  - Custom resources (resources added via the ADK)

BT measure User Actions - third party hot spot

This measure counts the slowest host of a user action over the configured threshold.

Third Party Content dashboard

The **Third Party Content** dashboard shows information about third-party content and its impact on page load times. Click **Start Center > User Experience > Third Party Content Performance** to open the dashboard.



Third Party Content dashboard

Impact vs W3C Resource Timing

The JavaScript Agent collects two different types of third-party content metrics depending on the capabilities of the browser: Impact and W3C Resource Timings.

Impact

Impact is measured by the Third Party module that uses the browser's JavaScript engine capabilities. The Third Party module tracks identifiable resources loaded from domains other than the page's own domain on the load time of the web page, and records their start and stop time. Impact is defined as the time the moment the page is loaded until the specific resource onload event fires. During this time, the browser tracks the following for each resource:

- **Browser Wait time**: The time spent in a browser queue waiting for an available network connection.
- **DNS Lookup**: Time to resolve any DNS name to an IP address.
- **Connecting**: Time required to create a TCP connection.
- **Redirect time**: Any time spent due to browser redirects (301, 307, …).
- **Request time**: Time spent on sending the request to the server.
- **Waiting time**: Time spent waiting for a response from the server.
- **Response time**: Time to read the entire response from either the server or the cache.
- **Execution/Render time**: Any time the browser needs to execute or render the resource.

This tracking works with all browser families and versions. However, the Third Party module tracks only a few types of resources, while others are ignored. Accuracy is limited to the accuracy of time stamps in the browser's JavaScript engine.

W3C Resource Timings

The browser measures W3C Resource Timings, so they are more accurate. The measured time is from the moment the resource is queued for downloading in the browser's download queue until the moment the resource download is complete. This is similar to the impact measure, except that it does not include execution/render time.

The browser must support the W3C Resource Timings API to use this metric.
See http://caniuse.com for a current list of browsers that support W3C Resource Timings. Almost all resources can be tracked if the browser supports resource timing. Sometimes AppMon can retrieve additional information, such as whether the resource was cached or failing.

The following table compares Impact and W3C resource timings.

	Impact	W3C Resource Timings
JS Agent Module	3rd Party Module	Resource timing module (requires 3rd party module)
Measured by	JavaScript Agent	Browser
Browser Support	All browsers	See http://caniuse.com for a current list of supported browsers.
Accuracy & data quality	Varying, dependant on browser used.	High

Detected Resources	• Images referenced in <img> and <input type="submit"> tags • Scripts referenced in <script> tags	All resources except those referenced in HTML5 audio and video elements[1]
What is measured?	The time the JavaScript agent initializes until the onload event of the resource triggers.	The time from when the resource queues in the browser for fetching until resource loading completes.

[1]) Due to a misinterpretation of the W3C specification, Chrome currently does not include resources fetched from the cache with the Resource Timings.

W3C resource timing metrics

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

W3C Resource Timings

User Action PurePaths

Related pages

User Action timings

Third-party content analysis

Web performance optimization for UEM

Monitor streaming media activity

Configure geographic location

External links

Performance metrics - W3C specification

Navigation timing API by browser

Overview

Most modern browsers provide performance metrics for important steps involved in loading pages, such as domain lookup (DNS), establishing connections to HTTP servers (TCP), and securing connections (SSL). The AppMon JavaScript Agent can automatically collect those metrics.

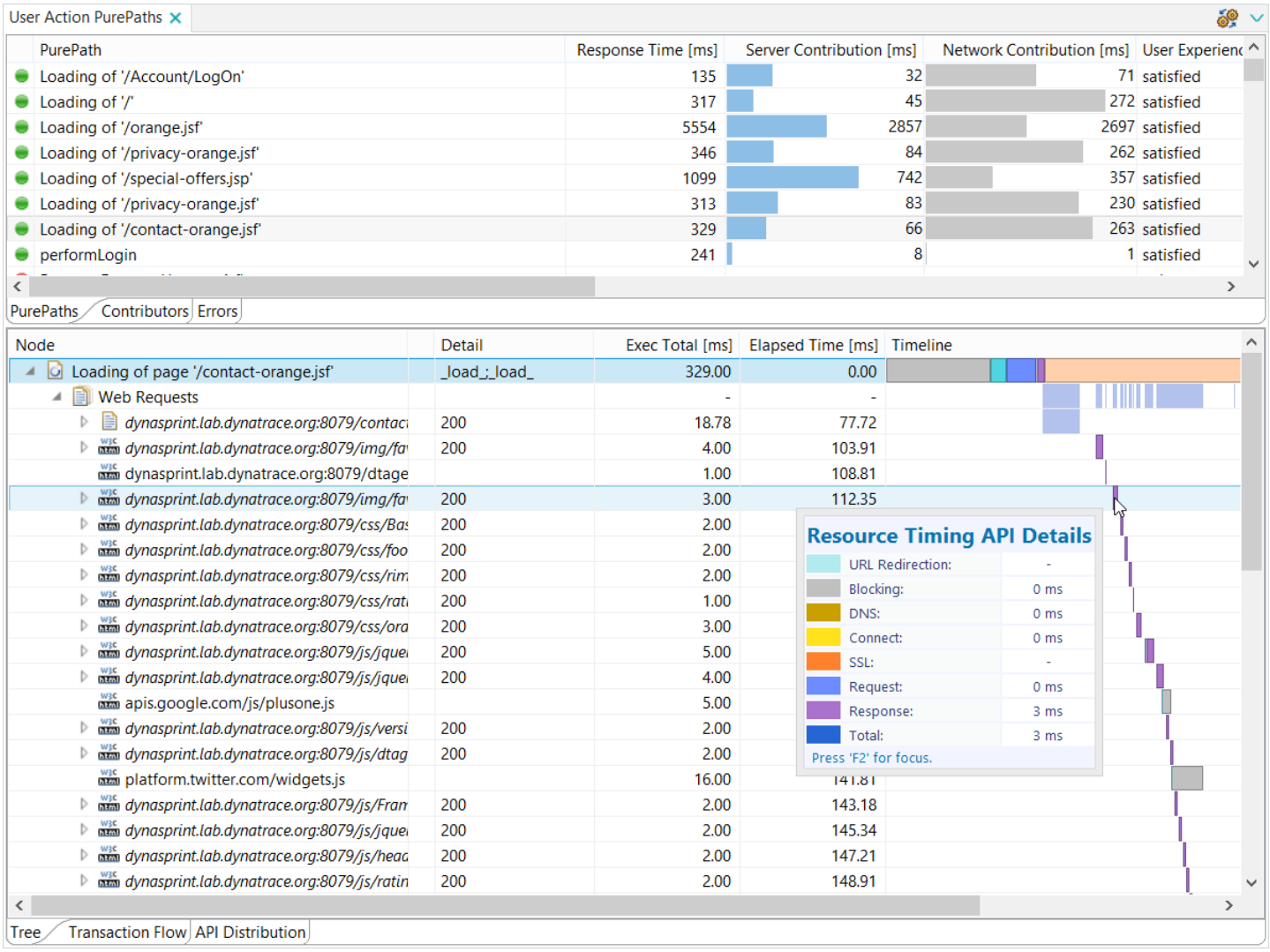
The W3C specification contains detailed information about the way browsers have to calculate those metrics. See http://caniuse.com/nav-timing for a current list of browsers supporting the API.

W3C Resource Timings

AppMon collects all W3C navigation timing metrics by default in all supported browsers by default. You can disable it, by de-selecting the **Capture resource information** checkbox, under **System Profile > User Experience > tab for the required application > W3C resource timings**

User Action PurePaths

Resource timing details are visible in the details of page or frame load actions in the **User Action PurePaths** dashlet. A waterfall visualization of the durations of different steps involved in loading a page is provided in the **Timeline** column, as shown in the following graphic.



See Web Performance Optimization for UEM for more information about setting up UEM to produce full waterfall charts.

Measures

The available measures are based on metrics provided by the resource timing API.

Resource timing steps

The following measures are used to chart the duration of specific steps in the page loading process.

Measure	Description	Definition in terms of W3C specification
DNS	Time spent or resolving domain names.	`window.performance.domainLookupEnd - window.performance.domainLookupStart`
Connect	Time spent establishing a socket connection from the browser to the web server.	`window.performance.connectEnd - window.performance.connectStart`
SSL	Time spent establishing a secure socket connection from the browser to the web server.	`window.performance.connectEnd - window.performance.secureConnectionStart`
URL Redirection	Time spent following HTTP redirects.	`window.performance.redirectEnd - window.performance.redirectStart`
Request	Time spent waiting for the first byte of the document response.	`window.performance.responseStart - window.performance.requestStart`
Response	Time spent downloading the document response.	`window.performance.responseEnd - window.performance.responseStart`
Total	Time between the response being delivered and the `OnLoad` event.	`window.performance.loadEventEnd - window.performance.domLoading`

Resource timing landmarks

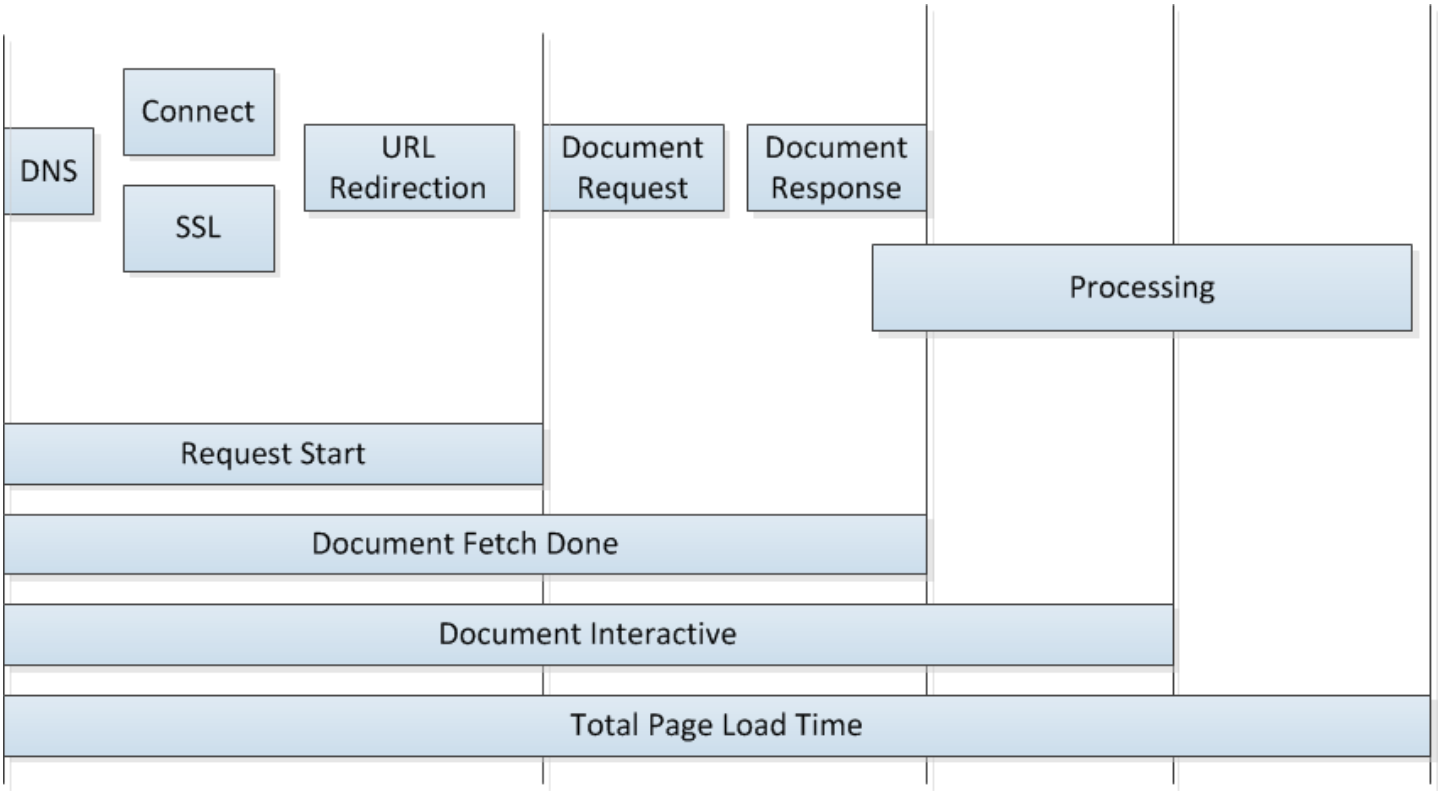
The following measures chart the total time from a user clicking a link or button until the browser reaches a specific point in the resulting page loading process.

Measure	Description	Definition in terms of W3C specification
Request Start	Time until the document request can be sent. This includes DNS, connect, SSL, and URL redirects.	`window.performance.requestStart - window.performance.navigationStart`

Document Fetch Done	In addition to everything until Request Start, this includes the time until the document (HTML) completely downloads from the HTTP server.	`window.performance.responseEnd - window.performance.navigationStart`
Document Interactive	Time until the document becomes interactive. This includes all steps completed for Document Fetch Done.	`window.performance.domInteractive - window.performance.navigationStart`

The following example timeline illustrates the measures described above.

Visualization of Measures



Page Performance dashboard

Resource timing measures display in the Page Performance dashboard. Click **Start Center > User Experience > User Action Performance and Health > Page Performance** to open the Page Performance dashboard.

Page Performance shows data of easyTravel in timeframe: last 24 hours

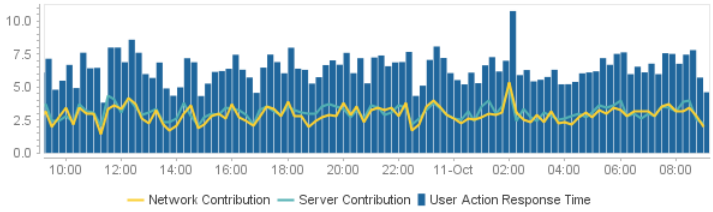
Apdex of User Action

Represents the apdex for user actions per application, which is a number from 0 to 1. An apdex of 1 is best and all below 0.5 is bad.

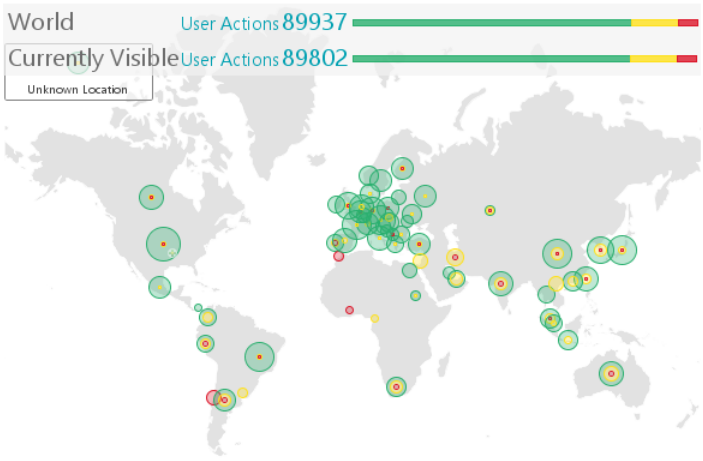


User Action Time Contribution

Displays the server contribution and the network contribution in relation to the user action response time.

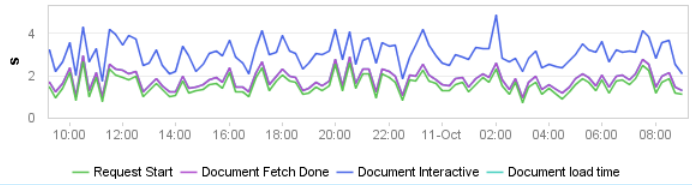


Global Performance Overview



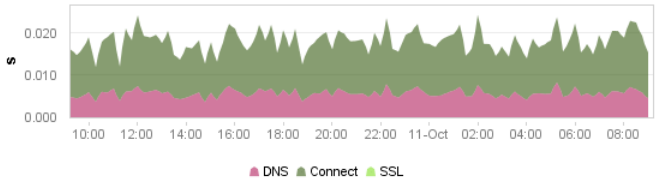
Document Load Breakdown

Average duration until the browser reaches a specific point in the page loading lifecycle. These landmarks are based on the W3C Timing Specifications.



Connect Performance

Average durations of specific steps performed by the browser while loading pages. These metrics are based on the W3C Timing Specifications.



Slowest User Actions

Analysis stopped by serv... Visible Timeframe: yesterday 07:36 to yesterday 14:52

Name	Response Time [s]	Bandwidth
Display SearchJourneyActivity	0.01	-
AppStart easyTravel mobile	0.02	-
click on "8"	0.13	-
click on "24"	0.14	-
click on "7"	0.15	-

Most Frequent User Actions

Analysis stopped by ... Visible Timeframe: yesterday 07:36 to yesterday 14:52

Name	Count	Response Ti...	Bandwidth
click on "Search"	1325	3.38	-
/orange.jsf	875	11.79	-
/contact-orange.jsf	499	8.16	-
click on "Next"	448	6.57	-
/privacy-orange.jsf	428	10.34	-

Manual XHR framework instrumentation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Related topics

Action groups advanced configuration

CDN content analysis

Resource summary configuration

JavaScript ADK and JavaScript Ajax ADK

Synthetic Monitoring integration

UEM-PurePath integration for Synthetic Monitoring

Perceived render time

Third-party content analysis

User Action timings

Visually complete metric

W3C resource timing metrics

How to instrument a web application

Monitor streaming media activity

Analyze visit details

Configure geographic location

Configure streaming media monitoring

Firewall constraints for UEM

Support for IE 8 and lower

The JavaScript agent leverages property listeners to find out the exact moment a framework completely initializes. This technique lets the agent instrument several functions provided by the framework needed for capturing and correlating webrequests and user actions. However, property listeners are not available in older versions of Internet Explorer. In older versions, the earliest automatic initialization event is the end of the onload event, when frameworks have most likely finished initialization. This is sufficient for many cases, but sometimes framework components are used during the onload event. To capture these function calls, you must manually trigger the JavaScript agent framework instrumentation. For most frameworks, it's sufficient to inject a small JavaScript snippet right after the framework's script tag. For AngularJS and Prototype, additional effort is required.

```
<head>
        <meta...></meta>
    <script type="text/javascript" src="dtagent99_xj3_1234.js"></script>
    <script type="text/javascript" src="framework.js"></script>
    <script type="text/javascript">window.dT_&&dT_.initFramework&&dT_.initFramework();</s
 <script>/* other scripts \*/</script>
</head>
```

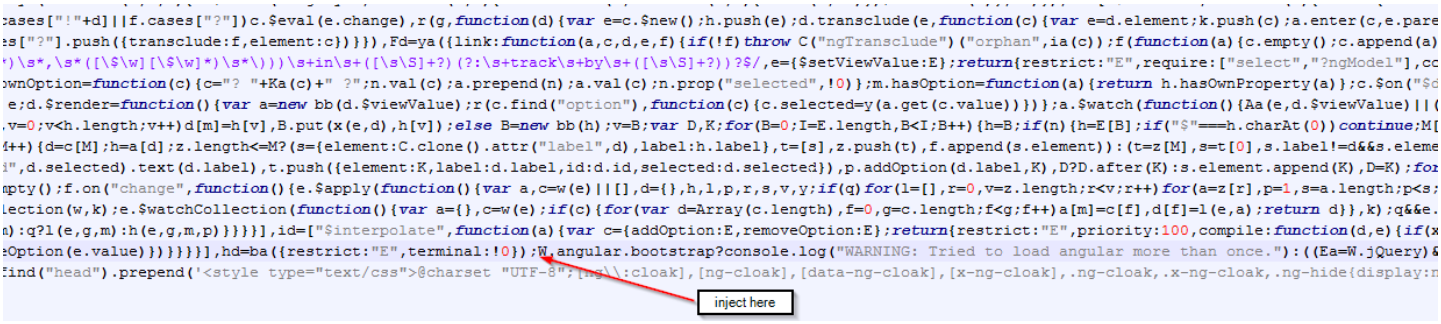
Ajax detection modules requiring snippets

> **Note**
>
> Snippets are required simply for supporting older browsers like IE 8 and lower unless otherwise specified (such as Angular 2, which requires a snippet to support it in ANY browser).

AngularJS (< 2)

This framework comes bundled with jqLite, a minimal version of jQuery. The agent must initialize right after jqLite has finished initialization. For the debug version of AngularJS, inject the snippet right before `jqLite(document).ready`. The following screenshot shows where it has to be injected in the minified AngularJS version (`*angular.min.js`).

```
window.dT_&&dT_.initAngular&&dT_.initAngular();
```



AngularJs (>= 2)

Angular 2 is essentially a completely different framework from Angular 1. Due to the bundling nature and multiple factors leading to the initialization of the framework, it cannot be automatically instrumented. As a result, a code snippet is also needed here. This is required to support Angular 2 in all browsers. The snippet looks like the following, where the parameters `http` and `Headers` are required, which are imported from Angular.

```
if(window.dT_ && dT_.initAngularNg){dT_.initAngularNg(http,Headers);}
```

This must be inserted in an angular2 app component initialization file, where the http functionality is imported for use, ideally in the constructor for the application component where Http is passed. A typical example might look like this:

```
import {Http,Headers} from '@angular/http';
import {Component} from '@angular/core';

@Component({
    selector: 'my-app',
    template: '<h2>My Angular 2 App</h2>'
})
export class AppComponent{
        // HttpModule should be an import in the NgModule which bootstraps this component
    constructor(http: Http) {
        // dynaTrace angular2 instrumentation
        if(typeof dT_!='undefined' && dT_.initAngularNg){dT_.initAngularNg(http,Headers);
    }
}
```

Note

This example is based on the Angular 2.0.0 official release and thus may not reflect the correct way to initialize an Angular 2 application on older versions. You may need to ensure the `http` and `Headers` are valid objects and contain relevant methods (foe example, `http` should include `get` and `post`) to use them, as their contents may differ depending on how you initialized the application or between versions. This can be tested by issuing an XHR from the constructor, for example to verify that they work in that location before passing them to the `initAngularNg` method.

Dojo

For Dojo, inject the snippet after `dojo.js` script-tag.

```
window.dT_&&dT_.initDojo&&dT_.initDojo();
```

ExtJS

For ExtJS, inject the snippet after `ext-all.js` script-tag.

```
window.dT_&&dT_.dtInitExtJS&&dT_.dtInitExtJS();
```

jQuery

For jQiery, inject the snippet after `jquery.js` script-tag.

```
window.dT_&&dT_.initJQuery&&dT_.initJQuery();
```

MooTools

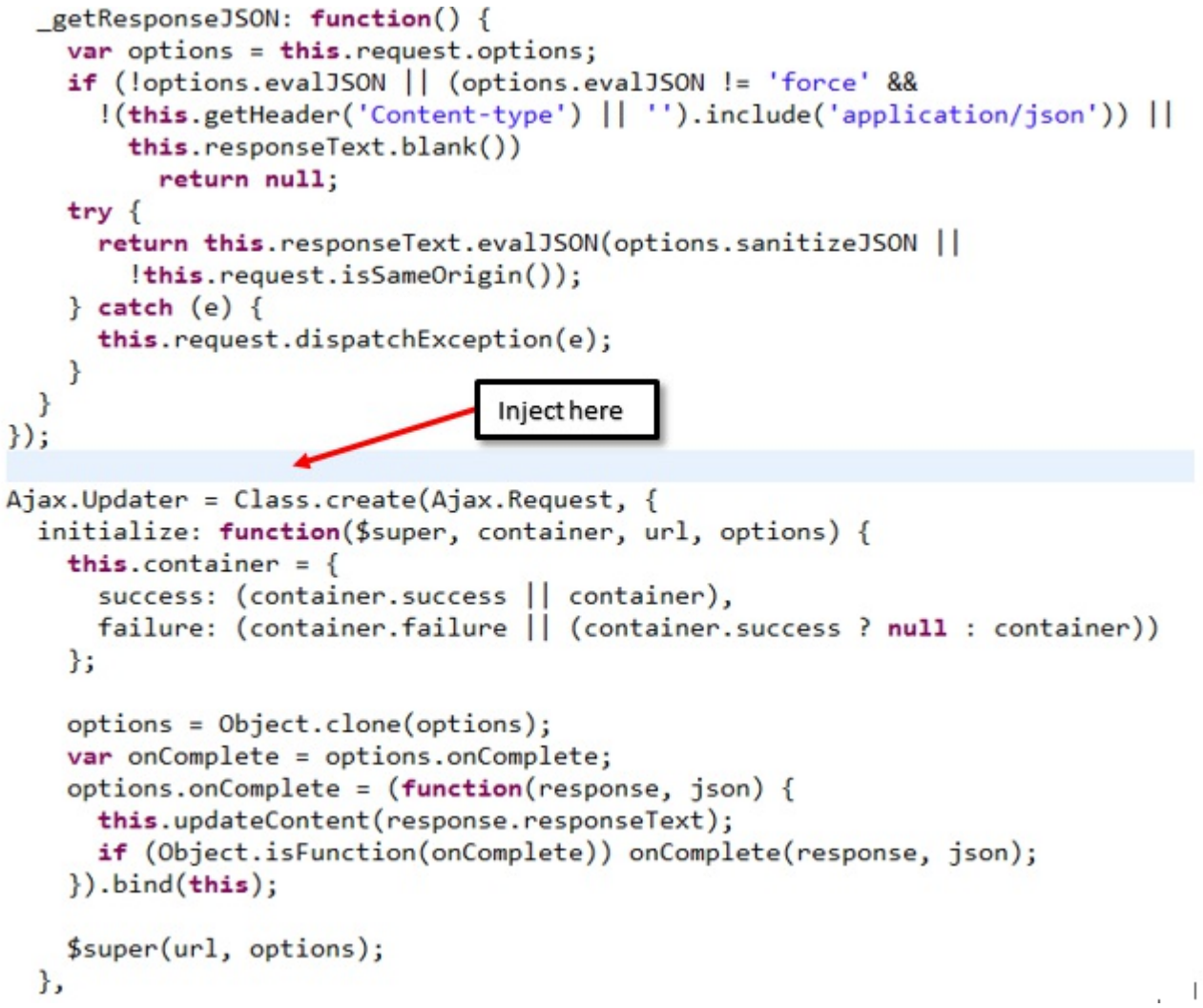
For MooTools, inject the snippet after `mootools.core.js` script-tag.

```
window.dT_&&dT_.initMooTools&&dT_.initMooTools();
```

Prototype

For Prototype, inject the snippet before `Ajax.Updater` call in the `prototype.js` file. See the screenshot for the location in the prototype file.

```
window.dT_&&dT_.initPrototype&&dT_.initPrototype();
```



FAQ

Q: What happens if the snippet of code is not added?

A: If the snippet is not added, the JS Agent cannot find out if the respective framework has been completely loaded and if that function hooking can be started.
Therefore actions such as AJAX actions are not captured in those browsers. Load and click actions still trigger redirects (as of anchor tags or form submits).

Visually complete metric

On this page

Overview

Single page Applications

Correlate user performance with business metrics

End user analytics

Related pages

User Action timings

Third-party content analysis

Web performance optimization for UEM

Monitor streaming media activity

Configure geographic location

W3C resource timing metrics

Overview

Visually complete is a point-in-time metric that measures when the visual area of a page's content has finished loading. Visually complete timing happens substantially earlier than the typical page load time, and optimizing the visually complete timing is more valuable then optimizing other page load timings when targeting the user experience. Visually complete has the following benefits:

- Lets you see exactly how long it takes your end users to see the information they are looking for.
- Provide a true business-relevant metric from the perspective of the user.
- 100% visibility to actual real user experiences, regardless of device.
- Align IT and business to connect user experience with business outcomes.
- Accelerate performance improvement.
- Optimize decisions across development, operations and user journey.

Along with the speed index, these metrics represent the most accurate picture of real page load timings.

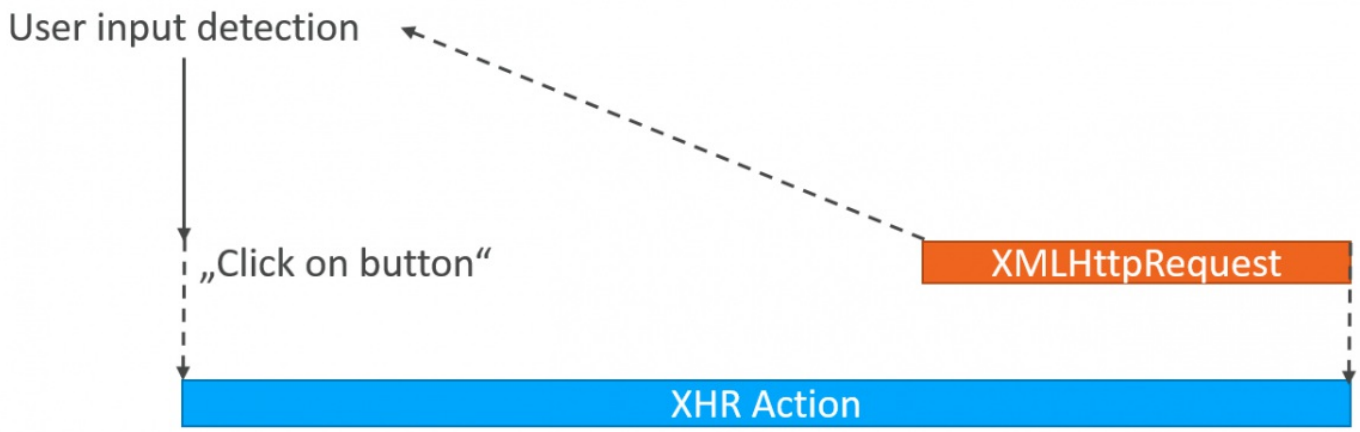
Visually complete shows you the exact time your real users spent waiting for above-the-fold content to load completely for any location, device, operating system and browser type. It can also be combined with powerful waterfall charts to understand and tune the rendering of a web page.

The Speed Index marks the average time that visible parts of a page display. It is useful for comparing the user experience of various pages. The lower the number, the better the user experience of the page. The Speed Index is only available for page load actions.

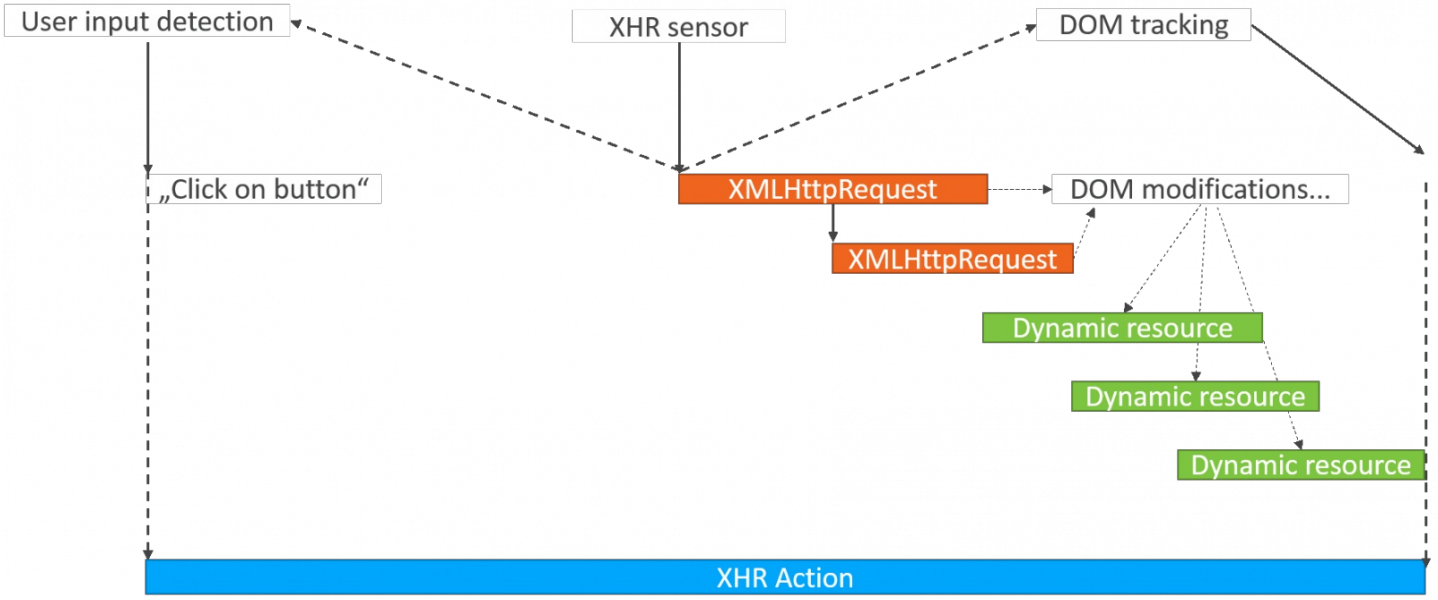
For Appmon Spring 2017 and later, the JavaScript agent captures both visually complete and speed index timings. Enable both visually complete and speed index capture for a System Profile by selecting the **Visually Complete** check box in the System Profile - User Experience settings.

Single page Applications

For single page application frameworks like Angular, visually complete gives even more accurate timings for user actions. Previously on application load, captured user interactions could be correlated with XHR requests being fired afterwards, as shown in the following:

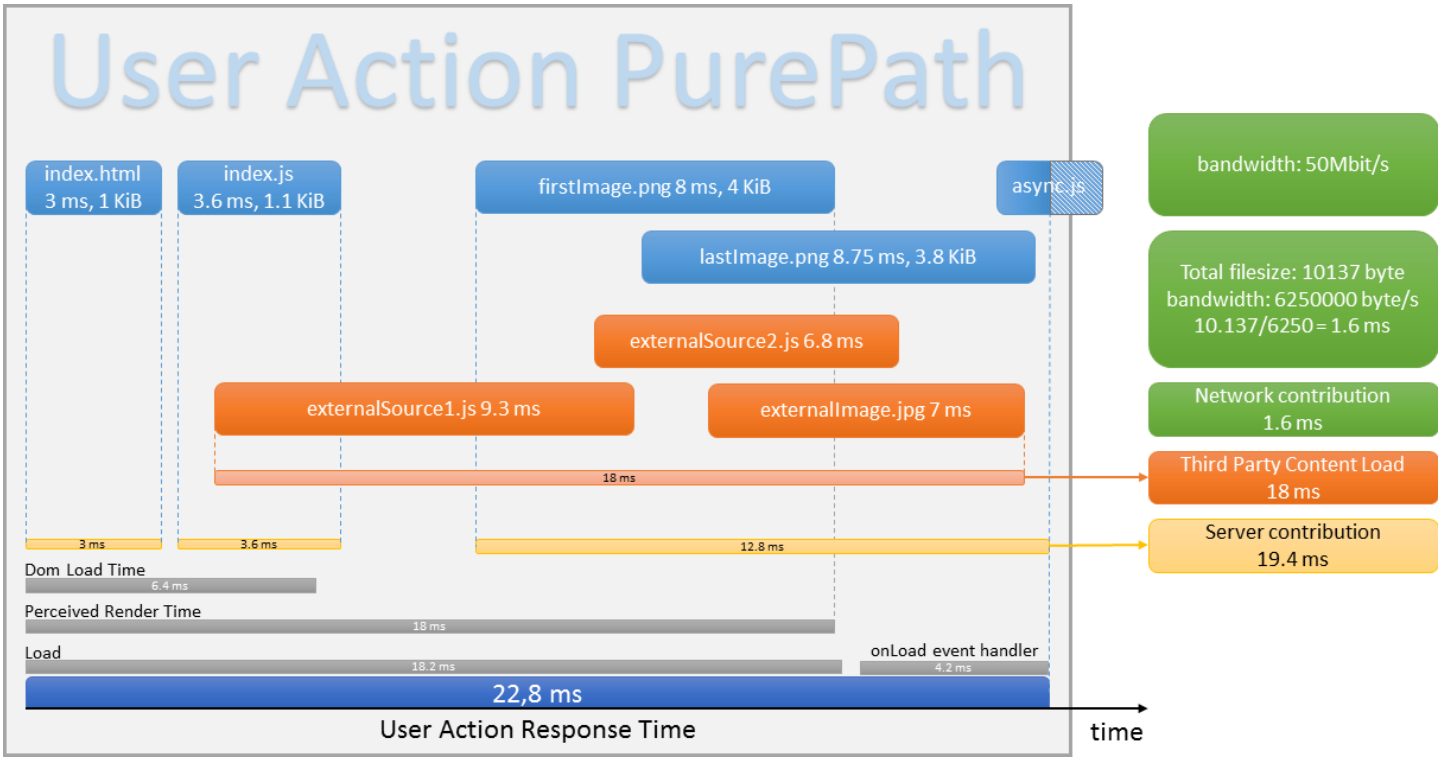


With visually complete, timing no longer stops when the XHR-request response comes back. Because of this, you can get additional insight into dynamic resource loading triggered by JavaScript-based DOM-element changes from the XHR call response.



Correlate user performance with business metrics

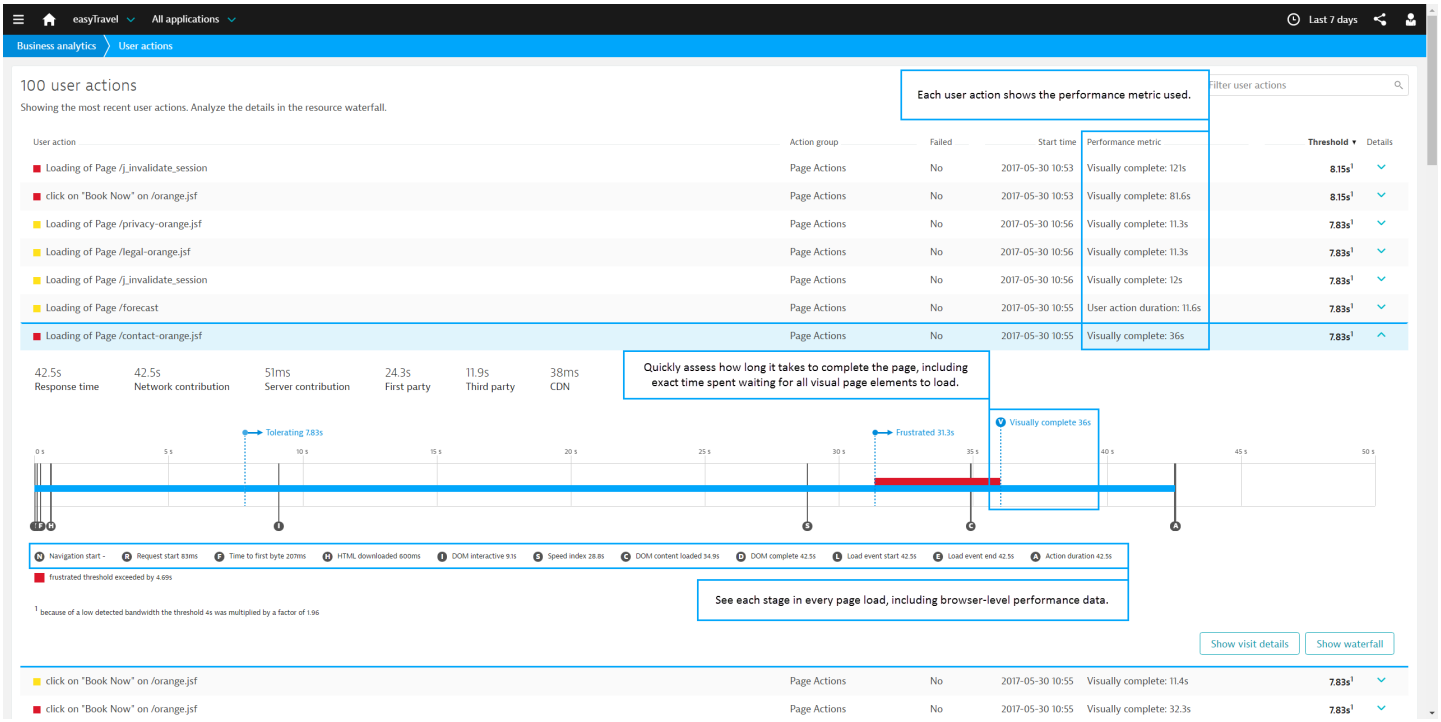
Performance is one of the main drivers for user experience, which impacts business outcomes. AppMon and the visually complete metric gives you this analysis out of the box. For example, the following graphic shows how bounce rate is impacted by performance or Frontend JavaScript errors.



End user analytics

Visually complete and speed index display for each step in a user action.

In AppMon Web, select **Visits**, then select the you want to analyze. Click **Show visit details** to view the actions for that visit. Click on an action step to see contribution timings and the step's action timeline. The action timeline shows timings for all the relevant action stages, including speed index and visually complete. When the performance metric used for the step is user action duration or visually complete, it is highlighted at the top of the timeline.



User experience management administration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

UEM for Web Applications

Troubleshooting

UEM for mobile applications

Related pages

Configure geographic location

Configure streaming media monitoring

Firewall constraints for UEM

UEM Health Check

User experience management

JavaScript Agent

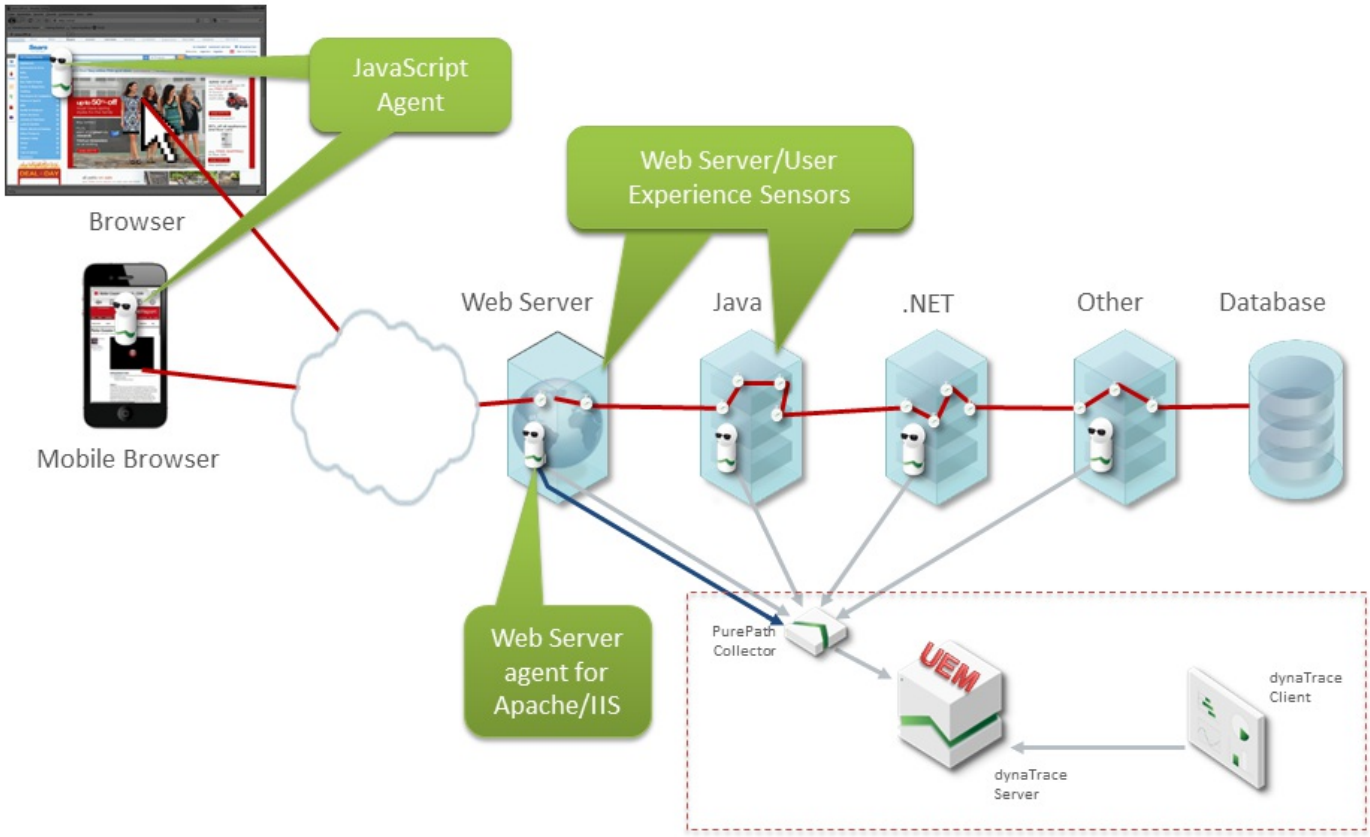
Sensor properties

Web Server Master Agent configuration

Use User Experience Management (UEM) administration to configure AppMon components for UEM. AppMon has many default configurations. This page describes specific components you must configure to successfully manage user experience. It also discusses common troubleshooting steps.

UEM for Web Applications

The central component in UEM for web applications is the JavaScript Agent. It gathers user experience data and sends it back to an instrumented web server. The Agent inside the web server intercepts the data and sends it to the AppMon Server.



To execute the JavaScript Agent, inject JavaScript code in the HTML response stream that your web server delivers. When you enable UEM in a web site, you see the code that includes JavaScript Agent in your web site's HTML code.

```
<meta name="ajs-static-resource-uri-prefix" content="/community/s/en/2166/30//_">

<script type="text/javascript" src="/dtagent42_np3_3139.js" data-
dtconfig="rid=RID_-237470873|rpid=-1849903288|domain=compuware.com|doNotInstrument=clk,mup,blr,chg|tp=200,50,3|doNotDetect=mup,
kyu,scr,kyd,clk,sbm,mdw,dcl"></script><script type="text/javascript">
    // Deprecated global variables. To be removed in a future version of Confluence.
    var contextPath = '/community';
</script>
```

You must inject a server-side Agent to enable UEM in your web site. This is either a web server Agent (Apache or IIS) or a Java Agent. Also, you must place an active User Experience Sensor in the corresponding Agent group.

You must consider the injection and compression points and instrument early in the process. Inject the Agent in your application server and send uncompressed JavaScript even if your web server or a front tier such as a load balancer compresses the outgoing content.

Troubleshooting

There are many components involved in User Experience and it can be confusing. This section discusses common issues, and guides you through a basic UEM scenario. Most of the following settings are configured correctly by default. If you migrate System Profiles or if your system does not accommodate the standard settings, you may have to make additional configuration settings.

Step 1: Execute UEM health check

You can perform a Basic UEM Heath Check if your UEM is not working. See UEM Health Check for more information.

Step 2: Verify that the JavaScript agent is injected in your web site

Open your web site with a browser and view the page source. You should see a `<script>` that includes `dtagent55_<featurehash>_xxx.js`, where xxx is an internal version number, as shown in the following.

```
<meta name="ajs-static-resource-url-prefix" content="/community/s/en/2166/3077_">

<script type="text/javascript" src="/dtagent42_np3_3139.js" data-
dtconfig="rid=RID_-237470873|rpid=-1849903288|domain=compuware.com|doNotInstrument=clk,mup,blr,chg|tp=200,50,3|doNotDetect=mup,
kyu,scr,kyd,clk,sbm,mdw,dcl"></script><script type="text/javascript">
    // Deprecated global variables. To be removed in a future version of Confluence.
    var contextPath = '/community';
</script>
```

If you find the corresponding JavaScript code, then the injection is successful. If not, continue to the following section:

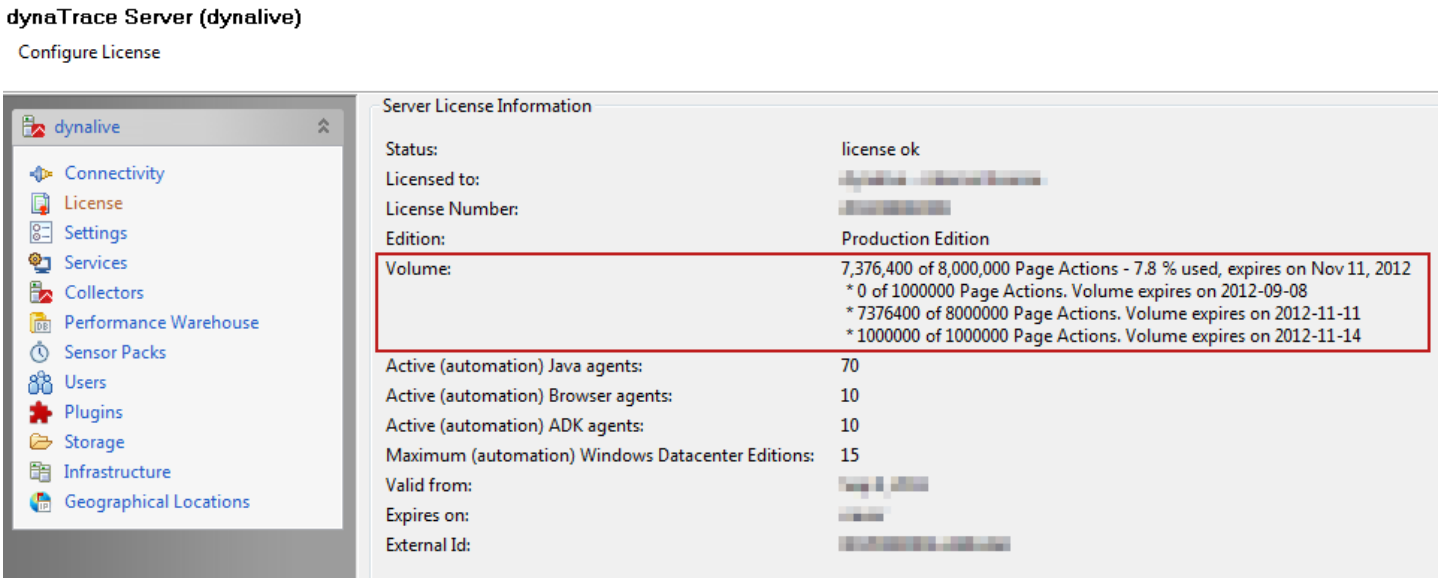


Use Internet Explorer with enhanced security configuration to disable the JavaScript functionality. Click **Internet Options > Security** to add the web site to the trusted site list.

If you find the corresponding JavaScript code, then the injection is successful. If not, continue to the following section.

Use Internet Explorer with enhanced security configuration to disable the JavaScript functionality. Click **Internet Options > Security** to add the web site to the trusted site list.

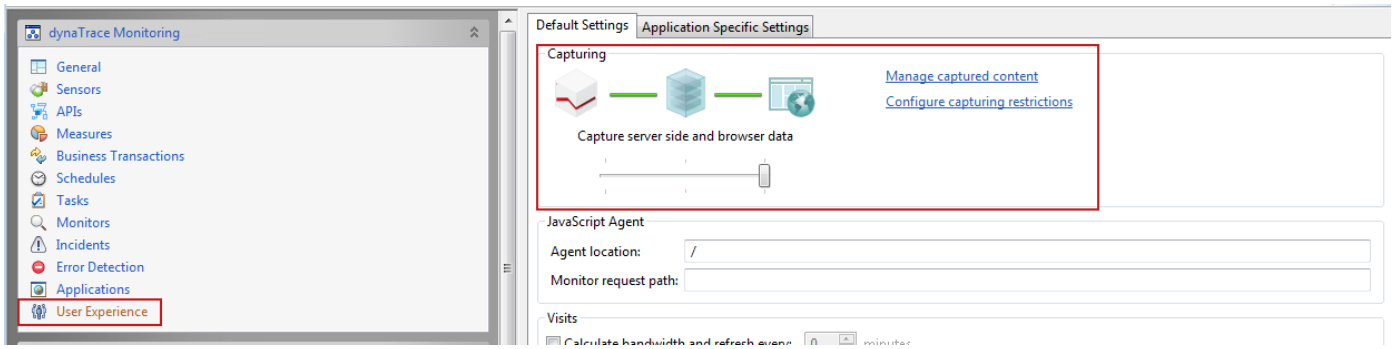
- **Verify that your license is valid**
  You need a license with valid user action volume.
  Click **Settings > Dynatrace Server** to open the **Manage Dynatrace Server** dialog box in the AppMon Client, and then go to the License tab.



- **Verify that server-side agents are connected**
  Open the **Agents Overview** dashlet from the Cockpit and verify that the expected agents are connected, active, and map to the right System Profile.

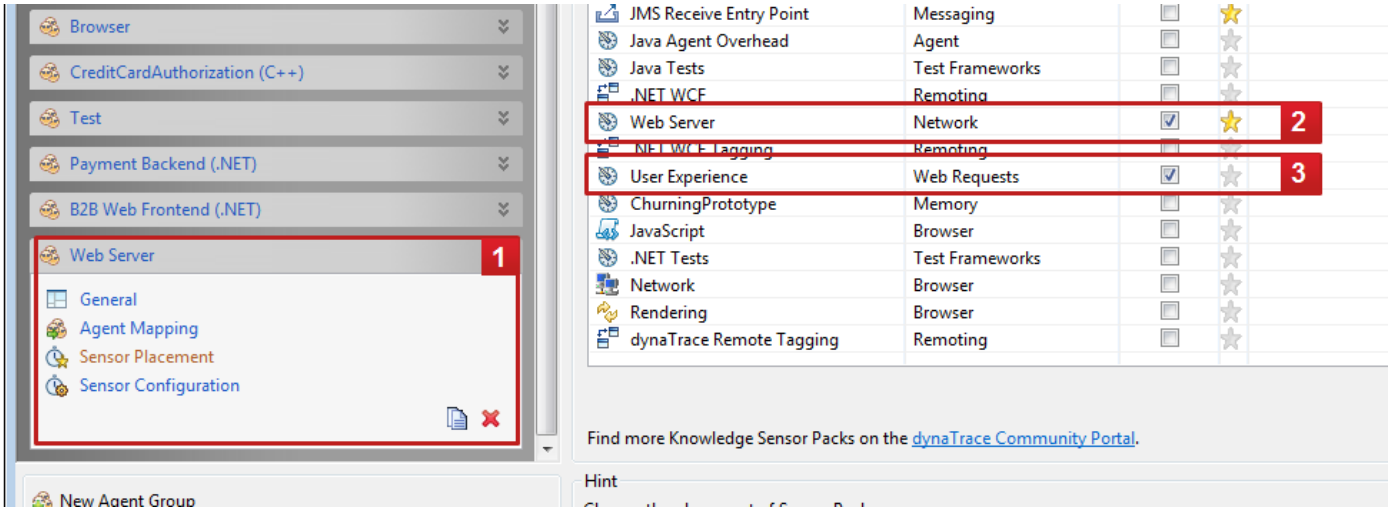
Also check to make sure that the agent captures data. Open the **PurePaths Dashlet** dashlet, browse to your web page, and verify that there are server-side PurePaths.

- **Verify that UEM is enabled**
  Choose **System Profile > User Experience** System Profile - User Experience and click the specific application tab. Make sure the capturing slider is at *Capture server side, mobile and browser data (UEM) data for this application*.

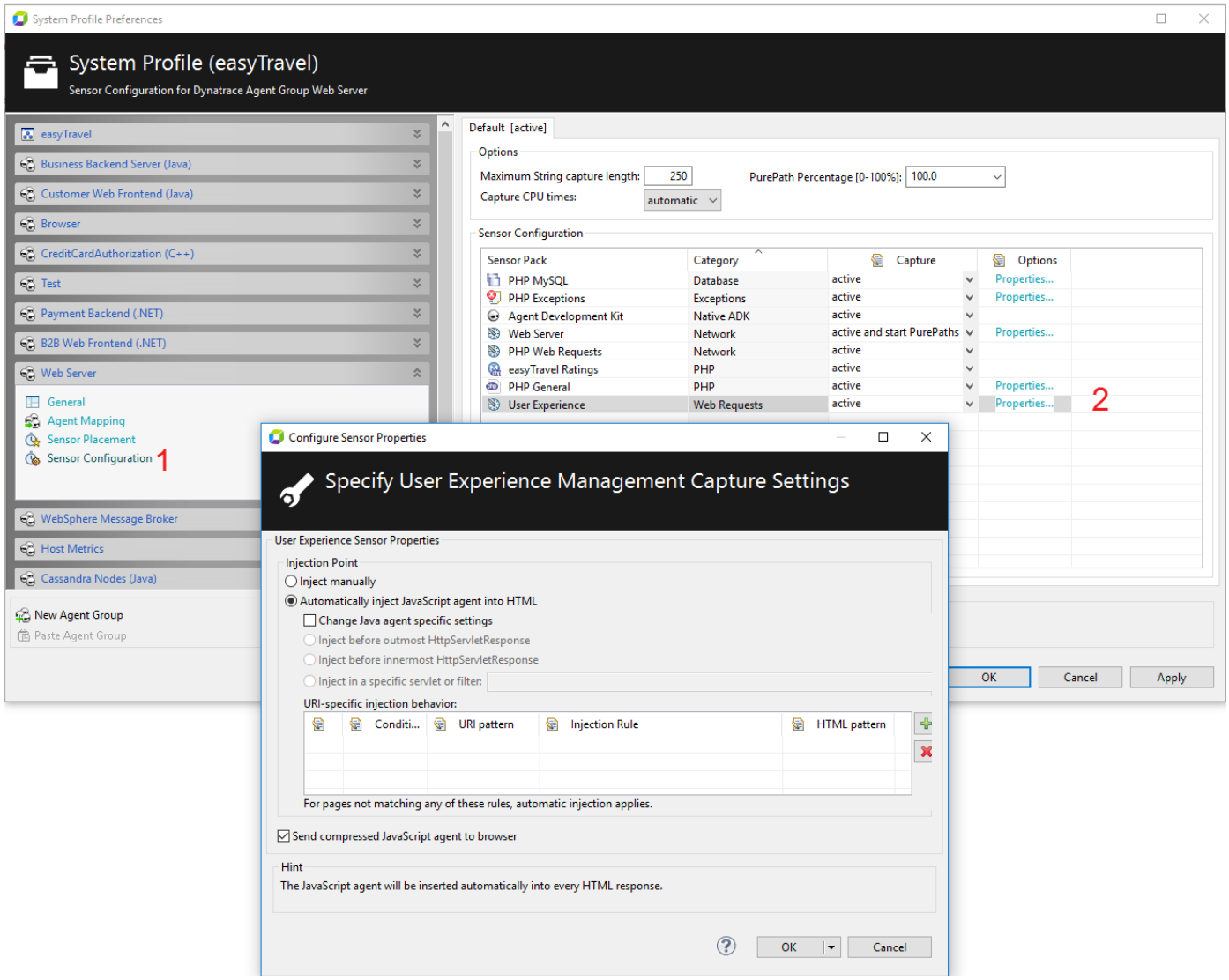


- **Verify that the necessary sensors are placed**
  For UEM to work, the User Experience sensors must be placed and active in the agent groups of the server-side agents. The server-side Agents must inject the JavaScript agent.

  To verify, choose **System Profiles > Sensors** and click the tab for the corresponding agent group. The User Experience sensor must be in the table and active. If not in the list, it is not placed. To place it, in the System Profile Preferences for the applicable system profile, select **Sensor Placement** for the specific application and select the **Place** check box for **Web Server** and **User Experience** in the Sensor Placement list. Restart all applicable Agents after placing the sensor. If the sensor is not in the list of placeable sensors, your license may not support UEM.



- **When you inject on a Java application server, change the injection mode**
  You can configure the position in the servlet/filter chain where the User Experience sensor that injects the JavaScript Agent on a Java application. To configure this, click **System Profile > Web Server agent group > Sensor Configuration > User Experience**. For description of properties, see SensorProperties - UserExperience.



- **Verify that the web server agent and the web server module can communicate**
  If you get the following error messages in the logs of the web server agent and or the web server module, change the file system permissions on the file `<DT_HOME>/agent/conf/dynaTraceWebServerSharedMemory` as shown in the following, so that both processes can write.

```
warning [native] Could not open file C:/Program Files/dynaTrace/dynaTrace
6.2.0/agent/conf//dynaTraceWebServerSharedMemory: Access is denied.
severe  [native] Could not create or attach to shared memory segment. Using
        private data instead.
severe  [native] This means the shared memory segment used for communication
        between the web server agent and the web server module could not be
        created or attached to. Try deleting the file and make sure the agent has
        write permissions to it. The backing file used was: C:/Program
        Files/dynaTrace/dynaTrace 6.2.0/agent/conf//dynaTraceWebServerSharedMemory
```

- **If you use IIS, be sure the web server Agent module is enabled for each site**
  See Microsoft IIS Web Server Agent Configuration for more information.
- **Verify that the HTML page is valid:**
  Look at the flow diagram to verify that your HTML page matches the injection rules. Look for unmatched delimiters and wrong HTML structuring.
  Examples:
  Unmatched delimiters (double quotes):

```
<meta http-equiv="X-UA-Compatible content="IE=EmulateIE9" />
<link type="text/css" rel="stylesheet" href="...." media="all">
```

Mistyped delimiter / double quote:

```
<meta http-equiv="X-UA-Compatible? content="IE=EmulateIE9" />
<link type="text/css" rel="stylesheet" href="...." media="all">
```

Wrong tag before <html> tag:

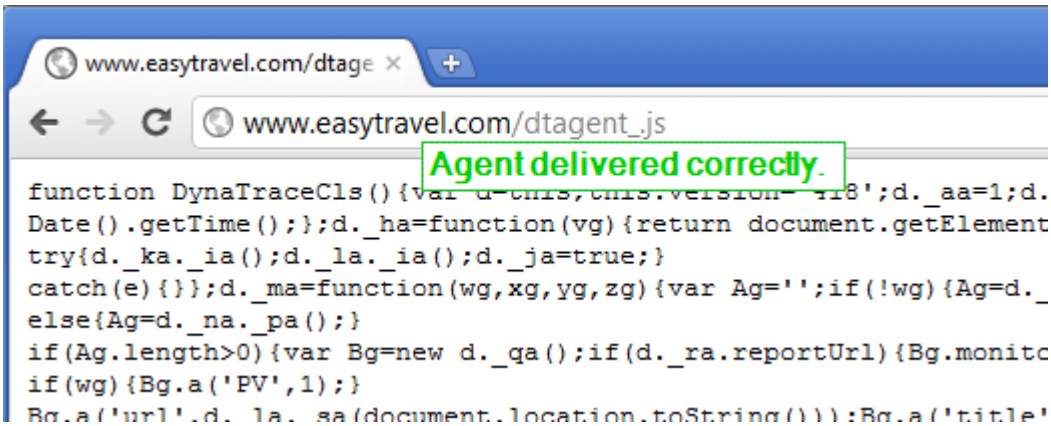
```
<style type="text/css">
<html>
...
```

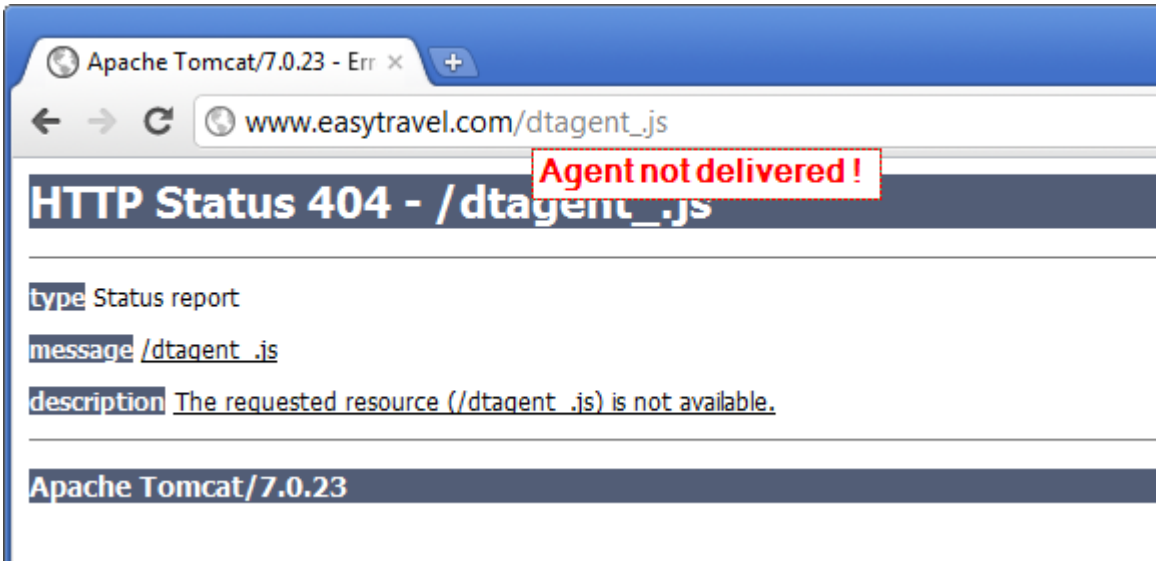
Step 3: Check your firewall constraints

The web server injects the JavaScript Agent into your website, but some firewall settings may block communications with the AppMon Server. See Firewall Constraints for UEM for more information. UEM Health Check typically detects issues related to the firewall.

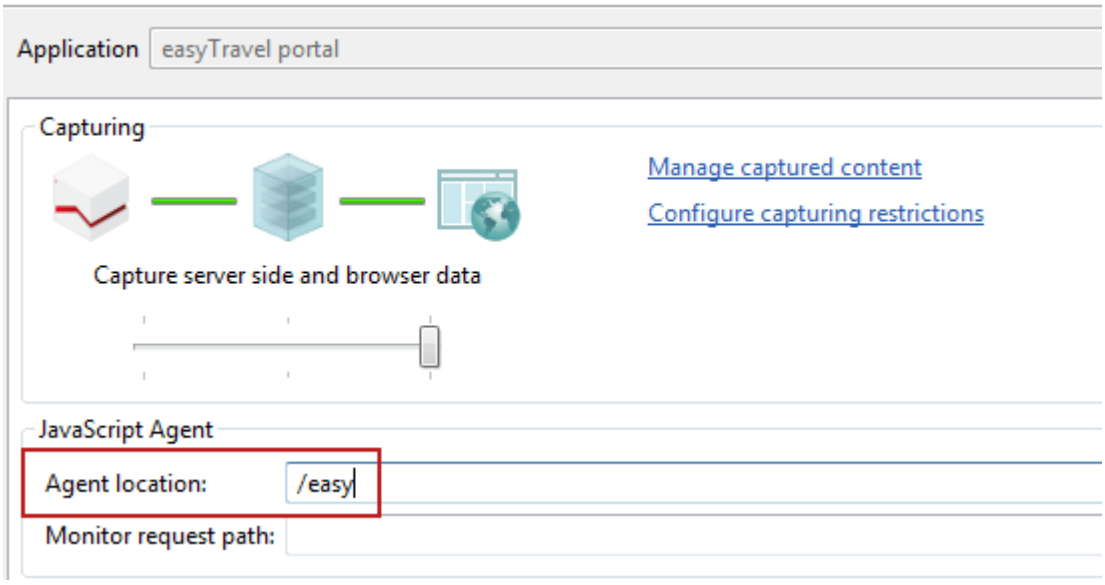
Step 4: JavaScript Agent is injected, but still no user actions or visits are created

- **Verify that the JavaScript Agent is available**
  Download the JavaScript Agent (FireFox supports hyperlinks in its page source browser). If this does not work, the request cannot reach a server-side agent.



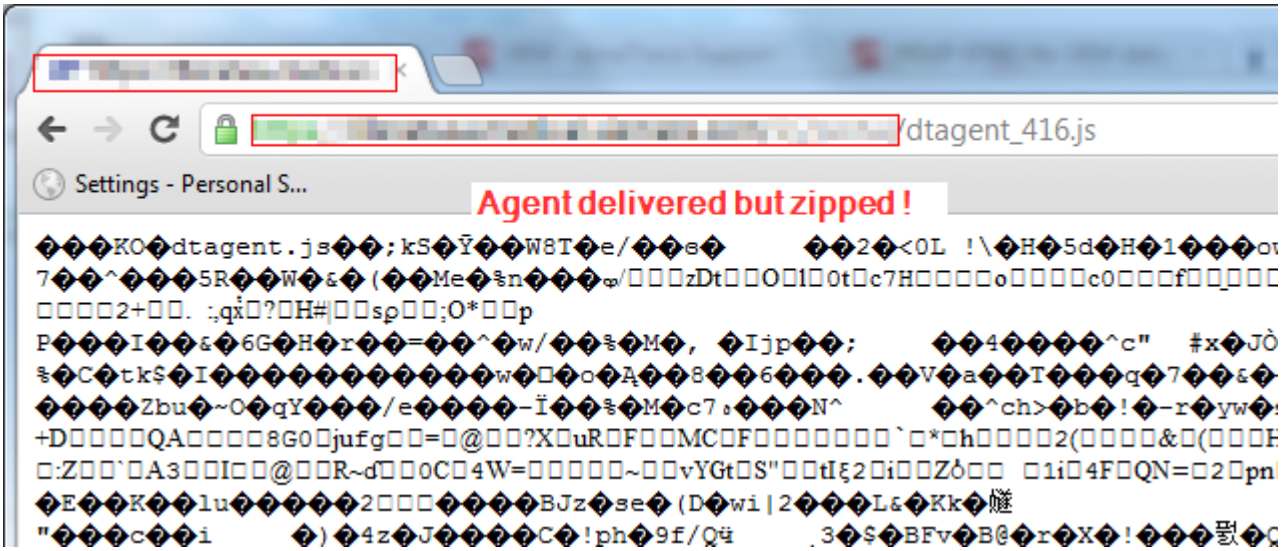


Make sure the request reaches a server-side agent and that there are no URL rewriting rules which could distort the request URL. To configure a different path, go to the System Profile - User Experience dialog box.



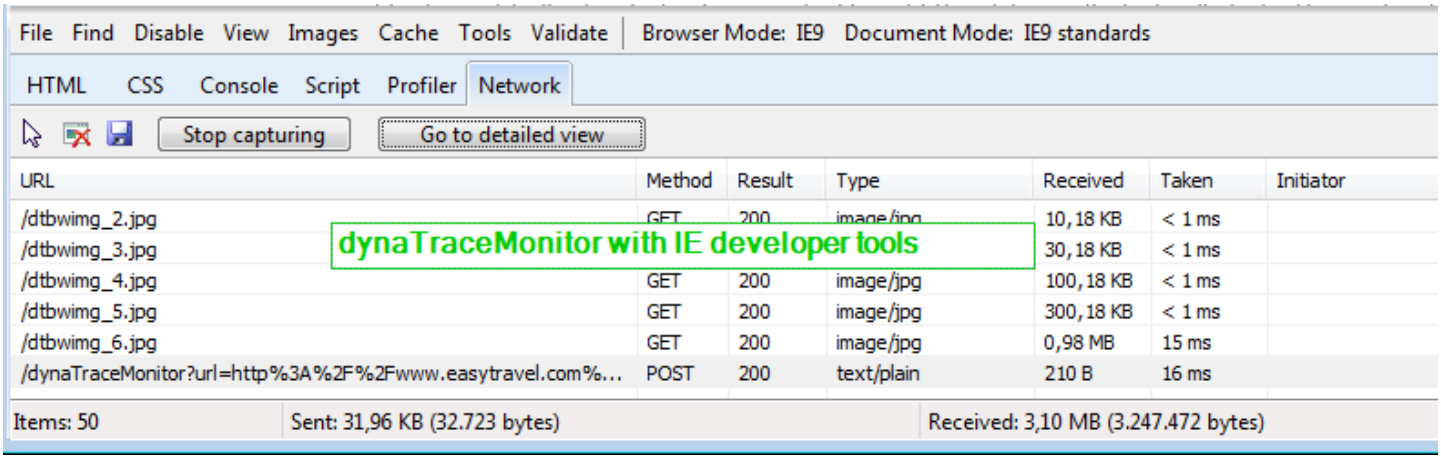
- **Verify that the JavaScript Agent is not double-zipped**
  To keep the network overhead small, the `dtagent_###.js` file is zipped to the browser. Although the content encoding header is set correctly by the AppMon Agent, some applications or web servers seem to ignore this and compress the `dtagent_###.js` file again. As a result, the browser cannot interpret the double-zipped JavaScript file and UEM does not work.

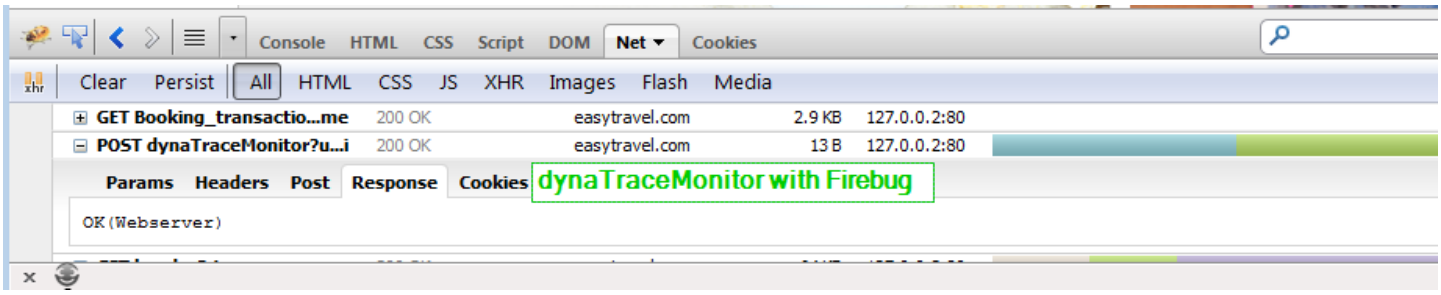
The screenshot below shows a double-zipped agent file when you access it via the address bar in the browser.

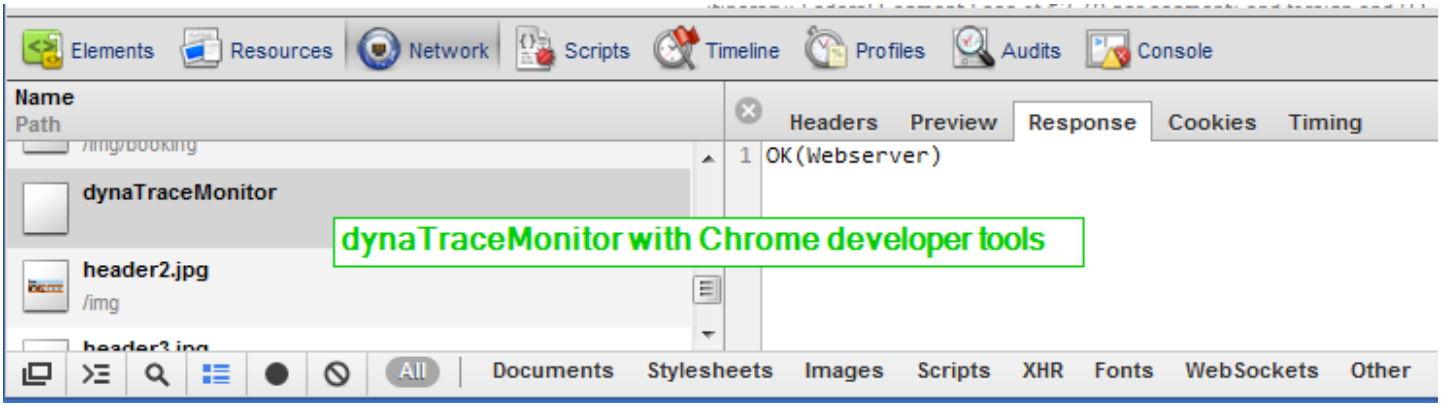


You can bypass this problem by deactivating zipped agent delivery in the User Experience sensor. To do it, right click the system profile, select **Configure System Profile > Web Server agent group > Sensor Configuration > User Experience**, and uncheck the **Send compressed JavaScript agent to browser** checkbox.

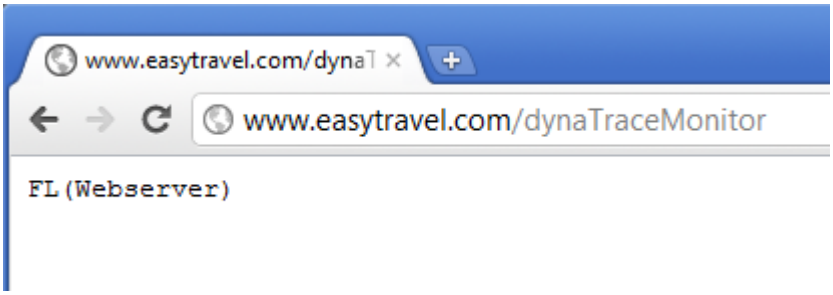
- **Verify that monitor requests performed by the JavaScript Agent are correctly handled by server-side agents:**
  Check your browser if the XHR signal is sent five seconds after page load (for example, with Firebug in Firefox, or with Developer tools in IE or Chrome).



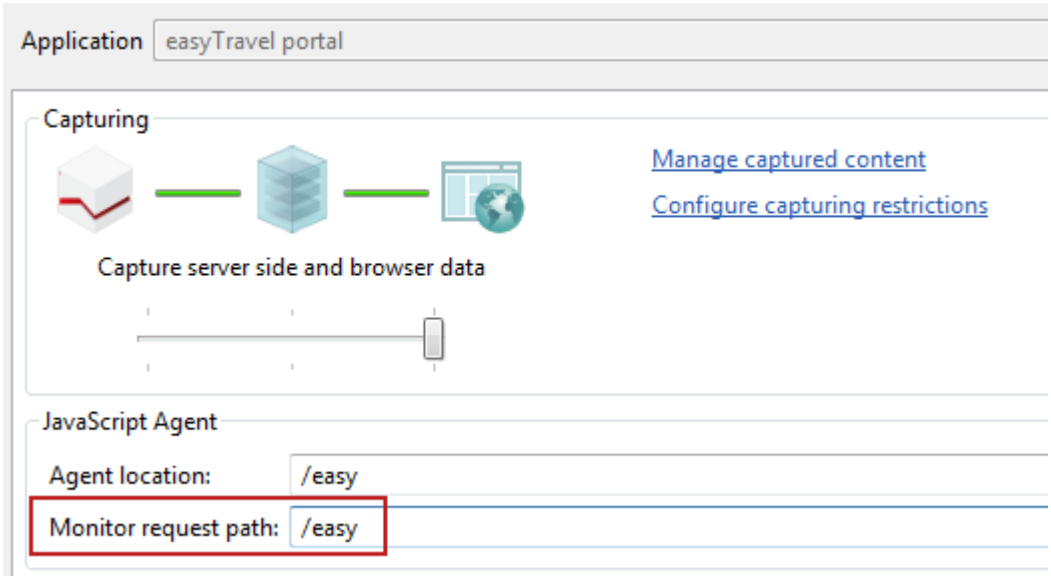




Browse to your web site and replace the last URL path part with `dynaTraceMonitor`. The response should be text only - either `FL(Java)` or `FL(Webserver)`.

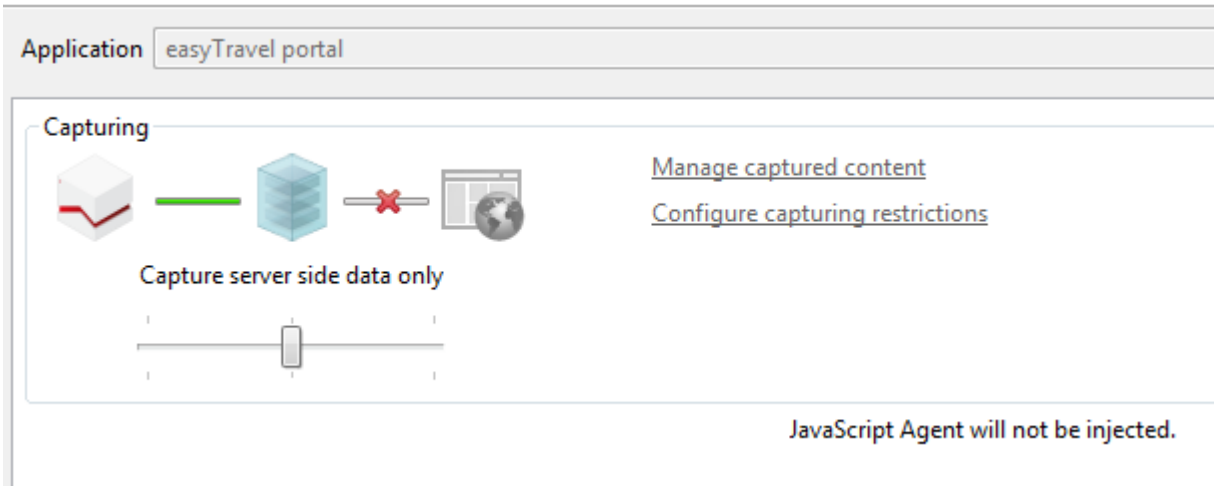


Use the System Profile - User Experience dialog box to configure a different path where the agent should send its monitor requests.



Step 5: UEM causes JavaScript errors, page layout and/or functionality is broken

- **Verify that failures are really caused by UEM**
  To verify that the JavaScript Agent causes the failures, disable UEM and try again.



- **Disable capturing settings**
  Disable the capturing settings in the Web applicaitons section of the **System Profile - User Experience** dialog and try again. Contact Support if settings break in your application.

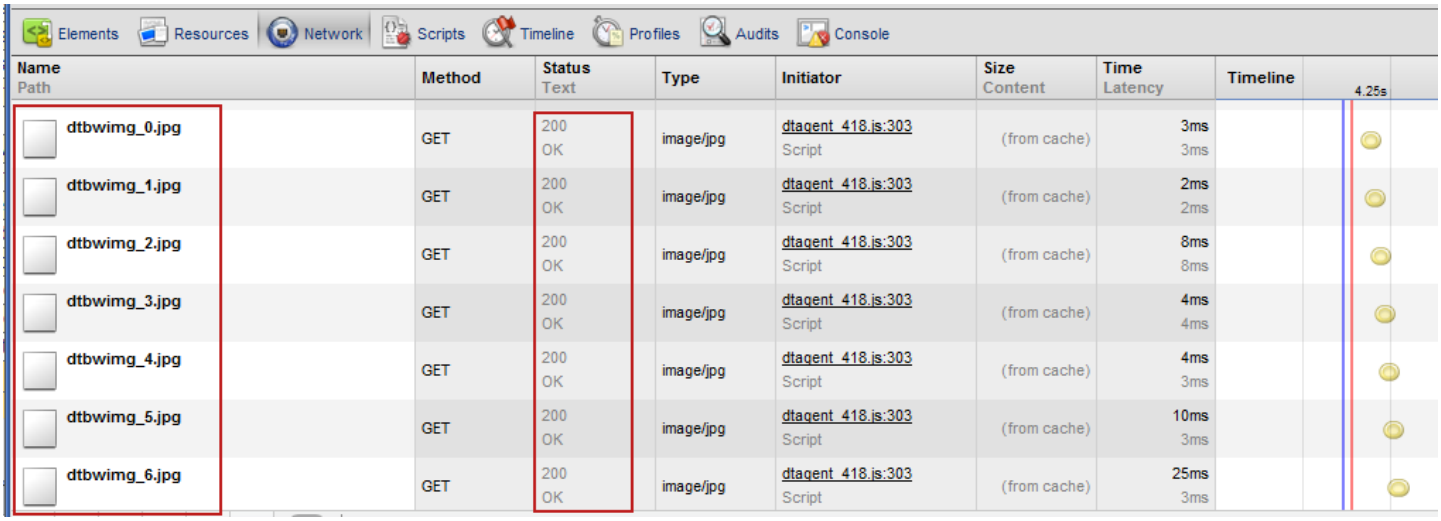
Step 6: Only loading user actions are captured but no user actions, web requests are missing in user actions

- **Verify capturing settings**
  Choose **System Profile > Manage captured content** to ensure that events are captured, and support for the used framework is activated.
- **No user input can be associated with action**
  Auto-suggest features often use `setTimeout` to trigger requests. The JavaScript Agent does not detect these actions.
- **Web requests are missing in user actions**
  Web requests may be missing for the following reasons:
  - The Web request is performed after load using an asynchronous script and the request cannot be correlated. Use ADK instead.
  - The Web request is performed by an Ajax framework which is not enabled or not supported by AppMon. To solve this instrument via the JavaScript Ajax ADK.
  - The request maps to another application.

Contact Support if the request should be part of a user action.

Step 7: Bandwidth is not detected

- **Verify that bandwidth detection is enabled**
  In the System Profile Preferences for the applicable system profile, select **User Experience**. Select the appropriate application tab and check settings in the Bandwidth calculation section and verify that bandwidth calculation is properly set. Also check any application specific settings. See UEM FAQ for detailed info.
- **Verify request/response of bandwidth XHRs**
  Check your browser if the bandwidth XHRs are requested, and the response is acceptable (for example, with Firebug in Firefox, or with Developer tools in IE or Chrome).



Step 8: Conflict with instrumented event handlers

Event handlers, which are instrumented by the JavaScript agent, can conflict with the JavaScript code on the web page.

To find out whether this is the problem, follow these steps:

1. Go to your System Profile and click **User Experience settings > Manage captured content**.
2. Disable all check boxes in **Instrumented event handlers**.
3. Save the System Profile and refresh the web page.

If you do not get a JavaScript error, the problem may be the event capturing.

If you turn off the instrumentation and do not lose user actions, leave it off or contact Support. The instrumentation replaces original event handlers with wrapper functions and captures information. When it replaces the original event handler, it can be problematic. The following code shows how the wrapping is done.

Why is the instrumentation problematic? The instrumentation replaces original event handlers with wrapper functions and captures information. When it replaces the original event handler, it can be problematic. The following code shows how the wrapping is done:

```
if(htmlElement.onclick){
  var originalOnclick=htmlElement.onclick;
  htmlElement.onclick=function(){
    [...capturing code...]
    originalOnclick.apply(this,arguments);
  }

}
```

When is instrumentation problematic?

- Properties are set on the event handler function.

```
 <script>
  var buttons=getElementByTagName('button');
  for(var i=0;i<buttons.length;i+){
    if(buttons[i].onclick){
      buttons[i].onclick.myProperty='button'+i;//sets a property on the function
    }
  }

  function btnClick(e){
     var e=e || window.event
     var target = e.target || e.srcElement;
     alert(target.onclick.myProperty); //this will throw an error when onclick is wrappec
  }
</script>
<button onclick="btnClick()">Button</button>
```

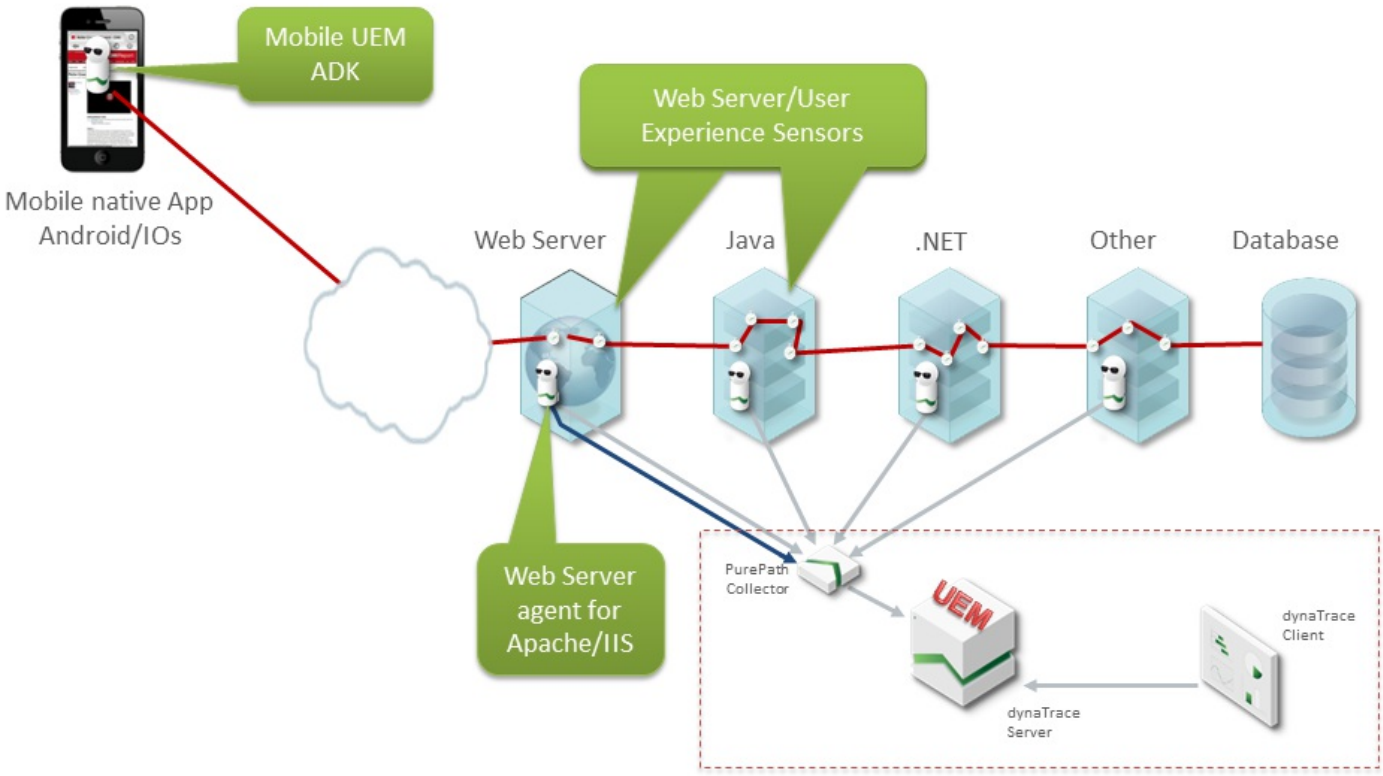
- The event handler is checked via `getAttribute` on the respective DOM node. When a direct property accessor such as `object.propertyX=value` sets a property on a browser, the `getAttribute('propertyX')` function does not return the expected value. The instrumentation overrides original event handlers this way and can cause JavaScript errors.
- There are several other reasons why wrapping does not work. This depends on web application implementation and the frameworks and tools it uses.

Contact Support if you have another issue not covered by the above information.

UEM for mobile applications

Use the Mobile App ADK for Android and iOS to capture transactions that start within mobile apps. To indicate a user action's start and stop, call the library with a simple API in the code of a mobile app. The captured data is sent to a server-side Agent. It is handled analogously to the data that the JavaScript Agent sends as described above. See Mobile Agent for more information. Use the Mobile App ADK for Android and iOS to capture transactions that start within mobile apps. To indicate a user action's start and stop, call the library with a simple API in the code of a mobile app. The captured data is sent to a server-side Agent. It is handled analogously to the data that the JavaScript Agent sends as described above. For more information, see Mobile Agent .

UEM for Mobile Applications Diagram



Firewall constraints for UEM

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

HTTP requests

HTTP headers

Cookies

Troubleshooting

Related pages

User experience management

UEM Health Check

UEM uses some HTTP technologies to send performance data from the browser to the AppMon Server. To do this, JavaScript Agent code injects into your HTML pages. This code communicates with the web server or Java Agents, which forward collected data to the AppMon Server. To fully enable UEM, however, you must verify that your firewalls, proxies, and web servers are configured properly.

HTTP requests

To function fully, the web server or Java Agents must recognize the following requests of a browser and deliver them to the browser.

- `dtagent_xxx.js` is the JavaScript Agent. Typically, this is from the absolute root directory (the code inserted is `<script href="/dtagent_xxx.js" />`). You can configure it in the User Experience section of the System Profile dialog box. Use an absolute URL, if possible, because browser or any proxy servers should cache this file.

- `dynaTraceMonitor` is the monitor signal the JavaScript Agent sends back to the server. This is sent relative to the current page. You can configure it in the User Experience section of the System Profile dialog box. This setting also dictates where the image requests (`dtbwimg_x.jpg`) used for bandwidth detection should go.

HTTP headers

The User Experience sensor uses the following HTTP headers.

- `rproxy_remote_address`, `True-Client-IP, X-Client-Ip`, `x-forwarded-for`, `x-http-client-ip` (in this order): Determine the IP of the client (also for geo-locating). If none of these headers is set, AppMon uses the remote peer's IP address.
- `X-Forwarded-Host, X-Host`, and `Host`: Determine the page host domain (also for application autodetection).
- `x-dtreferer`: In some cases the JavaScript agent has to set this header and this header is used on the web / Java agent instead of the referer header.
- `x-dtPC`: In case of parallel XHR requests the JavaScript agent has to set this header which is used by the correlation to link web requests correctly to user actions.

Cookies

The JavaScript Agent and the web server and Java Agents use the following cookies:

- `dtCookie`: Session cookie used to identify user sessions. This cookie is valid as long as the browser process is running. Do not confuse this with HTTP-sessions of web applications or visits. Both usually have timeouts.
- `dtPC`: Page context cookie used to link web requests to user actions.
- `dtLatC`: Latency of monitor signals.
- `dtSA`: Source action cookie. On browsers that do not support sessionStorage.

Troubleshooting

Use UEM Health Check to detect firewall issues and communication between the injected JavaScript Agent and the AppMon Server. See UEM Health Check for more information.

UEM Health Check

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Basic checks

Application checks

Related pages

User experience management administration

Manage UEM visits volume

System Profile - User Experience

Sensor properties

How to get Support

Important

Make sure that the HTTP headers `x-dtPostBody`, `x-dtHealthCheck`, and `x-dtAgentId` do not get blocked in your environment. The UEM Health Check needs them to work properly.

In the AppMon Client Start Center, select **Monitoring > UEM Health Check > Check your UEM configuration** to run a UEM Health Check.

Basic checks

When you perform a UEM Health Check, AppMon executes a number of basic checks. These do not depend on any application-specific settings. AppMon also executes a check when applications are not selected for the UEM Health Check process.

License check

- **Volume Expired**: If the UEM visits volume expires, you must redeem another voucher or purchase additional volume. See Manage UEM Visits Volume for more information.

- **Volume Exhausted**: If the number of page visits in the license is expired, see Volume Expired to redeem or purchase a UEM Volume.
- **No UEM Volume Included in License**: To purchase a UEM Volume, see Volume Expired.

- **No License Information Available**: If the check does not determine any licensing information, click here to contact support.

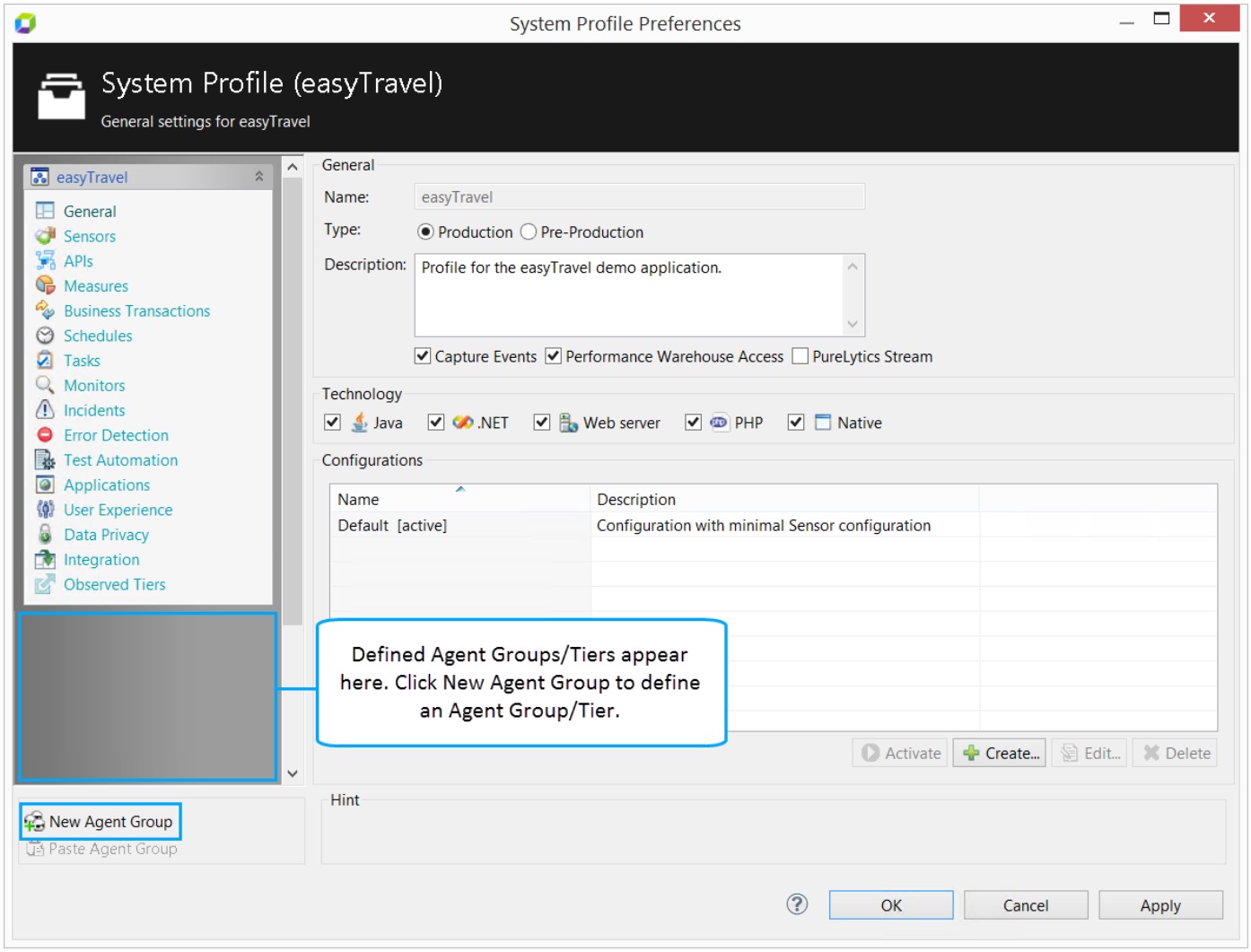
- **License State Error**: If the license is expired, invalid, not activated, locked to a different machine, or unknown, click here to contact support.

Agent check

This checks the availability of Agent Groups in the System Profile, and the mapping of Agents to Agent Groups. (Tiers)

- **Broken Sub-Agents**: If there are configuration problems such as those related to shared memory, then the sub-Agent reports as broken. Find the reason and details in the Agent logs.

- **No Agent Group / Tier**: Agent Groups / Tiers are not defined for the System Profile. To open System Profile preferences, double-click a System Profile or click **Edit System Profile** in the Configuration Panel (if available) and define at least one Agent Group / Tier with a mapping for your Agents.



- **No Agent Connected**: An Agent is not connected to the Agent Group. Use the following tasks to diagnose the issue:
  - Check the mappings in your Agent Groups and make sure that at least one Agent is connected for each defined application.
  - Make sure that at least one Agent Group has an active and placed User Experience Sensor with at least one Agent connected.

Application checks

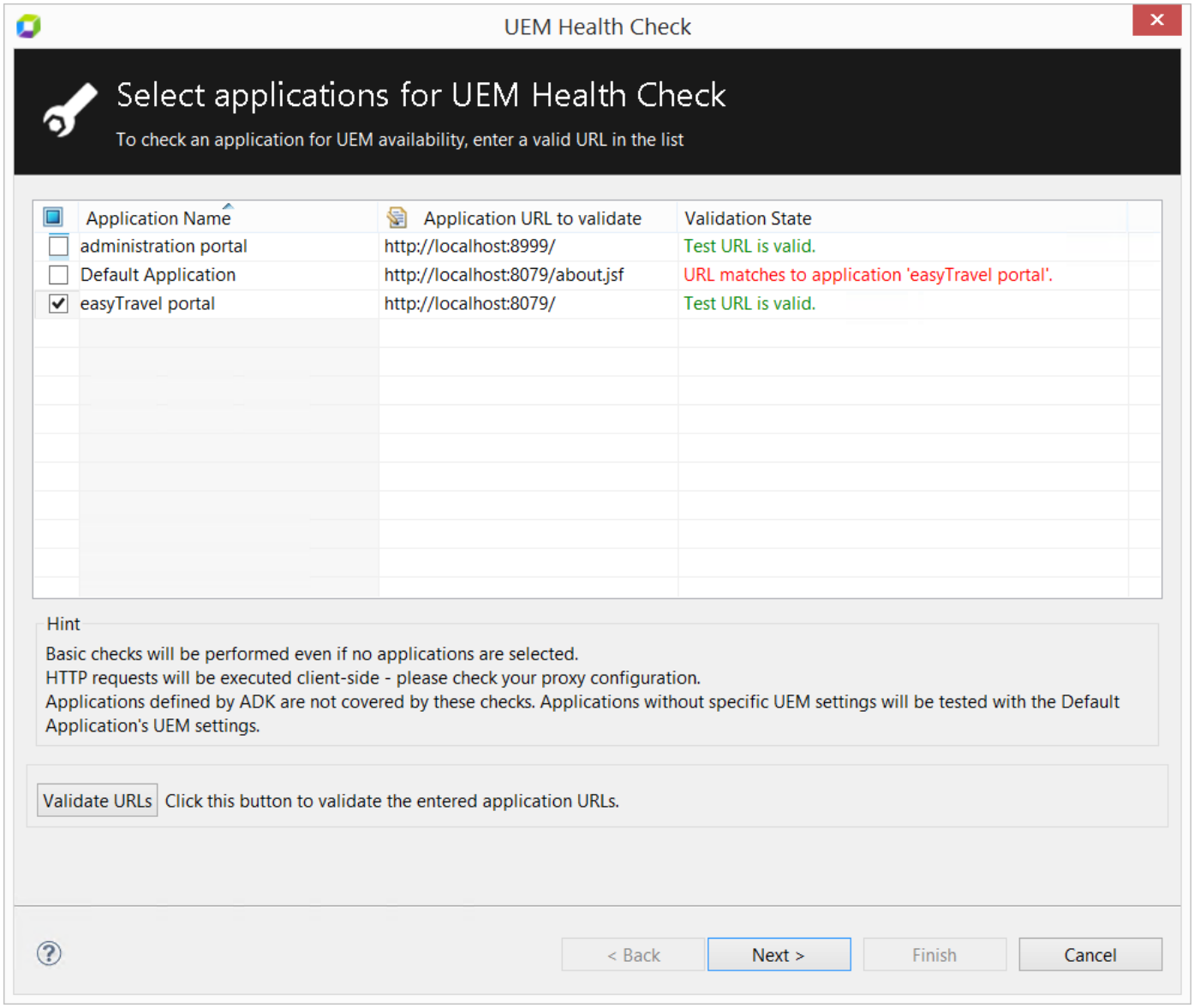
For each application that you select in the UEM Health Check wizard, AppMon performs the following application-specific checks:

- UEM Enabled Check
- Injection Check
- Agent Version Check
- JavaScript Agent Delivery Check
- DynatraceMonitor Request Check

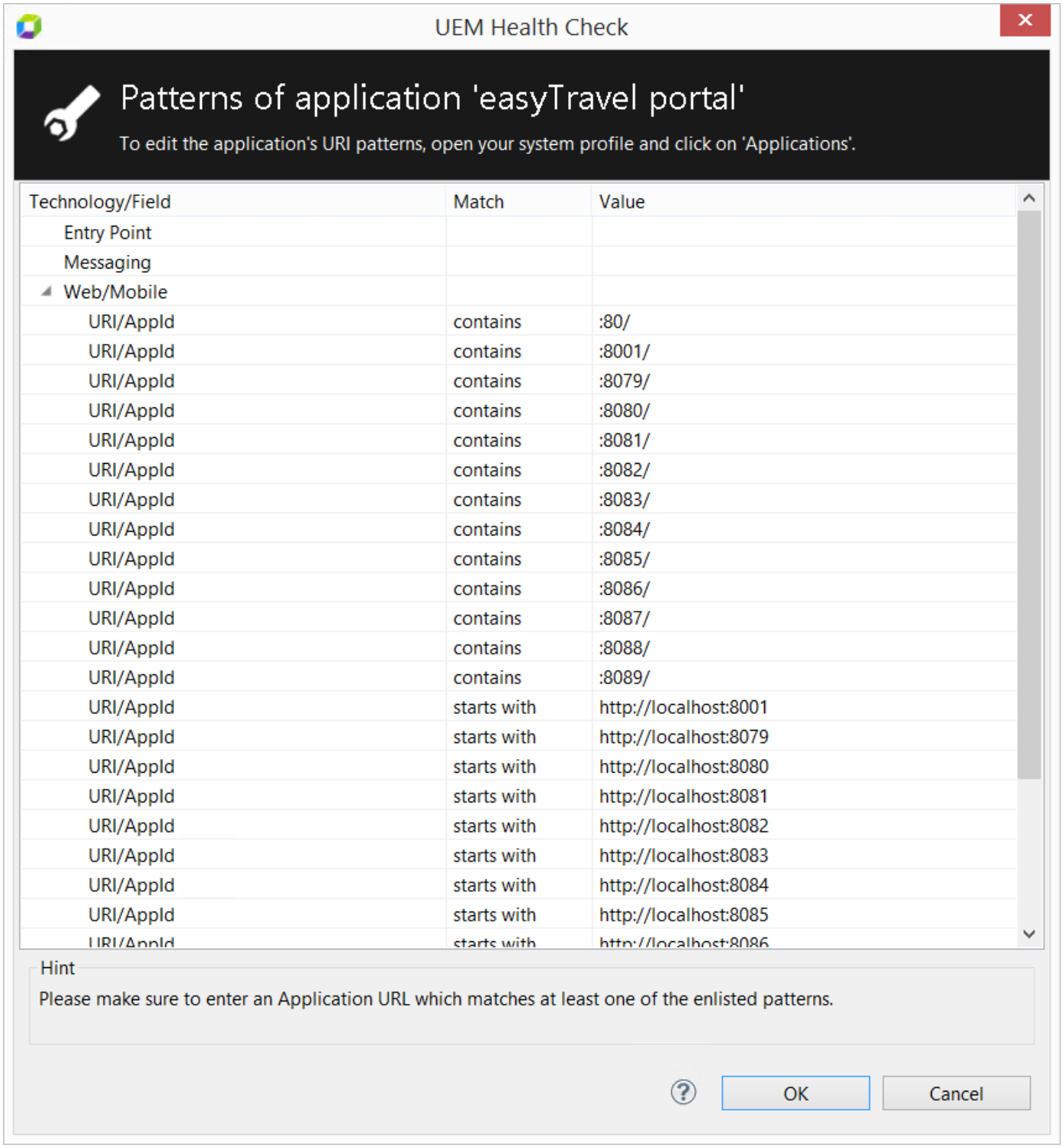
If one of these application-specific checks terminates with an error, AppMon skips all dependent checks.

Example workflow

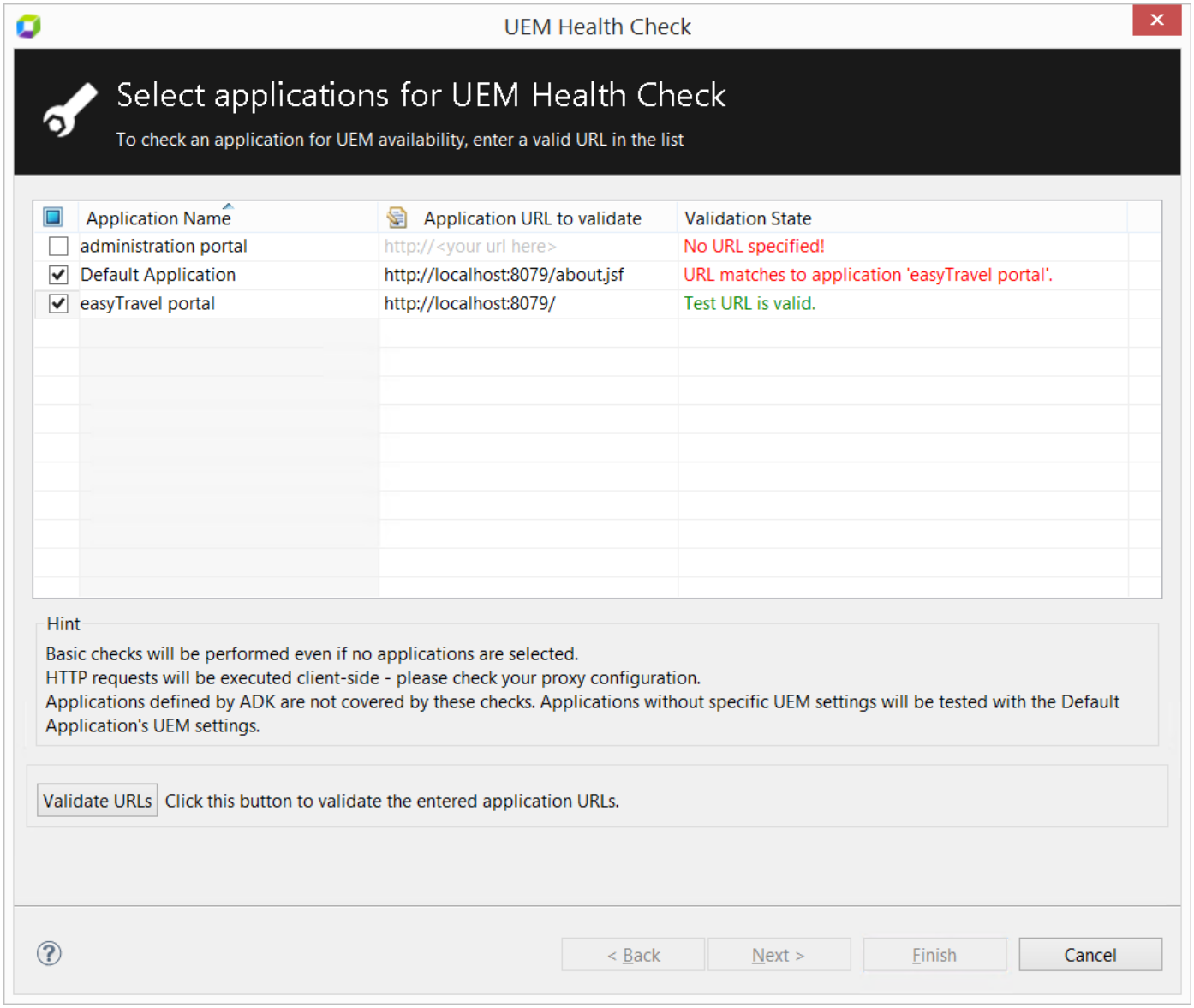
1. Select an application for EUM health check and validate the URLs.



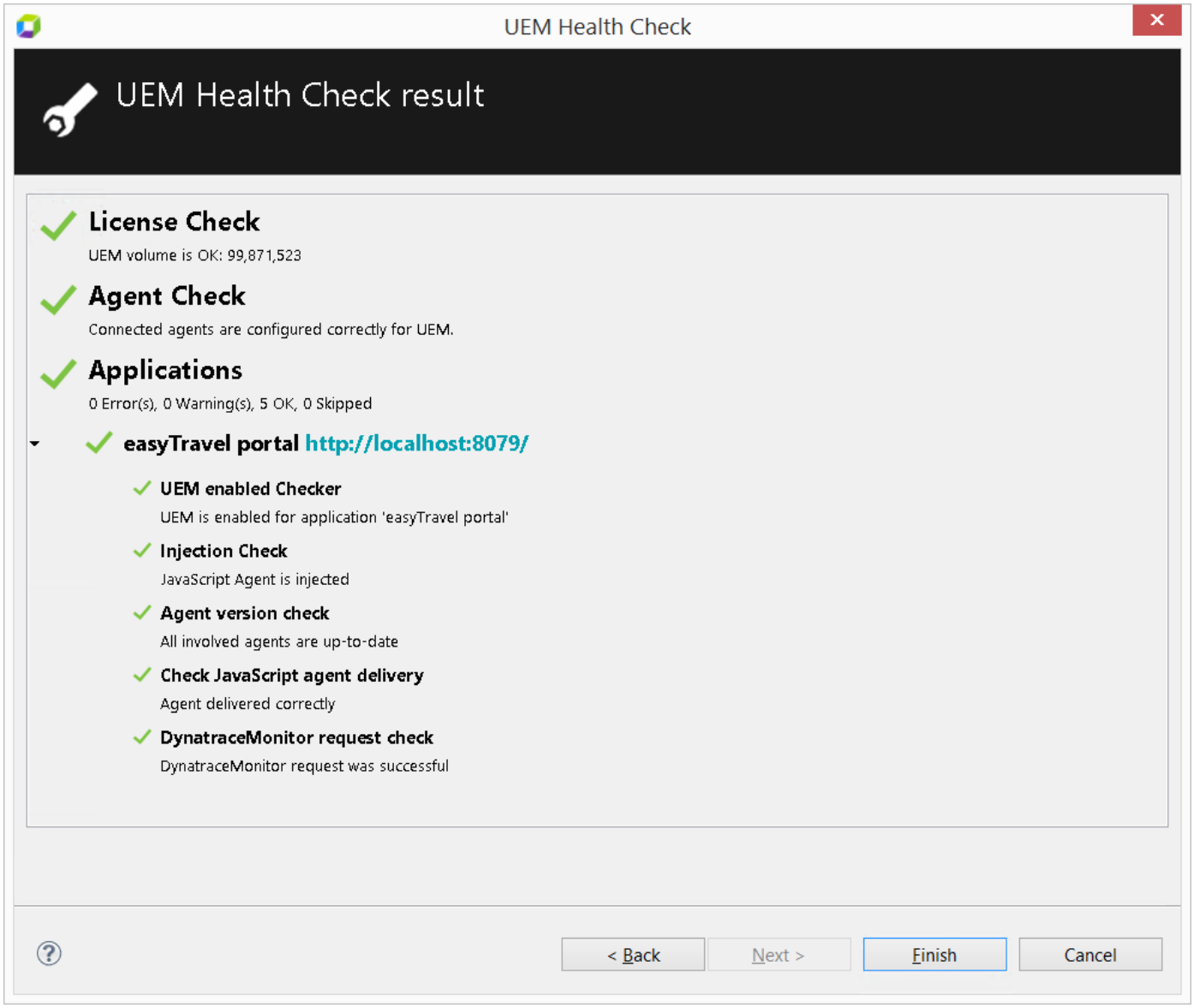
2. URI patterns for the selected applications.



3. Clear the selection for the application with no URL specified.



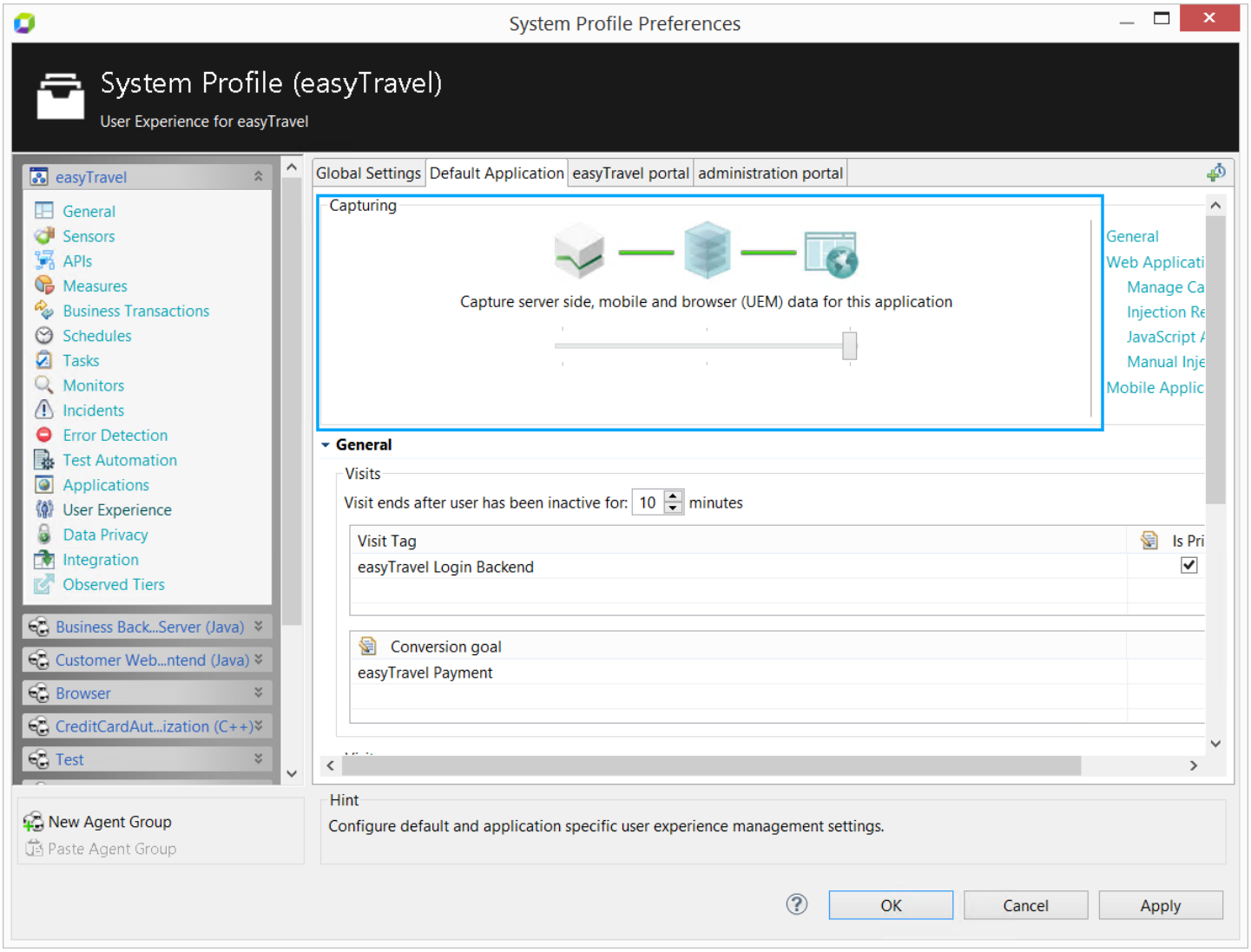
4. UEM Health Check result



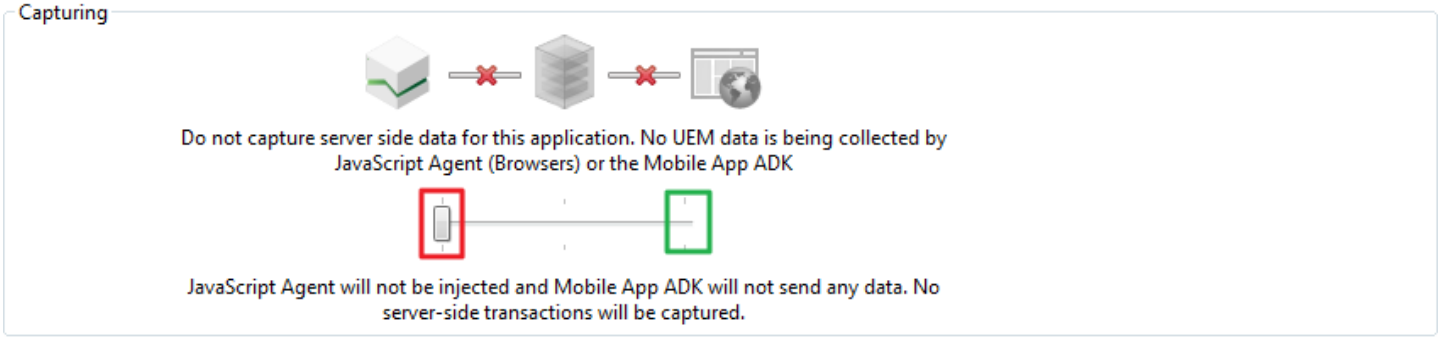
UEM enabled check

This check makes sure that capturing UEM data from the browser is enabled. It fails if capturing is enabled only for server-side data, or if capturing is disabled completely.

Change settings in **System Profile preferences > User Experience**.

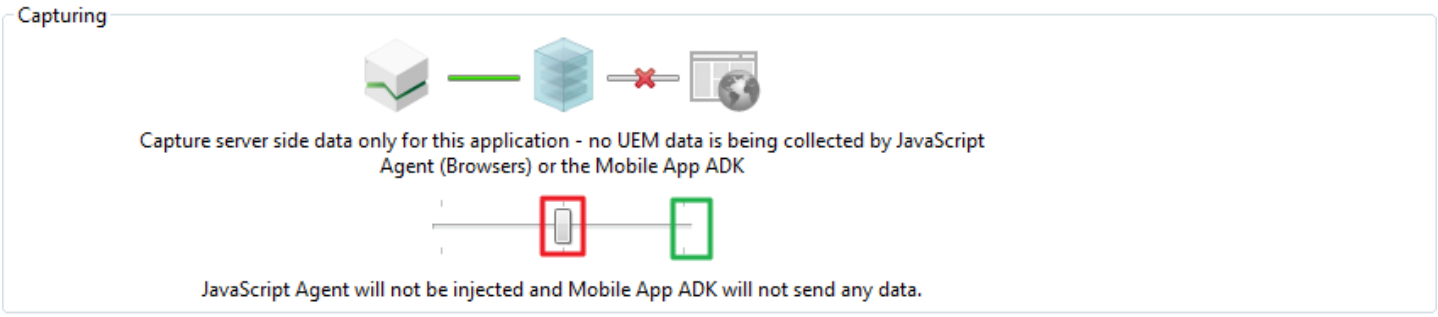


- **UEM Disabled**: UEM capturing is disabled completely for the checked application.

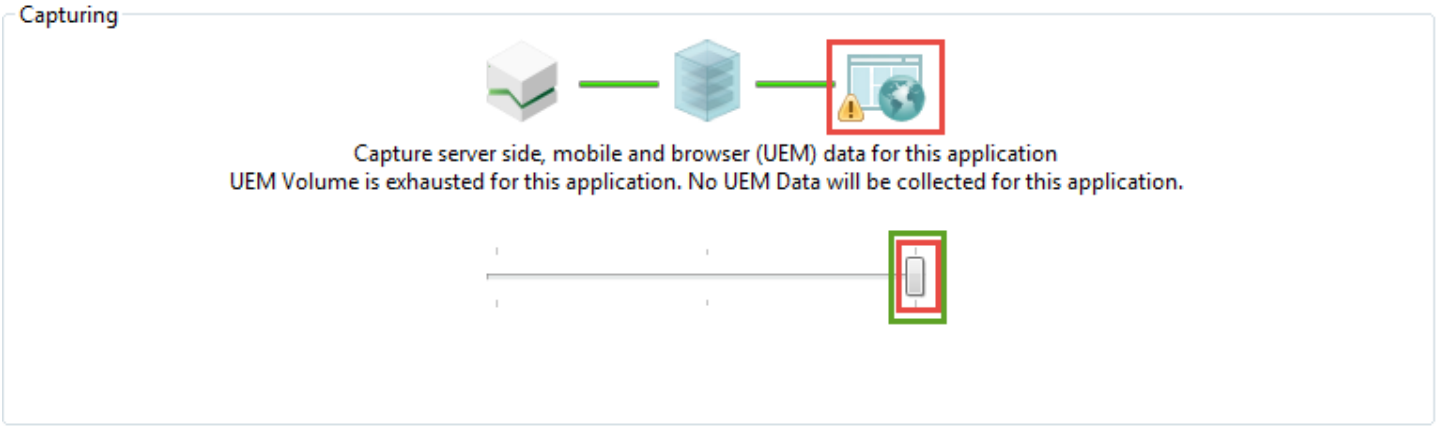


Make sure the slider is in the desired position.

- **Server Side Data Only - No Browser (UEM) Data**: UEM capturing is set to capture only server-side data for the checked application.



- **Server Side Data Only - UEM Volume Exhausted for Application**: UEM data capturing is turned off because there is no more UEM volume available for this application. This means either all remaining volume is reserved for other applications or that a reservation which applies to this application has been exhausted.



Injection check

This check verifies that the JavaScript Agent is injected correctly. Inject the JavaScript Agent into all pages manually or automatically. For automatic injection, a number of options restrict the injection.

Configure JavaScript Agent injection in **System Profile preferences > User Experience**. To configure for all applications, click the **Default application** tab and make your configuration settings Under JavaScript Agent. For Individual applications, Select the application specific tab. See System Profile - User Experience > JavaScript Agent for more information.

The JavaScript Agent is correctly injected if the `src-attribute` of the first script-tag in the HTML response body matches a certain file name pattern.

To Health Check each application, enter a URL. Click **Start Center > Monitoring > Check your UEM configuration** under **UEM Health Check**. If you have a web server master Agent, the connected sub-Agent may be from an earlier version and reports a different configuration.

Errors

The injection check may fail with the following messages:

- **Invalid Test URL**: The specified test URL is not in a valid format.
- **Agent Matches to a Different System Profile**: The provided validation URL does not match the current System Profile. Provide a validation URL which belongs to the profile.
- **Error During Health Check Request**: The injection check requests the test page, but fails. Verify that you are connected to the Internet and that you can connect to the URL.
- **Error Requesting Test Page**: The HTTP request to the specified test page returns with an HTTP status code other than 200 (OK). Check the URL. You may have to specify a proxy or a firewall may block requests.
- **UEM Sensor Inactive**: The affected Agents' User Experience Sensor is set to an inactive state. Open the Agent's Sensor configuration and set the User Experience Sensor's capturing mode to active.
- **Server not Instrumented by Agent**: Verify that the test page URL points to a server with an AppMon Agent installed. If you have an instrumented Java-based application server such as Apache Tomcat, set the agent's User Experience sensor to active.
- **Test Page is Invalid HTML**: To inject the JavaScript Agent, the page must be valid HTML. The Agent tags the start of an HTML syntax violation automatically. The JavaScript Agent injects properly even if the document contains a violation tag. The document may have been assembled by multiple injected servlets and/or filters.
- **No Injection Point Found on Test Page**: No injection point found for JavaScript Agent on this page, likely due to invalid HTML or unexpected end in transmission.
- **Test Page Does Not Have Content Type Header**: No content-type header value is set, so the JavaScript Agent is not injected in the response body.
- **Test Page Has an Invalid MIME Type**: Page does not have one of the two accepted MIME-Types: text/html or application/xhtml+xml. The JavaScript Agent is not injected. Verify that the URL points to an HTML page and not to an image, `swf,` or other non-HTML content.

- **Web or Application Server Agent Does Not Have a JavaScript Agent Available for Injection**: The Agent may not have a connection to the Master Agent or to a Collector. Verify that the respective Agent appears in the **Agent Overview** Dashboard.

- **Test Page is a Special Page**: The specified URL for the test page points to a page that servers use internally, for example “bea_wls_” used by BEA Weblogic, where the JavaScript Agent is not injected. Specify a different test page URL.

- **Test Page is Special**: The test URL belongs to a special web request by the Web Server Agent, such as the AppMon Agent or bandwidth test images. This cannot verify the correct injection of the JavaScript Agent. Specify a different URL.

- **Test page is Not HTML**: The test page is not HTML, based on the file extension (for example `.gif`, `.jpg`, `.png`). Specify a different URL.

- **Manual Injection is Enabled; No Agent Has Been Injected by Developer**: Verify that the JavaScript Agent is included in the HTML page and that it is the first script tag on the page.

- **JavaScript Agent Capturing is Disabled in Application Configuration**: In the applicable System Profile Preferences, select **> User Experience > Application Specific Settings tab** to configure a JavaScript Agent injection for all applications or for each one separately.

- **JavaScript Agent Capturing is Disabled in Sensor configuration**: In the applicable System Profile Preferences, select the desired Agent Group and choose **Sensor Placement** to configure a JavaScript Agent injection.

See Sensor Properties > User Experience for more information.

- **JavaScript Agent Injection is Disabled but JavaScript Agent is Injected**: The JavaScript Agent is turned off for this web or app server, however the JavaScript Agent is still injected. This may be due to the following reasons:
  - The page is cached somewhere. Verify that the HTML pages for this application are not cached, or UEM may not work.
  - The JavaScript Agent was injected manually. Disable manual injection. For more information, see Sensor Properties > User Experience.

- **Injection Disabled Because Dynatrace Header is On the Request**: If the request and response, respectively, has an AppMon header, this is used for a Health Check.

- **Invalid Page Encoding**: There is no injection due to incompatible encoding (must be UTF-8 or UTF-16 compatible).
- **Injection Threshold Reached**: There is no injection because the content length is too long.
- **Chunked Transfer Encoding**: The injection is disabled for pages that are delivered using chunked transfer encoding.
- **Agent Already Injected**: There is no injection because the JavaScript Agent is already injected. This could be due to a (redundant) manual injection attempt.
- **UEM Disabled on Agent**: UEM is disabled in the Agent configuration. The configuration is not communicated correctly between the server and Agent. This is because the Agent configuration (as stored on the server) is checked before the Injection check is executed, and this test did not fail.
- **Server Agent Terminated Unexpectedly**: This may be a programming error. Click here to contact support.
- **JavaScript Agent URL/File Name Malformed**: The URL of the injected JavaScript Agent is malformed. The correct URL / file name of the JavaScript Agent should look like this: `dtagent_ax_6000500005176.js`. This may occur if you inject manually, or delete a JS plugin.
- **JavaScript Agent has wrong version format**: The URL of the injected JavaScript Agent does not have the correct format. A JavaScript agent URL >= 6.5 should look like this: `dtagent_ax_6000500005176.js`. All lower versions should look like: `dtagent630_ax_5176.js`.
- **UEM Capturing Disabled on Agent**: UEM is enabled in the application's UEM settings, but the responding Agent does not retrieve the changed settings. Verify that the communication between the AppMon Server and the Agent, and/or any Collectors works correctly.
- **Servlet-Specific Injection: Servlet not Provided**: The injection point of the User Experience Sensor is set to a specific servlet or filter. This may not exist, or the associated URL pattern is not part of the provided validation URL. Verify that the communication works between the AppMon Server and Agent.

- **Servlet-Specific Injection: Response-Header Missing**: The injection point of the UEM Sensor is set to a custom servlet that is different from the one that responds to the Health Check request. This can occur when the Agent's UEM settings are not current. Verify that the communication between server and Agent works.

- **Web Server: UEM License Expired**: The Web Server Agent is not injected because its license is expired. Please extend or renew your license.

- **Failed for Unknown Reasons**: If AppMon could not detect automatically why the injection did not work, the following reasons may explain why:
  - The test URL does not belong to this application. Make sure that the test URL is correct.
  - The page is cached somewhere such as a proxy or a CDN. Make sure that HTML pages for this application are not cached, else UEM may not work.
  - The UEM Health Check was not able to determine why the injection failed. Click here to contact support.
- **Wrong Version Format**: The injected JavaScript Agent has the wrong version format. AppMon 6.5 and later has an updated new version format. See JavaScript Agent Configuration for more information. This may be a programming error. Click here to contact support.

Warning

The test may fail if the JavaScript Agent is not injected correctly. It may also return a number of warnings when the Agent is correctly injected. These warnings are a sign of a condition that may prevent AppMon from tracking visitors.

- **Server Agent Does Not Support Health Check**: The Agent does not support the Health Check feature. This may occur if an earlier version of the Agent is installed. The Health Check can execute, but it may not find all the errors in the configuration. Update the server with the latest version of the Agent.

- **Injection Percentage Restrictions are Enabled**: The injection of the JavaScript Agent is limited by percentage restrictions. The Agent is only injected in the specified percentage of requests.

- **Browser Injection Restrictions are Enabled**: Injection restrictions based on browser type and version are enabled so certain browsers are not tracked.

- **Do Not Track Header is Enabled**: Most modern browsers support this functionality.

- **Injection Restrictions Based on IP Ranges are Enabled**: Only users with IP addresses that match the specified rules are tracked. The AppMon Client IP is excluded when you use a normal browser.

- **Manual Injection is Enabled**: The application developers should manually inject the JavaScript Agent in all pages to be captured.

- **Multiple JavaScript Agents Found**: If more than one JavaScript Agent is found in the validation page, the requested validation page is probably injected manually, by the application developers, and automatically, by the web server or application server.

- **AgentId Not On Response Header**: The connected Agent(s) ID is not on the Health Check's response header. This occurs if the Agent is not updated automatically to version 5.5 or later.

- **Discard An Injection Due to an Unexpected Flush() Call**: Some Java applications and frameworks use flush() to send portions of an HTML page to the browser before its entire content is generated. Rarely, this makes it necessary to discard an injection in the flushed portion of the HTML code. In this case, the JavaScript Agent gets injected at a later, potentially sub-optimal position.

Agent version check

Use this to check the Agent version in the Health Check's web request.

- **Agent(s) Not Up-To-Date**: The Agents listed in the UEM Health Check run an earlier version than the AppMon Server and were not updated properly by the bootstrap Agent. Verify that the communication works between the AppMon Server and Agents.

- **The Health Check's HTTP Request Returned an Invalid Response From the Connected Agents**: The following are some possible causes.
  - Missing response header which should contain the IDs of the connected Agents. This can happen when one of the connected Agents runs on a version earlier than 5.5.
  - Invalid response header modified by the connected Agents.
  - Agent that is unknown to the AppMon Server. Agent is connected to a different AppMon Server or none at all.

- **Master Agent Unknown or Not Found**: One of the connected sub-Agent's master Agent is either not available or unknown to the server.

- **Versions For Sub-Agent and Master-Agent Do Not Match**: The master Agent (e.g. Web Server Agent) and one of the connected sub-Agents run on different versions. The following are some possible causes.
  - After installing an AppMon Update, the Web Server Agent or the master Agent are not restarted or are unable to retrieve the latest Agent version from the AppMon Server.
  - If you use IIS under Windows Vista, or later, the installation directory `C:\Program Files` may cause a shared memory problem.

JavaScript Agent delivery

This check verifies that a valid JavaScript Agent is injected into the requested web page.

- **The Agent is not delivered**: The Agent URL is not reachable. The may be due to one of the following:
  - The Agent path is not handled by a web server with an installed Agent. A typical case is a Java server with an application that runs in a sub-path such as `/app/` but no root application is available. The JavaScript Agent fails when requested from the root path.
  - The Agent path is not handled by the servlet engine itself, but by a custom part of the app server. (WebSphere sometimes does that for static content).
  - The Agent path is blocked by a firewall.

- **Agent Empty**: The Agent URL is reachable but does not return content. Some firewalls or proxy servers return empty content, instead of a 404 HTTP error. Configure your firewall or proxy server to correctly serve the JS Agent URL.

- **Agent is Gzipped But was Requested Uncompressed**: This occurs when a web server compresses the content for all requests, although the content was not requested with the *Accept-encoding: gzip* header enabled. This may be problematic for clients who do not support gzipped content. This can be caused by a servlet filter, an Apache module, or an ISAPI filter that does not respect the *Accept-encoding* header correctly.

- **Invalid Agent Content**: The delivered Agent contains invalid JavaScript content which may trigger JS errors on the injected page and, therefore, is not a valid JS Agent. This can be caused by a Web server returning a default page instead of a HTTP 404 error when a page is not found. Typically, this is because the requests to the JavaScript Agent are not allowed by a firewall, proxy, or the web server.

- **Gzipping is Disabled**: Sending JavaScript Agent in a compressed format is deactivated in the UEM Sensor configuration. This leads to more overhead when downloading the JavaScript Agent, but this setting may be necessary to avoid double gzipping of the Agent.

- **Agent not Gzipped Although Requested Compressed**: This occurs when the *Accept-encoding* header is not passed to the AppMon Agent. It may not be a problem. It may indicate that gzipping does not work for this application. This may impact performance.

- **Agent is not Gzipped Although Gzip Header is Set**: The Agent is requested as gzipped and the respective response header (*content-disposition)* is set, but the response does not contain gzipped content. There may be a problem in the web server configuration. This occurs when a web server compresses all responses and disregards the *Content-disposition response* header set to *gzip*.

Deactivate sending the JavaScript Agent gzipped in the UEM Sensor configuration, as a workaround.

- **JavaScript Agent is Gzipped Twice**: The delivered JavaScript Agent is gzipped twice. Contact your system administrator or support.

AppMon monitor request check

This check verifies if the monitor request path is valid. See JavaScript Agent - Configuration for more information.

In the applicable System Profile Preferences, select **> User Experience > Application Specific Settings tab** to configure a JavaScript Agent injection for all applications or for each one separately.

A request is valid when **both** the *http status code* and the *response body* match specific values.

- The HTTP status must always equal 200 (OK).
- The response body must match the following pattern with valid status values OK, FL and valid server values Java, Web server:

```
status(server)
```

The following is an example of a valid request:

```
HTTP Status: 200
Response Body: OK(Java)
```

The AppMon Monitor request check may fail with the following messages:

- **Response Code Invalid**: The HTTP request to the Monitor URL returned with an HTTP status code other than 200 (OK). Verify the specified Monitor URL. There may be a firewall that blocks requests to that URL.

- **Response Body Invalid**: The HTTP request to the specified Monitor URL was successful but returned with an invalid response body. This may be due to an invalid Monitor URL. The web server redirects to a default error page and has an HTTP status 200 (OK).

- **Request Body Invalid**: This error occurs if the Agent cannot read the parameters of the monitor signal request. This may cause a problem with UEM data, such as missing User Action PurePaths. Your firewall may block POST requests, or truncate or encode them. If you're not able to change these settings, select **Split large signals** inside your application's UEM settings. As a result, the requests are sent with the GET method.

- **Cookie Problem**: During the Health Check execution, there is a problem with cookies when you access the Monitor URL. The following are a list of causes:
  - The required AppMon cookie *dtCookie* cannot be created. This problem is discovered when the *Set-Cookie* header is not found on the response headers. Verify the cookie policies of your application server or webserver.
  - The found session-cookie is invalid. Update the Agents versions.
  - The cookie's domain is set to an invalid value.

- **Unsafe Cookie Content**: The injected application server may have a problem with unsafe cookie content; for example when a cookie contains characters like '<', '>' or '$'. The JavaScript agent is using a cookie named *dtSa*, which might be available if an action with a redirect happens. It stores information about the previous page, such as the encoded URL. Depending on the characters in that URL, the cookie might contain the problematic characters listed above. If the monitored application does not include switches between different subdomains, the agent can be configured to use SessionStorage instead of a cookie. For AppMon 6.5.10 and later, the **Disable subdomain source action support** setting is available in debug mode. Contact support for information on selecting this setting.

Known Problems

Using GZIP compression on server

This occurs when custom or multiple GZIP-filters are installed on the application server. Install one GZIP-filter only or use an application URL which does not use a GZIP-filter.

Configure geographic location

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Configuration - world map

Related pages

User Action timings

Third-party content analysis

User Action timings

Monitor streaming media activity

Perceived render time

Overview

AppMon captures the client IP address. For CDNs and load balancers, AppMon watches for the appropriate HTTP headers and converts this IP address to a geo location with a built in database. The IP address is determined as follows:

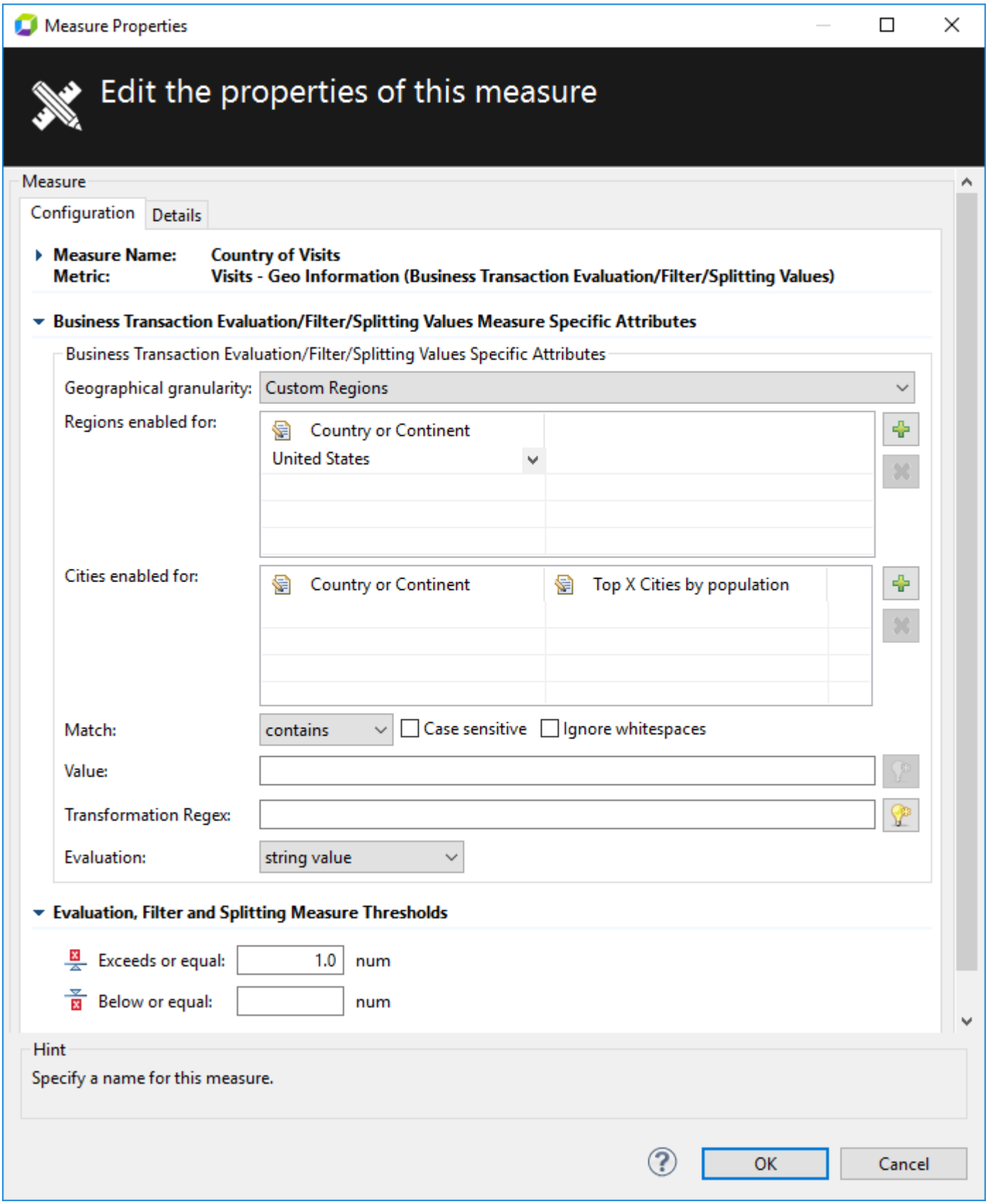
- http-requestheader field “rproxy_remote_address”
- http-requestheader field “True-Client-IP”
- http-requestheader field “X-Client-Ip”
- http-requestheader field “X-Forwarded-For”
- http-requestheader field “X-Http-Client-Ip”
- ip-address of requester

Locations may be assigned to internal IP addresses in the geographical locations section of the server settings. See Geographical Locations - Mapping of IP Addresses for more information. The geographical location is displayed in several dashlets. See Visits Dashlet, User Actions Dashlet, User Action PurePaths Dashlet and World Map Dashlet for more information.

Configuration - world map

The World Map data is based on business transactions Measures, whose grouping is by geographical location. To customize the desired level of detail within the World Map, select **Edit map granularity** from the context menu. The **World Map** dashlet uses out of the box measures, grouped by the **Country of Visit** Measure.

Measure Configuration Tab



Configure the desired geographical granularity by the **Measure Properties** dialog box shown above. The following are available options for geographical granularity.

- **Continent**: Only the continent name is returned.
- **Country**: The result of the measure is the country name.
- **Custom Regions**: The result is either a country name or `<country name> - <region name>` for all countries or continents listed in the Regions enabled for table.

Country and region names are taken from the internal geographical locations database using the ISO 3166-2 standard for the US and Canada, and the FIPS 10-4 standard for the rest of the world. The country is encoded using the ISO 3166-1 Alpha-2 country codes.

Configure streaming media monitoring

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

User Action timings

Third-party content analysis

User Action timings

W3C resource timing metrics

Configure geographic location

Watch and learn

Monitoring streaming media activity

External links

Audio support by browser

Video support by browser

Overview

You can configure the AppMon JavaScript Agent to collect information such as play time and rebuffer count. It also collects the streaming media rebuffer duration that is embedded in HTML pages and uses HTML5's audio and video tags. Additionally, if streaming media is embedded Flash or Silverlight but not HTML5 tags, it is possible to send information about media streams to the AppMon Server using the JavaScript ADK.

Enable streaming media for System Profile

The streaming media functionality is disabled for a System Profile by default. To enable it via the Start Center

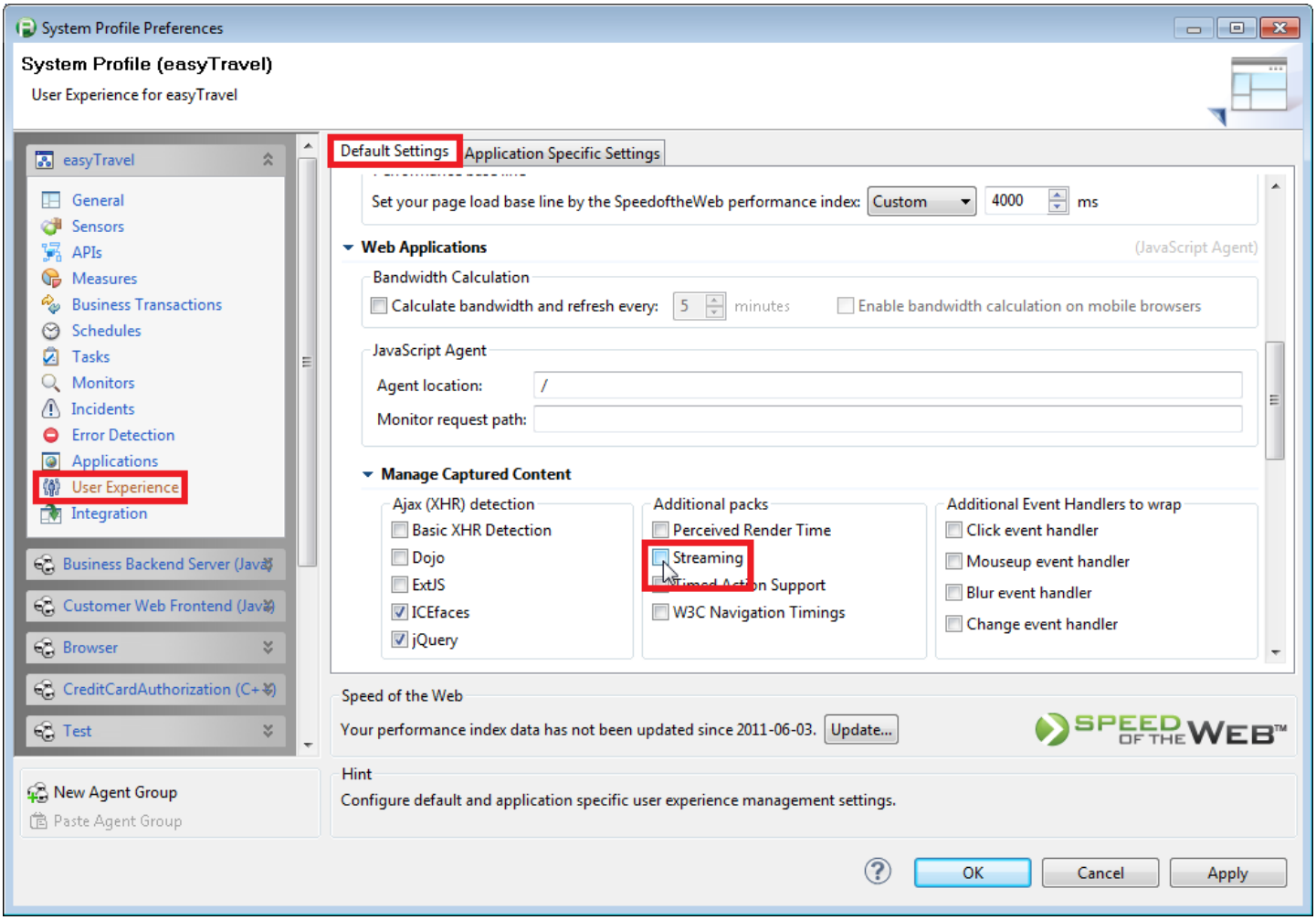


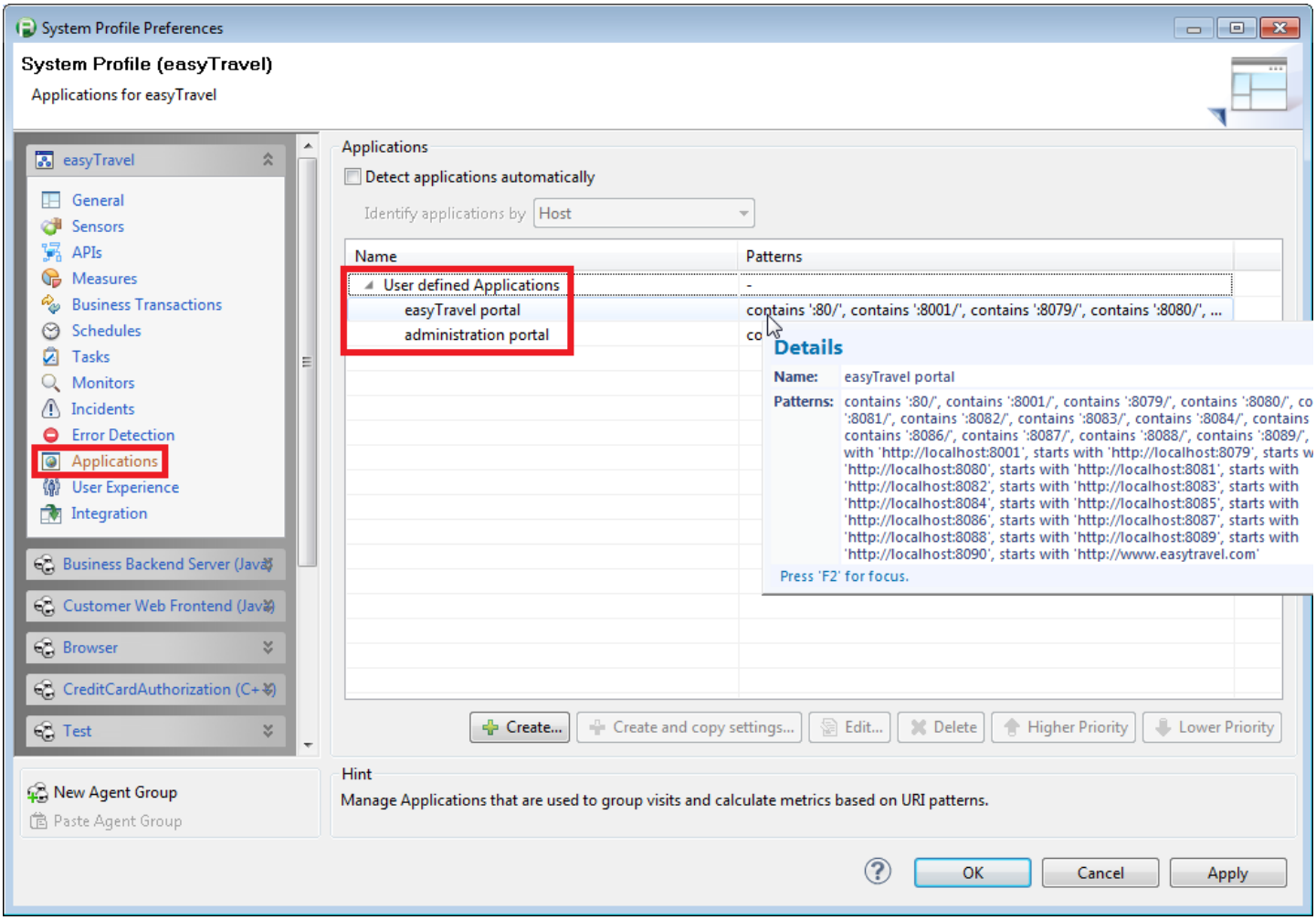
, click **Start Center > User Experience > Enable Streaming Media**. This creates all the necessary metrics (measures and business transactions) in the System Profile, and shows the link to the **Streaming Media** dashboard in the Start Center.

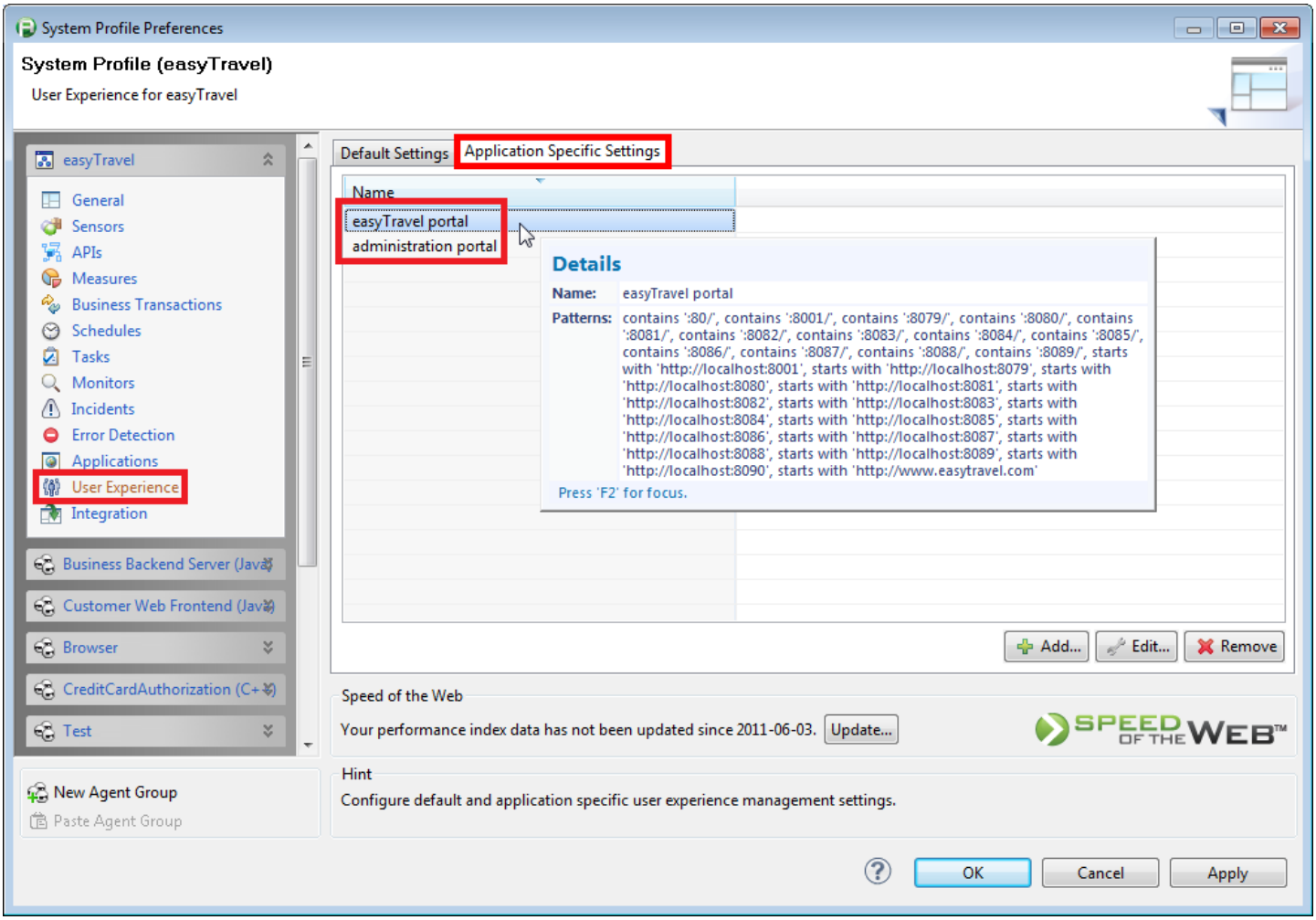
Enable streaming JavaScript plugin

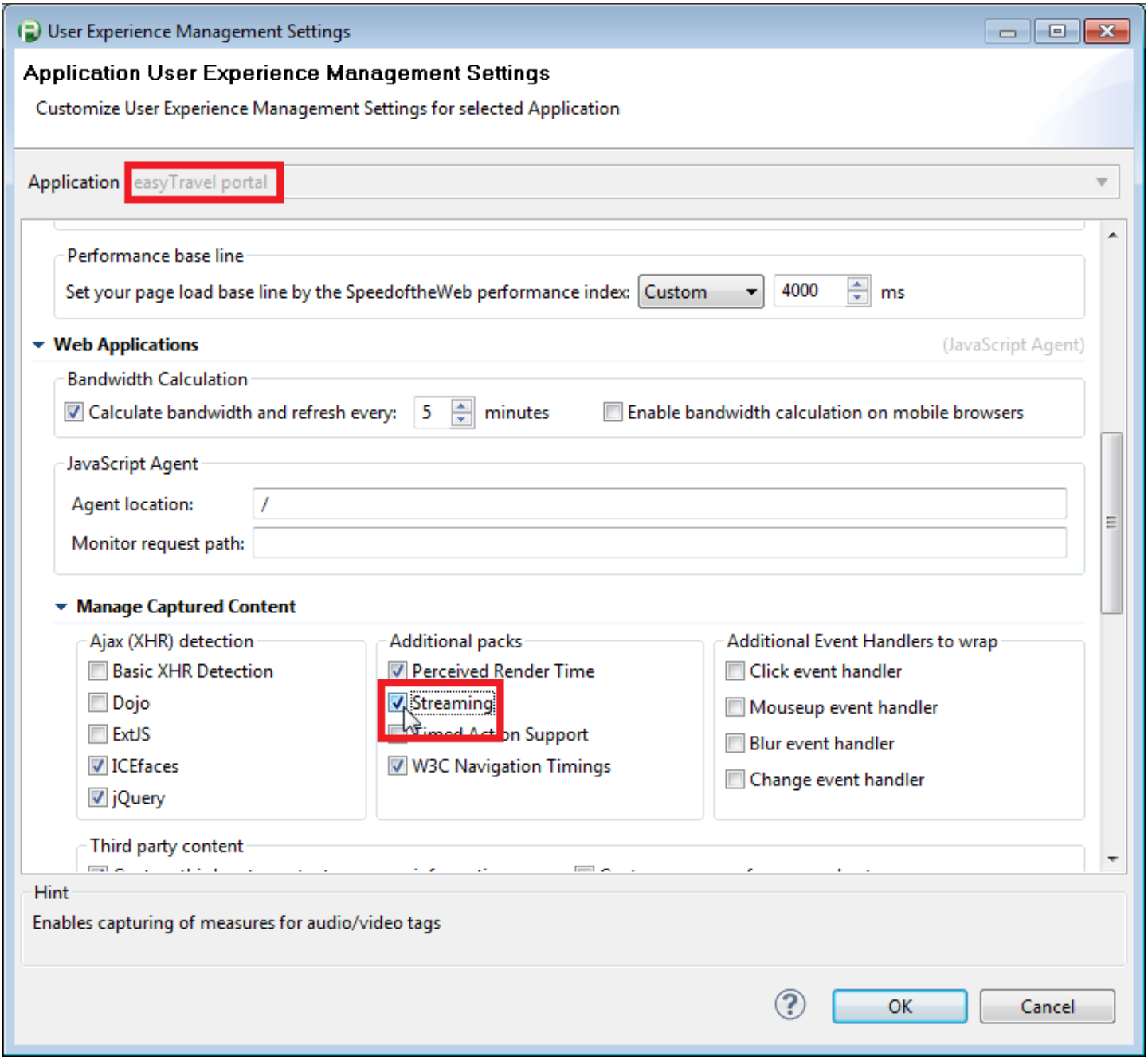
To collect streaming media information from HTML5 media tags, click **System Profile > User Experience > Default Application tab > Web Applications** section and select the **Streaming** check box. This delivers the JavaScript Agent with the streaming media plugin.

In the easyTravel example, there are two applications configured in the **Application Specific Settings** tab. This shows how to selectively capture streaming media data. Select the **Streaming** check box. This is finer granularity to spend the UEM volume wisely.


System Profile Preferences
System Profile (easyTravel)
User Experience for easyTravel
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Integration
Business Backend Server (Java)
Customer Web Frontend (Java)
Browser
CreditCardAuthorization (C++)
Test
New Agent Group
Paste Agent Group
Default Settings
Application Specific Settings
Set your page load base line by the SpeedoftheWeb performance index:
Custom
4000
ms
Web Applications
(JavaScript Agent)
Bandwidth Calculation
Calculate bandwidth and refresh every:
5
minutes
Enable bandwidth calculation on mobile browsers
JavaScript Agent
Agent location:
/
Monitor request path:
Manage Captured Content
Ajax (XHR) detection
Basic XHR Detection
Dojo
ExtJS
ICEfaces
jQuery
Additional packs
Perceived Render Time
Streaming
W3C Navigation Timings
Additional Event Handlers to wrap
Click event handler
Mouseup event handler
Blur event handler
Change event handler
Speed of the Web
Your performance index data has not been updated since 2011-06-03.
Update...
SPEED OF THE WEB
Hint
Configure default and application specific user experience management settings.
OK
Cancel
Apply


System Profile Preferences
System Profile (easyTravel)
Applications for easyTravel
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Integration
Business Backend Server (Java)
Customer Web Frontend (Java)
Browser
CreditCardAuthorization (C++)
Test
New Agent Group
Paste Agent Group
Applications
Detect applications automatically
Identify applications by Host
Name
Patterns
User defined Applications
easyTravel portal
administration portal
contains ':80/', contains ':8001/', contains ':8079/', contains ':8080/', ...
Details
Name: easyTravel portal
Patterns: contains ':80/', contains ':8001/', contains ':8079/', contains ':8080/', co
':8081/', contains ':8082/', contains ':8083/', contains ':8084/', contains
contains ':8086/', contains ':8087/', contains ':8088/', contains ':8089/',
with 'http://localhost:8001', starts with 'http://localhost:8079', starts w
'http://localhost:8080', starts with 'http://localhost:8081', starts with
'http://localhost:8082', starts with 'http://localhost:8083', starts with
'http://localhost:8084', starts with 'http://localhost:8085', starts with
'http://localhost:8086', starts with 'http://localhost:8087', starts with
'http://localhost:8088', starts with 'http://localhost:8089', starts with
'http://localhost:8090', starts with 'http://www.easytravel.com'
Press 'F2' for focus.
Create...
Create and copy settings...
Edit...
Delete
Higher Priority
Lower Priority
Hint
Manage Applications that are used to group visits and calculate metrics based on URI patterns.
OK
Cancel
Apply


System Profile Preferences
System Profile (easyTravel)
User Experience for easyTravel
easyTravel
General
Sensors
APIs
Measures
Business Transactions
Schedules
Tasks
Monitors
Incidents
Error Detection
Applications
User Experience
Integration
Business Backend Server (Java)
Customer Web Frontend (Java)
Browser
CreditCardAuthorization (C++)
Test
New Agent Group
Paste Agent Group
Default Settings
Application Specific Settings
Name
easyTravel portal
administration portal
Details
Name: easyTravel portal
Patterns: contains ':80/', contains ':8001/', contains ':8079/', contains ':8080/', contains ':8081/', contains ':8082/', contains ':8083/', contains ':8084/', contains ':8085/', contains ':8086/', contains ':8087/', contains ':8088/', contains ':8089/', starts with 'http://localhost:8001', starts with 'http://localhost:8079', starts with 'http://localhost:8080', starts with 'http://localhost:8081', starts with 'http://localhost:8082', starts with 'http://localhost:8083', starts with 'http://localhost:8084', starts with 'http://localhost:8085', starts with 'http://localhost:8086', starts with 'http://localhost:8087', starts with 'http://localhost:8088', starts with 'http://localhost:8089', starts with 'http://localhost:8090', starts with 'http://www.easytravel.com'
Press 'F2' for focus.
Add...
Edit...
Remove
Speed of the Web
Your performance index data has not been updated since 2011-06-03.
Update...
SPEED OF THE WEB
Hint
Configure default and application specific user experience management settings.
OK
Cancel
Apply



See System Profile - User Experience for more information.

If the streaming media items are not embedded in the page using HTML5 audio/video tags, the plugin cannot collect data for the media streams. In this case, the JavaScript ADK is used to collect information. You don't have to enable the **Streaming JavaScript Agent** plugin if you use the JavaScript ADK.

Supported browsers

The following is a list of supported browsers:

- IE 9+
- Chrome 18+
- Firefox 4+
- Safari 5+
- Opera 11+

See http://caniuse.com/audio and http://caniuse.com/video for more information.

Monitor user experience

World Map Dashlet
Monitor User Visits by Geographic Region
Monitor User Actions by Geographic Region
Use Top Findings to Analyze User Visits
Monitor Top Landing and Exit Pages
Monitor User Satisfaction
Monitor Streaming Media Activity
Trace User Activity Through All Channels
Find Specific User Visits
Web Performance Optimization for UEM

World Map dashlet

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Navigate the map

Customize the map

Drill down to dashlets

Geographical granularity

Related pages

User Experience dashlets

Client Errors dashlet

User Action PurePaths dashlet

User Actions dashlet

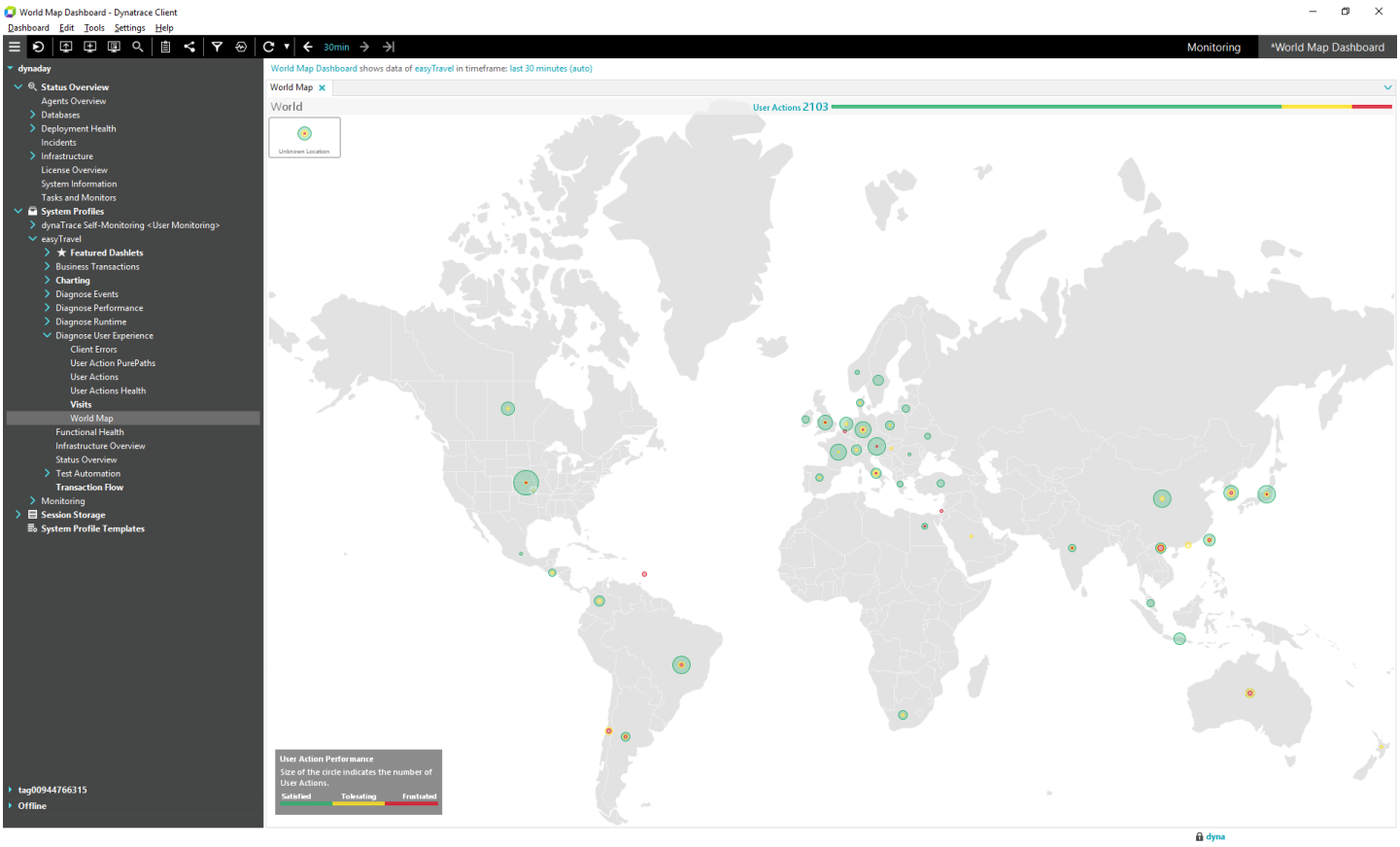
User Actions Health dashlet

Visits dashlet

Geographical Location Configuration

Overview

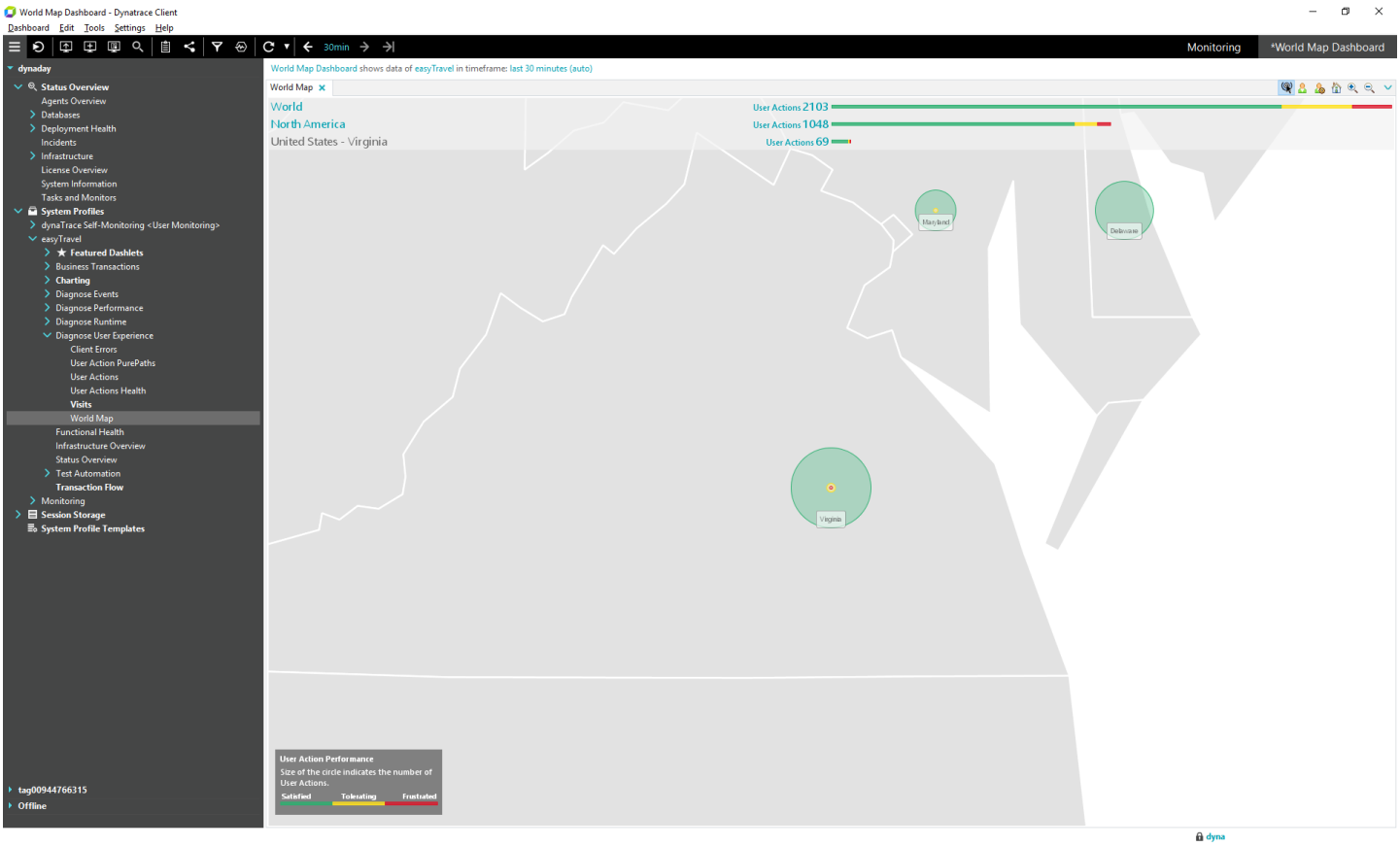
Use the **World Map** dashlet to visualize visits or user actions within a geographical representation of the world. Drill down into other dashlets and navigate and zoom to different locations.



World Map dashlet

Navigate the map

The dashlet toolbar lets you zoom in and out and back to the world view. You can also left-click to zoom and to drag the map. Hover over the map to highlight six clickable continents (Africa, Asia, Europe, North America, Oceania and South America). Click each to zoom.

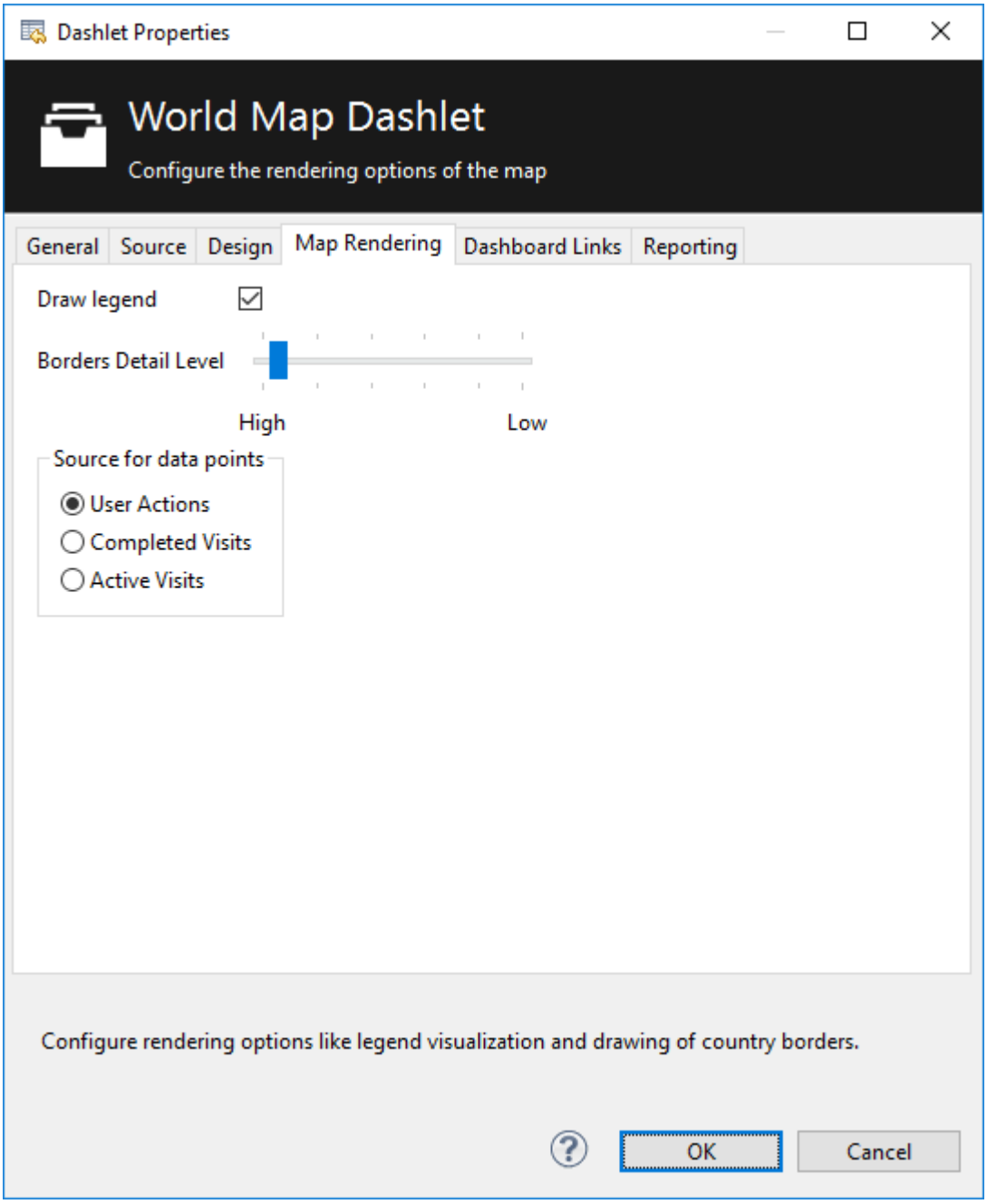


World Map Zoom Options

Click the map headers and a breadcrumb trail appears. Click on **World** and **North America** in the above image. The map zooms to the region. Click the middle mouse button to return to the whole world view.

Customize the map

Right-click in the dashlet and select **Dashlet Properties** to adjust the map rendering properties.



World Map Properties

You can switch off the legend in the lower left corner and adjust the borders zoom level. Switch the data source for the map's data points from **User Action** to **Completed Visits** or **Active Visits** in this dialog box. You can also use **Toggle Visits** in the toolbar.

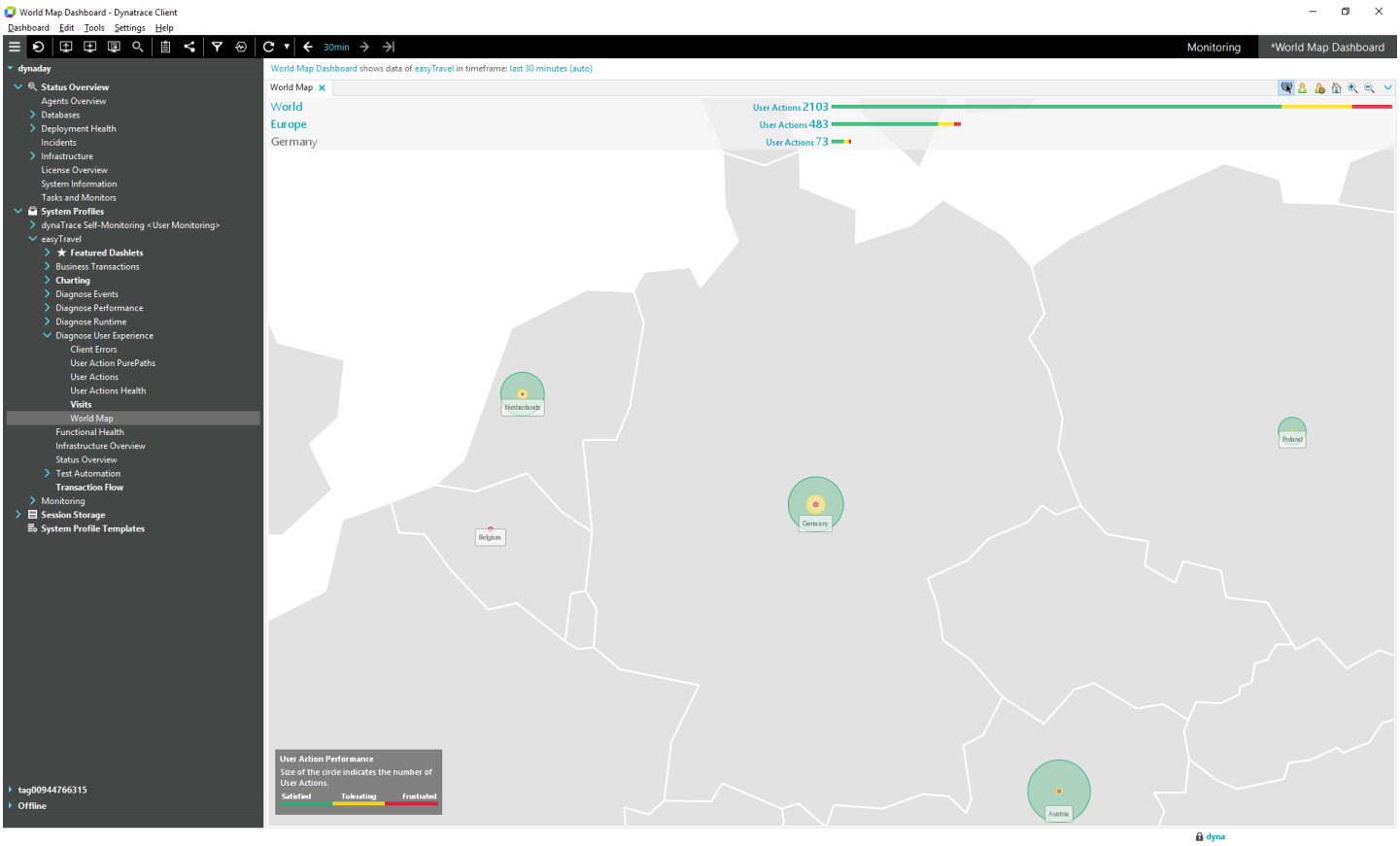
Drill down to dashlets

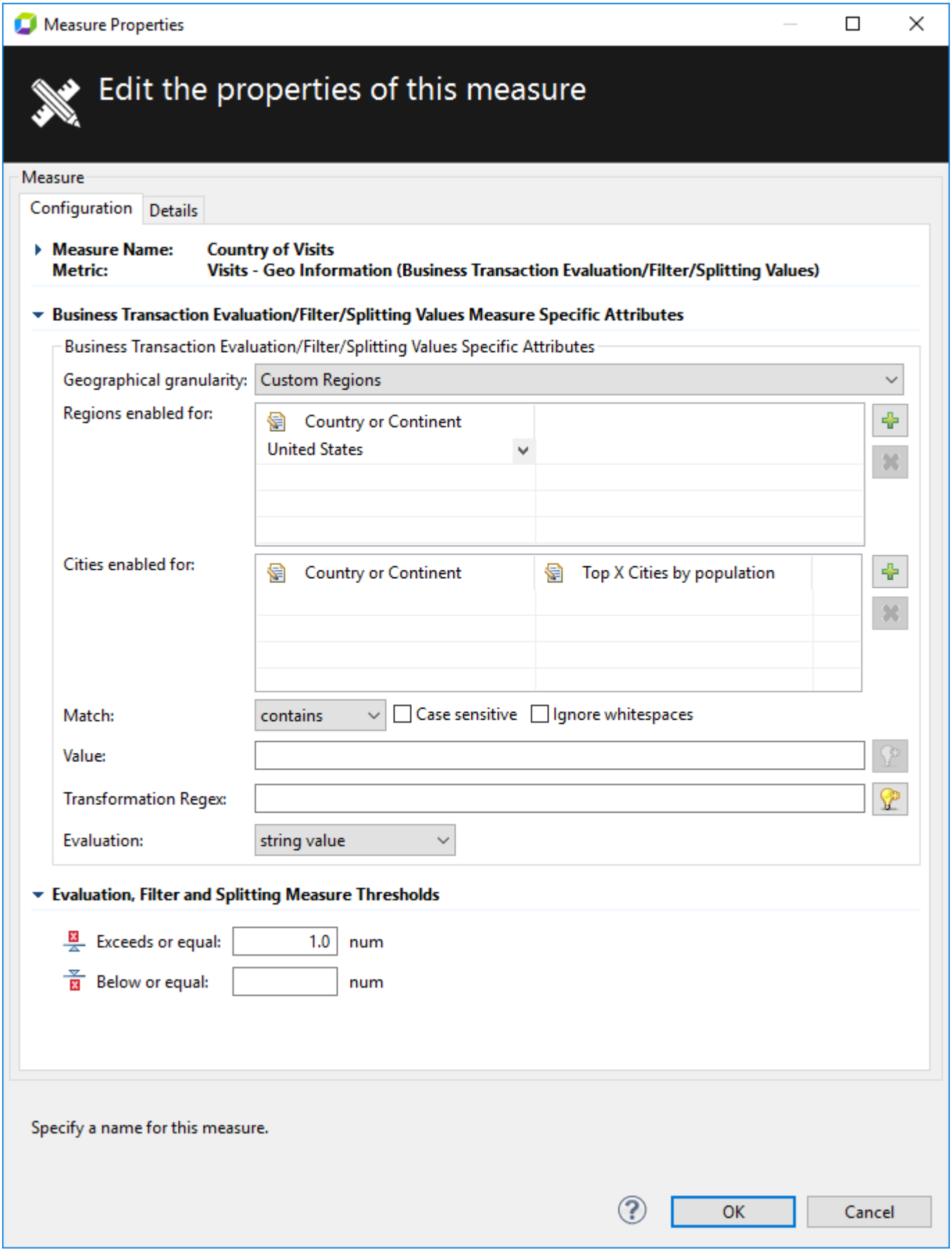
Right-click data points in the **World Map** dashlet to drill down to the **User Actions**, **Visits** dashlets. You can also click on the header bars to drill down to statistics for a region. Select **User Actions**, **Completed Visits**, or **Active Visits** numbers in the headers to drill down to the **User Actions**, **Completed Visits**, or **Active Visits** dashlets.

If the drill-down context menu is empty, you cannot drill down because there is no PurePath data available for the data point.

Geographical granularity

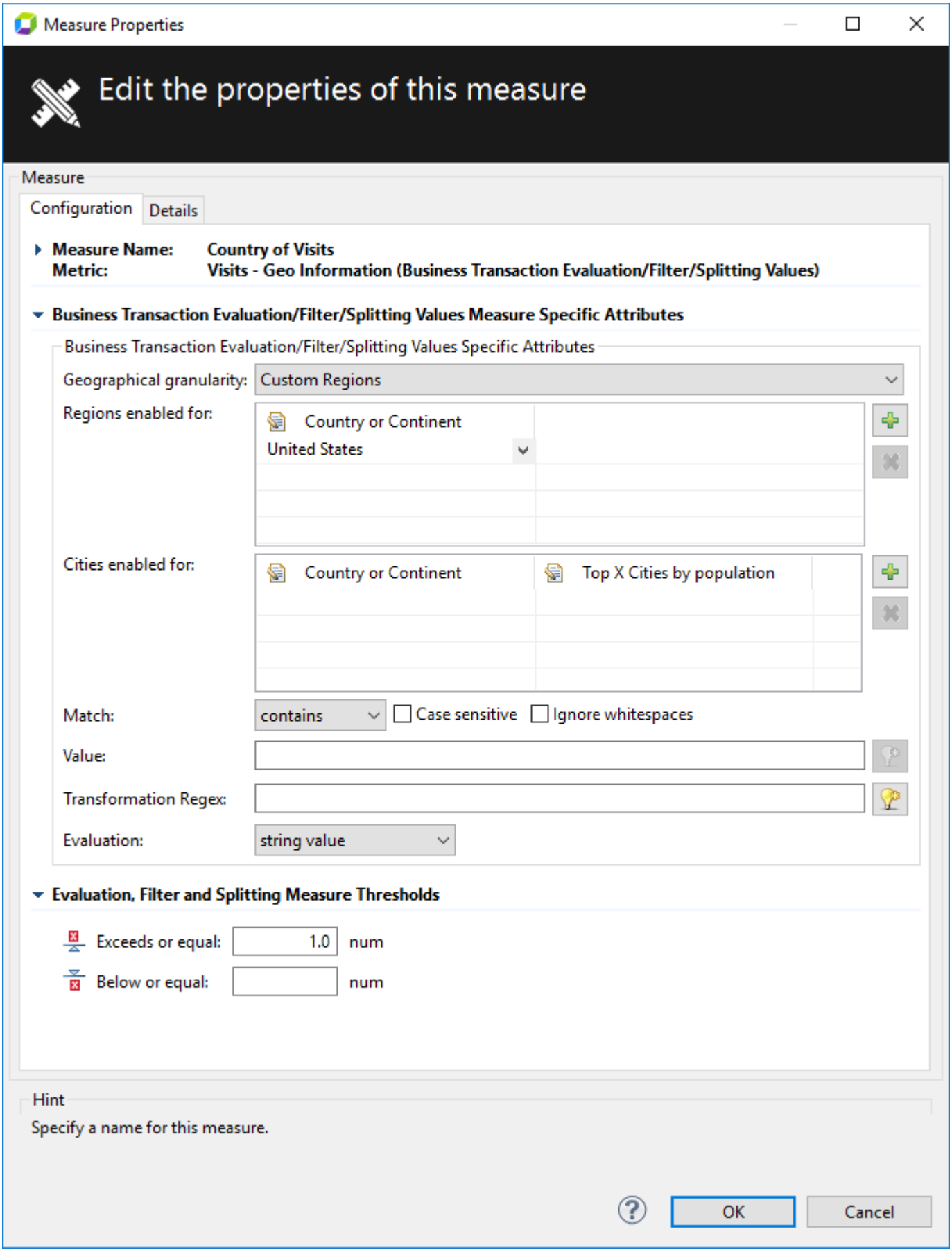
The default setting for geographical granularity for the World Map dashlet is set to region resolution for the USA and country resolution for the rest of the world. To edit this setting, click Edit map granularity in the context menu. A configuration dialog box appears. This dialog box configures the Measure that provides the country data for the dashlet. See Geographical Location Configuration for more information.

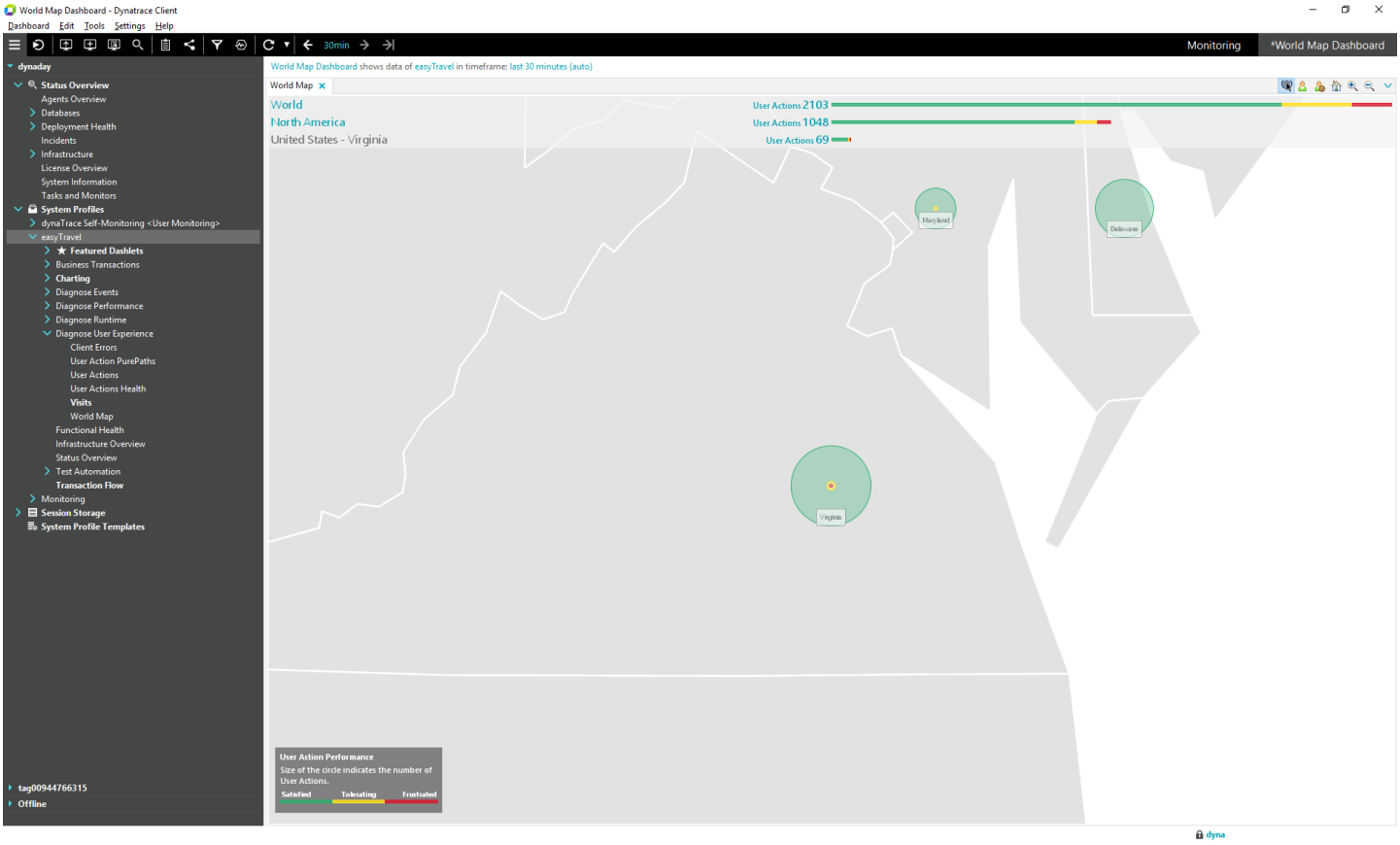



Measure Properties
Edit the properties of this measure
Measure
Configuration
Details
Measure Name: Country of Visits
Metric: Visits - Geo Information (Business Transaction Evaluation/Filter/Splitting Values)
Business Transaction Evaluation/Filter/Splitting Values Measure Specific Attributes
Business Transaction Evaluation/Filter/Splitting Values Specific Attributes
Geographical granularity: Custom Regions
Regions enabled for:
Country or Continent
United States
Cities enabled for:
Country or Continent
Top X Cities by population
Match: contains
Case sensitive
Ignore whitespaces
Value:
Transformation Regex:
Evaluation: string value
Evaluation, Filter and Splitting Measure Thresholds
Exceeds or equal: 1.0 num
Below or equal: num
Specify a name for this measure.
OK
Cancel

Top X cities

You can show the top X cities for certain regions of interest within the world map. Click **Edit map granularity** to fill in the regions of interest in the map. Additionally, the number to the right of the region restricts the number of cities that appear. The maximum number per region is 500.


Measure Properties
Edit the properties of this measure
Measure
Configuration
Details
Measure Name: Country of Visits
Metric: Visits - Geo Information (Business Transaction Evaluation/Filter/Splitting Values)
Business Transaction Evaluation/Filter/Splitting Values Measure Specific Attributes
Business Transaction Evaluation/Filter/Splitting Values Specific Attributes
Geographical granularity: Custom Regions
Regions enabled for:
Country or Continent
United States
Cities enabled for:
Country or Continent
Top X Cities by population
Match: contains
Case sensitive
Ignore whitespaces
Value:
Transformation Regex:
Evaluation: string value
Evaluation, Filter and Splitting Measure Thresholds
Exceeds or equal: 1.0 num
Below or equal: num
Hint
Specify a name for this measure.
OK
Cancel


World Map Dashboard - Dynatrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*World Map Dashboard
dynaday
Status Overview
Agents Overview
Databases
Deployment Health
Incidents
Infrastructure
License Overview
System Information
Tasks and Monitors
System Profiles
dynaTrace Self-Monitoring <User Monitoring>
easyTravel
Featured Dashlets
Business Transactions
Charting
Diagnose Events
Diagnose Performance
Diagnose Runtime
Diagnose User Experience
Client Errors
User Action PurePaths
User Actions
User Actions Health
Visits
World Map
Functional Health
Infrastructure Overview
Status Overview
Test Automation
Transaction Flow
Monitoring
Session Storage
System Profile Templates
tag00944766315
Offline
World Map Dashboard shows data of easyTravel in timeframe: last 30 minutes (auto)
World Map
World
North America
United States - Virginia
User Actions 2103
User Actions 1048
User Actions 69
Maryland
Delaware
Virginia
User Action Performance
Size of the circle indicates the number of User Actions.
Satisfied
Tolerating
Frustrated
dyna

Monitor user visits by geographic region

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

User experience management

Defining custom locations in the world map

World Map dashlet

Monitor User Actions by geographic region

Use top findings to analyze user visits

Monitor top landing and exit pages

Monitor user satisfaction

Trace user activity through all channels

Find specific user visits

Overview

Use the World Map to quickly visualize in real-time the user experience of your system at the world, region, and country level (for countries with large populations). It also displays key global metrics for a succinct understanding of how your users see application performance from their end. In AppMon Web, data includes:

- User experience index (Apdex rating)
- Average response time
- Average number of user actions per minute

- Failure rate of all actions

Access the World map using any of the following:

- AppMon Web: User Experience tile in the Business Analytics dashboard
- AppMon Client or AppMon Webstart Client: You can access the World Map any of the following ways:
  - Add a World Map Dashlet to any dashboard.
  - In the Start Center, select **User Experience > User Experience World Map** and click the User Experience World Map link.
  - In the Start Center, select **User Experience > User Experience Key Metrics** and click the UEM Key Metrics Dashboard link. The World Map is part of the key analytic metrics shown in this dashboard.
  - For the desired System Profile in the cockpit, select **Diagnose Performance > Diagnose User Experience > World Map**.
  - From the application's transaction flow page, click End User Experience to see Visit Count and related key visit metrics, and visits by client type in the End User Experience dashboard.

See World Map Dashlet for more information on using the World map in the AppMon Client or AppMon Webstart Client.

What's the User Experience Index?

The User Experience Index used in AppMon Web indicates how performance is perceived by an application's users, and categorizes visits into satisfying, tolerating, and frustrating visits. The index is based on the the following factors:

- **Apdex rating**, which is a universal standard for measuring the satisfaction of user action performance.
- **Errors or failures**, such as JavaScript errors or pages not found. A page that loads quickly may still have an error.
- **Entire user journey**. The last user action performed in the entire user journey is weighted more heavily.
- **User environment**. Poor bandwith is considered in the index calculation.

Users have a **satisfying** experience if:

- No action failed
- More than 50% of all actions were satisfying

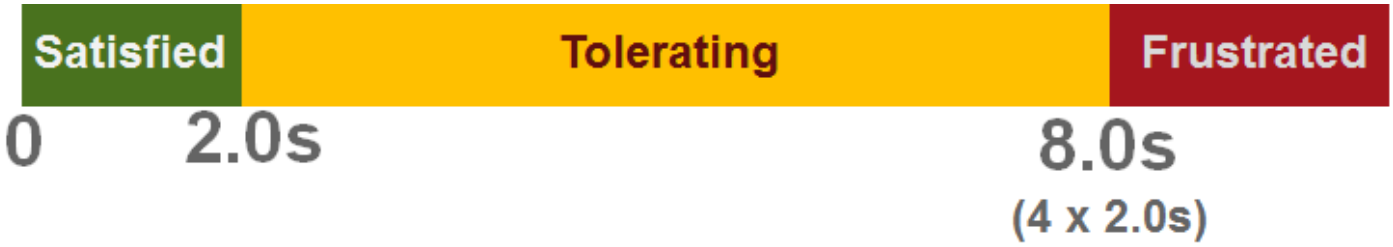
Users have a**frustrating** experience if:

- Their last action failed (“The Web site does not work - I’m leaving!”)
- Their last action was frustrating (“The Web site is too slow - I’m leaving!”)
- More than 50% of all actions were frustrating

Users have a **tolerating** experience if their experience is neither frustrating or satisfying, which means that:

- Their last action was not frustrating or failed
- Less than 50% of all actions were satisfying
- More than 50% of all actions were at least tolerating

For example, let’s assume you have an application-wide satisfied-threshold called page load baseline. The threshold for page load baseline is defined as two seconds. User actions are mapped to the following zones based on their response time:



If there are 10 satisfied actions, five tolerating actions and two frustrated actions, the result is a User Experience Index of (10 + 5 / 2) / 17 = 0.73.

Analyze user experience details in AppMon Web

Click anywhere in the User Experience tile on the Business Analytics default dashboard to open the User Experience World Map page.

The banner at the top of the page displays real time data for the total number of visits, average response time, third party content average load time per page, as well as percentage rates for failures, crashes, bounces, and conversions. This data always reflects the configured data scope and is contextual based on viewing the world or a specific region, country, or state.

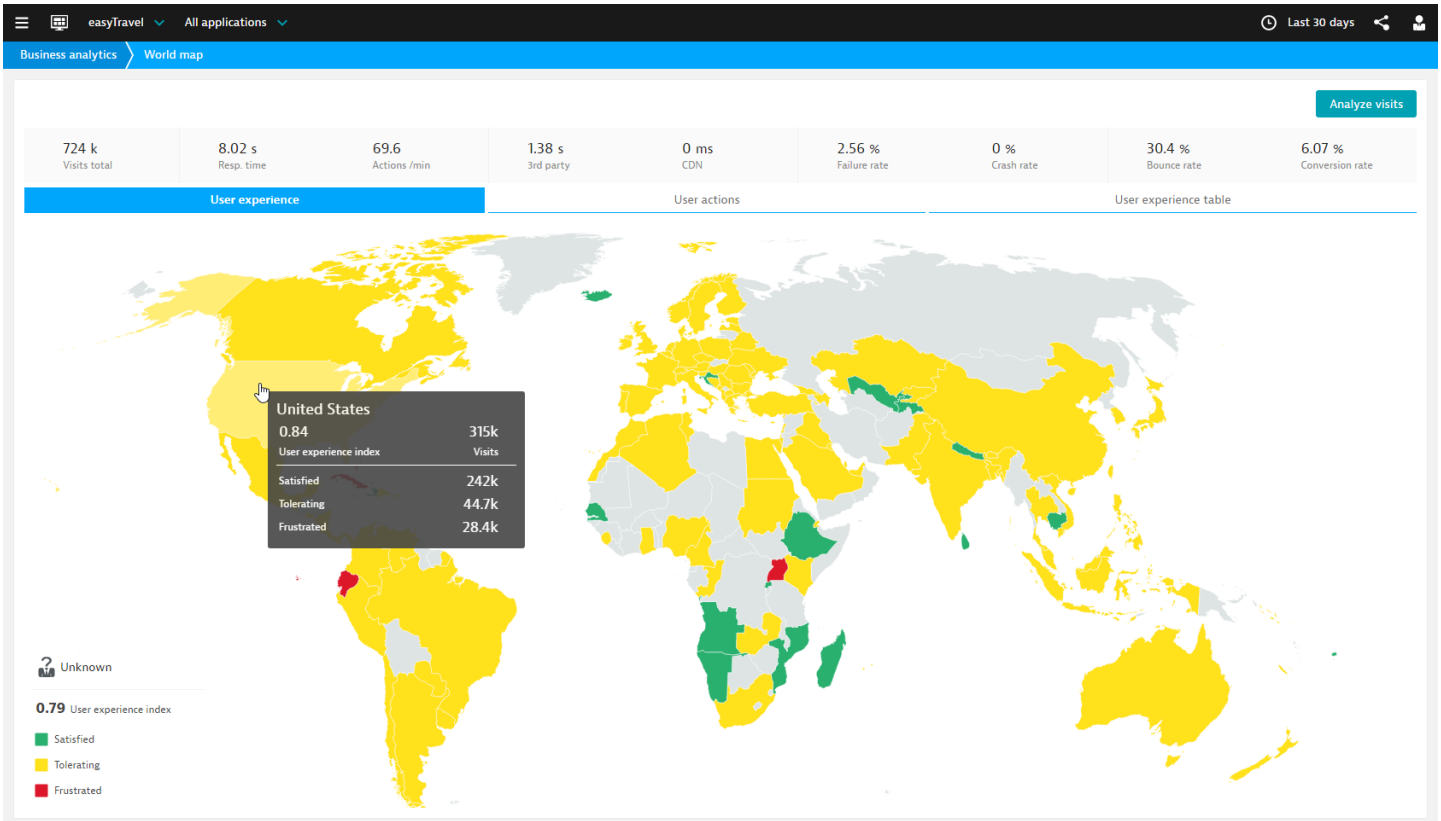
This page includes the **User experience**, **User actions**, and **User experience table** tabs. Each tab provides different data views for displaying the following key metrics that are aggregated at the country and state level.

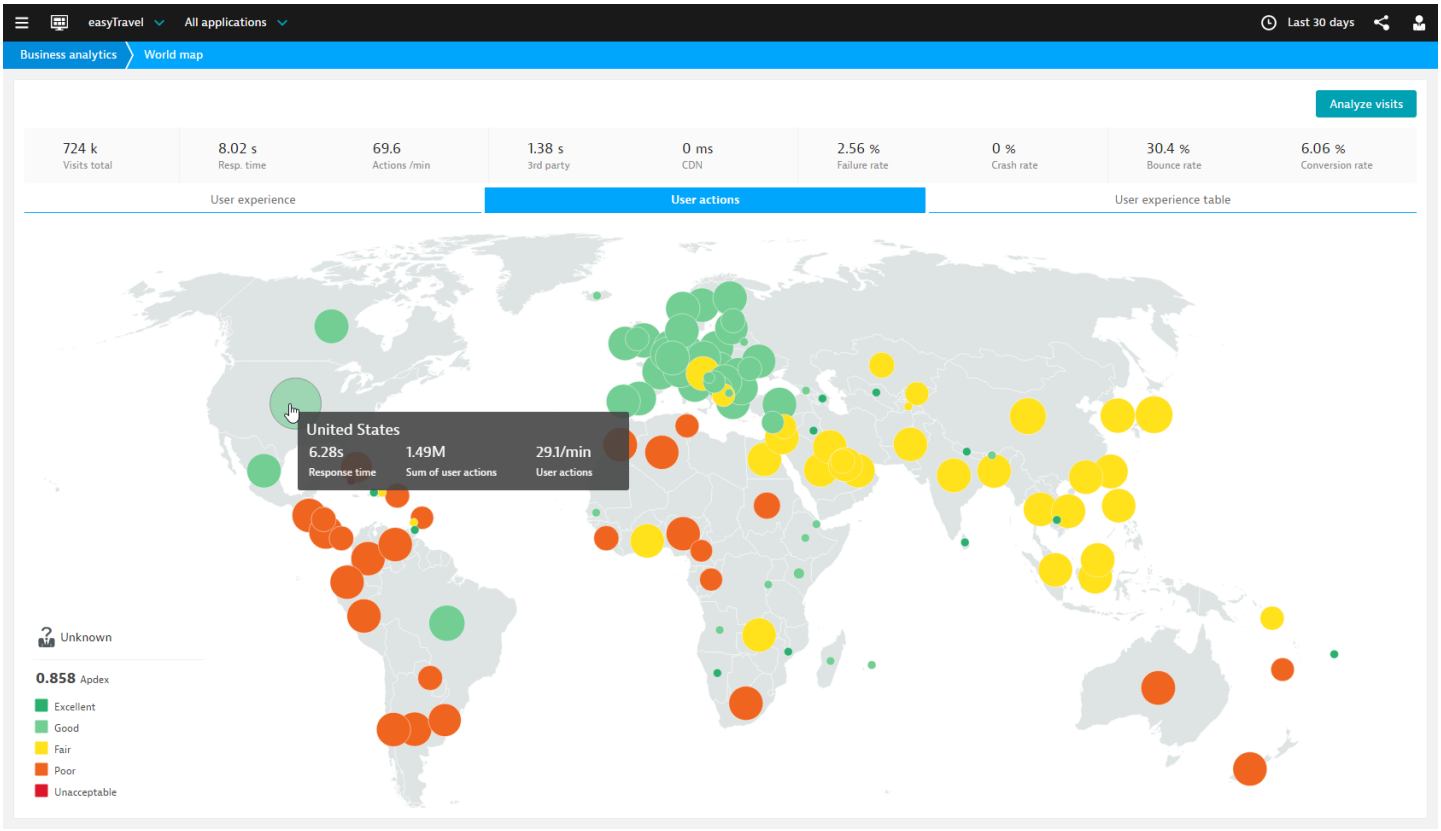
- User experience index (Apdex rating)
- Average response time
- Count of all user actions
- Number of user actions per minute

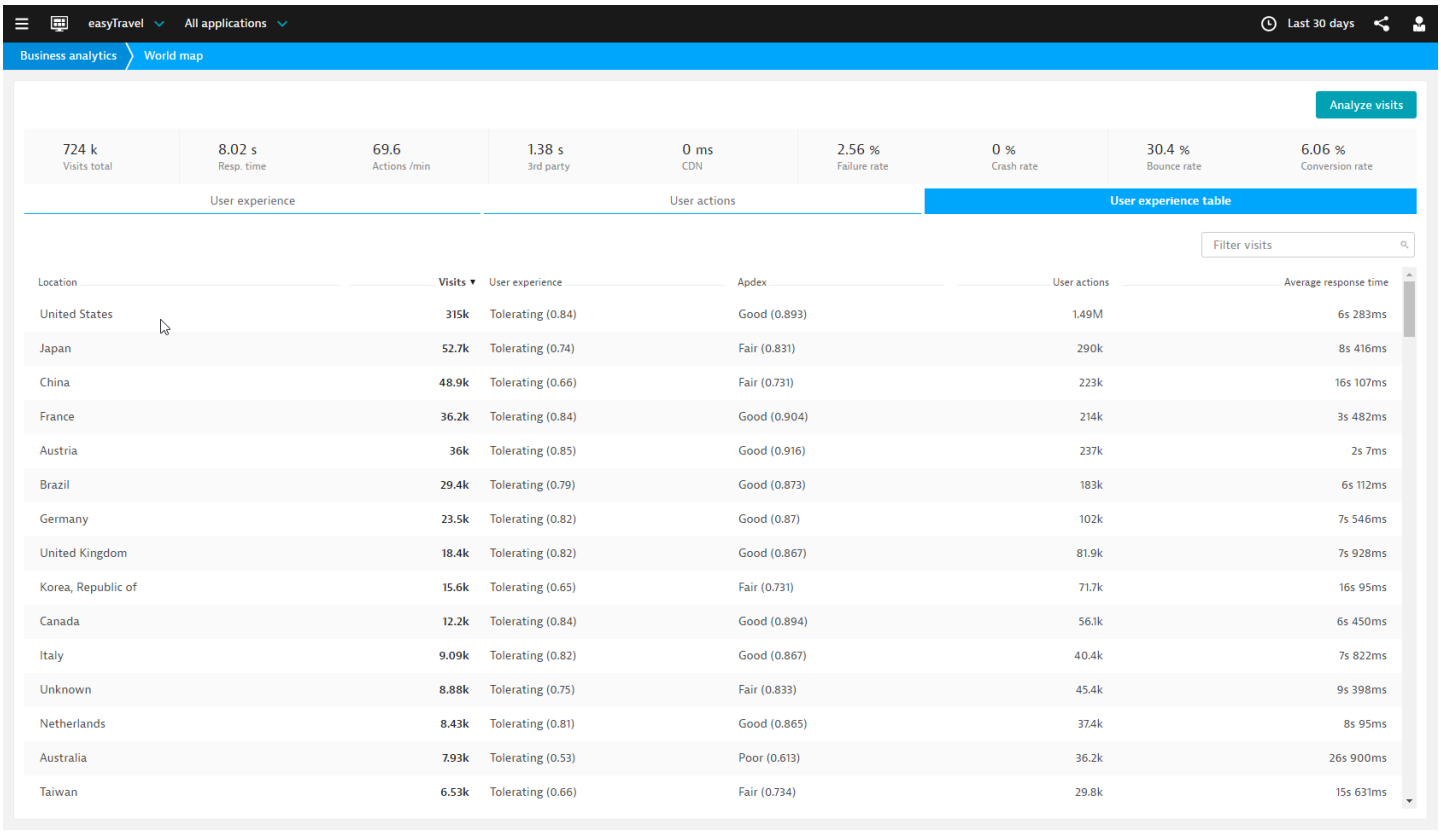
With both the **User experience** and **User actions** tabs, the contents of the corresponding User experience table reflects the level to which you are drilled down to. For example, if you are examining user actions, and select North America > United States > Michigan, the User experience table lists only the aggregated User Experience metrics for Michigan.

On the User experience or User actions tab, select a region or country, and then hover over a country or state to view a popup message containing these metrics.

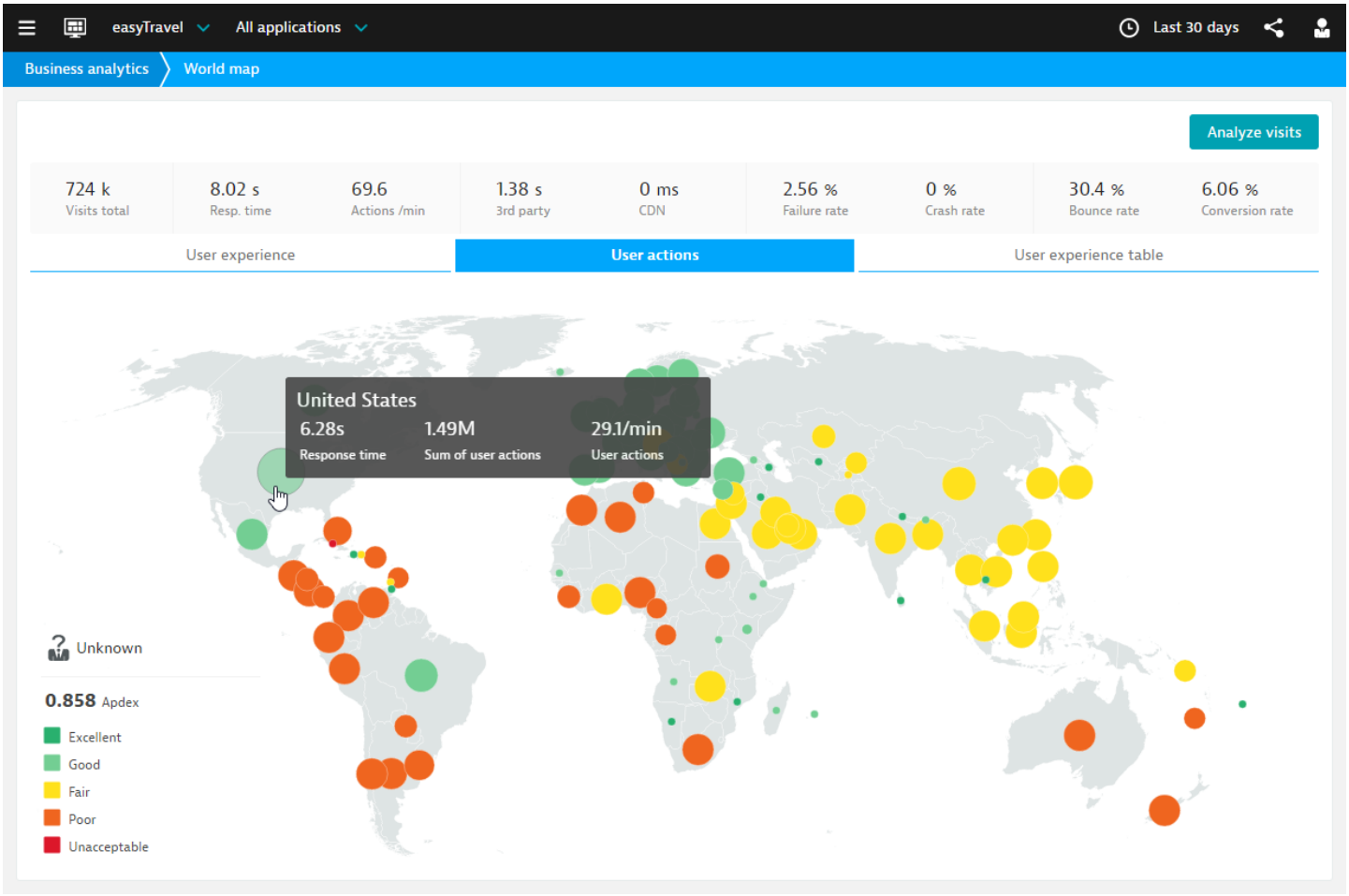
Three tabs allowyou to visualize user experience metric data or see a data-only view

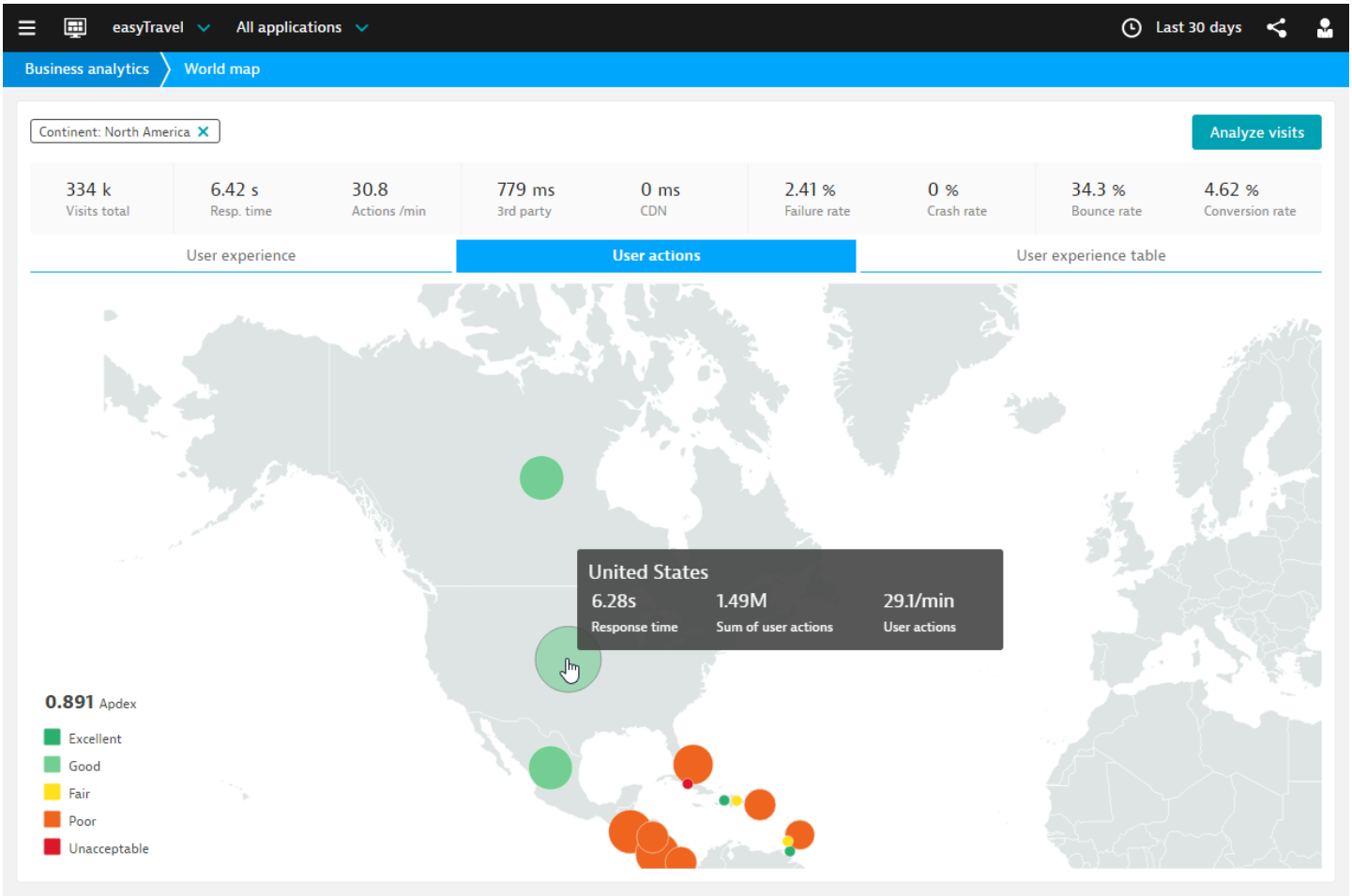


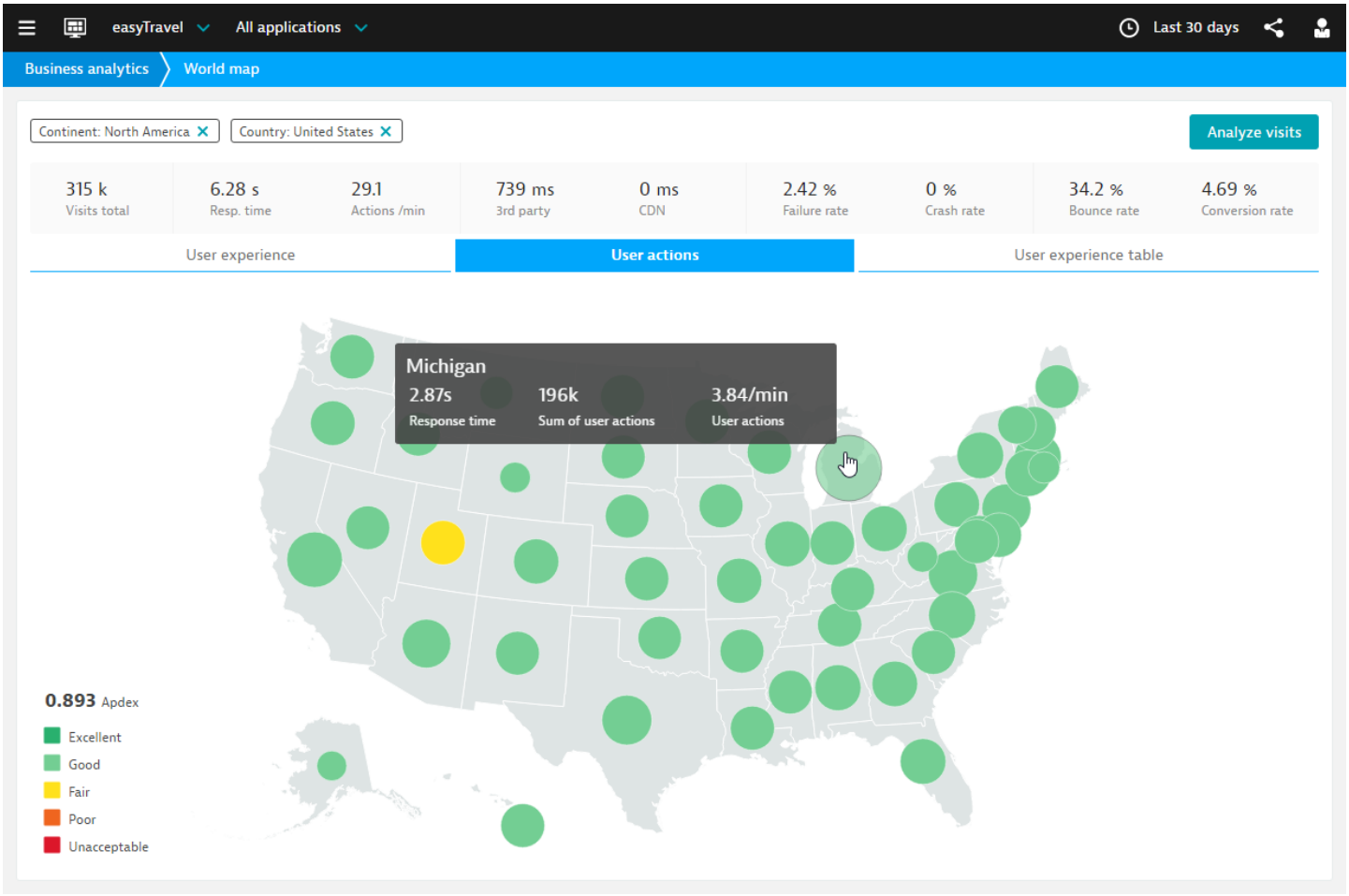


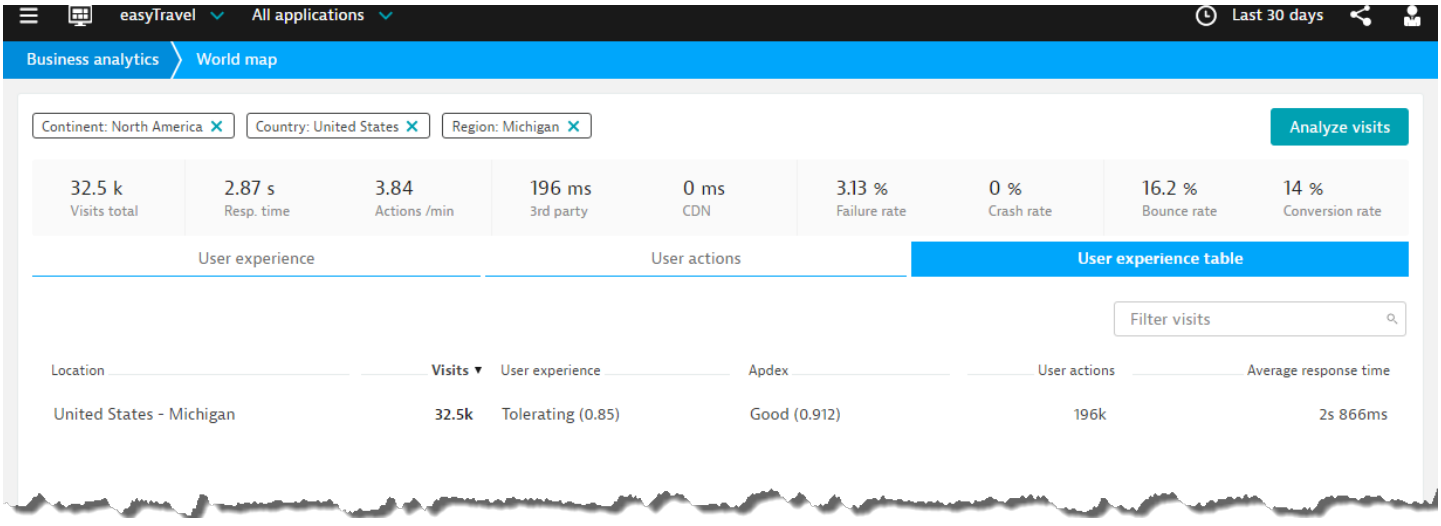


Visualize the context of user experience metrics at the world, region, or state level.


easyTravel
All applications
Last 30 days
Business analytics
World map
Analyze visits
724 k
Visits total
8.02 s
Resp. time
69.6
Actions /min
1.38 s
3rd party
0 ms
CDN
2.56 %
Failure rate
0 %
Crash rate
30.4 %
Bounce rate
6.06 %
Conversion rate
User experience
User actions
User experience table
United States
6.28s
Response time
1.49M
Sum of user actions
29.1/min
User actions
Unknown
0.858 Apdex
Excellent
Good
Fair
Poor
Unacceptable


easyTravel
All applications
Last 30 days
Business analytics
World map
Continent: North America
Analyze visits
334 k
Visits total
6.42 s
Resp. time
30.8
Actions /min
779 ms
3rd party
0 ms
CDN
2.41 %
Failure rate
0 %
Crash rate
34.3 %
Bounce rate
4.62 %
Conversion rate
User experience
User actions
User experience table
United States
6.28s
Response time
1.49M
Sum of user actions
29.1/min
User actions
0.891 Apdex
Excellent
Good
Fair
Poor
Unacceptable



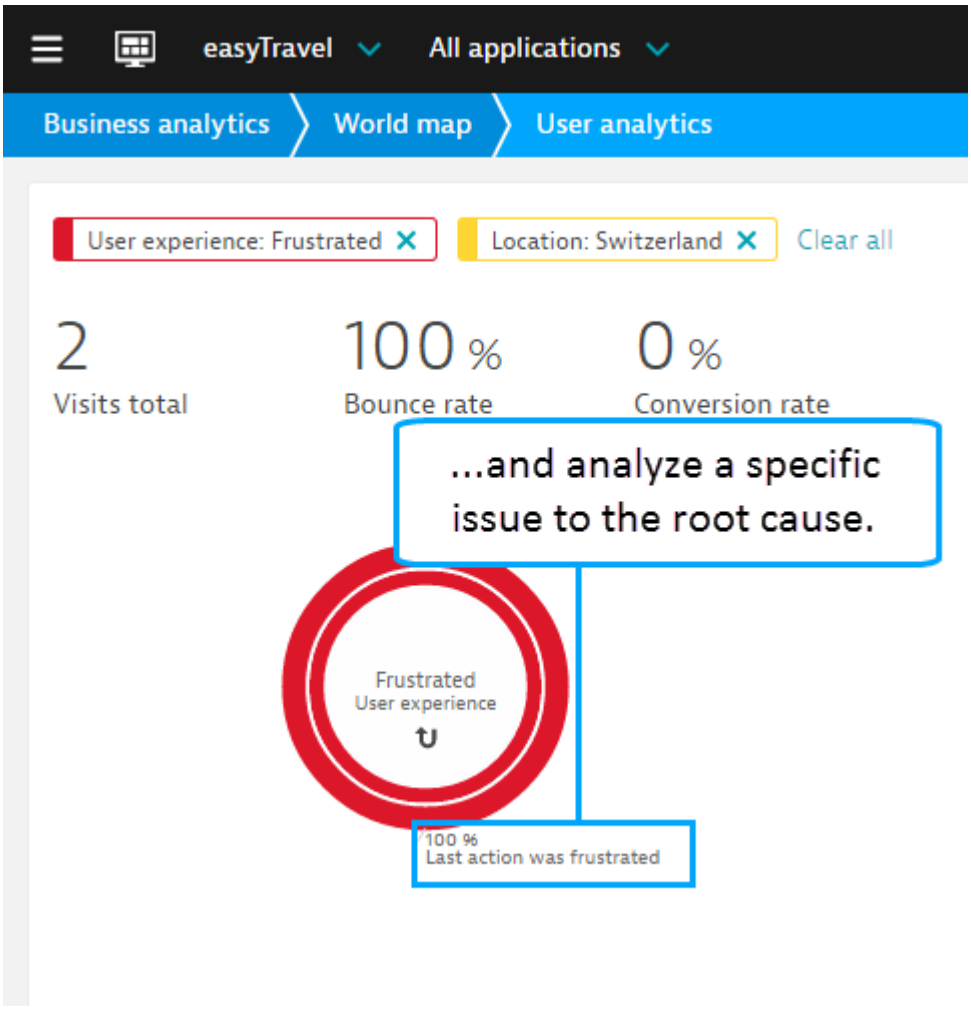


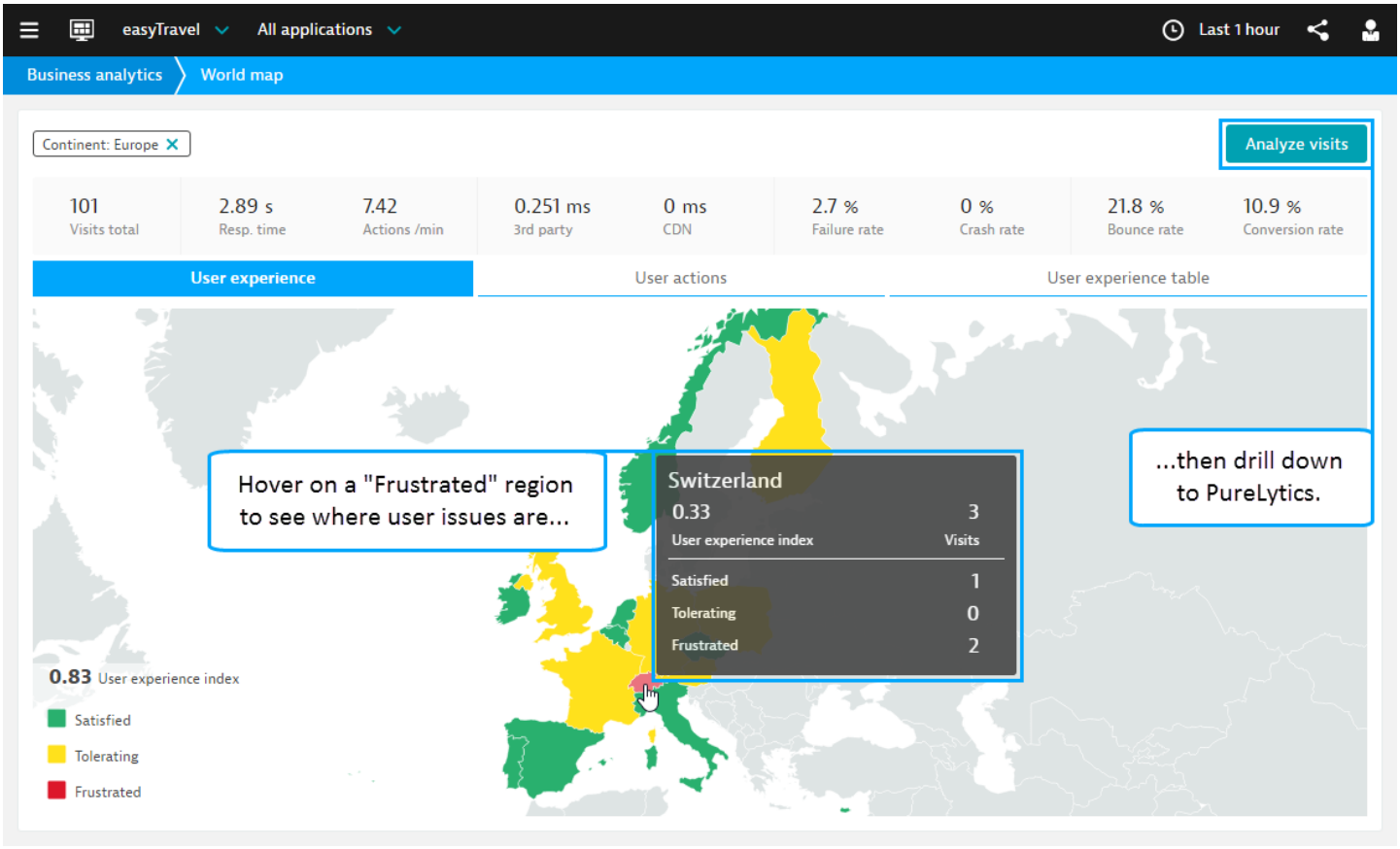
Note

The drilldown to specific states and provinces for selected countries requires that the **Country of Visits** business transaction splitting value is set up for each system profile. See Dynatrace Web Administration for more details.

Drill down to analyze a specific issue

When using the World Map to find areas where customers have issues, you can drill down into PureLytics data in the User Analytics dashboard to analyze specific issues. Select the region or country, then click Analyze Visits to see the visits in the User Analytics Web dashboard. Then filter User Experience to view only the frustrated issues. When you have filtered to find your specific visit, click **Inspect Filtered Visits** to start root cause analysis.




easyTravel
All applications
Last 1 hour
Business analytics
World map
Continent: Europe
Analyze visits
101
Visits total
2.89 s
Resp. time
7.42
Actions /min
0.251 ms
3rd party
0 ms
CDN
2.7 %
Failure rate
0 %
Crash rate
21.8 %
Bounce rate
10.9 %
Conversion rate
User experience
User actions
User experience table
Hover on a "Frustrated" region to see where user issues are...
Switzerland
0.33
User experience index
3
Visits
Satisfied 1
Tolerating 0
Frustrated 2
...then drill down to PureLytics.
0.83 User experience index
Satisfied
Tolerating
Frustrated

Monitor User Actions by geographic region

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

User experience management

World Map dashlet

Defining custom locations in the world map

Monitor user visits by geographic region

Use top findings to analyze user visits

Monitor user satisfaction

Monitor user satisfaction

Trace user activity through all channels

Find specific user visits

Overview

Use the World Map to quickly visualize in real-time the user experience of your system at the world, region, and country level (for countries with large populations). It also displays key global metrics for a succinct understanding of how your users see application performance from their end. In AppMon Web, data includes:

- User experience index (Apdex rating)
- Average response time
- Average number of user actions per minute

- Failure rate of all actions

Access the World map using any of the following:

- AppMon Web: User Experience tile in the Business Analytics dashboard
- AppMon Client or AppMon Webstart Client: You can access the World Map any of the following ways:
  - Add a **World Map Dashlet** to any dashboard.
  - In the Start Center, select **User Experience > User Experience World Map** and click the User Experience World Map link.
  - In the Start Center, select **User Experience > User Experience Key Metrics** and click the UEM Key Metrics Dashboard link. The World Map is part of the key analytic metrics shown in this dashboard.
  - For the desired System Profile in the cockpit, select **Diagnose Performance > Diagnose User Experience > World Map** .
  - From the application's transaction flow page, click End User Experience to see Visit Count and related key visit metrics, and visits by client type in the End User Experience dashboard.

What's the User Experience Index?

The User Experience Index used in AppMon Web indicates how performance is perceived by an application's users, and categorizes visits into satisfying, tolerating, and frustrating visits. The index is based on the the following factors:

- **Apdex rating**, which is a universal standard for measuring the satisfaction of user action performance.
- **Errors or failures**, such as JavaScript errors or pages not found. A page that loads quickly may still have an error.
- **Entire user journey**. The last user action performed in the entire user journey is weighted more heavily.
- **User environment** - Poor bandwith is considered in the index calculation.

Users have a **satisfying** experience if:

- No action failed
- More than 50% of all actions were satisfying

Users have a **frustrating** experience if:

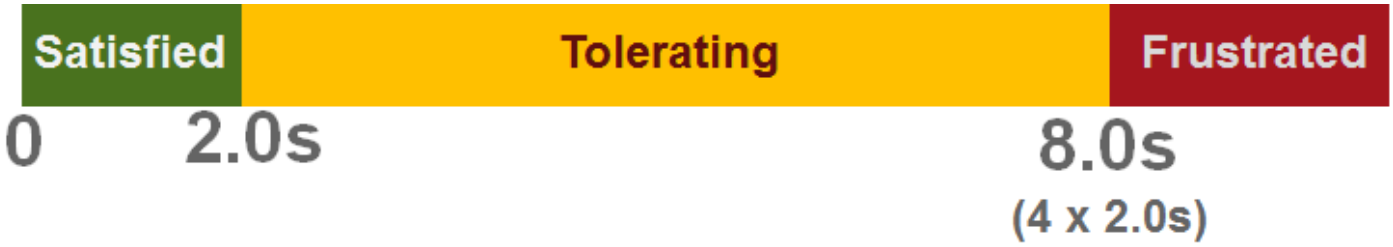
- Their last action failed ("The Web site does not work - I'm leaving!")

- Their last action was frustrating (“The Web site is too slow - I’m leaving!”)
- More than 50% of all actions were frustrating

Users have a **tolerating** experience if their experience is neither frustrating or satisfying, which means that:

- Their last action was not frustrating or failed
- Less than 50% of all actions were satisfying
- More than 50% of all actions were at least tolerating

For example, assume you have an application-wide satisfied-threshold called page load baseline. The threshold for page load baseline is defined as two seconds. User actions are mapped to the following zones based on their response time:



If there are 10 satisfied actions, five tolerating actions and two frustrated actions, the result is a User Experience Index of (10 + 5 / 2) / 17 = 0.73.

Analyze user experience details in AppMon Web

Click anywhere in the User Experience tile on the Business Analytics default dashboard to open the User Experience World Map page.

The banner at the top of the page displays real time data for the total number of visits, average response time, third party content average load time per page, as well as percentage rates for failures, crashes, bounces, and conversions. This data always reflects the configured data scope and is contextual based on viewing the world or a specific region, country, or state.

This page includes the **User experience**, **User actions**, and **User experience table** tabs. Each tab provides different data views for displaying the following key metrics that are aggregated at the country and state level.

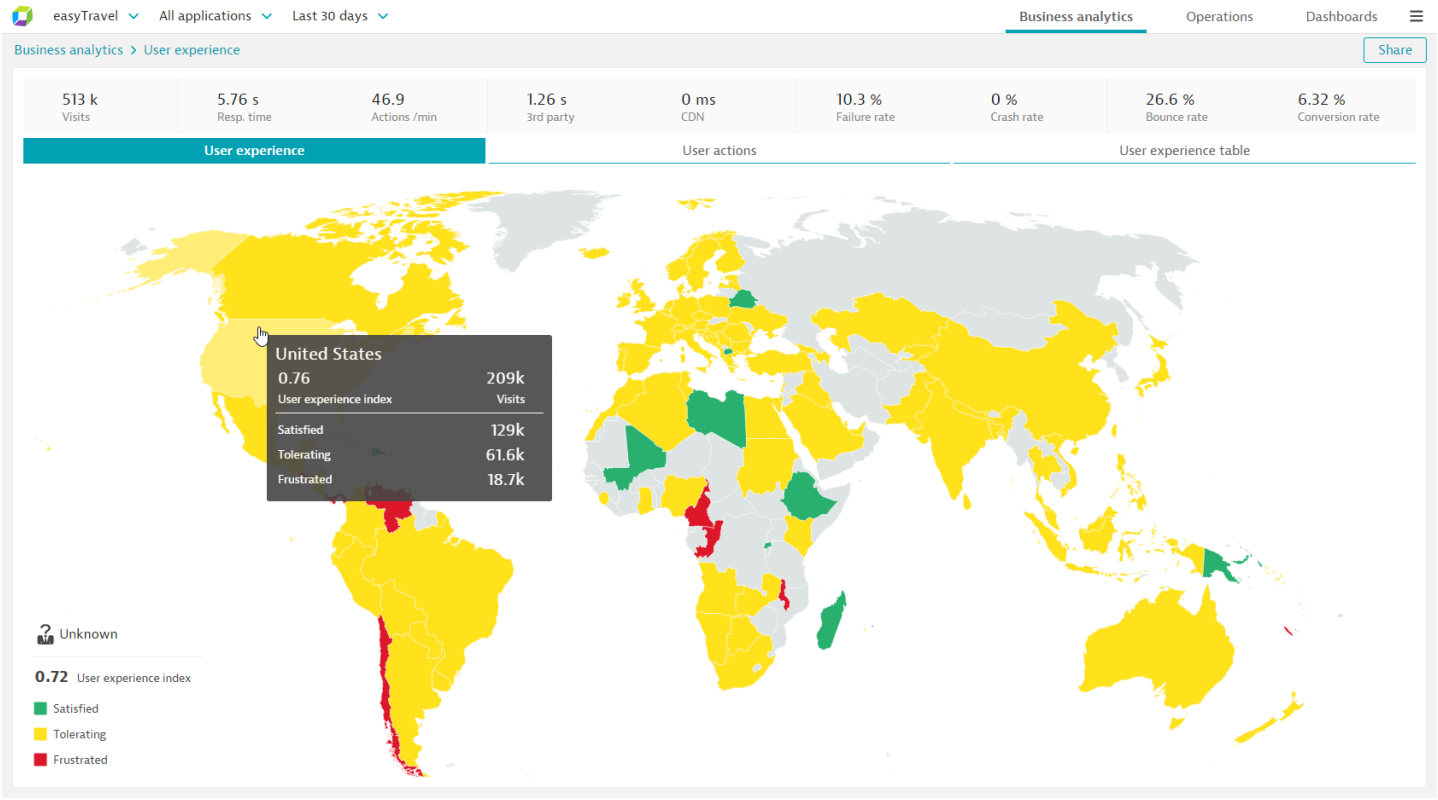
- User experience index (Apdex rating)
- Average response time

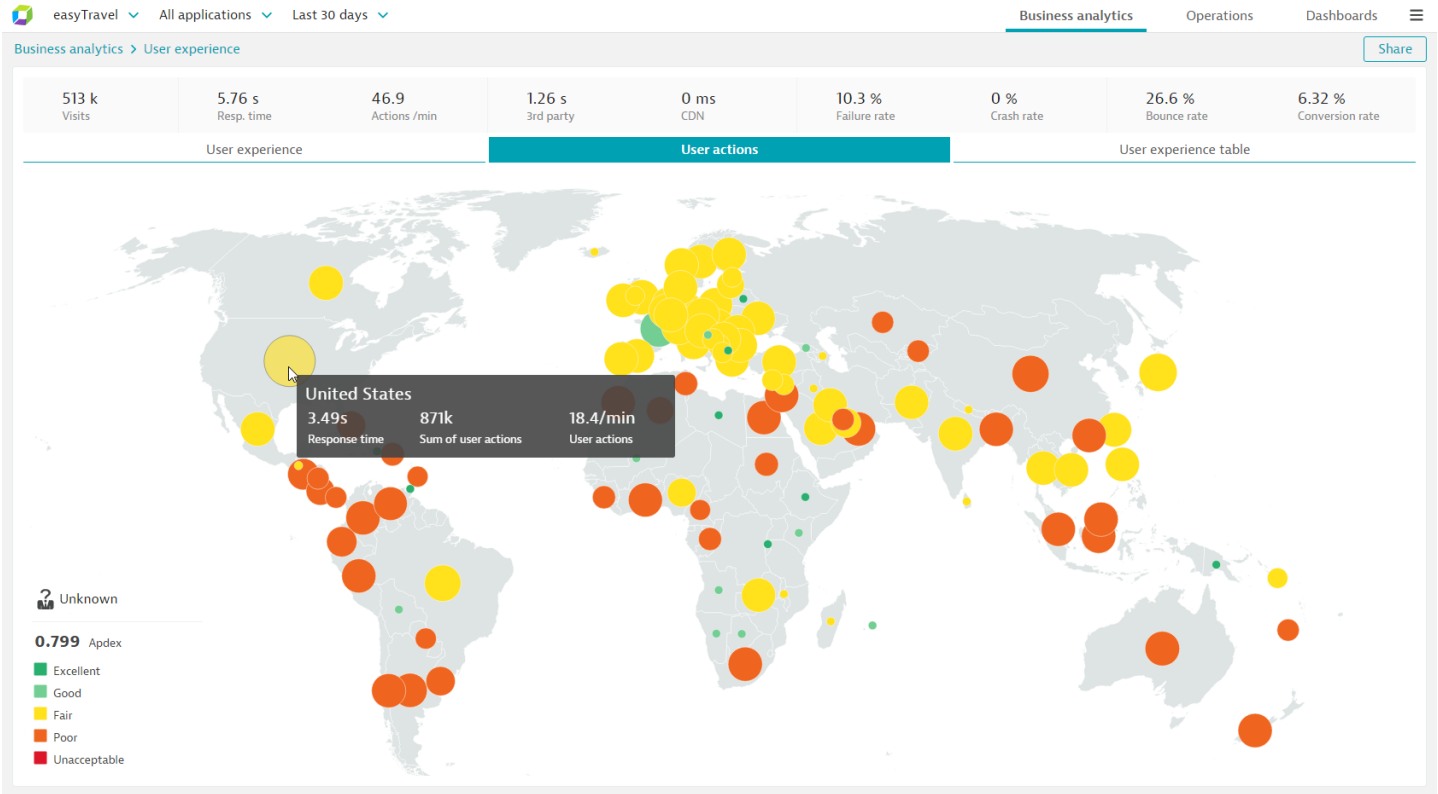
- Count of all user actions
- Number of user actions per minute

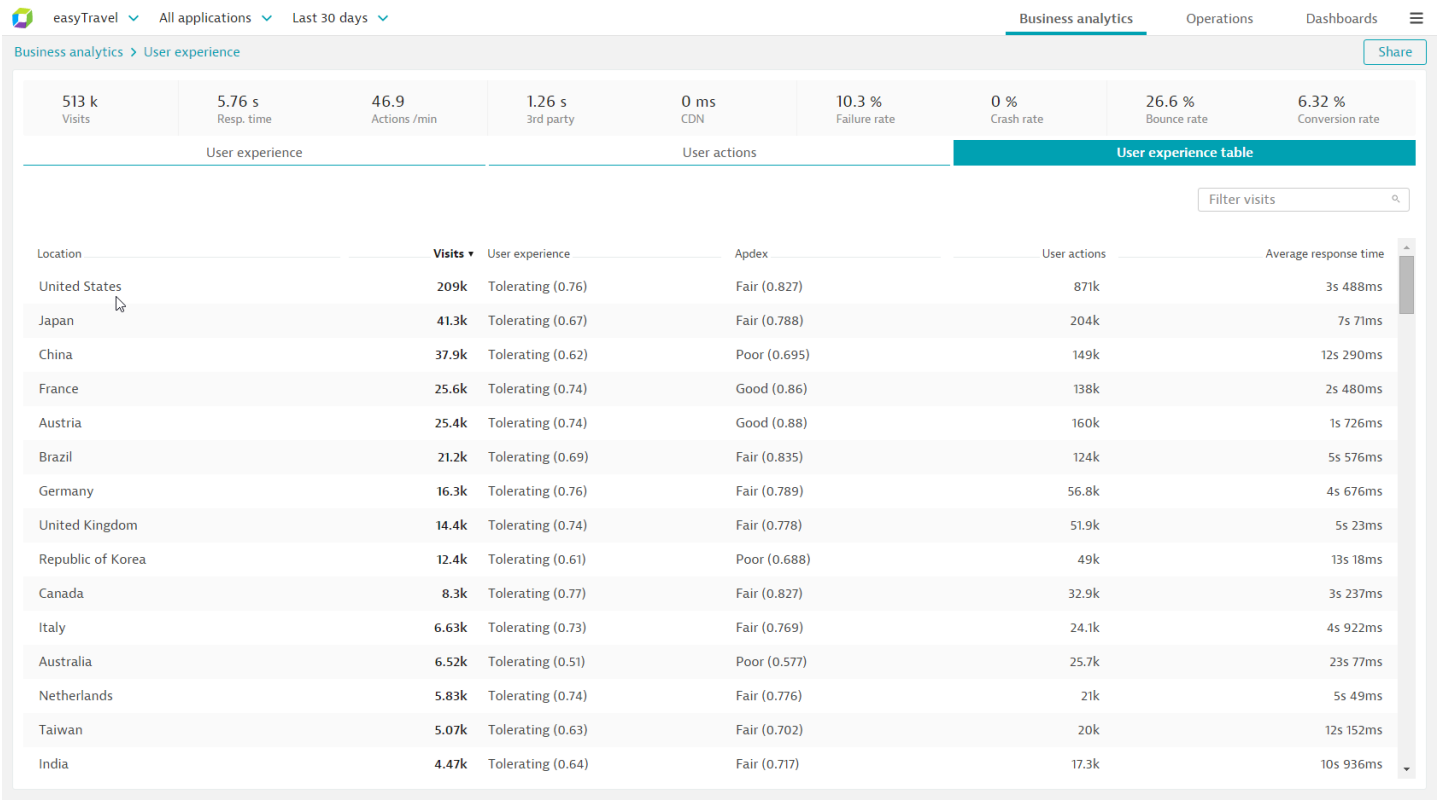
With both the **User experience** and **User actions** tabs, the contents of the corresponding User experience table reflects the level to which you are drilled down to. For example, if you are examining user actions, and select North America > United States > Michigan, the User experience table lists only the aggregated User Experience metrics for Michigan.

On the User experience or User actions tab, select a region or country, and then hover over a country or state to view a popup message containing these metrics.

Three tabs allowyou to visualize user experience metric data or see a data-only view

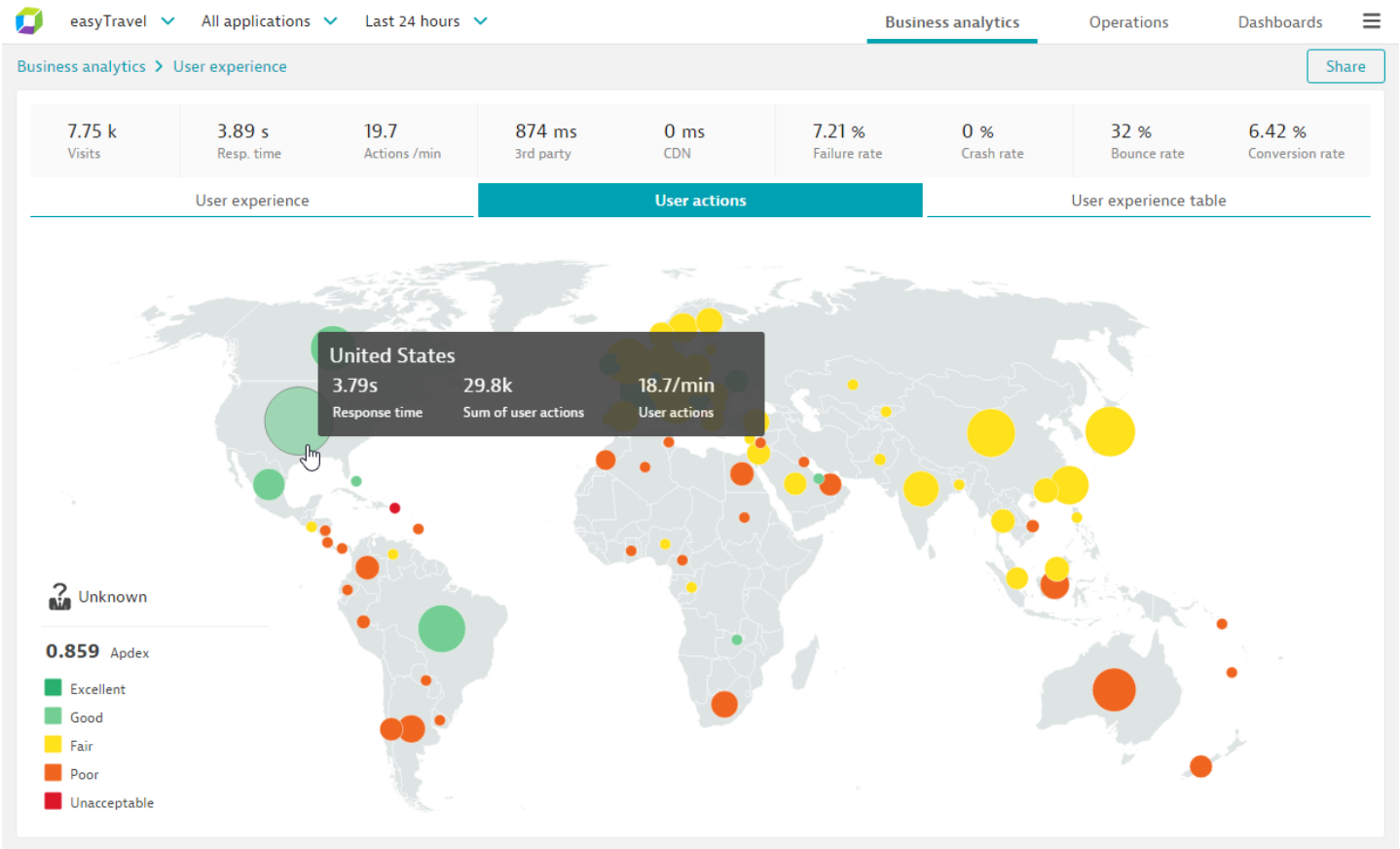


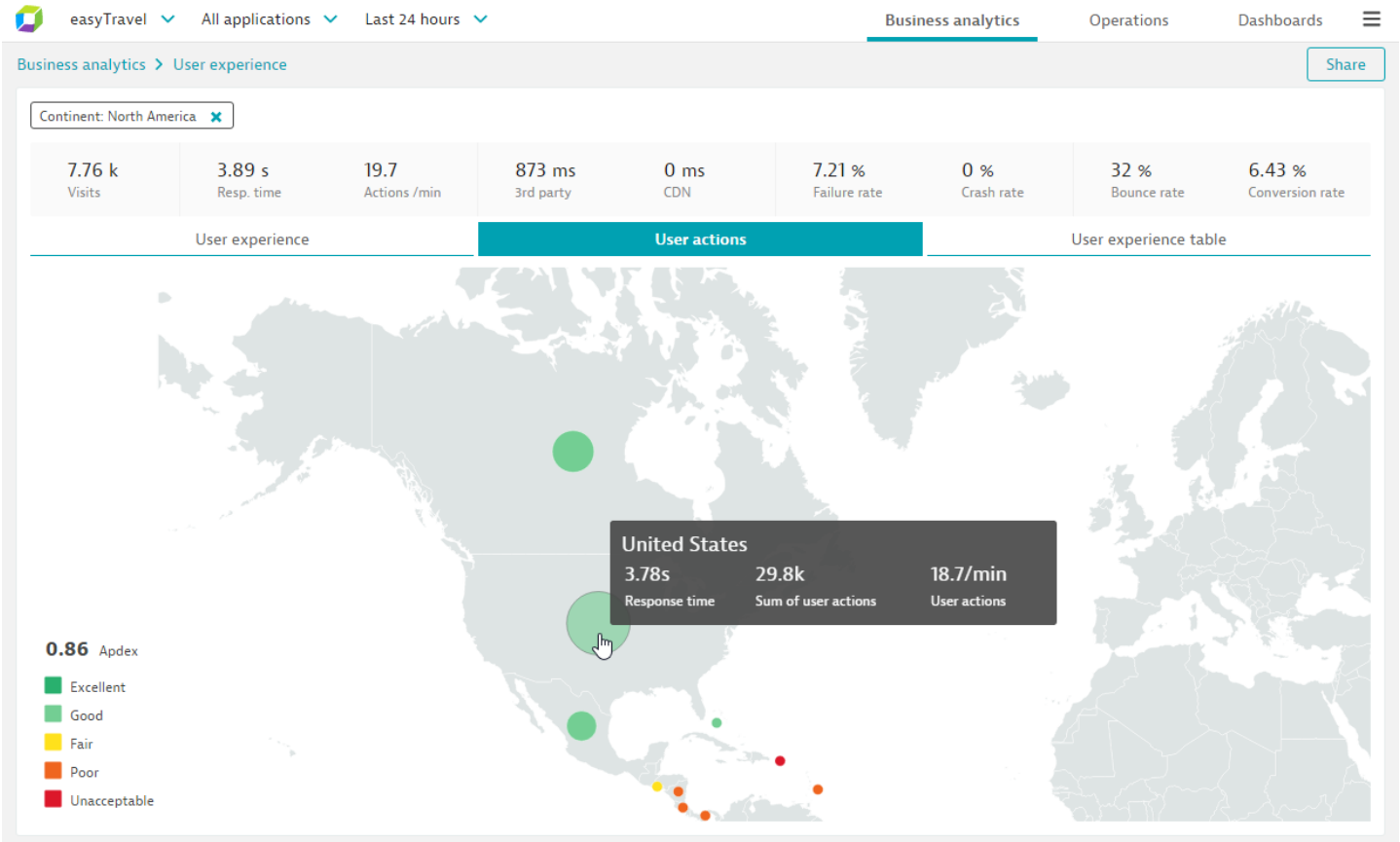


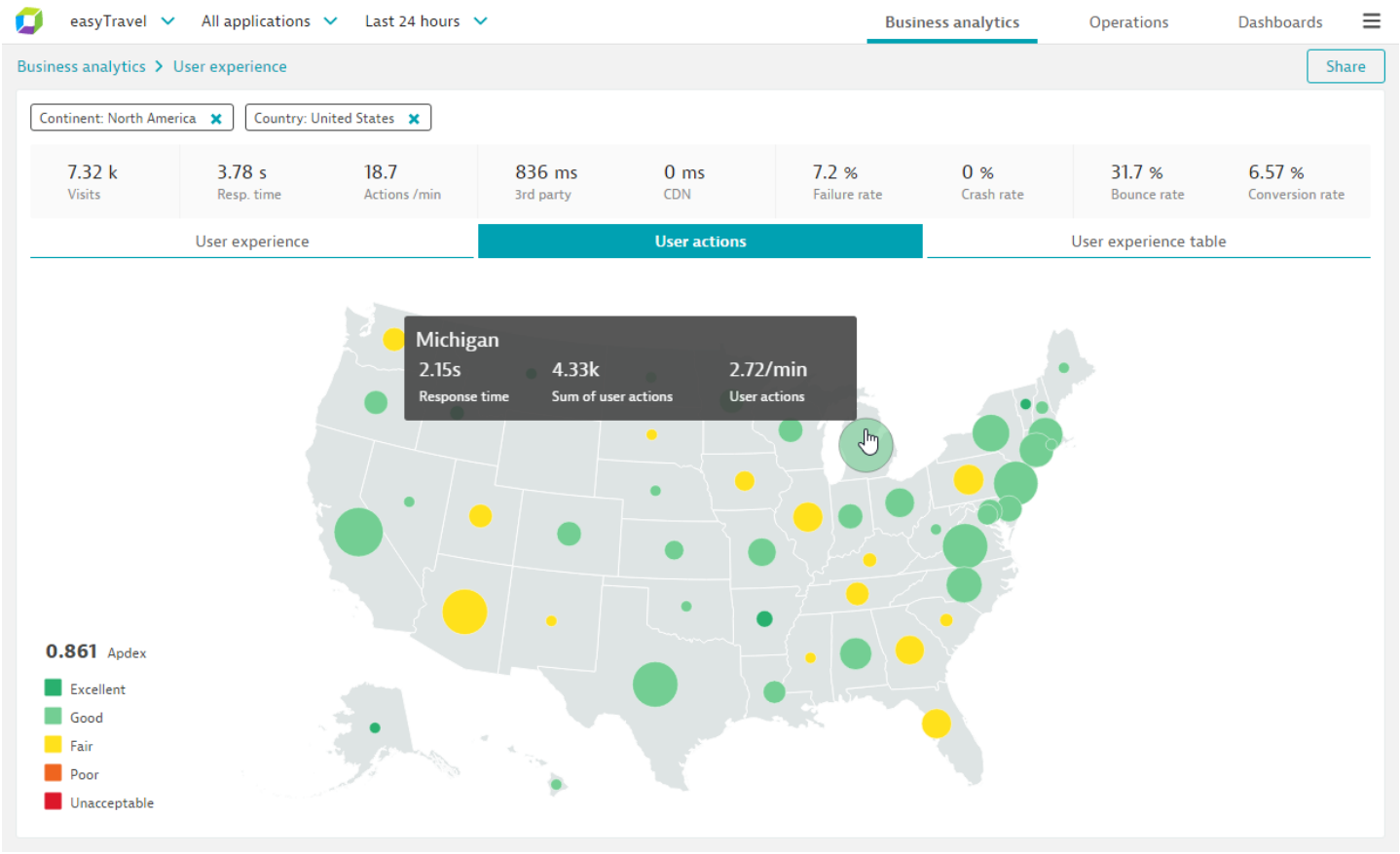


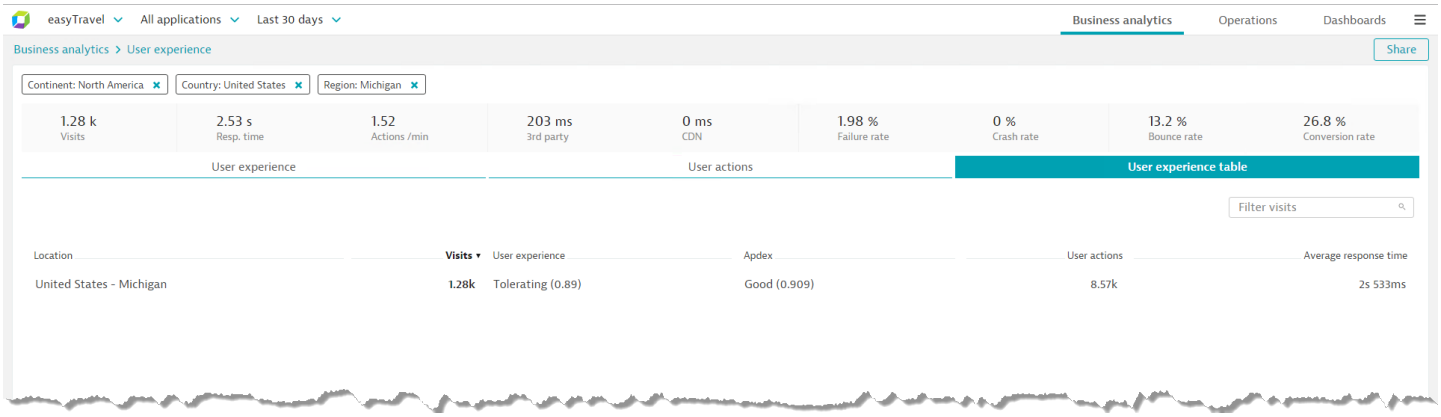
Location	Visits	User experience	Apdex	User actions	Average response time
United States	209k	Tolerating (0.76)	Fair (0.827)	871k	3s 488ms
Japan	41.3k	Tolerating (0.67)	Fair (0.788)	204k	7s 71ms
China	37.9k	Tolerating (0.62)	Poor (0.695)	149k	12s 290ms
France	25.6k	Tolerating (0.74)	Good (0.86)	138k	2s 480ms
Austria	25.4k	Tolerating (0.74)	Good (0.88)	160k	1s 726ms
Brazil	21.2k	Tolerating (0.69)	Fair (0.835)	124k	5s 576ms
Germany	16.3k	Tolerating (0.76)	Fair (0.789)	56.8k	4s 676ms
United Kingdom	14.4k	Tolerating (0.74)	Fair (0.778)	51.9k	5s 23ms
Republic of Korea	12.4k	Tolerating (0.61)	Poor (0.688)	49k	13s 18ms
Canada	8.3k	Tolerating (0.77)	Fair (0.827)	32.9k	3s 237ms
Italy	6.63k	Tolerating (0.73)	Fair (0.769)	24.1k	4s 922ms
Australia	6.52k	Tolerating (0.51)	Poor (0.577)	25.7k	23s 77ms
Netherlands	5.83k	Tolerating (0.74)	Fair (0.776)	21k	5s 49ms
Taiwan	5.07k	Tolerating (0.63)	Fair (0.702)	20k	12s 152ms
India	4.47k	Tolerating (0.64)	Fair (0.717)	17.3k	10s 936ms

Visualize the context of user experience metrics at the world, region, or state level.


easyTravel
All applications
Last 24 hours
Business analytics
Operations
Dashboards
Business analytics > User experience
Share
7.75 k
Visits
3.89 s
Resp. time
19.7
Actions /min
874 ms
3rd party
0 ms
CDN
7.21 %
Failure rate
0 %
Crash rate
32 %
Bounce rate
6.42 %
Conversion rate
User experience
User actions
User experience table
United States
3.79s
Response time
29.8k
Sum of user actions
18.7/min
User actions
Unknown
0.859 Apdex
Excellent
Good
Fair
Poor
Unacceptable


easyTravel
All applications
Last 24 hours
Business analytics
Operations
Dashboards
Business analytics > User experience
Share
Continent: North America
7.76 k
Visits
3.89 s
Resp. time
19.7
Actions /min
873 ms
3rd party
0 ms
CDN
7.21 %
Failure rate
0 %
Crash rate
32 %
Bounce rate
6.43 %
Conversion rate
User experience
User actions
User experience table
United States
3.78s
Response time
29.8k
Sum of user actions
18.7/min
User actions
0.86 Apdex
Excellent
Good
Fair
Poor
Unacceptable





Note

The drilldown to specific states and provinces for selected countries requires that the **Country of Visits** business transaction splitting value is set up for each system profile. See AppMon Web Administration for more details.

Use top findings to analyze user visits

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

Monitor user visits by geographic region

Monitor User Actions by geographic region

Monitor top landing and exit pages

Monitor top landing and exit pages

Monitor user satisfaction

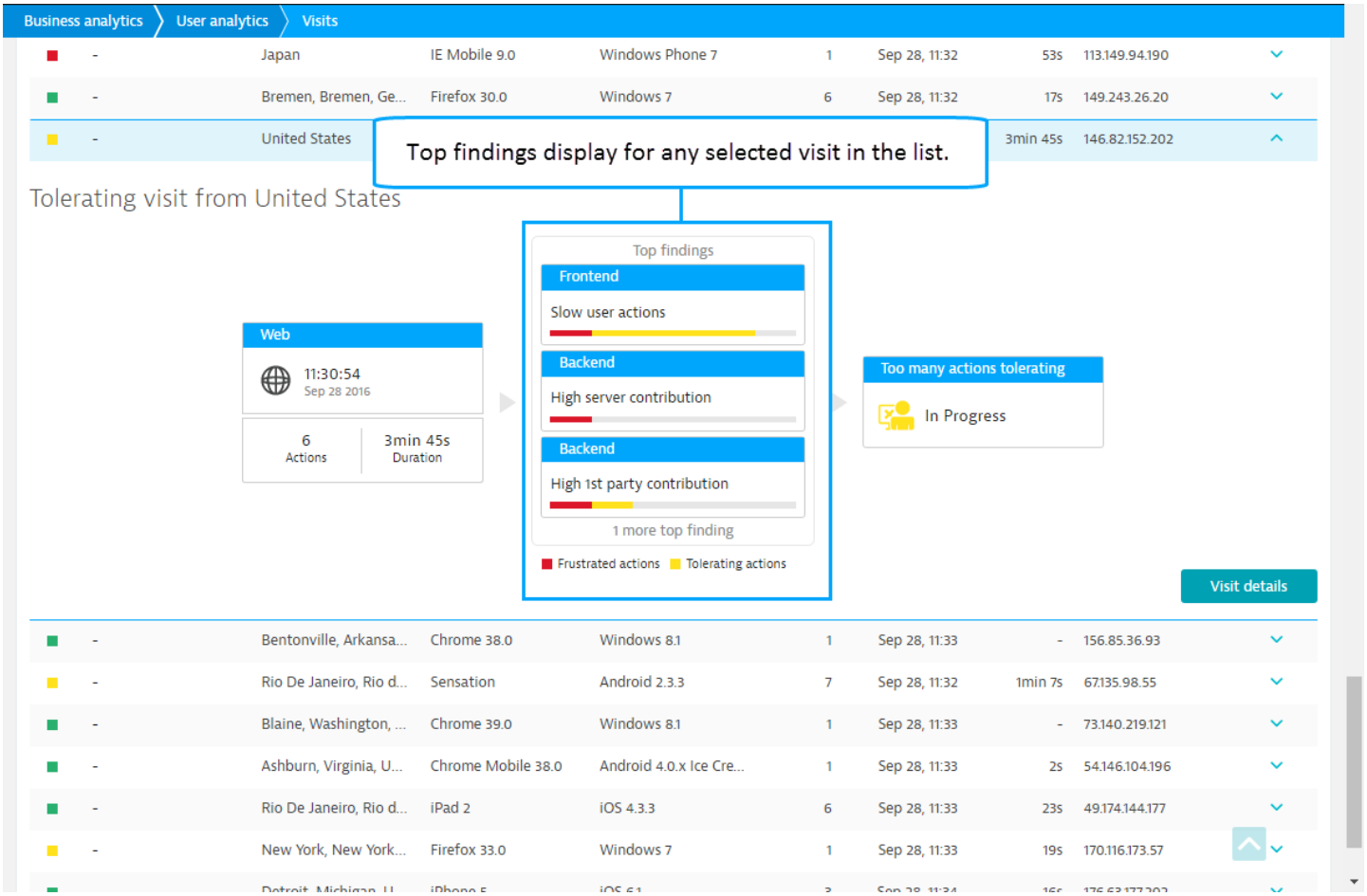
Trace user activity through all channels

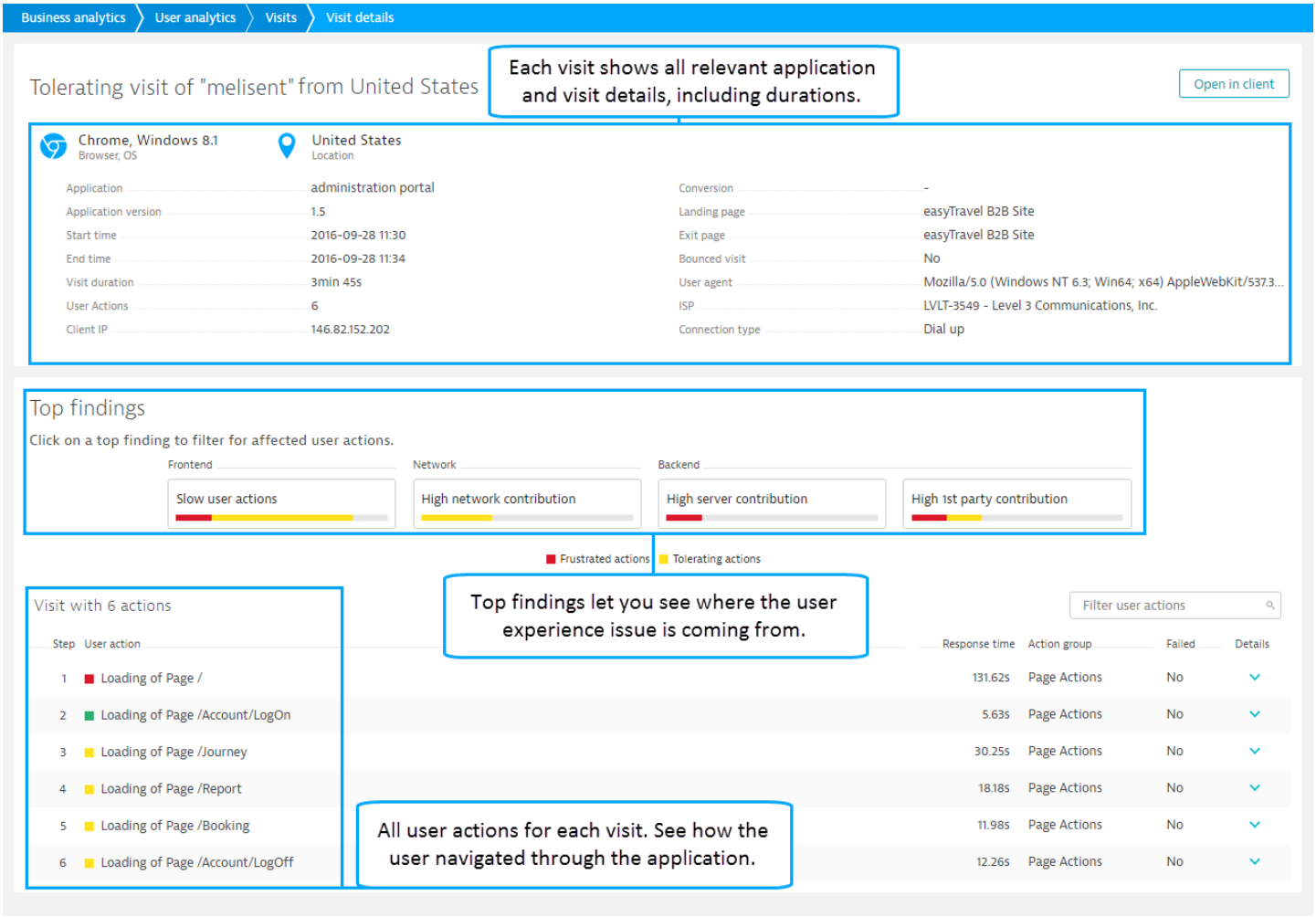
System Profile - User Experience

Overview

AppMon Web lets you easily identify the component or area of your application that causes bad user experience, and get to the root causes of poor user visit experiences. AppMon 6.5 and later analyzes every user action and highlights problematic patterns as top findings.

The top findings display on both the Visits and Visit Details Web dashboards instantly reveal a problem's origin, whether it is on the frontend, backend, third party content or one or more of these. This data is particularly useful to frontend developers to help diagnose and resolve user experience issues. Other relevant details on the Visit Details dashboard such as visit duration and user action details provide a complete picture of each visit so you can quickly assess and act on frustrated actions.


Business analytics
User analytics
Visits
Top findings display for any selected visit in the list.
Tolerating visit from United States
Top findings
Frontend
Slow user actions
Backend
High server contribution
Backend
High 1st party contribution
1 more top finding
Frustrated actions
Tolerating actions
Web
11:30:54
Sep 28 2016
6
Actions
3min 45s
Duration
Too many actions tolerating
In Progress
Visit details



Top finding types

These issue types can appear as top findings for a visit.

Frontend

- **Slow user actions**: User actions are marked as satisfied, tolerating or frustrated. Based on APDEX thresholds that can be configured per action group.
- **Client side errors**: Flags user actions containing JavaScript errors or other reported errors that triggered an error detection rule.

Network

- **High third party contribution**: Third party load times are a much higher part of the overall network time than normal.
- **High CDN contribution**: Indicates that available content delivery sources contributed to an abnormally high percentage of the total network time for the user actions.

- **High network contribution**: Indicates when bandwidth and/or request and response sizes make up a high percentage of total network time.

Backend

- **High server contribution**: Time spent on the server to process requests that impact user action response time is unusually high.
- **High 1st party contribution**: Time to load resources from the application domain and CDNs is an abnormally high percentage of load time.

Monitor top landing and exit pages

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

Monitor user visits by geographic region

Monitor User Actions by geographic region

Use top findings to analyze user visits

Monitor user satisfaction

Trace user activity through all channels

Find specific user visits

Drilldown to the AppMon Client

Overview

Use Top landing and exit page data to see where your main points of traffic generate, and how users interact with your web application based on scoped data. Landing and exit page data also helps you answer questions such as:

- Are visitors taking the actions intended by the application? Are some pages of my web application not being used as expected?
- Are users dropping off my web application from places other than anticipated drop off points? If so, could performance be playing a part?
- Where is the majority of my traffic coming from? Is an unsatisfying user experience a possible factor in traffic generation in some locations?
- Are user actions and performance consistent among all locations generating traffic?

- Do some pages in my application fail more than others?

Access landing and exit page data using the following:

- AppMon Web: User Experience tile in the Business Analytics dashboard.
- AppMon Client or AppMon Webstart Client: In the Start Center, select **User Experience > Landing Pages** and click the Landing Pages link.

Monitor landing and exit pages in AppMon Web

The Top countries, Top landing pages, and Top exit pages tiles in the Business analytics dashboard give you an overview of where your main points of traffic generate, and how users interact with your web application based on scoped data.

Click on any one of these tiles to view details in their respective detail views.

Lists in any of the detailed views for these tiles can be sorted by any of the data columns in the list. For long lists on any of these views, you can use the filter box to streamline the list and quickly find specific landing pages, exit pages, or countries by name.

Top countries

The Top countries tile of the Business analytics dashboard lists the countries that have had visits to the application, listed by highest number of visits by default. For each country with a visit, The Top countries detail view shows the location and number of visits along with the user experience score (rating and number), the APDEX rating, the total number of user actions, and average response time.

Top landing pages and top exit pages

The Landing Pages and Top Exit Pages tiles both lists the countries that have had visits to landing and exit pages, listed by highest number of visits by default. The detailed views for both show the number of visits for each listing and the average response time, along with the bounce rate, conversion rate, and percentage of failed transactions for each landing or exit page.

Analyze visit details

On this page

Overview

Waterfall data

Analyze PurePaths and drilldown

Related pages

User Action timings

Third-party content analysis

Web performance optimization for UEM

Monitor streaming media activity

Configure geographic location

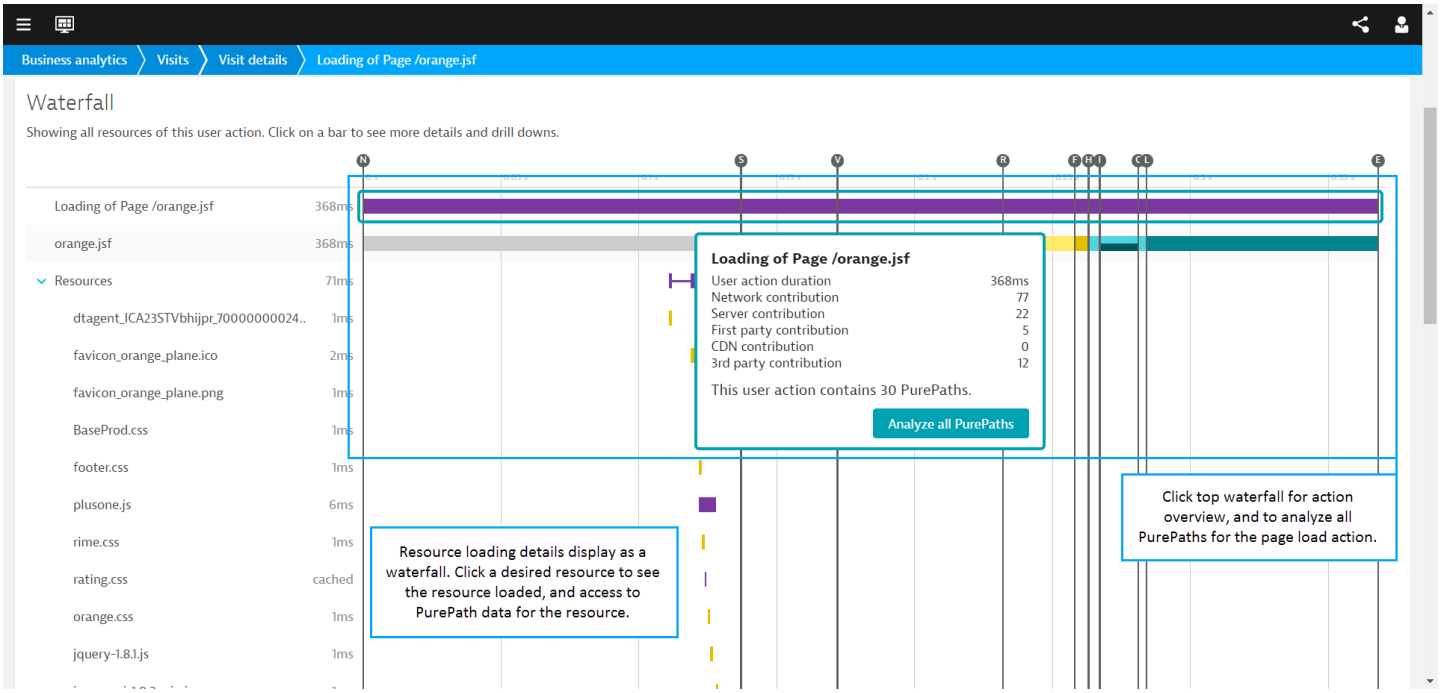
W3C resource timing metrics

Overview

For AppMon May 2017, visit details include a waterfall chart that shows timings for all the resources loaded for a selected user action. This lets you see at a glance exactly how the page loaded, resource loading with respect to performance metrics, and what resources have the biggest impact on page load performance.

Waterfall data

The top bar in the page load timeline at the top of the waterfall provides a quick view of contribution to the overall page load time. Click it to see contribution details, a count of the PurePaths in the user action, and a link to analyze all of the action's PurePaths.



Select the bar of any resource in the waterfall to see the following for that resource:

- A link to the view the loaded resource.
- General load time and resource sizing information.
- Resource timing form performance measures.

Note that detailed W3C resource timing metrics are not available for cross-origin resources.

Resource timing steps

The following measures are used to chart the duration of specific steps in the page loading process.

Measure	Description	Definition in terms of W3C specification
DNS	Time spent or resolving domain names.	`window.performance.domainLookupEnd - window.performance.domainLookupStart`
Connect	Time spent establishing a socket connection from the browser to the web server.	`window.performance.connectEnd - window.performance.connectStart`
SSL	Time spent establishing a secure socket connection from the browser to the web server.	`window.performance.connectEnd - window.performance.secureConnectionStart`
URL Redirection	Time spent following HTTP redirects.	`window.performance.redirectEnd - window.performance.redirectStart`

Request	Time spent waiting for the first byte of the document response.	`window.performance.responseStart - window.performance.requestStart`
Response	Time spent downloading the document response.	`window.performance.responseEnd - window.performance.responseStart`
Total	Time between the response being delivered and the `OnLoad` event.	`window.performance.loadEventEnd - window.performance.domLoading`

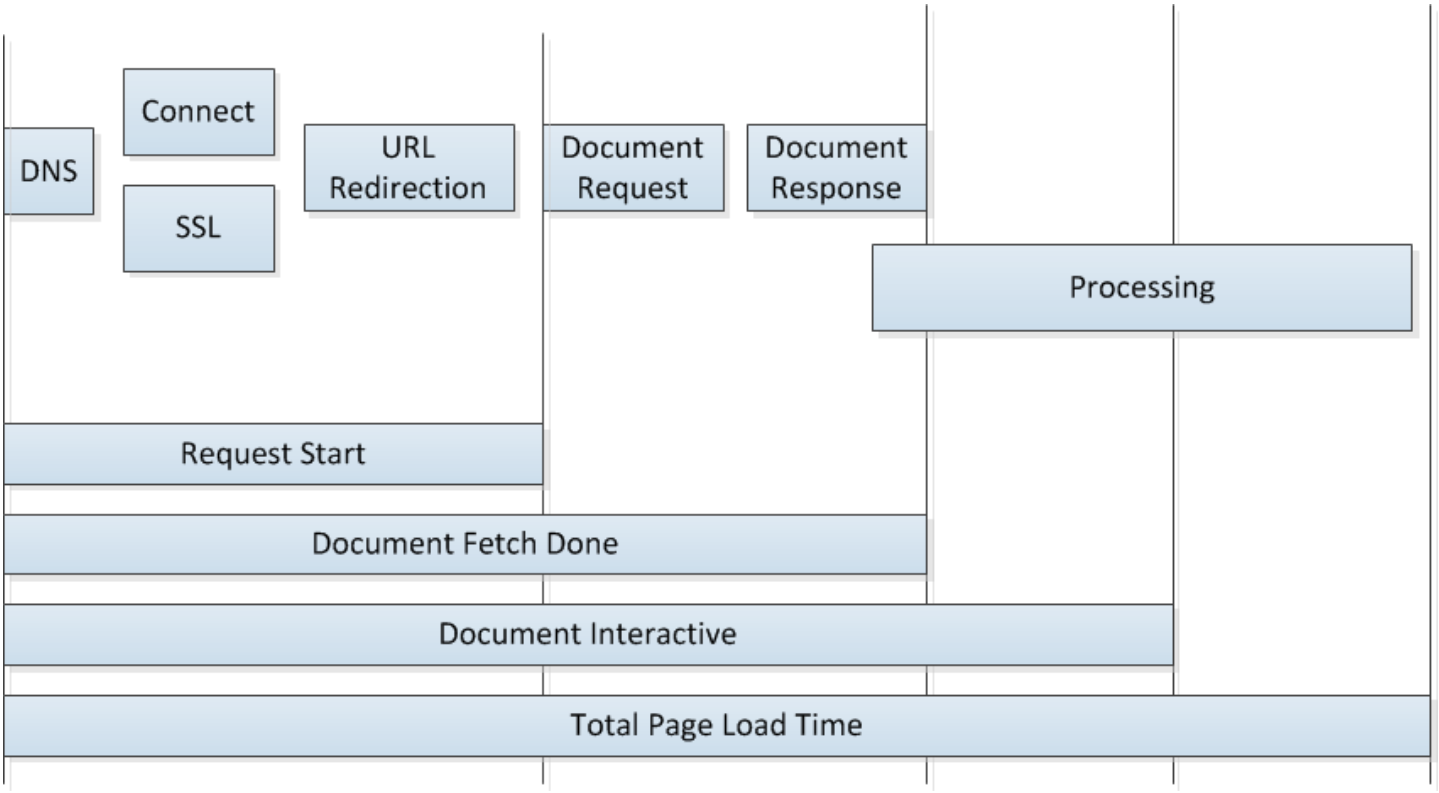
Resource timing landmarks

The following measures chart the total time from a user clicking a link or button until the browser reaches a specific point in the resulting page loading process.

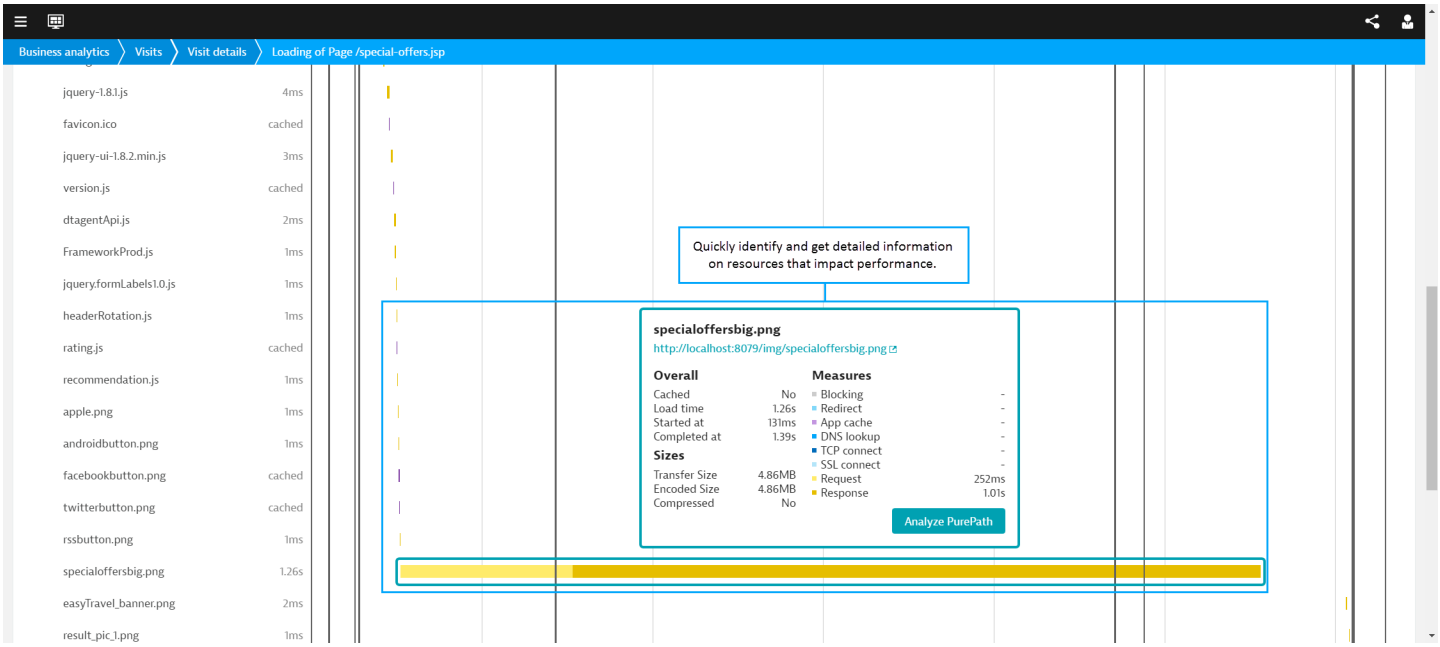
Measure	Description	Definition in terms of W3C specification
Request Start	Time until the document request can be sent. This includes DNS, connect, SSL, and URL redirects.	`window.performance.requestStart - window.performance.navigationStart`
Document Fetch Done	In addition to everything until Request Start, this includes the time until the document (HTML) completely downloads from the HTTP server.	`window.performance.responseEnd - window.performance.navigationStart`
Document Interactive	Time until the document becomes interactive. This includes all steps completed for Document Fetch Done.	`window.performance.domInteractive - window.performance.navigationStart`

The following example timeline illustrates the measures described above.

Visualization of Measures



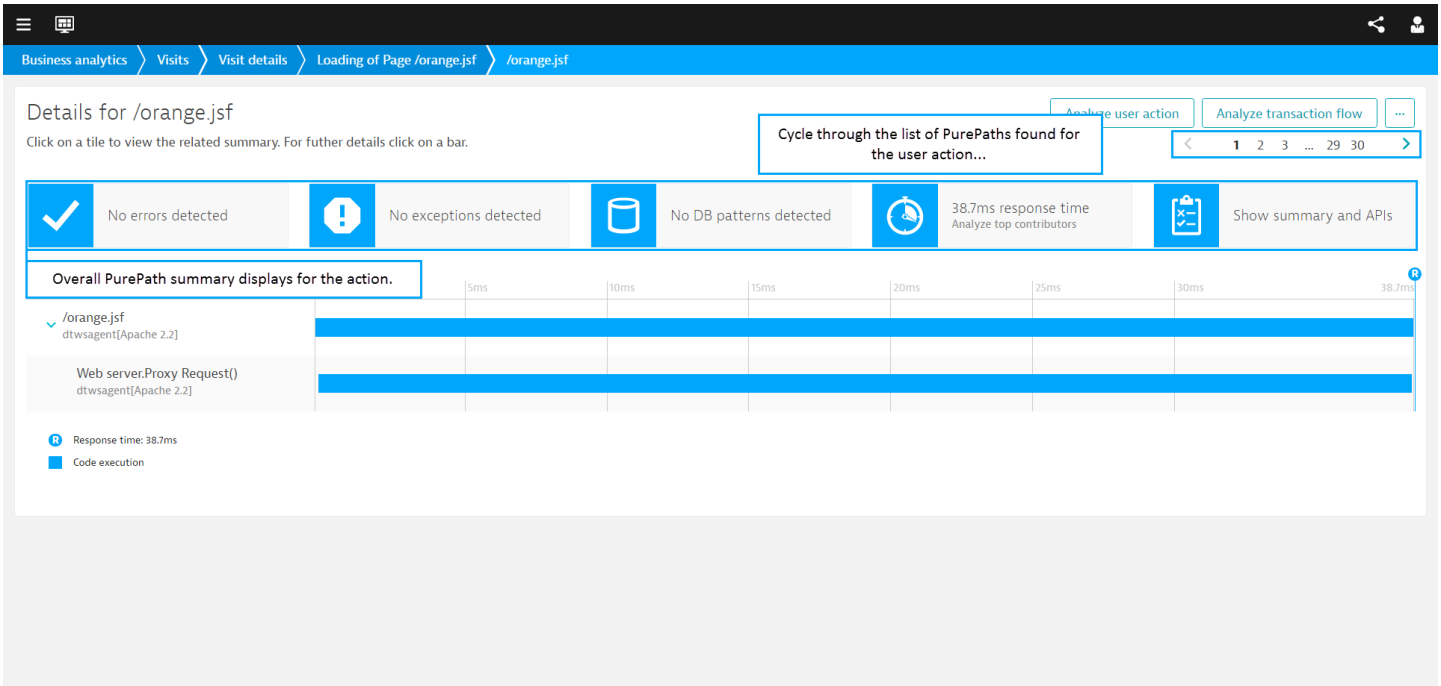
You can use the waterfall to easily identify resource loading bottlenecks.



Analyze PurePaths and drilldown

The waterfall gives you full visibility from end user down to the Purepath.

When analyzing an action, you can either analyze PurePaths for the whole action or the Purepaths for an individual resource. To see all the Purepaths, click the top bar in the waterfall page load timeline, then click **Analyze all PurePaths** for an overall PurePath summary and to cycle through the list of PurePaths found for that action.



When analyzing a Purepath, you can click on any request in listed for the resource to see down to the method and code level data.

Monitor user satisfaction

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

Web dashboards

Use top findings to analyze user visits

Trace user activity through all channels

Monitor top landing and exit pages

Find specific user visits

Visits dashlet

PureLytics

PureLytics Stream

Overview

Appmon User Satisfaction uses PureLytics data to provide the full context of user experience for your applications. Use the User satisfaction data to:

- See at-a-glance the current state of how application performance is perceived by its users.
- Quickly and efficiently filter millions of user visits using intuitive and visual filters to help you precisely identify problems.
- Search for specific visits and pass visit details along to support or engineering to accelerate problem resolution.
- Drill down from visits to see the full visit PurePath data in the AppMon Client.

Access the User Action data using any of the following:

- AppMon Web: Select **User Analytics** from the navigation pane.
- AppMon Client or AppMon Webstart Client: Find user satisfaction data in any of the following:
  - Add a World Map Dashlet to any dashboard.
  - In the Start Center, select **User Experience > User Experience World Map** for user satisfaction data by geographic region.
  - The Visits dashboard, available by selecting Visits in the Add Dashlet dialog box or by selecting **Diagnose User Experience > Visits** for a selected System Profile.

User Satisfaction tile in AppMon Web

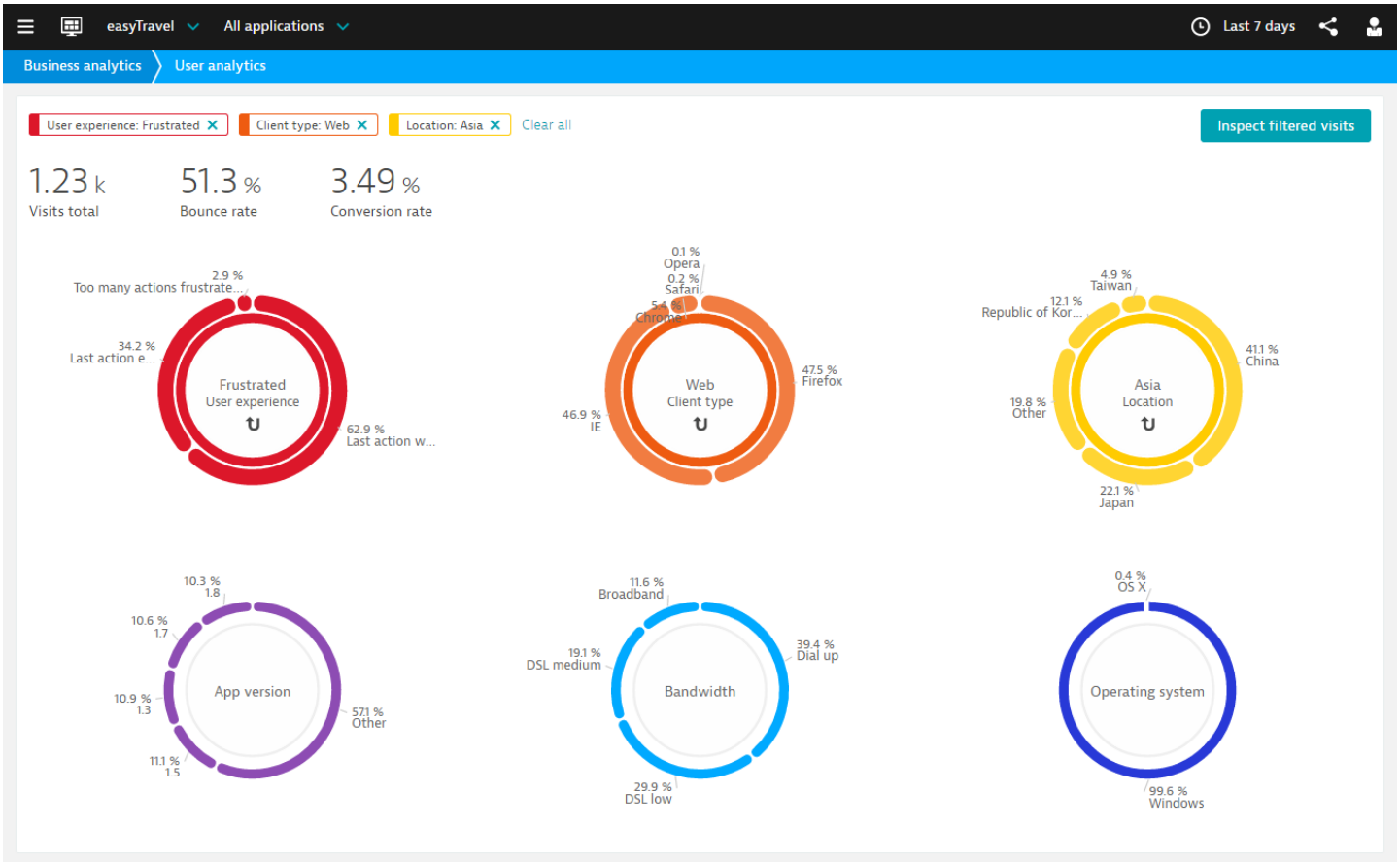
From the User satisfaction tile in AppMon Web, you can drill down to open more informative views that help you quickly understand and respond to user experience performance issues.

Filter user visits for precise problem identification

Clicking the **User satisfaction** tile opens the full screen **User satisfaction** view from which you can filter by the following visit properties:

- User experience index
- Client type
- Location
- Application version
- Bandwidth
- Operating system

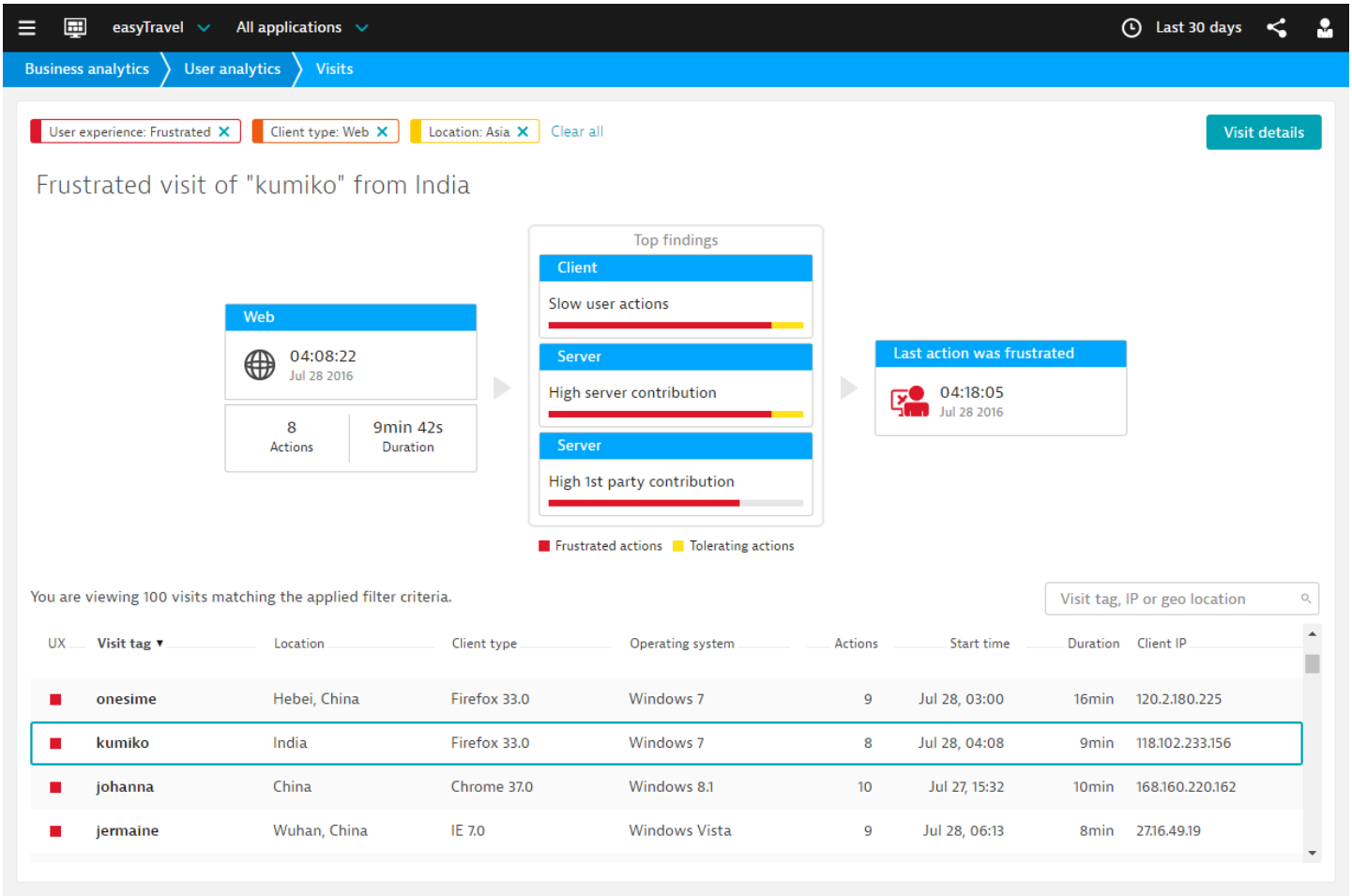
To select a filter, click the section of the ring that you want to filter by. Selected filters display at the top of the screen and the visits total, bounce rate, and conversion rate update instantly. For example, in the following screenshot, all user visits made in the last 24 hrs for the easyTravel system profile are filtered by **Frustrated** users, **Web** clients, and the **Asia** region with the total of 154 visits, a 40.3 percent bounce rate, and a 26 percent conversion rate displayed.



Check the filtered visits

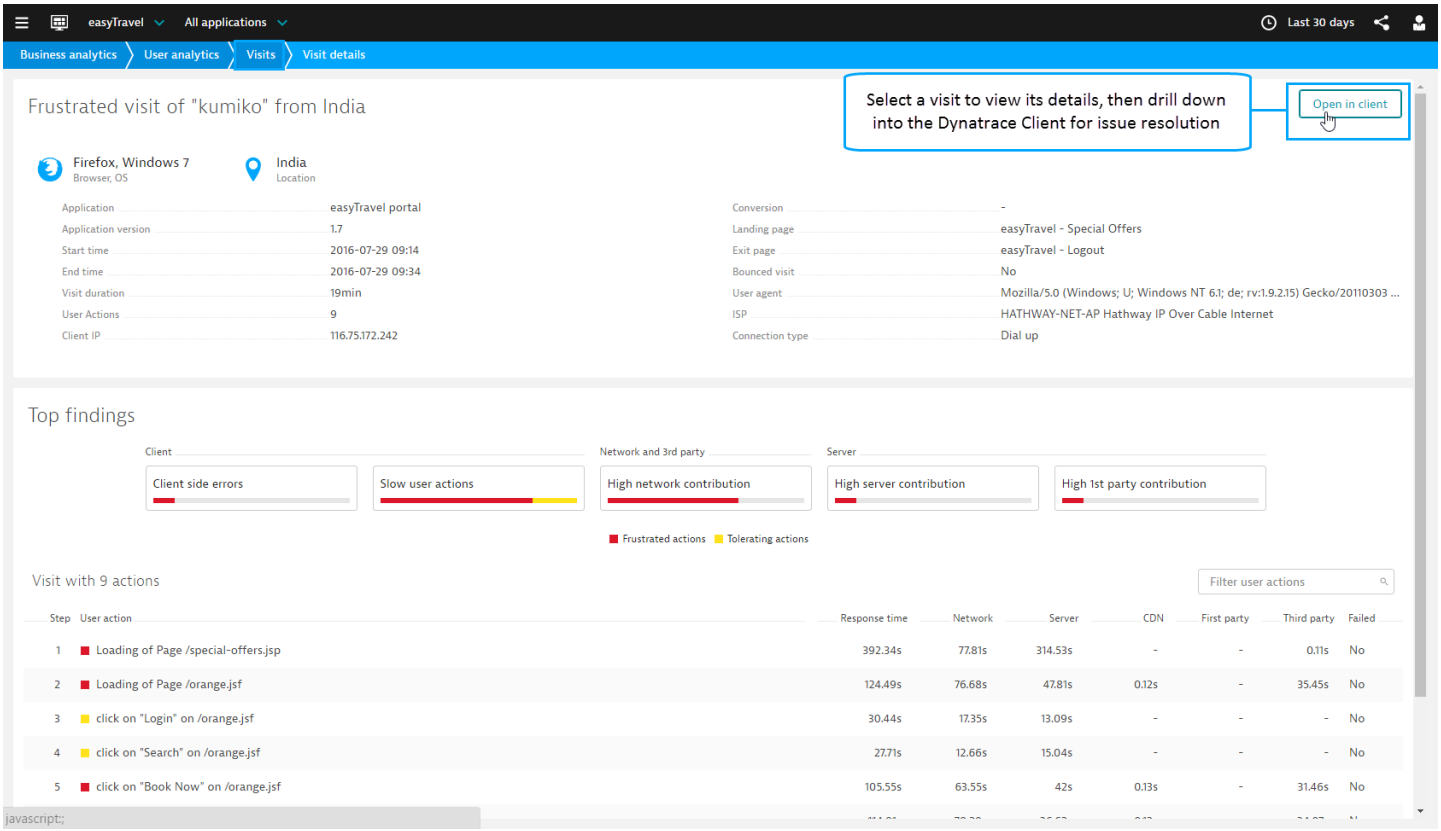
In addition to seeing the total number of visits, bounce rate, and conversion rate for the selected filters, you can click the **Inspect filtered visits** button to view the specific details of up to 50 visits that match the selected filter(s).

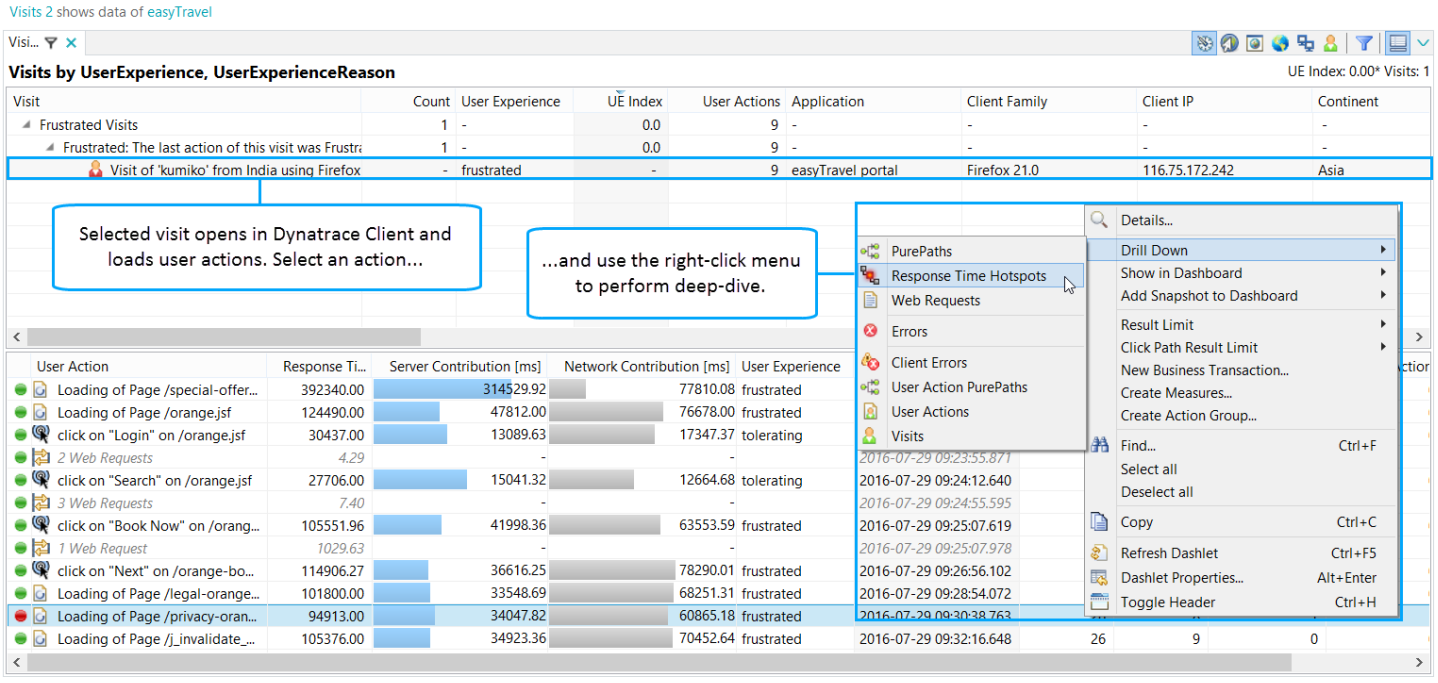
> **Note**
>
> The Visits list is **not the entire list of visits** that match the selected filter(s), but a **random sampling** of up to 50 matching visits.



Drill down in the AppMon Client

You can further scrutinize a visit by selecting it and clicking the **Open in client** button to analyze visit details in the AppMon Client. For visits, you can check performance, click path and success details for each user action that made up the visit. From there, you can focus on specific steps for drill down activities. See Visits Dashlet for more information.


easyTravel
All applications
Last 30 days
Business analytics
User analytics
Visits
Visit details
Frustrated visit of "kumiko" from India
Select a visit to view its details, then drill down into the Dynatrace Client for issue resolution
Open in client
Firefox, Windows 7
Browser, OS
India
Location
Application easyTravel portal
Application version 1.7
Start time 2016-07-29 09:14
End time 2016-07-29 09:34
Visit duration 19min
User Actions 9
Client IP 116.75.172.242
Conversion -
Landing page easyTravel - Special Offers
Exit page easyTravel - Logout
Bounced visit No
User agent Mozilla/5.0 (Windows; U; Windows NT 6.1; de; rv:1.9.2.15) Gecko/20110303 ...
ISP HATHWAY-NET-AP Hathway IP Over Cable Internet
Connection type Dial up
Top findings
Client
Client side errors
Slow user actions
Network and 3rd party
High network contribution
Server
High server contribution
High 1st party contribution
Frustrated actions
Tolerating actions
Visit with 9 actions
Filter user actions
Step User action Response time Network Server CDN First party Third party Failed
1 Loading of Page /special-offers.jsp 392.34s 77.81s 314.53s - - 0.11s No
2 Loading of Page /orange.jsf 124.49s 76.68s 47.81s 0.12s - 35.45s No
3 click on "Login" on /orange.jsf 30.44s 17.35s 13.09s - - - No
4 click on "Search" on /orange.jsf 27.71s 12.66s 15.04s - - - No
5 click on "Book Now" on /orange.jsf 105.55s 63.55s 42s 0.13s - 31.46s No
javascript:;


Visits 2 shows data of easyTravel
Visits by UserExperience, UserExperienceReason
UE Index: 0.00* Visits: 1
Visit Count User Experience UE Index User Actions Application Client Family Client IP Continent
Frustrated Visits 1 - 0.0 9 - - - -
Frustrated: The last action of this visit was Frustra 1 - 0.0 9 - - - -
Visit of 'kumiko' from India using Firefox - frustrated - 9 easyTravel portal Firefox 21.0 116.75.172.242 Asia
Selected visit opens in Dynatrace Client and loads user actions. Select an action...
...and use the right-click menu to perform deep-dive.
User Action Response Ti... Server Contribution [ms] Network Contribution [ms] User Experience
Loading of Page /special-offer... 392340.00 314529.92 77810.08 frustrated
Loading of Page /orange.jsf 124490.00 47812.00 76678.00 frustrated
click on "Login" on /orange.jsf 30437.00 13089.63 17347.37 tolerating
2 Web Requests 4.29
click on "Search" on /orange.jsf 27706.00 15041.32 12664.68 tolerating
3 Web Requests 7.40
click on "Book Now" on /orang... 105551.96 41998.36 63553.59 frustrated
1 Web Request 1029.63
click on "Next" on /orange-bo... 114906.27 36616.25 78290.01 frustrated
Loading of Page /legal-orange... 101800.00 33548.69 68251.31 frustrated
Loading of Page /privacy-oran... 94913.00 34047.82 60865.18 frustrated
Loading of Page /j_invalidate_... 105376.00 34923.36 70452.64 frustrated
2016-07-29 09:23:55.871
2016-07-29 09:24:12.640
2016-07-29 09:24:55.595
2016-07-29 09:25:07.619
2016-07-29 09:25:07.978
2016-07-29 09:26:56.102
2016-07-29 09:28:54.072
2016-07-29 09:30:38.763
2016-07-29 09:32:16.648 26 9 0
PurePaths
Response Time Hotspots
Web Requests
Errors
Client Errors
User Action PurePaths
User Actions
Visits
Details...
Drill Down
Show in Dashboard
Add Snapshot to Dashboard
Result Limit
Click Path Result Limit
New Business Transaction...
Create Measures...
Create Action Group...
Find... Ctrl+F
Select all
Deselect all
Copy Ctrl+C
Refresh Dashlet Ctrl+F5
Dashlet Properties... Alt+Enter
Toggle Header Ctrl+H

Monitor streaming media activity

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Display Streaming Media Data

Streaming media measures

Business transacton evaluation/filter/splitting values

Streaming Media dashboard

Troubleshooting

Limitations

Related pages

User Action timings

Third-party content analysis

User Action timings

W3C resource timing metrics

Configure geographic location

Watch and learn

Monitoring streaming media activity

External links

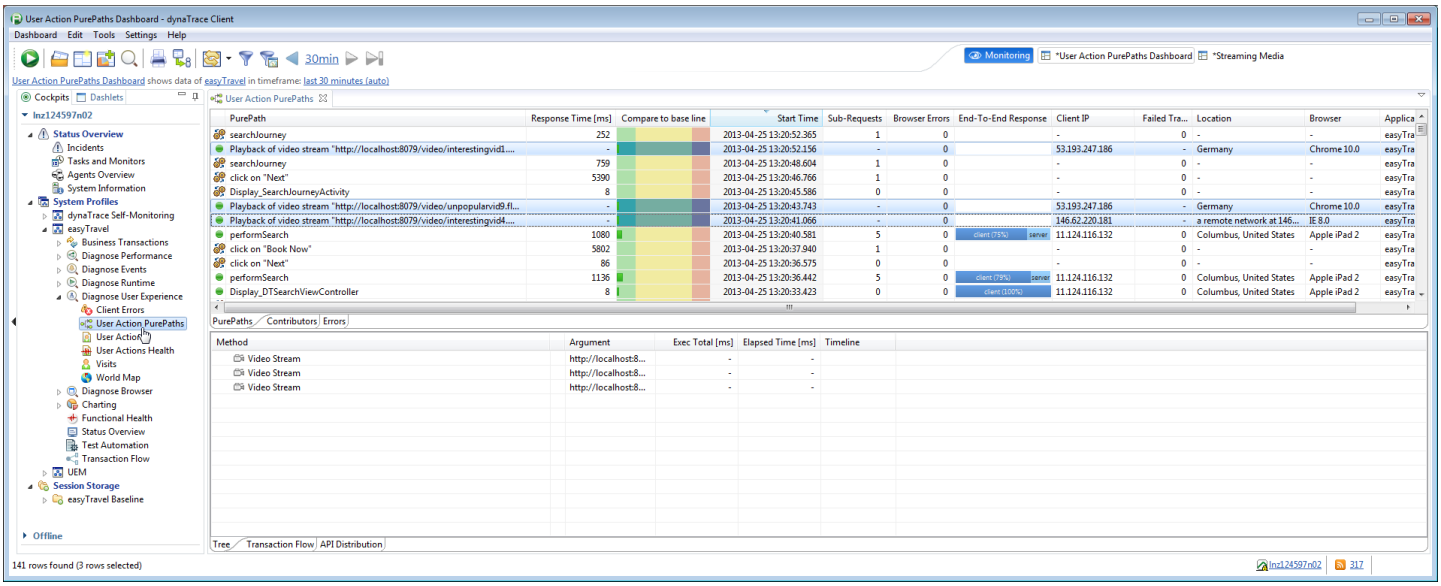
Audio support by browser

Video support by browser

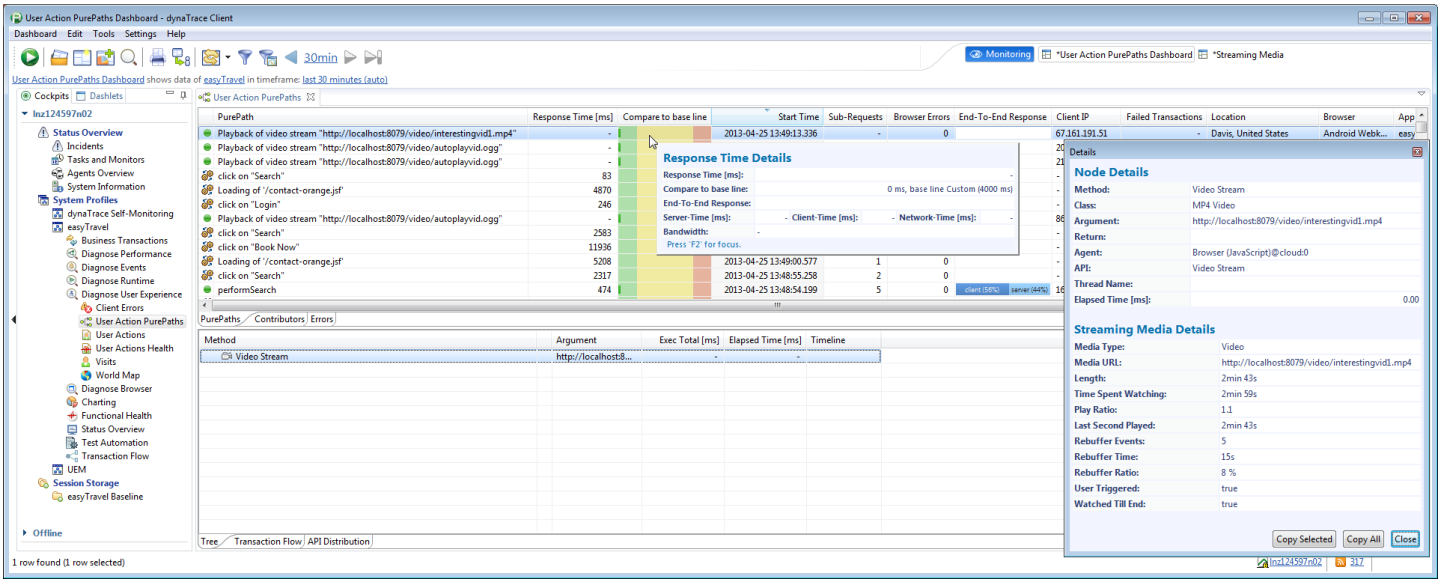
Display Streaming Media Data

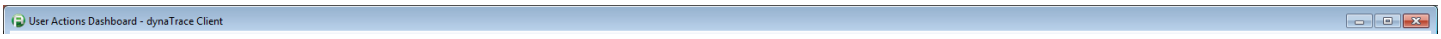
The streaming media information appears in the **User Action PurePath** dashboard as PurePaths, starting with **Playback of** followed by the video stream URL. These PurePaths only have a single action node for the related media stream.

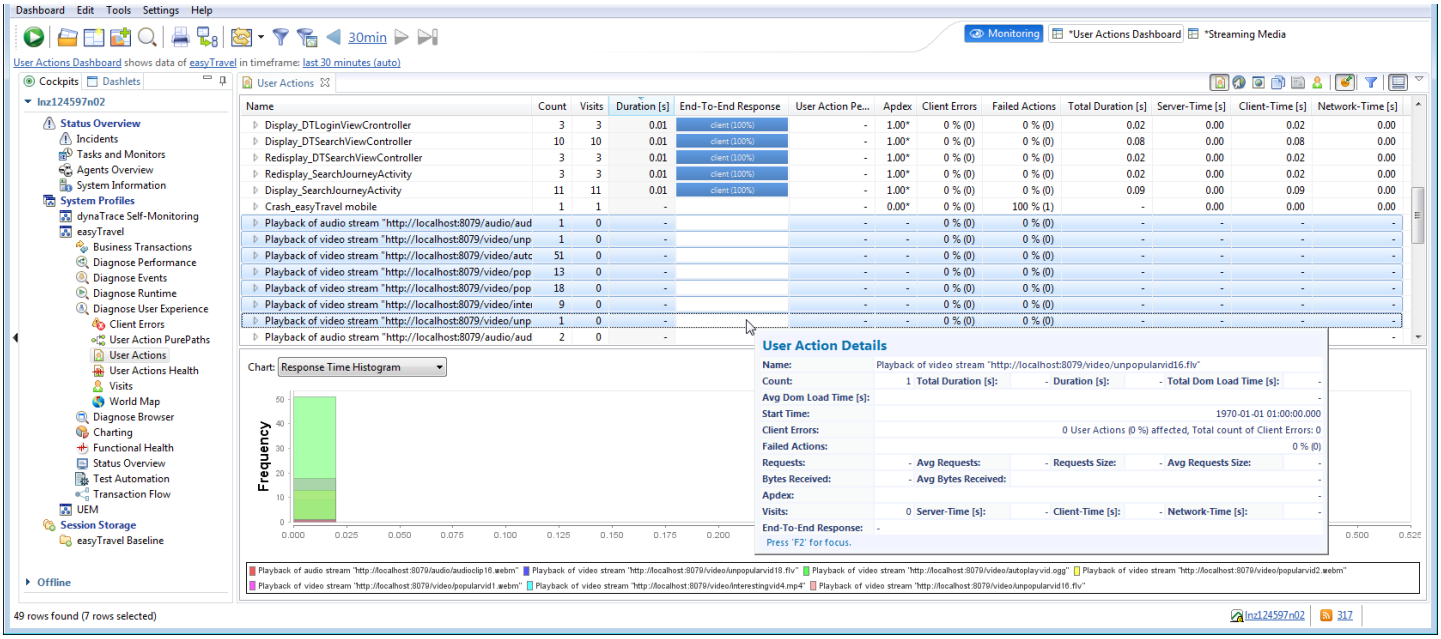
User Actions PurePath Dashboard

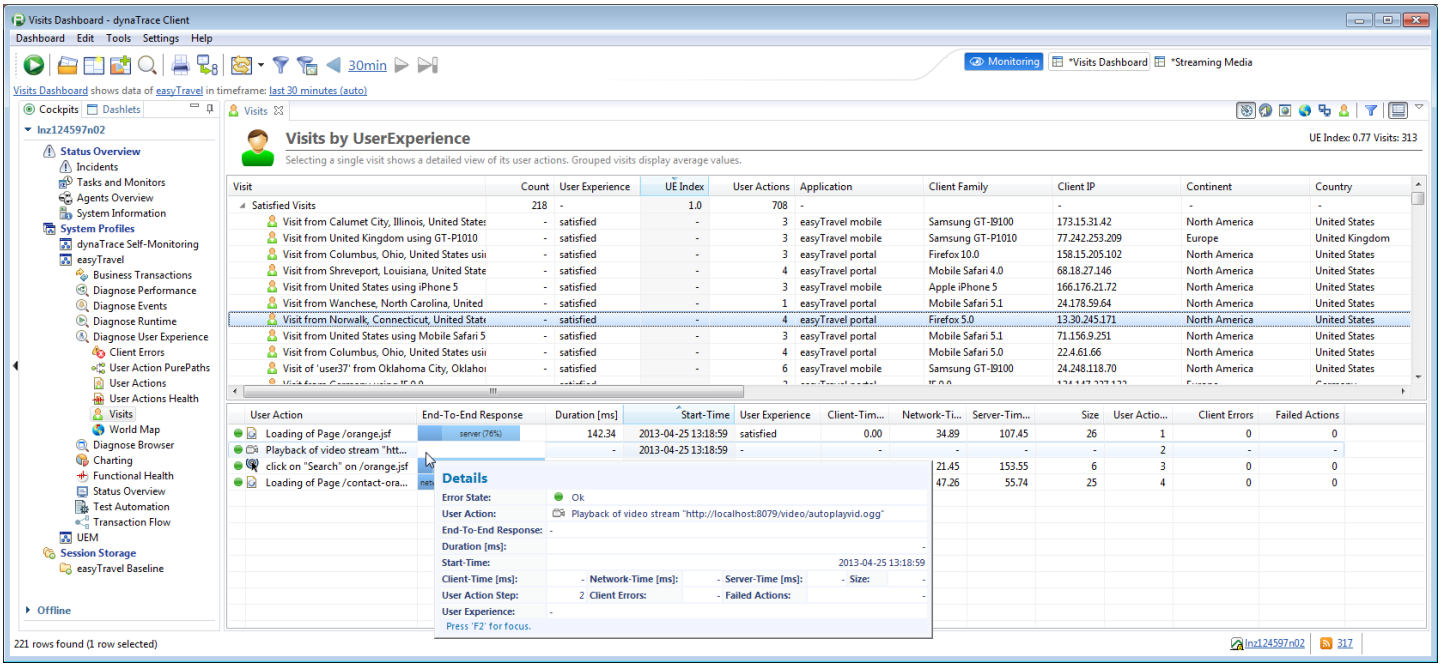


Right-click on the media stream action and select **Details**, or hover over a line to see the details of the media stream. The streaming media node also appears on the **User Actions** dashlet and the **Visits** dashlet.









Streaming media measures

After you enable the streaming media functionality, the following measures are available based on streaming media metrics:

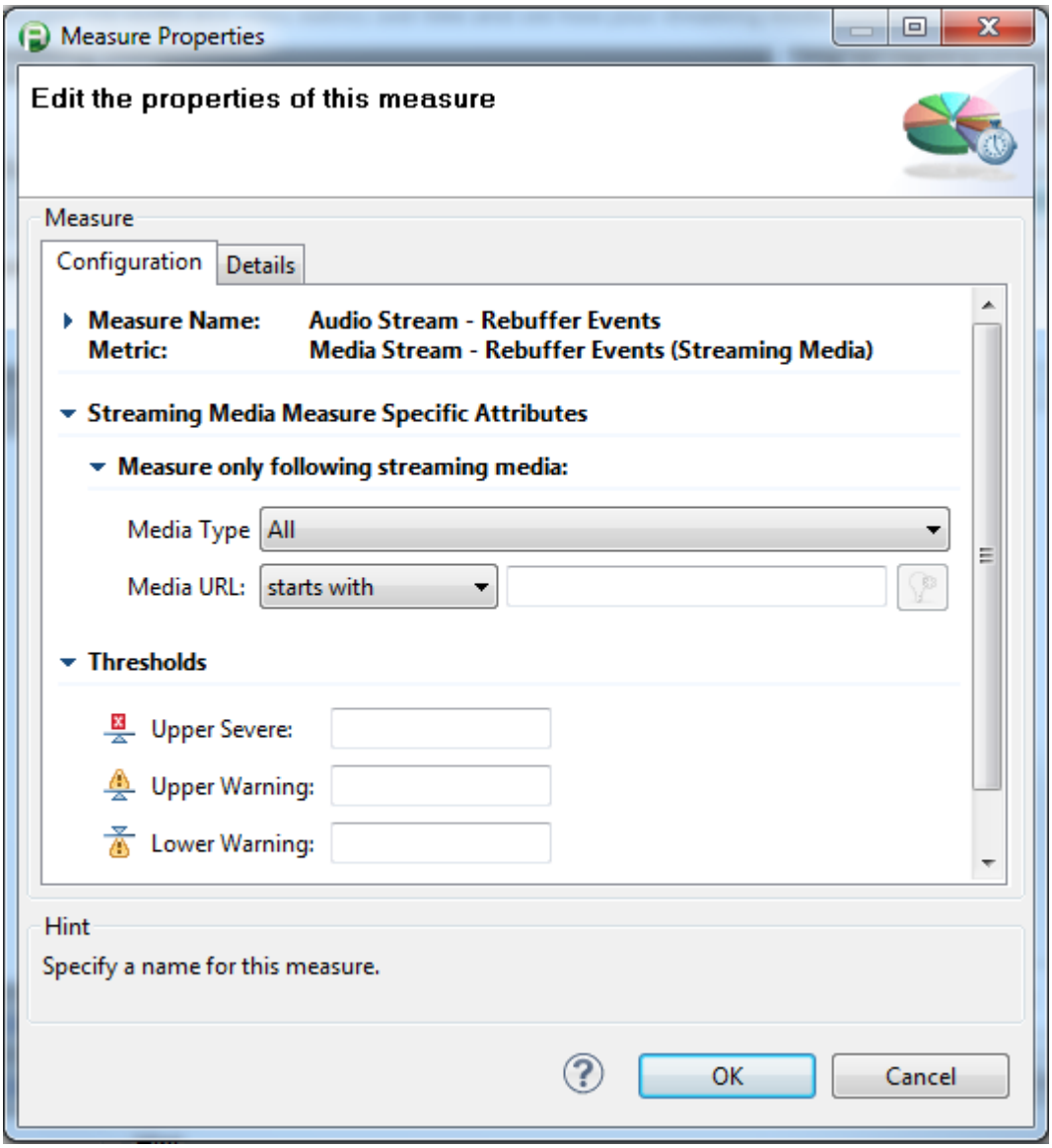
Name	Description
Audio Stream - Count	The number of times an audio stream is played
Audio Stream	

Audio Stream - Last second played	How far (by percentage) the last listened to second is in the streaming audio. The percentage of the audio stream skipped is not taken in account.
Audio Stream - Length	Length of streaming audio items (Time it takes to watch from start to finish once)
Audio Stream - Play Ratio	The number of times a streaming audio item is consumed during one session (compared to its length). A play / length ratio higher than one = Stream, or part of it, is consumed multiple times. A play / length ratio lower than one = Stream isn't watched till the end, or parts of the streams are skipped.
Audio Stream - Rebuffer Events	The number of times the user waits for the audio buffering before playback can resume.
Audio Stream - Rebuffer Ratio	The percentage of the time the user waits for buffering out of the entire audio stream watching time.
Audio Stream - Rebuffer Time	The time the user waits until more of the audio is buffered before playback can resume.
Audio Stream - Time Spent Listening	The time spent consuming an audio stream. This can exceed the length if the user watches all or parts of the media multiple times.
Video Stream - Count	The number of times a video stream is played
Video Stream - Last second watched	How far (by percentage) the last watched second is into the streaming video. Percentage of the video stream skipped not taken in account.
Video Stream - Length	The length of the streaming video. The time it takes to watch from start to finish once.
Video Stream - Play Ratio	The number of times a streaming video is consumed during one session. A play / length ratio higher than one = Stream, or part of it, is consumed multiple times. A play / length ratio lower than one = Stream isn't watched till the end, or parts of the streams are skipped.
Video Stream - Rebuffer Events	The number of times the user waits for the buffering of the movie before playback can resume.
Video Stream - Rebuffer Ratio	The percentage of the time the user waits for buffering out of the entire video stream watching time.
Video Stream - Rebuffer Time	The time the user waits until more of the movie is buffered before playback can resume.
Video Stream - Time Spent Listening	The time spent watching a video stream. This can exceed the length, if the user watches all, or parts, of the video multiple times.

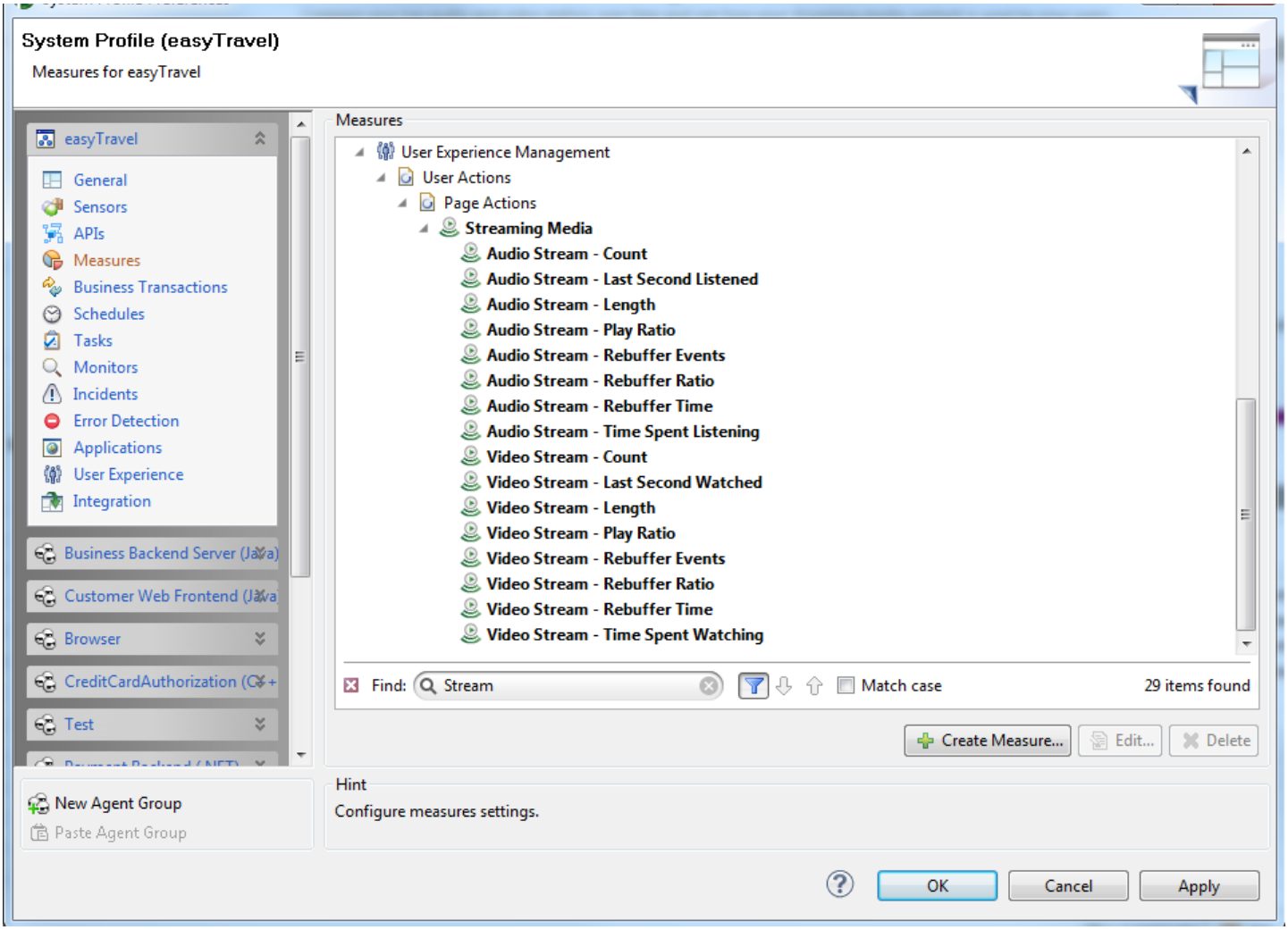
Additional Measures can be created based on user preferences. For each Measure, you can limit the

Additional Measures can be created based on user preferences. For each Measure, you can limit the following:

- The values counted to a certain type (for example, all/audio/video).
- The values based on the media stream URI (for example, only count media items delivered from a certain folder).





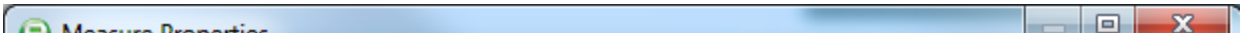


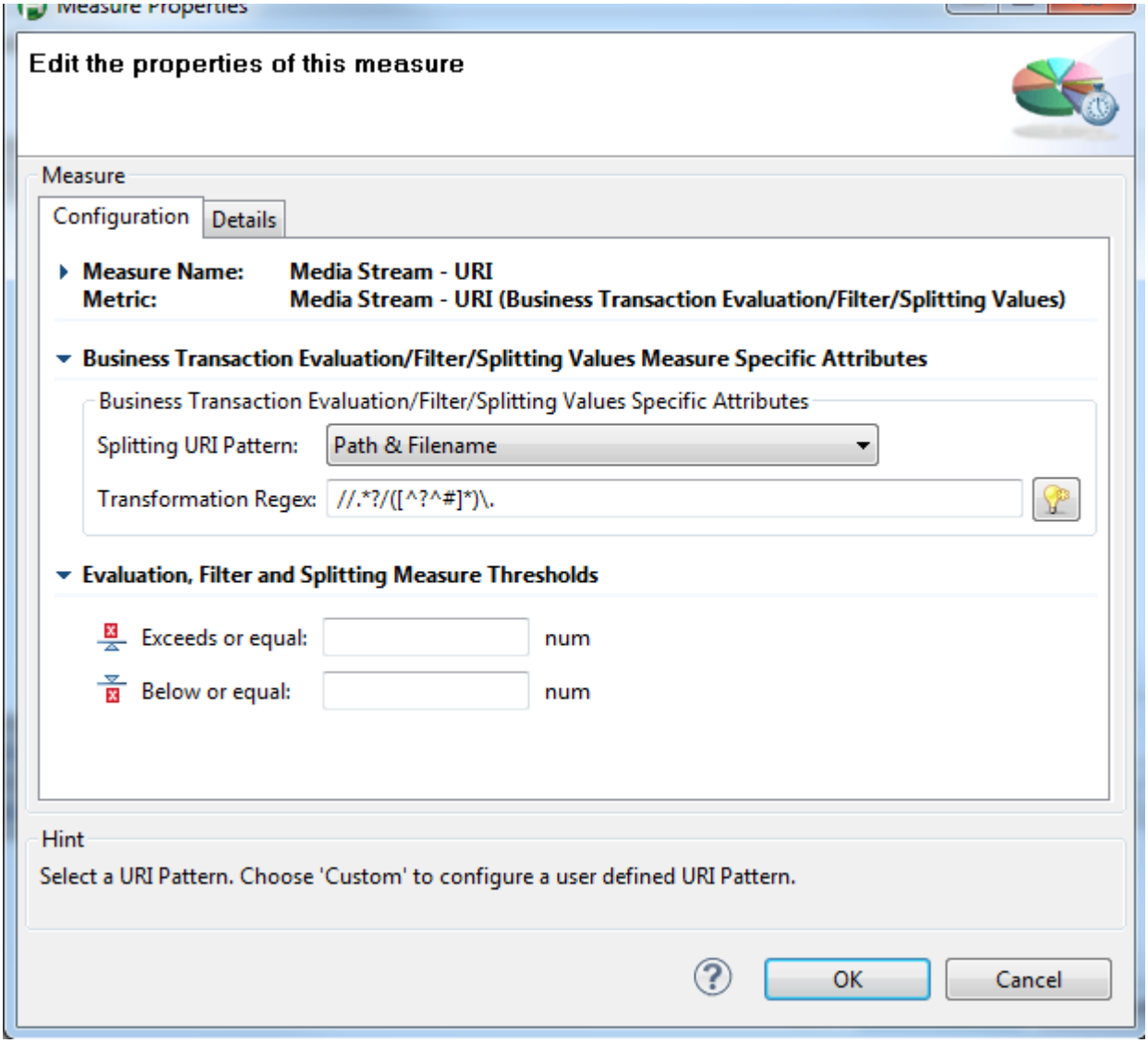
Business transacton evaluation/filter/splitting values

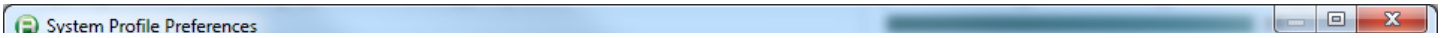
Name	Description
Media Stream - URI	Represents the URI of the media stream

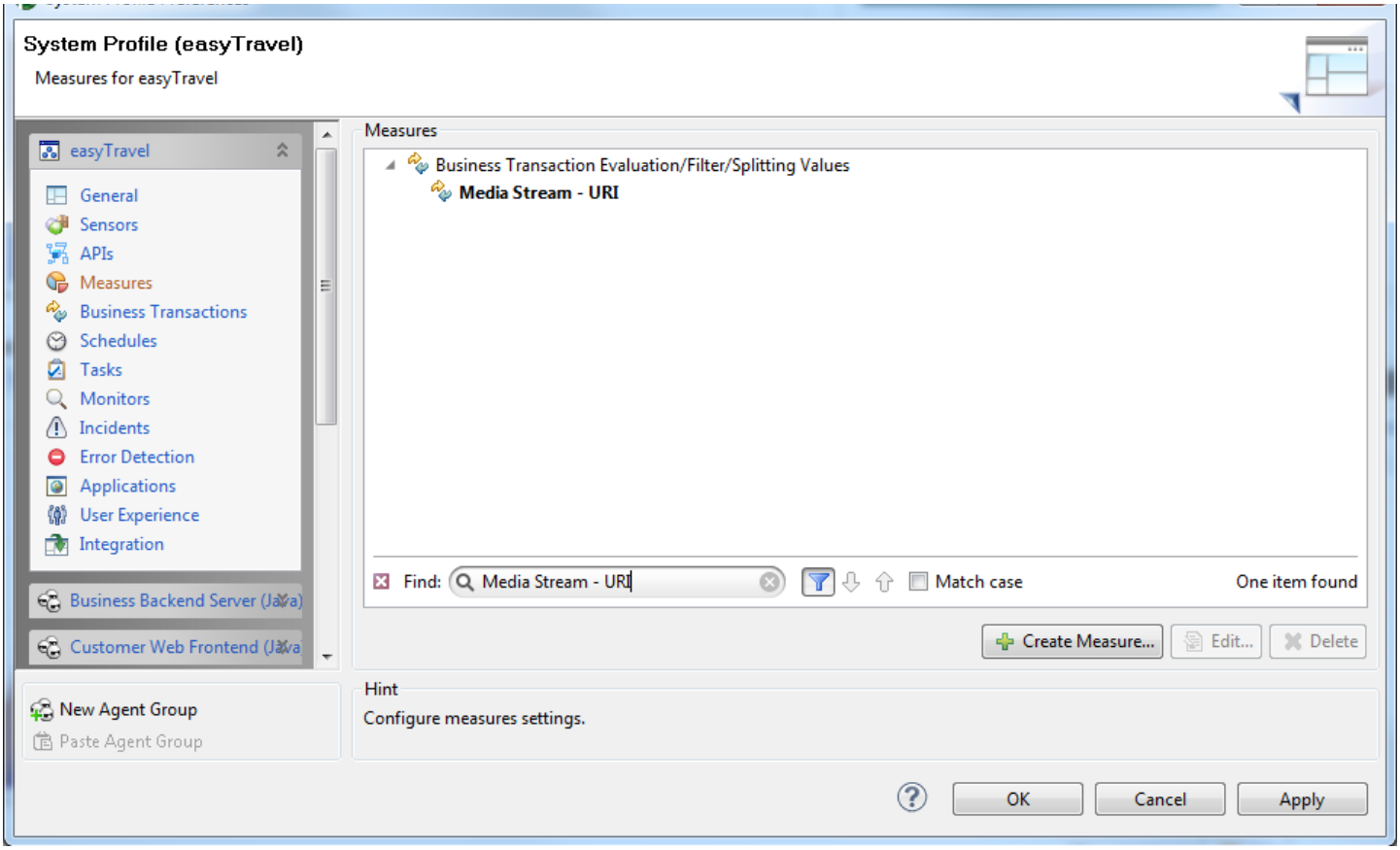
You can configure the splitting Measure to transform the media URI based on a regular expression. Use this to unify different files or to shorten the URLs.

By default the path information and the file name is used for the splitting value. AppMon discards the domain for brevity and the file ending to unify different formats for different browsers.



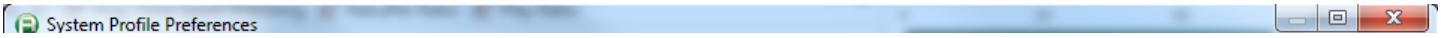

Measure Properties
Edit the properties of this measure
Measure
Configuration
Details
Measure Name: Media Stream - URI
Metric: Media Stream - URI (Business Transaction Evaluation/Filter/Splitting Values)
Business Transaction Evaluation/Filter/Splitting Values Measure Specific Attributes
Business Transaction Evaluation/Filter/Splitting Values Specific Attributes
Splitting URI Pattern: Path & Filename
Transformation Regex: //.*?/([^?^#]*)\.
Evaluation, Filter and Splitting Measure Thresholds
Exceeds or equal: num
Below or equal: num
Hint
Select a URI Pattern. Choose 'Custom' to configure a user defined URI Pattern.
OK
Cancel

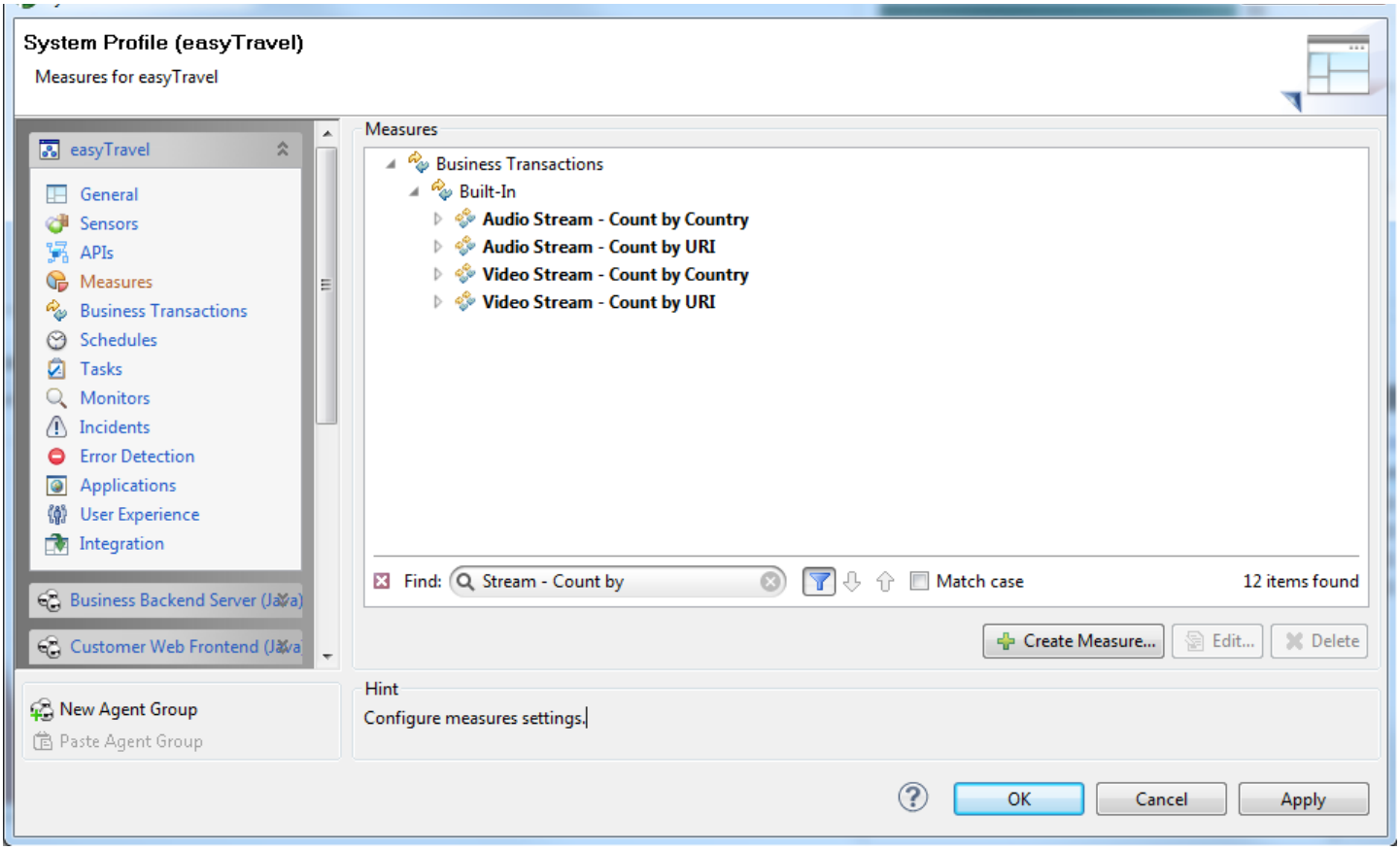

System Profile Preferences



Business Transactions

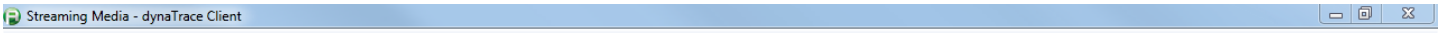
Name	Description
Audio Stream - Count by Country	The count of audio streams consumed split by country of visitor
Audio Stream - Count by URI	The count of audio streams watched split by audio stream URI
Video Stream - Count by Country	The count of video streams watched split by country of visitor
Video Stream - Count by URI	The count of video streams watched split by video stream URI

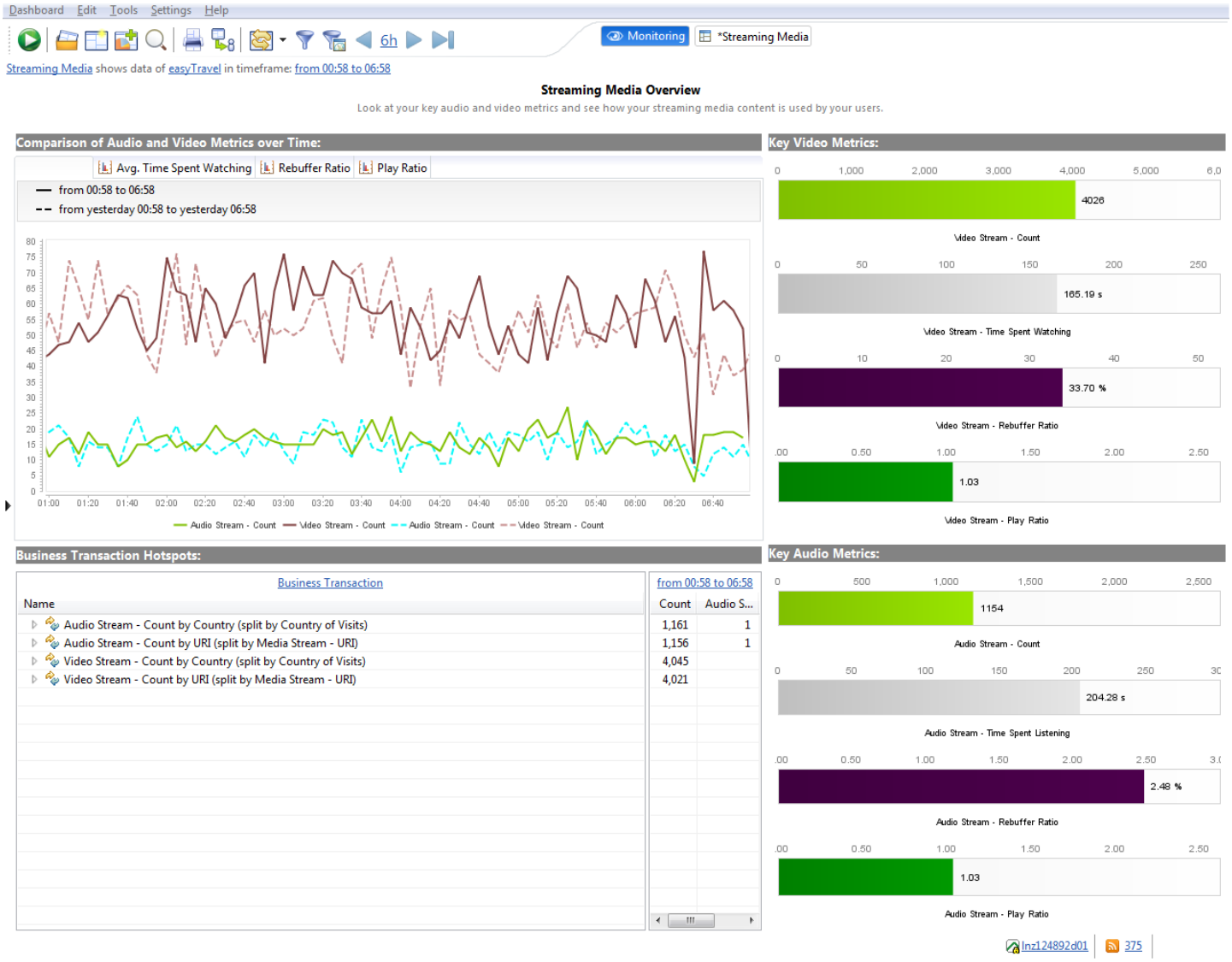


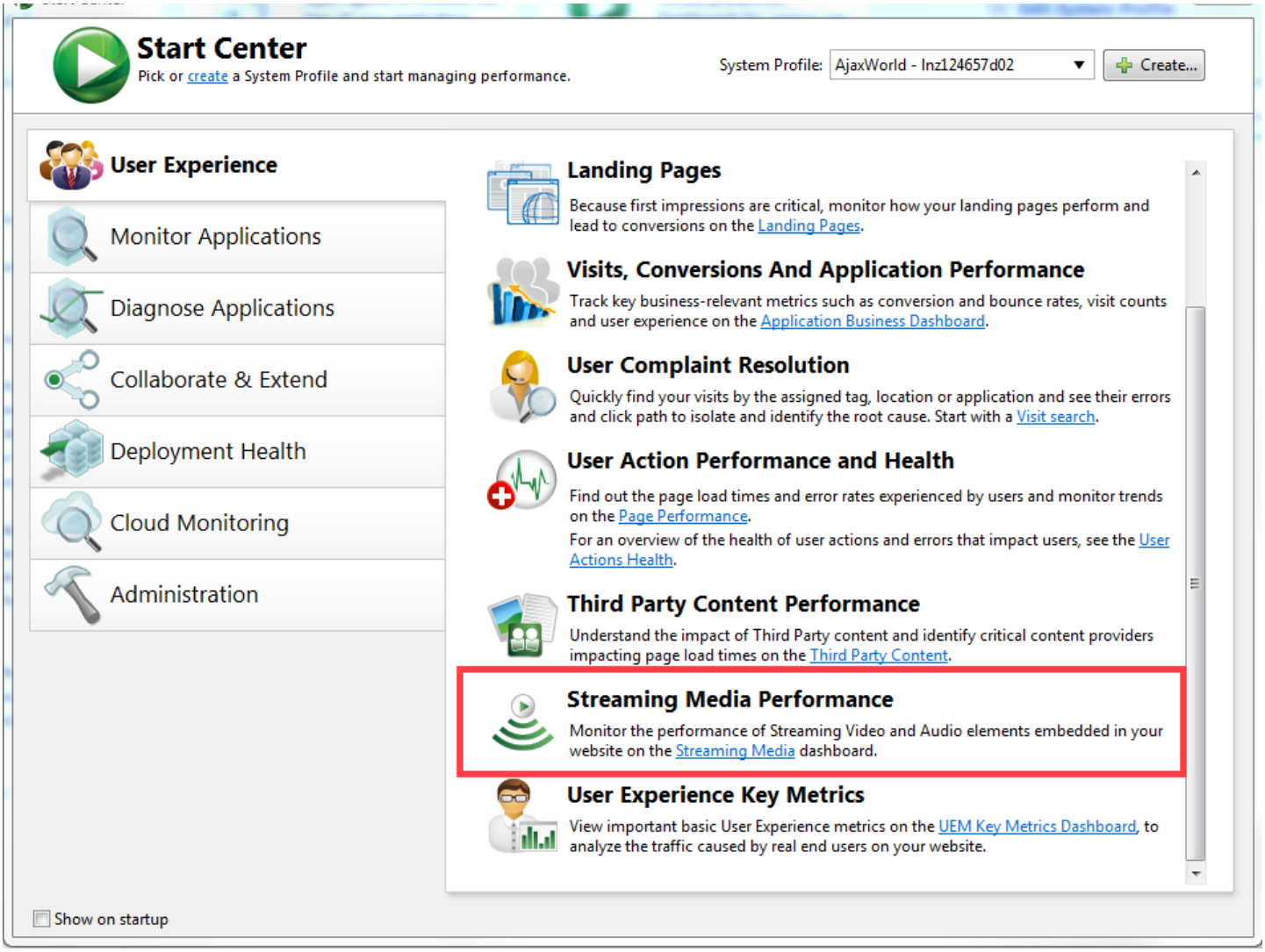


Streaming Media dashboard

The Streaming Media dashboard shows information about media streams and their performance. Click **Start Center > User Experience > Streaming Media Performance** and click the Streaming Media link to open the dashboard.




Dashboard Edit Tools Settings Help
6h
Monitoring
*Streaming Media
Streaming Media shows data of easyTravel in timeframe: from 00:58 to 06:58
Streaming Media Overview
Look at your key audio and video metrics and see how your streaming media content is used by your users.
Comparison of Audio and Video Metrics over Time:
Avg. Time Spent Watching
Rebuffer Ratio
Play Ratio
from 00:58 to 06:58
from yesterday 00:58 to yesterday 06:58
Audio Stream - Count
Video Stream - Count
Audio Stream - Count
Video Stream - Count
Key Video Metrics:
4026
Video Stream - Count
165.19 s
Video Stream - Time Spent Watching
33.70 %
Video Stream - Rebuffer Ratio
1.03
Video Stream - Play Ratio
Business Transaction Hotspots:
Business Transaction
from 00:58 to 06:58
Name
Count
Audio S...
Audio Stream - Count by Country (split by Country of Visits) 1,161 1
Audio Stream - Count by URI (split by Media Stream - URI) 1,156 1
Video Stream - Count by Country (split by Country of Visits) 4,045
Video Stream - Count by URI (split by Media Stream - URI) 4,021
Key Audio Metrics:
1154
Audio Stream - Count
204.28 s
Audio Stream - Time Spent Listening
2.48 %
Audio Stream - Rebuffer Ratio
1.03
Audio Stream - Play Ratio
lnz124892d01
375
Start Center



Troubleshooting

The following lists some common reasons why no streaming media information is collected:

Streaming plugin not enabled

The JavaScript Agent Streaming plugin is not enabled, or is enabled for the wrong application. Verify this by looking at the JavaScript Agent URL that is injected into the source code of the page with a media stream. The hash of features has to contain an “s”.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xh
```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xh
<html xmlns="http://www.w3.org/1999/xhtml"><head>
        <title>easyTravel - One step to happiness</title>

        <meta content="text/html; charset=UTF-8" http-equiv="Content-Type" />

        <meta content="INDEX, FOLLOW" name="robots" />

        <meta content="dynaTrace Software GmbH" name="author" />

        <script type="text/javascript" src="/dtagentdbg55_bijnps3_5176.js" data-dtconfi
        <link href="img/favicon_orange_plane.png" rel="apple-touch-icon" />

        <link href="css/BaseProd.css" rel="stylesheet" type="text/css" />
        <link href="css/footer.css" rel="stylesheet" type="text/css" />
        <link href="css/rime.css" rel="stylesheet" type="text/css" />
        <link href="css/rating.css" rel="stylesheet" type="text/css" />
```

Media streams are embedded using Flash or Silverlight

When media items on the page are not embedded using HTML5 audio/video tags, the JavaScript Agent cannot detect them. In this case the JavaScript ADK must be used to collect the information about the media streams.

Limitations

Opera 11 does not report correct timestamps

Opera 11 does not provide timestamps for events fired by audio or video elements. The values for Opera 11 might not be as accurate as in other browsers if a large amount of JavaScript code is executed on a page.

Very short buffer periods can not be detected

Very short buffer periods cannot be detected, because some browsers do not behave as specified in the standard.

Older versions of chrome report wrong values

When a lot of buffering occurs, older versions of Chrome report wrong values. Chrome reports that the

When a lot of buffering occurs, older versions of Chrome report wrong values. Chrome reports that the current position of playback is much larger than the actual duration of a video. For example, the current progress value is reported to be 90 seconds, but the actual duration of the video is only 27 seconds. These wrong values can not be filtered out, because Chrome does not know the actual duration due to heavy buffering.

Trace user activity through all channels

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

Monitoring Overview dashboard

Monitor user satisfaction

Use top findings to analyze user visits

Monitor top landing and exit pages

Find specific user visits

Overview

Use the **Omni channel** tile on the **Operations** dashboard to understand the distribution of your users and their experience by channel - web, mobile web, and mobile app. This provides an instant **head-to-head comparison** for the most important channel metrics, as well as a Time Frame count of non-human-generated visits.

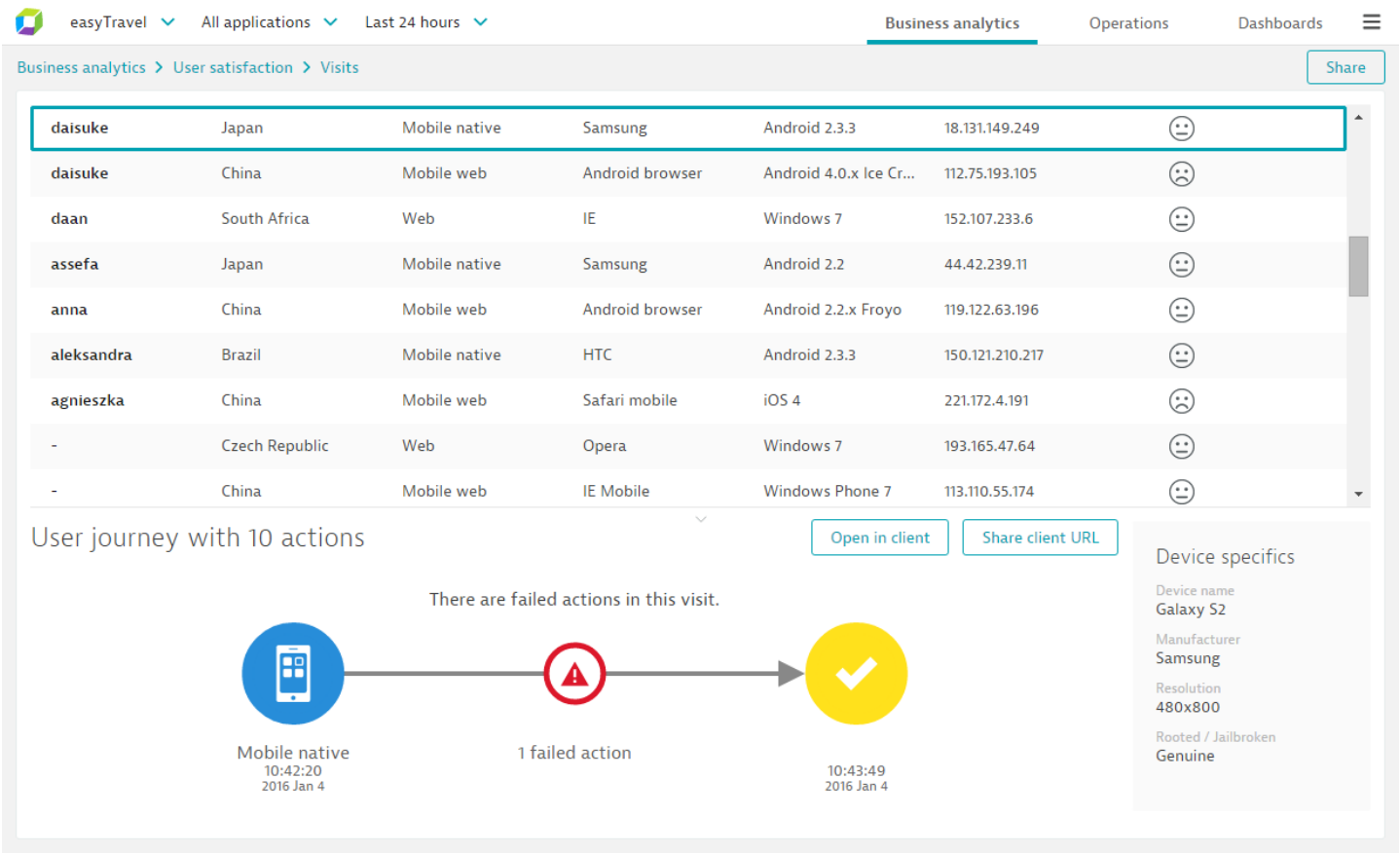
The corresponding Omni channel detail view charts other key user experience metrics, such as the total number of visits, the user experience index, conversion rate, and bounce rate by channel. Metrics display for both selected Time Frame and for the previous week.



Reviewing landing pages and exit pages

You can check the breakdown of the landing pages and exit pages for the visits breakdown in the OmniChannel detail view. This gives you an understanding of how your users work through the application, the application performance they experience, and whether users are interacting with the application as expected. See Monitor Top Landing and Exit Pages for more information.

Find specific user visits

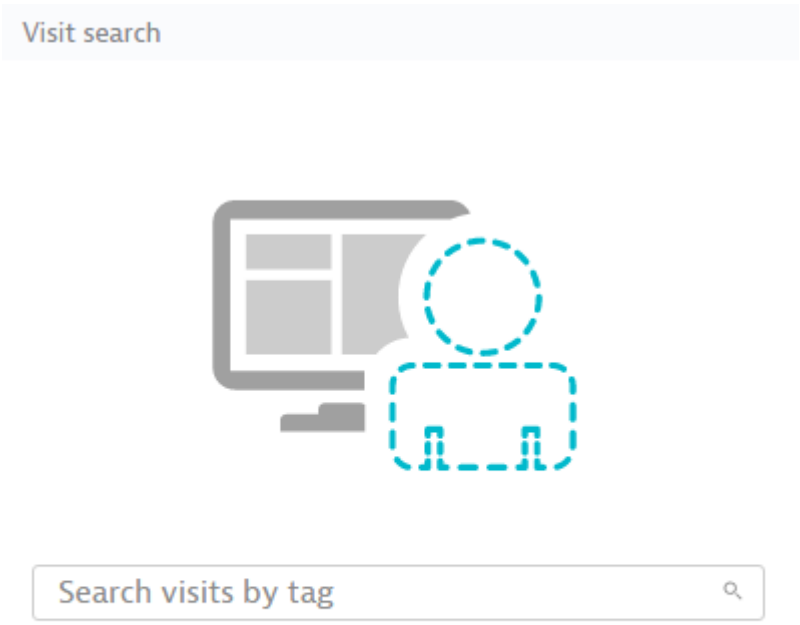


From the Visits view, you can also open the AppMon Client and drill down to view PurePath data and specific user action details of the selected visit. Learn more about the Drilldown to the AppMon Client.

> **Note**
>
> You must set up a visit tag in UEM configuration for the System Profile to find visits in a search based on user visit values. Assign a tag to a visit by creating a measure that extracts data, such as a user name, from an HTTP parameter or method argument of a PurePath. See System Profile - User Experience for configuration details.

You can also search from the Visits view in the User Satisfaction tile. Click the User Satisfaction tile to open the User Satisfaction view, then click the **Inspect filtered visits** button at the top right of the page to open the visits view. Type a visit tag in the search box and press **Enter** to find visits based on the tag.

The Visits view appears and displays up to 50 matching visits.



The Business Analytics dashboard on the AppMon Web Home screen includes a tile that lets you find specific user visits. To find a specific user visit, type in the tagged visit value in the search box in the Visit search tile and press **Enter**.

Web performance optimization for UEM

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Related pages

System Profile - User Experience

Java Agent configuration

W3C resource timing metrics

Mobile UEM

Visually complete metric

Overview

You can set up User Experience in AppMon to obtain data that helps developers optimize Web performance in your applications.

Users who formerly used the Browser Diagnostics native add-on for Windows Internet Explorer and Firefox can use this solution on any browser that executes JavaScript to capture UEM visit data and W3C resource timings and actions implemented with common Java frameworks, covering every visitor end-to-end. Capture W3C resource timings to provide full waterfall details for each visit that is set to send waterfall details for analysis. Capture visually complete metrics for full business-relevant timings from the perspective of the user and waterfall chart details for user actions in a visit. These provide critical data to help you optimize your web application for optimal performance.

Getting end-to-end traceability is as simple as installing the JavaScript Agent on your servers. Web Server and User Experience sensors inject the Agent automatically. For mobile applications, compile the agent into the mobile application for full end-to-end traceability and UEM data. See Java Agent Configuration for more information in installing and configuring JavaScript Agents, and Mobile Agent for details on compiling and using the JavaScript Agent for mobile applications.

Set up web performance optimization for UEM

Once the JavaScript Agent is installed and user Experience sensors are in place, you can setting up web performance optimization for UEM. This includes the following:

- Enable UEM for the applicable System Profile.
- Verify the JavaScript Agent Agent injects properly into the web site.
- Enable JavaScript Support for the Application's frameworks.
- Enable W3C Resource capturing.
- Optionally enable metadata capturing

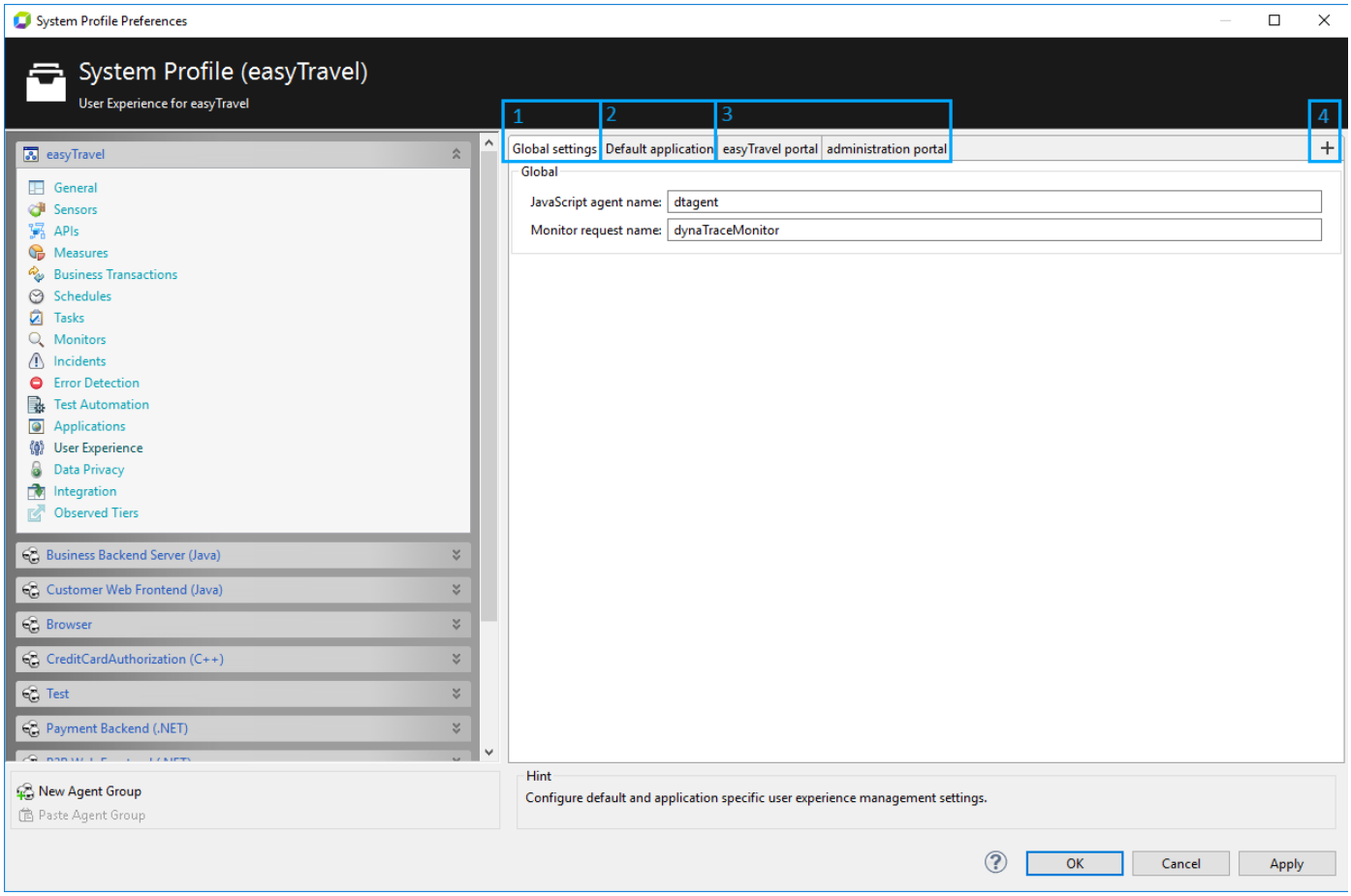
Enable UEM

1. In the AppMon Client, double-click the desired profile with the target application, then in the System Profile Preferences, choose **User Experience**.

2. Select the tab corresponding to the application you want to configure.

   Learn how

New system profiles contain a single configuration for the Default application (2). Click the **Add specific application** button (4) to copy the configuration for a specific application (3). Any application that does not have a specific configuration uses the Default application configuration.



3. Configure UEM Capturing and General settings as needed to enable UEM data capturing for the target application(s).

4. Click **Apply**.

Ensure JavaScript Agent injection into the web site

Installing the JavaScript Agent and creating the appropriate UEM sensors provides injection into your application, and compiling the agent into your mobile application code provides injection into your mobile application. You can configure additional JavaScript Agent injection settings to ensure you're using the correct JavaScript Agent version, and injection method.

1. In the User Experience tab of the System Profile settings for the desired System Profile, scroll to and expand the Web applications section.
2. Configure settings as needed to set the JavaScript Agent version to inject, and to set the injection method.

See System Profile - User Experience Agent injection section for more information.

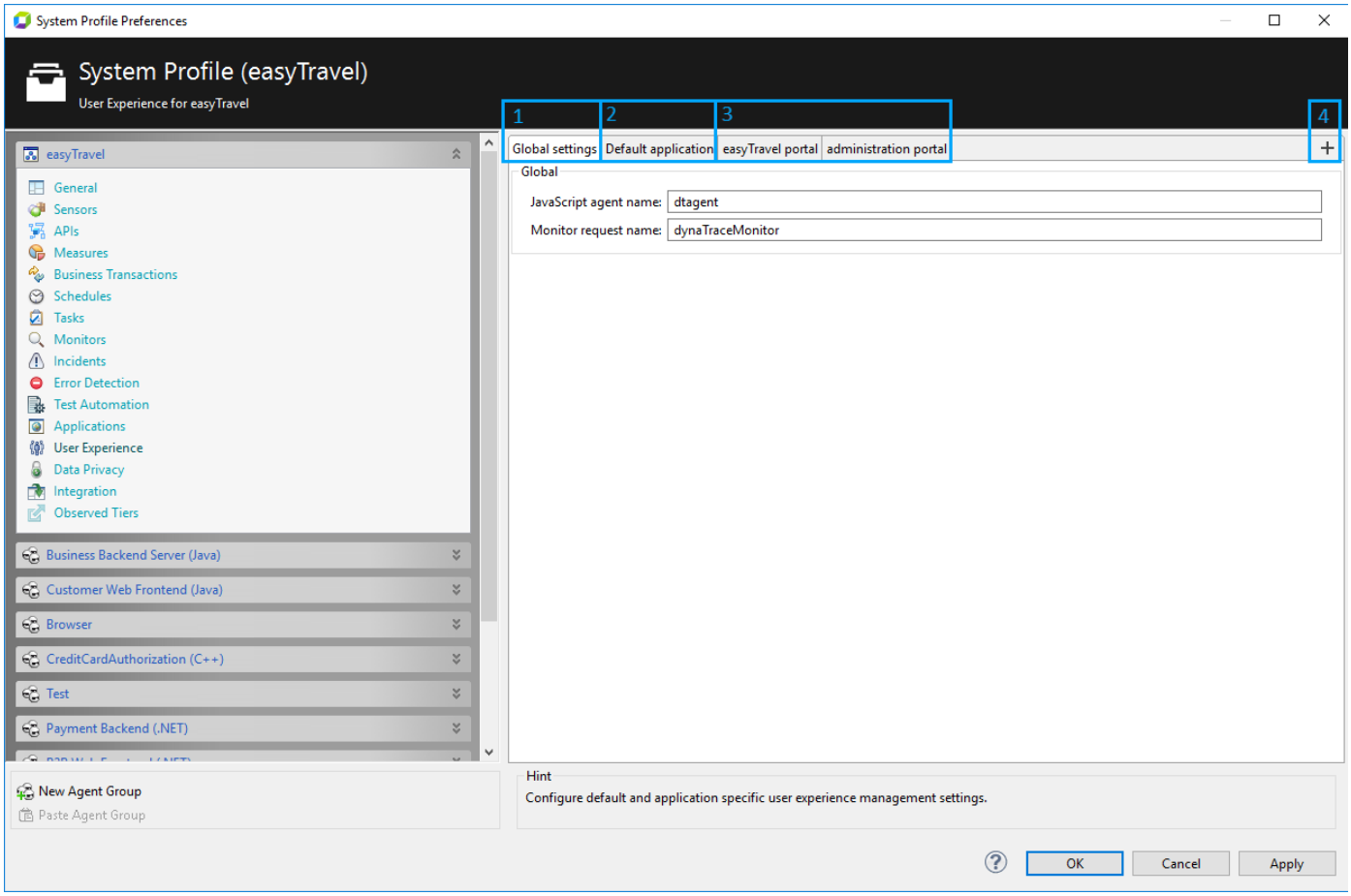
Enable JavaScript support for your frameworks

To enable JavaScript support for your supported frameworks:

5. With the desired system profile, select the tab corresponding to the application you want to configure in the **User Experience** tab.

   Learn how

   New system profiles contain a single configuration for the Default application (2). Click the **Add specific application** button (4) to copy the configuration for a specific application (3). Any application that does not have a specific configuration uses the Default application configuration.



6. Scroll to the **Web applications > Ajax (XHR) detection** section and select appropriate check boxes for your supported Java frameworks. Consult with your developers if you need to know which frameworks are used. See Ajax (XHR) detection and additional packs for more information.

> **Tip**
>
> If no selections are available that cover your supported frameworks, select the **Basic XHR Detection**, which is useful for covering most frameworks.

7. Under Additional Packs, additionally select **JavaScript Errors** and **Perceived Render Time** to capture error and render time data.

8. Click **Apply**.

Enable W3C resource capturing

To capture full waterfall details:

1. in the **User Experience** tab for the desired System profile Preferences, navigate to the **Web applications > W3C resource timings** section.
2. Select **Capture Resource Information** to enable resource capturing of the web performance data that is useful in performance optimization.

See the W3C Resource Timings section in System Profile - User Experience for more information.

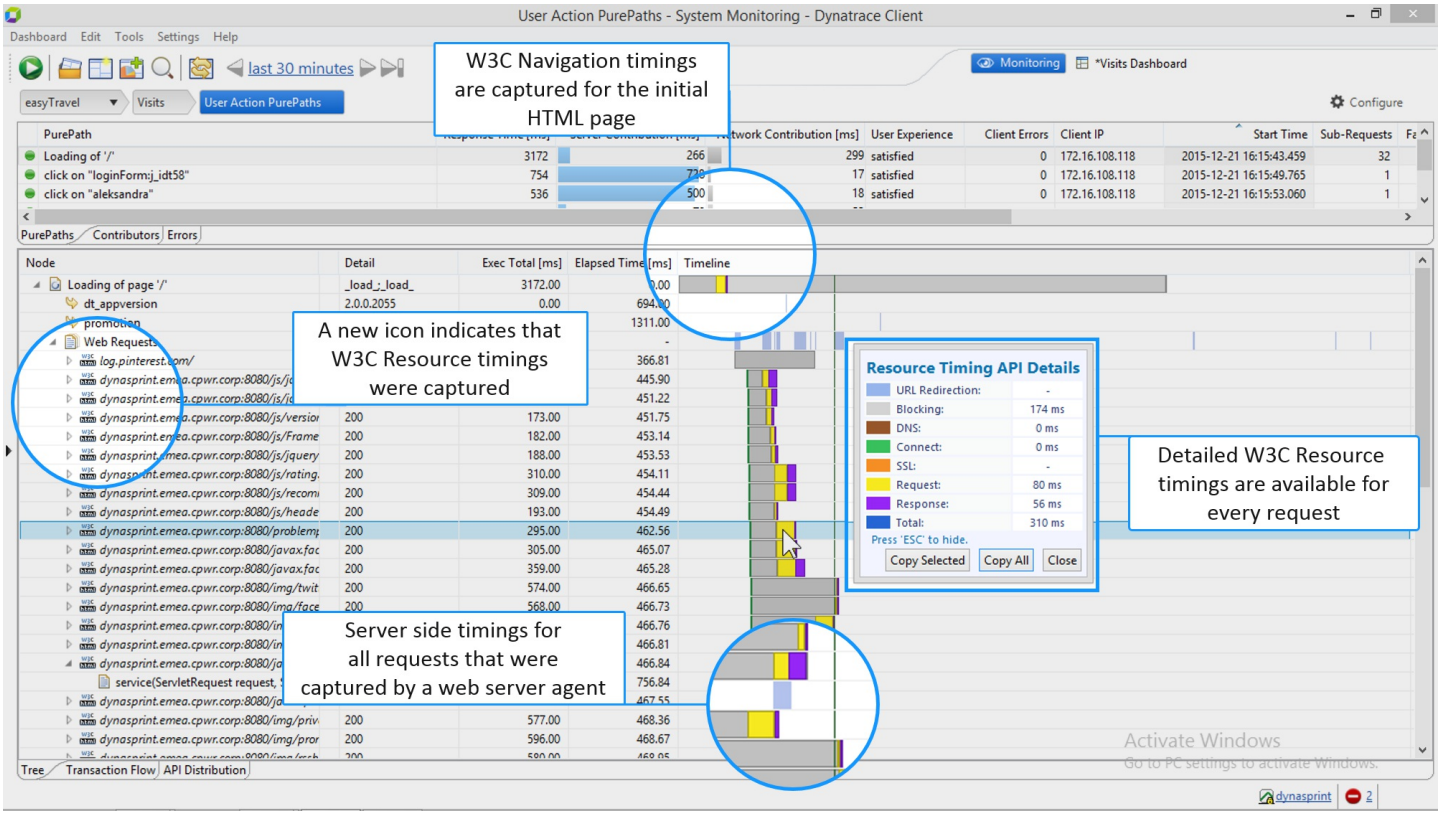
Enable metadata capturing

Optionally configure metadata capturing to help provide a more complete web performance picture. You can capture metatag information, execute CSS selectors, and capture information on HTML controls and JavaScript variables. Enabling metadata capturing along with W3C resource capturing is useful in click-through testing scenarios to capture rendered screen elements or dynamically generated content, such text box content or a calculated sum, part of the data to aid in making performance optimization decisions.

See System Profile - User Experience Metadata section to learn how.

Seeing full waterfall details

Once you set up web performance optimization for UEM, you can see full resource and deep-drive metrics for all configured visits as a waterfall chart in the User Actions PurePath dashboard.



Use UEM on any application

In situations where you do not have access to a particular web application to install the JavaScript Agent on it, but still want to capture UEM visit data and W3C resource timings for waterfall details, you can set up the web server as a reverse proxy, then direct your browser to use the reverse proxy. This lets you leverage AppMon's powerful UEM data gathering capabilities for any web application.

To get web pages performance with UEM for any website, you can install any Web Server such as Apache and use it as a proxy server. So for example, if set up Apache as a reverse proxy and install an Agent for it, then tell the Browser to use that proxy, when you execute page web requests, the web server acting as a proxy injects the JavaScript agent on these pages. This method is also useful when you need to test an application but do not have access to the application.

See Set up a Forward or Reverse Proxy to learn how.

How to instrument a web application

On this page

Agent injection

Agent Configuration

Automatic injection Agent tag placement rules

Manual injection details

Monitor signal behavior

Related pages

How does UEM work?

JavaScript Agent

How to instrument a hybrid app

How to instrument an Android app

How to instrument an iOS app

System Profile - User Experience

The JavaScript Agent is the central component in UEM for web applications. It collects user experience data in the browser and sends it to a web or application server, where an AppMon agent forwards the data to an AppMon server.

The JavaScript Agent is a script that automatically injects into web pages (of mime type `text/html`) using either Java or Web Server Agents, or manually at design time if not otherwise possible.

Agent injection

Injecting Agents is what instruments a web site. This consists of:

- Injecting an inline tag.

- Injecting a tag with a `src`-attribute. The inline tag must be injected as the very first script and consists of essential instrumentation parts of the JavaScript Agent. For the best capturing experience, the second tag should be injected right after the inline tag, but before any other JavaScript.

Asynchronous injection

For AppMon 6.5 and later, the JavaScript Agent injects asynchronously by default using the `defer` script tag attribute. Refer to MDN Documentation for more information.

Asynchronously loaded scripts do not block page loads. However, since the Agent cannot redefine any library variables once initialized (like jQuery or AngularJS), the `defer` script tag attribute also used for any third party frameworks to be instrumented. Also, the JavaScript Agent must be available before those frameworks start initializing, otherwise it is not possible to capture actions triggered by scripts.

You can always disable asynchronous injection by clearing the **Load the JavaScript Agent asynchronously** check box in **[System Profile] > User Experience > Web applications**. See System profile - User Experience for more information.

Automatic injection

The following is a sample tag-combination automatically injected by the Agent as the first script tags in `<head>` at delivery of the web page – with settings according to the configuration in **[System Profile] > User Experience**:

```
<script type="text/javascript">(function(){var a=window;a.dT_?
a.console&&a.console.log("Duplicate agent injection detected, turning
off redundant initConfig."):window.dT_||(window.dT_=
{cfg:"tp=0,50,10|lab=1|reportUrl=dynaTraceMonitor|agentUri=/ajax/dtagent
|auto=1|domain=#DOMAIN#|rid=RID_1640928385|rpid=1646377341|app=#APP#"})}
();...</script> <script type="text/javascript"
src="/dtagent_bdpx_0000000.js"></script>
```

The first tag (without `src`-attribute) is the agent's *initialization code*. It must be delivered inline and executes synchronously. It performs early-wrapping of native browser functions and any available third party libraries to ensure reliable capturing.

The second tag requests the part of the JavaScript Agent that can be loaded asynchronously and therefore won't block a page from being loaded. The filename defines it's version and which parts of the Agent should be delivered:

- `dtagent` is the JavaScript Agent Name set in **[System Profile] > User Experience > Global Settings**.
- `700000000GABuildNotYetKnown` is the AppMon Agent version (major, minor, revision, buildnumber).
  Each part of the version string consists of four characters, trimming leading zeroes, such that it is read like this: `7 0000 0000 GABuildNotYetKnown`, which translates to `7.0.0.GABuildNotYetKnown`.
- `bdpx` is the feature hash, and shows information captured by the Agent, such as bandwidth, dojo, perceived render time and basic XHR detection.

Manual injection

If agents cannot be injected automatically, you can insert the initialization tag manually at design time as the first script tag in `<head>`. Since the initialization tag and it's contained code varies depending on your application settings, it can easily be retrieved by using the Server REST Interfaces API using `https://<AppMon server>:8021/api/v1/profiles/<systemprofile>/applications/<application>/`

The initialization code performs operations to load the JavaScript Agent (either with or without the `defer` attribute) and updates it's configuration if required. It is not necessary to update the initialization tag every time you change the UEM configuration, though adjustments are applied faster for new visitors if you do.

Agent Configuration

Injected Agents use configuration settings in the System Profile - User Experience vertical tab, generically:

```
<script type="text/javascript"
src="/dtagent_<featurehash>_<version>.js"></script>
```

- `<featurehash>`: A set of characters that identifies which Agent modules to include and activate. When only the basic Agent is activated, the feature hash is empty resulting in two adjacent underscores in `dtagent__<version>.js`. The following modules are delivered with AppMon:

- Ajax (XHR) detection modules:
  - ActiveX XHR Detection (v)[1]
  - Angular JS (g)
  - Basic XHR Detection (x)
  - Dojo (d)
  - ExtJS (e)[2]
  - ICEfaces (i)
  - jQuery (j)
  - MooTools (m)
  - Prototype (o)
- Additional packs:
  - JavaScript Errors (q)
  - Perceived render time (p)
  - Speed Index (S)
  - Streaming (s)
  - Timed Action Support (t)
  - User Timings (T)
  - Visually Complete (V)
- Community packs (debug mode only):
  - Gomez page and group IDs (z)
  - JS agent async core module
  - JS agent initconfig module
  - Windchill (1)

[1] ActiveXObject support (required by IE 6, supported by IE 7 and later) is split from the basic XMLHttpRequest module to reduce file size and performance overhead. The ActiveXObject support module is only needed if the instrumented application utilizes ActiveXObjects to send requests from the JavaScript side that have to be captured. Every AppMon supported browser supports XMLHttpRequests, so most applications only need basic XMLHttpRequest detection. If uncertain whether you need ActiveXObjects support, both modules can be enabled.

[2] Some ExtJs-Functions use deferred callbacks when they send XHRs. AppMon does not capture these requests because it can't create a link to a user action. Activate the Timed Action support To avoid this, activate the Timed Action support. It extends user actions to track deferred functions. Ext.FormPanel is a known ExtJs function that must have Timed Action (t) support.

[3] The resource timings modules takes advantage of the W3C Resource Timing API. However, not all browsers currently support it, thus the JavaScript Agent performs timing checks based on instrumented html tags and gathers information about load times of external resources. This approach can cause some overhead, so be sure to set resource timing limits within your application configuration.

The following shows a configuration example with activated Dojo and jQuery JavaScript packs:

```
<script type="text/javascript" src="/dtagent_dj_700000000
GABuildNotYetKnown.js"></script>
```

Automatic injection Agent tag placement rules

The following set of rules specifies the criteria that determine a suitable location within an HTML document to inject the JavaScript Agent:

- **(<?xml?> rule)** If an <?xml …?> specification is encountered:
  - Ignore it and continue to scan the document.
  - Overrules (non-<meta> tag rule), (non-<head> tag rule) and (initial tag rule).
- **(initial tag rule)** If a tag appears before <html> that is not <!DOCTYPE …>, <html>, <link>, <meta>, <script>, or <style>:
  - Abort without injecting.
- **(<html> tag rule)** When <html> is encountered:
  - If a potential injection point is found earlier, inject there, and do not scan further (if there are multiples, the earliest injection point is used).
  - Otherwise continue to scan the document.
- **(<!DOCTYPE> rule)** If <!DOCTYPE …> doesn't specify an HTML (is not <!DOCTYPE HTML … >):
  - Abort and do not inject.
- **(comment rule)** If a <!- comment -> is encountered:
  - Ignore it and continue to scan the document.
  - Overrules (non-<meta> tag rule) and (non-<head> tag rule).
- **(<title> tag rule)** If <title> is encountered:
  - Ignore everything until </title>, then continue to scan the document.
  - Overrules (<body> tag rule).
- **(<body> tag rule)** If <body> is encountered:
  - Document scan stops.
  - Any potential injection point found earlier is used (in case there were multiple, the earliest injection point is used).
  - If no rule provided an injection point earlier, inject after <body>.
- **(<script src> tag rule)** If a <script …> tag is found within <head> that has a `src="..."` attribute:
  - If a (potential) injection point is found earlier, inject there, and do not scan further (if there are multiples, the earliest injection point is used).
  - Otherwise, inject before this <script> and do not scan further.
  - Overrules (non-<meta> tag rule), (<base> tag rule).

- **(<link> tag rule)** If a <link …> tag is found within <head> that is not inline (doesn't have an `href="data:..."` attribute):
  - If a (potential) injection point is found earlier, inject there, and do not scan further (if there are multiples the earliest injection point is used).
  - Otherwise, inject before this <link> and do not scan further.
  - Overrules (non-<meta> tag rule), (<base> tag rule).
- **(flush rule)** (Java only) when flush is called on the injecting stream/writer outside of <head>… </head> and a conditional injection awaits confirmation:
  - Discard the conditional injection point, propagate the flush and continue to scan the document for a new injection point.
  - Overrules any conditional injection point found before by the (non-<head> tag rule).
- **(flush in <head> rule)** (Java only) when flush is called on the injecting stream/writer within <head>… </head> and conditional injection awaits confirmation:
  - Keep the injection point and continue to scan the document.
  - Disregard the flush (don't propagate it).
- **(<base> tag rule)** If <base …> is encountered:
  - Inject after the <base …> tag.
  - Overrules any conditional injection point found before by the (non-<meta> tag rule), (unclosed <meta> rule), (non-<head> tag rule).
- **(non-<meta> tag rule)** If a tag is found within <head> that is neither <meta> nor <title>:
  - Inject before it (conditional injection).
  - Continue to scan the document, in case this injection choice gets overruled.
- **(non-<head> tag rule)** If a tag is found after <html> but before <head> that is neither <head> nor <body>
  - Inject before it (conditional injection).
  - Continue to scan the document, in case this injection choice gets overruled.
- **(unclosed <meta> rule)** If a </head> arrives after a <meta> that doesn't get closed (either by a closing </meta> tag or by the XML-style <meta … /> tag):
  - Add </meta> followed by the injection, both before the </head> (conditional injection).
  - Continue to scan the document, in case this injection choice gets overruled.
- **(</head> tag rule)** on </head>:
  - If a (potential) injection point is found earlier, inject there, and do not scan further (if there are multiples the earliest injection point is used).
  - Otherwise, continue to scan the document.
- **(end of file rule)** When end of file is reached (and scanning did not terminate before that):
  - Do not perform an injection.
  - Overrules any conditional injection point found before by the (non-<meta> tag rule), (unclosed <meta> rule), (non-<head> tag rule).

- **(parse error rule)** When the document's contents doesn't appear to exhibit the basic structure expected from HTML tags and attributes, it makes further parsing futile:
  - Document scan stops.
  - Any potential injection point found earlier is used (in case there were multiple, the earliest injection point is used).
  - If no rule provided an injection point earlier, do not perform an injection.

Manual injection details

If the JavaScript Agent can't be auto-injected by the Java or Web Server Agent at page delivery, you can still insert the Agent initialization script tag manually at design time.

- If the server providing the Agent and the one getting the monitoring data are different (domains), you must set the Monitor Request Path and select the **Send the Dynatrace monitor request to a foreign domain (CORS)** check box. For Cordova / PhoneGap make sure this reference resolves via the domain whitelist and CORS is disabled. See PhoneGap Whitelist Guide for more information.
- You must also switch the Injection Point to manual in the respective **<System Profile> > Agent Group / tier > Sensor Configuration > User Experience sensor > Properties**.

Dynamically served Agent

As mentioned above, use the Server REST Interfaces API to inject the JavaScript Agent initialization tag via this URL:

```
https://<AppMon server>:8021/api/v1/profiles/<systemprofile>/applications/<application>/javascriptagent/initcode
```

The initialization tag performs operations to load the correct JavaScript Agent and update the application's UEM configuration as it changes.

Statically served Agent

For less traffic to the instrumented server, you can manually download the JavaScript Agent file and locate it anywhere on your server. As the agent file differs depending on your configuration, the correct file for your application can be retrieved via the Server REST Interfaces API:

```
https://<AppMon server>:8021/api/v1/profiles/<systemprofile>/applications/<application>/javascriptagent
```

Be sure to configure the agent location accordingly, so the initialization tag is capable of building the correct agent tag.

For statically served Agents you just need to inline the initialization tag. Upon configuration changes, just switch the agent file.

How does it work?

After UEM configuration changes, the initialization code is created containing all necessary information about the current configuration for each application. It can then be retrieved via the REST API.

If a page is injected with the JavaScript Agent initialization code, it checks if the browser already stored some application information and creates the JavaScript Agent script tag. It is then injected directly after the initialization tag and starts working immediately.

Performance notes

- **Overhead**: Since the initialization tag is inlined into a page, there's no additional request that has to be made by the browser. The configuration updates are retrieved as soon as an action is sent.
- **Caching**: Since the JavaScript Agent is cached for about a week, there is a chance that the initialization tag loads it with outdated settings. In this case the first page load is tracked using the old settings. A beacon is sent and in response the app / web server Agent provides new configurations which are then stored in the browser's local storage. Upon the next page load these settings are loaded and the initialization code is capable of creating an updated JavaScript Agent tag.
- **Filesize**: Depending on the application configuration the obfuscated initialization code's size is about 6.65kb.
- **Browser support**: Internet Explorer 6 and 7 have no implementation of Local or Session Storage and therefore do not automatically update upon configuration changes. However, as soon as the cached page ages, the initialization code updates to the latest configuration revision.

Scenarios

There are various ways the initialization code is executed, depending on the visitor.

- **New visitor**: First, the initialization code is loaded, which loads the JavaScript Agent depending on the current configuration.
- **Recurring visitor**: The full page is loaded from cache. The initialization code checks the browser's local storage for configuration entries. If there are none, it takes the last known configuration and loads the JavaScript Agent from cache. Any new settings sent with the beacon get stored in the local storage.
- **Recurring visitor with changed settings**: The full page is loaded from cache. The initialization code checks the browser's local storage for configuration entries. If there are any, it checks if they are newer than the ones contained in the initialization code and takes the latest settings to create the JavaScript Agent tag. If feature hash and version are the same, the JavaScript Agent is loaded from cache.
- **Recurring visitor which shouldn't be tracked**: If a visit should not be tracked due to visit percentage settings, the JavaScript Agent still initially injects using the initialization code and sends a beacon containing a load action. The application / web server Agents receive this beacon and decide if the visit should be tracked or not. If not, the beacon is ignored and the response contains a flag. Should the JavaScript Agent receive such a flag, every communication shuts down and no more beacons are sent until the browser session ends.

Monitor signal behavior

The JavaScript Agent sends the gathered information in the form of POST requests. This behavior is recommended for most servers and requires that POST requests are allowed on dynaTraceMonitor*.

In most cases signals are sent as soon as actions are finished. This action starts with a user input that triggers a request and ends as soon as all of the request's callbacks are finished. The load-action occurs on each iframe and full page load and is sent on every page, if user inputs exist or not.

These actions trigger signals to be sent immediately after the actions are finished, in contrast to *medium priority signals*. These are triggered by timeless and lesser important actions. These medium priority actions are bandwidth, streaming, and resource information, as well as reported values like errors or custom strings. If a signal is queued, this information is appended and doesn't create a separate request.

If multiple actions happen in a small time frame, such as a user action during page load, all of those actions are sent in a single signal.

For longer actions, a preview signal is sent as medium priority signal. Actions are sent only if all of their children and siblings are finished. If, for example, an action is opened and a second one is opened (which automatically is a child of the first one), the first one won't be sent until the second one is finished.

Support for cross-origin resource sharing (CORS)

Reporting from a non-instrumented host

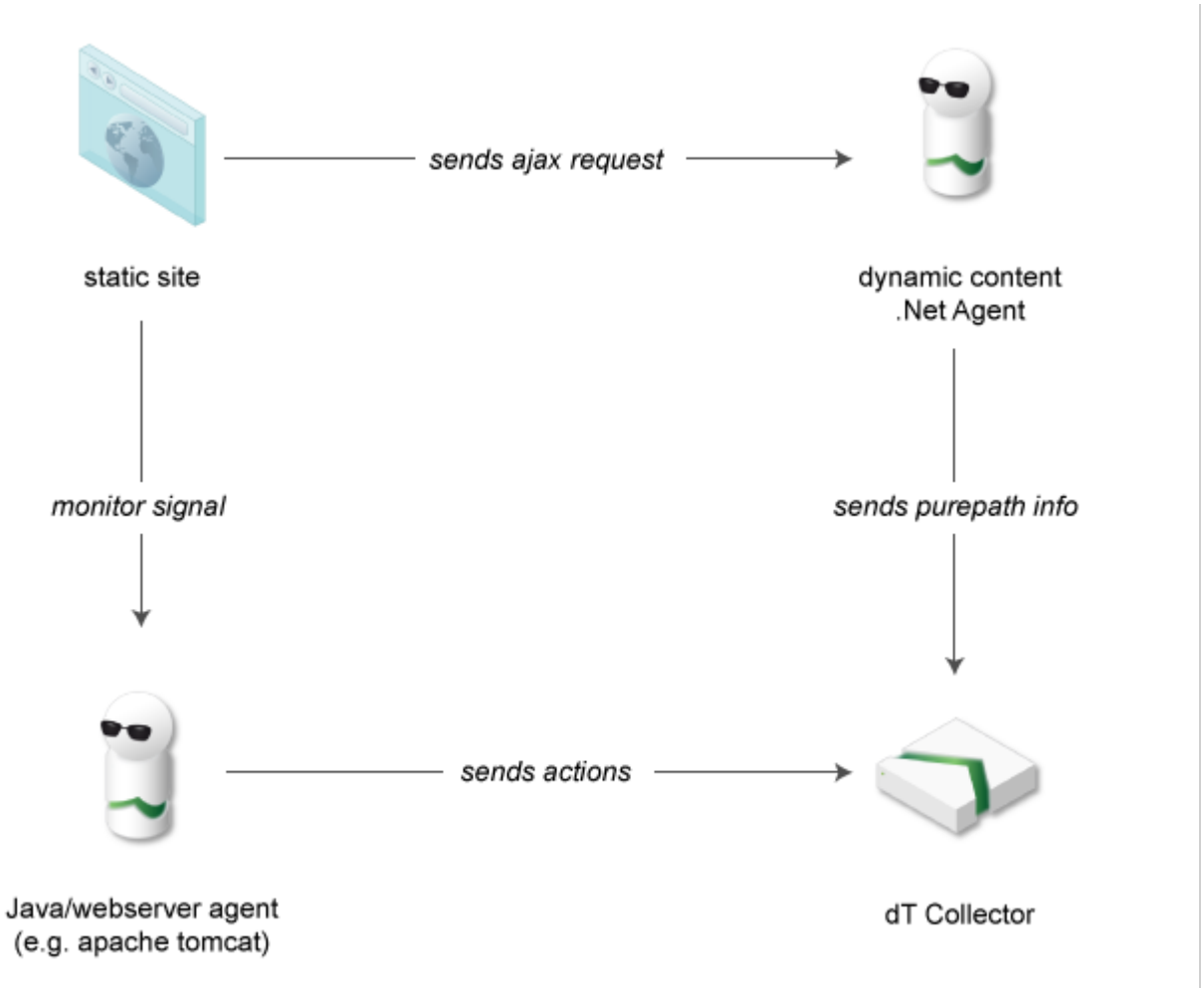
AppMon enables injection of the JavaScript Agent on web servers that are not instrumented. If a website is hosted on a web server where UEM is not active, the JavaScript Agent has to report to a different domain, which can cause cross domain problems. If you use a CDN and don't want to send the monitor signal back through the CDN, you can send the monitor signal via CORS (Cross Origin Resource Sharing).

To enable support for CORS, start the Appmon Client and select the **Send AppMon monitoring request to foreign domain** in **[System Profile] > User Experience vertical tab > [application] horizontal tab > Advanced configuration** section.

Be sure to set the *Monitor request path* configuration to a location (use an absolute path that starts with `http`) that is UEM-enabled (for example by a Java Agent).

Monitoring CORS requests

Web requests can only be linked to user actions if cookies are available. In case of CORS requests, cookies aren't sent per default. Therefore correlation between user actions and CORS web requests can only be achieved by using the withCredentials property of the XMLHttpRequest object. This applies for an environment like this:



Restrictions

Due to access-control-restrictions it is not possible to capture OPTIONS requests like those sent as CORS preflight requests. Capturing those requests would require sending a AppMon cookie to the receiver of the request, which is not allowed. This could result in User Action PurePaths that have the correct timing values, but are missing the time the OPTIONS request took before the real CORS request fires. See MDN for more information about CORS preflight requests.

Mobile UEM

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Supported platforms

Configure AppMon

Integrate with the application

Link web requests

Configure mobile data collection

Define measures

Analyze data

Overhead

Related pages

How to instrument a hybrid app

How to instrument an Android app

How to instrument an iOS app

Mobile symbolication service

Set up system components

Licensing

User experience management administration

Edit a System Profile

System Profile - Measures

Downloads

Latest update

AppMon UEM's mobile application support lets you track visitors, actions, and action PurePaths triggered by your iOS and Android application. To trace user activity, you must add mobile agent method calls to your mobile application's source code. When the mobile agent is embedded in the application, user experience data can be analyzed in the same way as for users browsing your UEM enabled web site.



Mobile app ADK workflow

Supported platforms

- A recent Apple iOS (armv7, armv7s, arm64 or later architecture, iOS 6 or later).
- Android version 2.3 or later.

See Release notes for details.

Configure AppMon

The setup is similar to the setup for UEM:

- Perform basic AppMon setup. See the appropriate topics under Set up System Components.
- Apply the UEM license, including Visits Volume. See Licensing.
- Install an Agent on the application server or web server. See User Experience Management Administration.

Integrate with the application

Se the relevant topics linked below for detailed information on integrating the Mobile Agent into your mobile application.

Native Applications

- iOS Agent Setup and Instrumentation
- Auto-Instrumentation for iOS
- Android Agent Setup and Instrumentation
- Auto-Instrumentation for Android

Hybrid applications

See How to instrument a hybrid app.

Link web requests

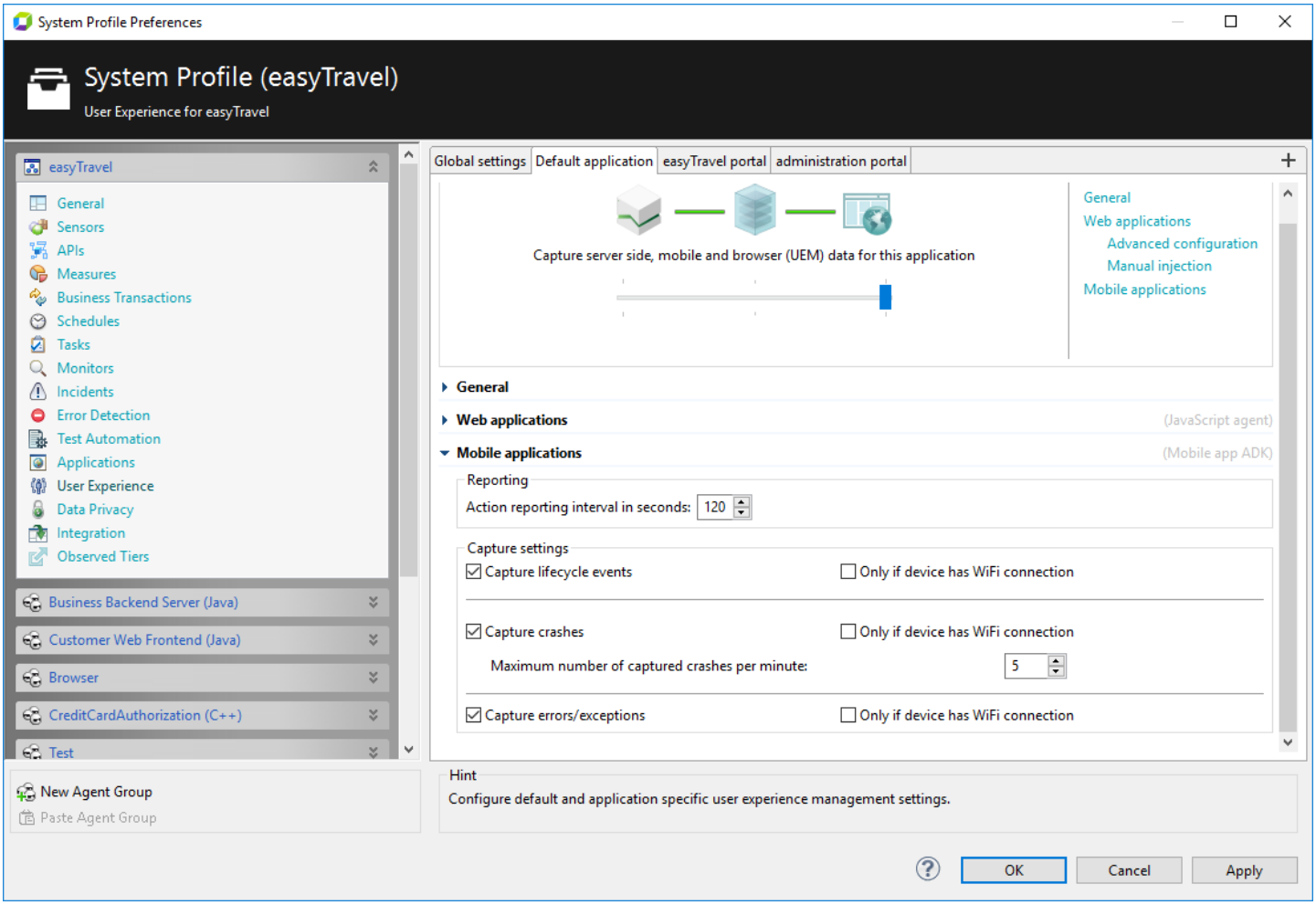
If a web request is tagged and the server side is instrumented, the server side web request PurePath links to the mobile User Action PurePath.

- If the web request is tagged on the same thread where a currently open action was started, it is always linked to this action.
- If the web request is tagged in a different thread, it is only assigned to an action if the web request occurs during the time frame of the action and the action is left in the same thread. Because AppMon currently does not capture the latency between the web request started on the mobile side and received on the server side, linking by thread ID and time frame is not guaranteed to work as expected for higher-latency scenarios.

Configure mobile data collection

In the **User Experience** item of the **System Profile Preferences** dialog box, you can adjust mobile data capture settings. To access these settings, right click your system profile and select **Edit system profile > User experience > <application tab> > Mobile applications**. See the Mobile applications section in System Profile - User Experience to learn about configuring mobile application settings.

To limit overhead, a maximum depth of 10 nested mobile actions is evaluated. Deeper nested actions are skipped and not shown in the User Action PurePath.



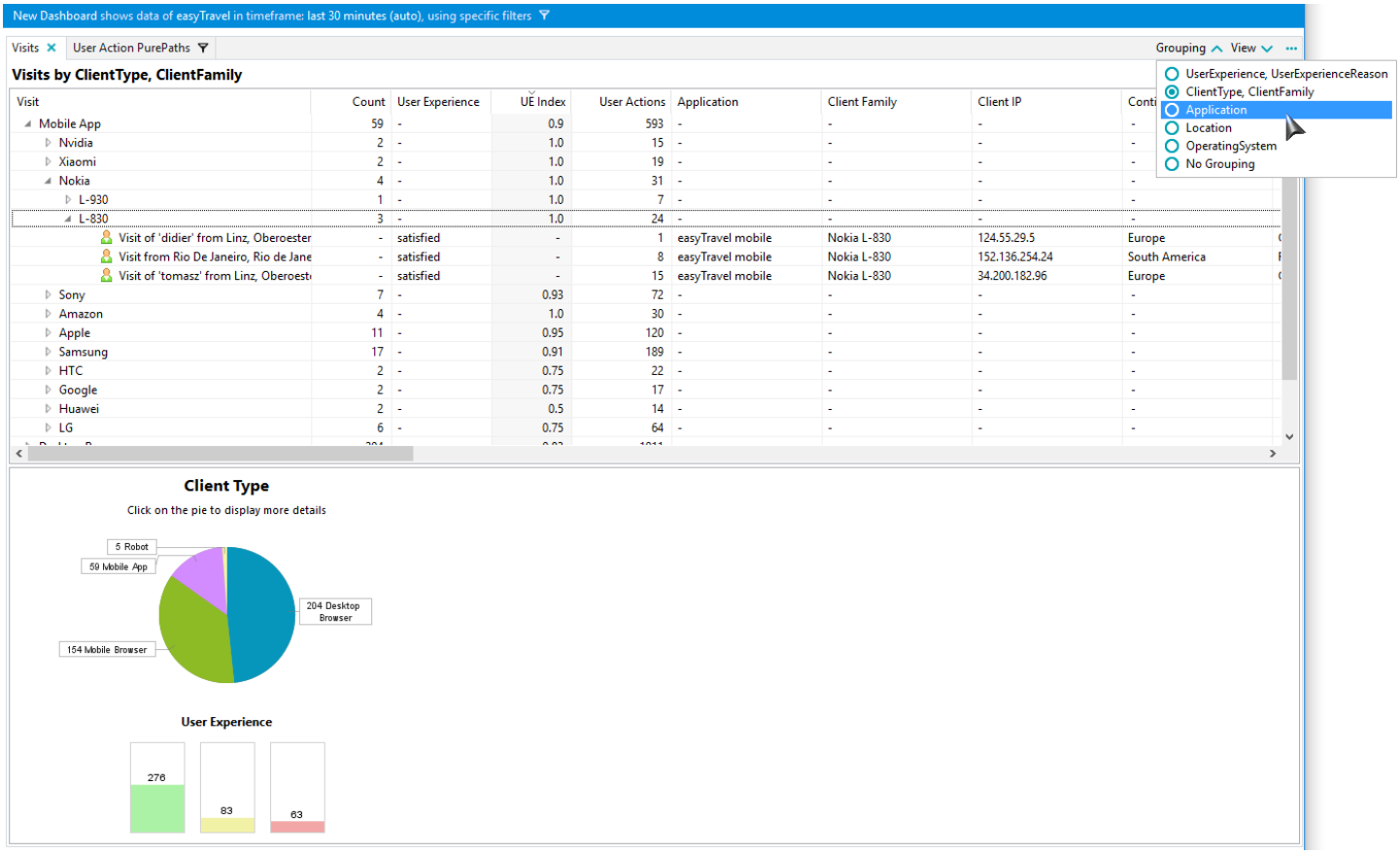
User experience settings - mobile applications

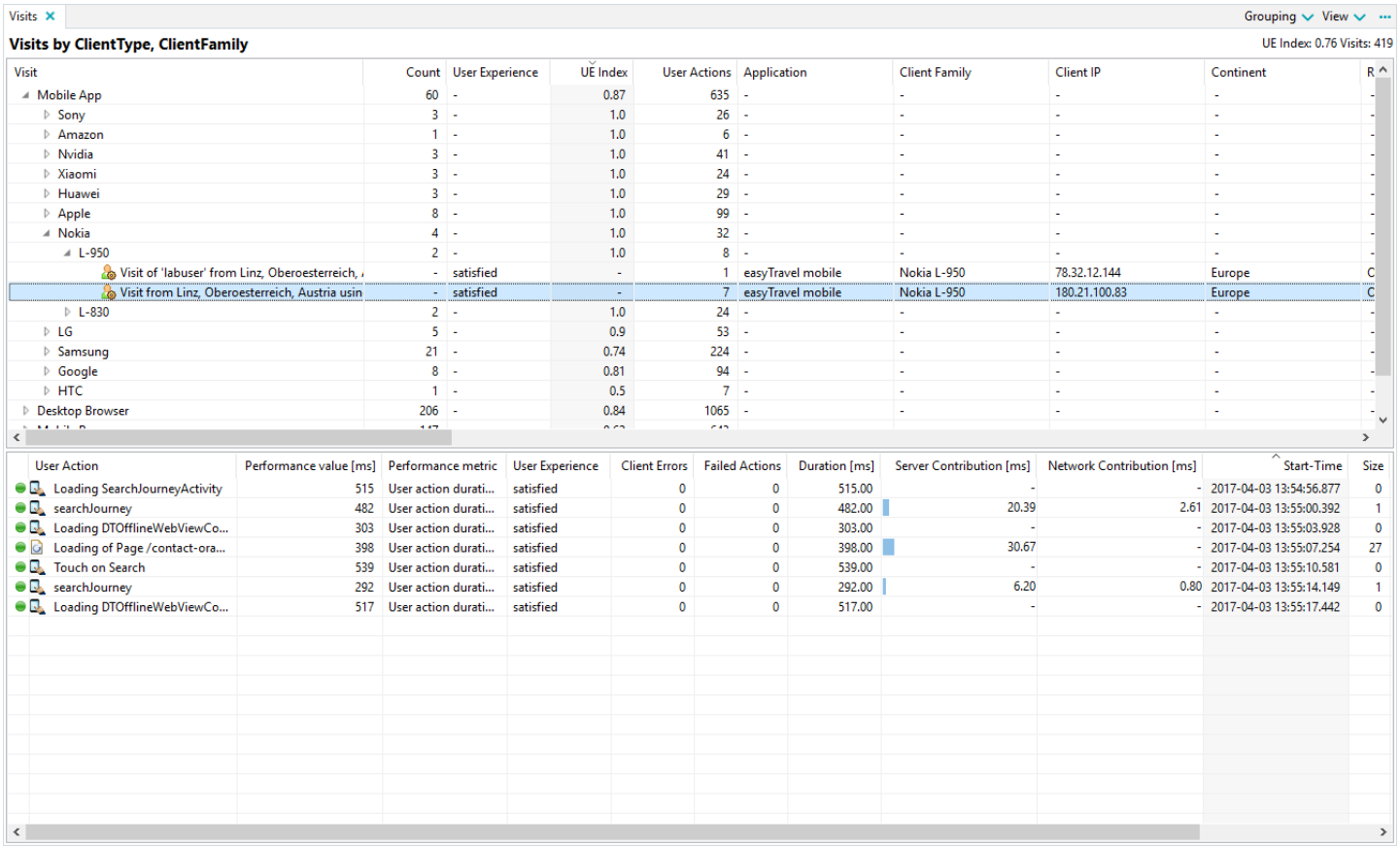
Define measures

To evaluate the results returned by the instrumented mobile application, subscribe new Measures for Mobile Actions in the **Measure Configuration** dialog box. To access it, right click your system profile and select **Edit system profile > Measures**, then expand **User Experience Management > User Actions > Mobile Actions** and double click the required measure. See Edit a System Profile and System Profile - Measures.

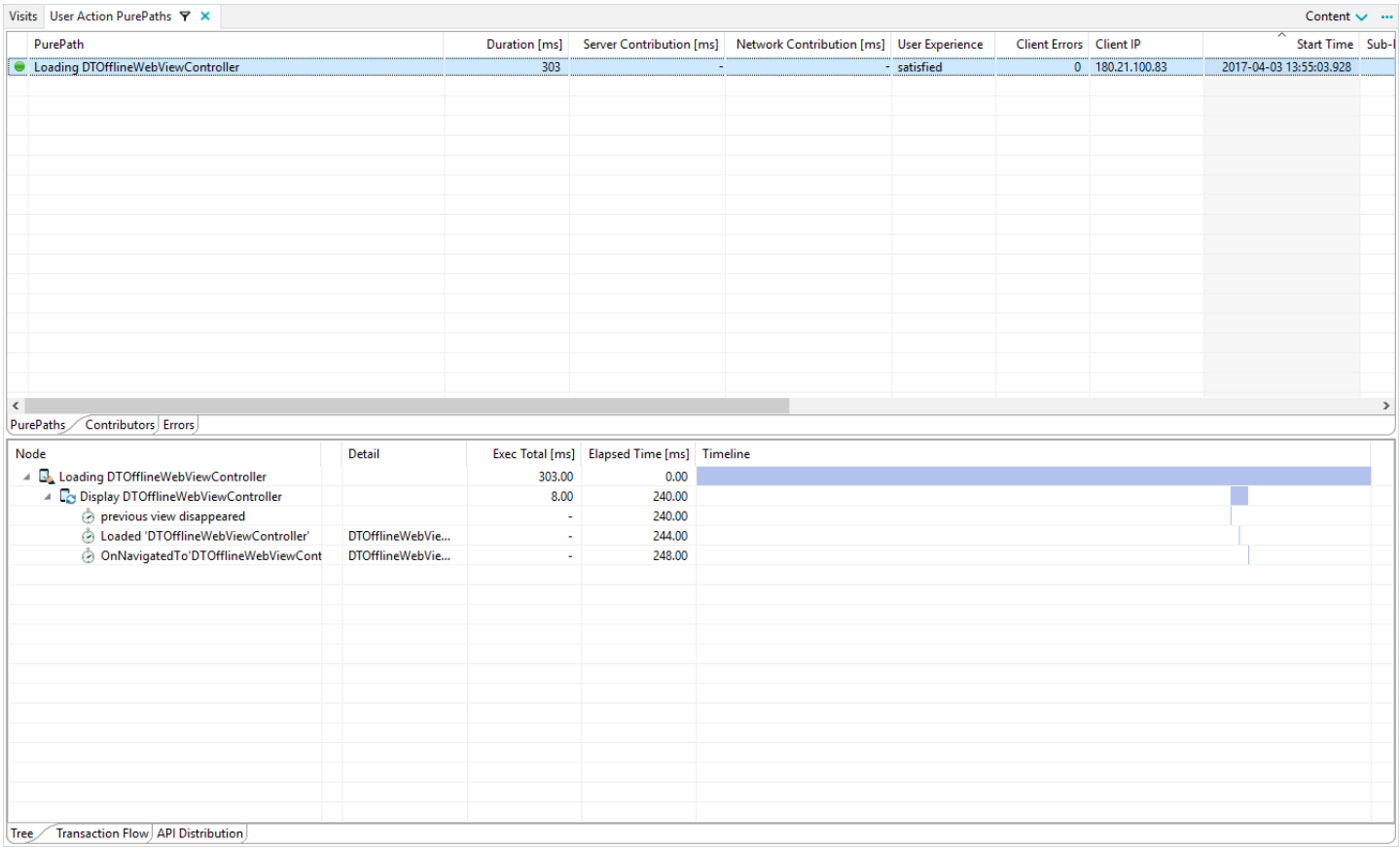
Analyze data

When you have configured AppMon for mobile data collection, the **Visits Dashlet** displays mobile visits, identified by mobile device, and shows all user actions per visit. To easily identify mobile visits, click **Grouping** and select either the **Client type, Client family** or **Application**



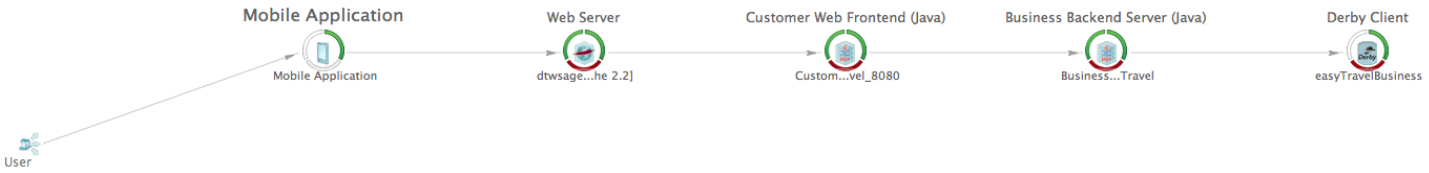


Right click a user action, and select **Drill down > User action PurePaths** to open the **User Action PurePaths Dashlet** linked to server-side PurePaths.



User Action PurePaths

The **Transaction Flow dashlet** displays a visualization of your PurePaths, including PurePaths for mobile devices. To open it, right click a PurePath you want to analyze and select **Drill down > Transaction Flow**.



Transaction Flow Dashlet

Zoom in to see where most of the time is spent.

The icons that identify mobile actions and events are defined in the following table.

Symbol	Mobile Action/Event
	Mobile lifecycle action (e.g. app start, display and redisplay of a view).

	Timestamp for events of a lifecycle action.
	Mobile action.
	Named event.
	Report value event.
	Report error event.
	Crash with attached crash report.
	Recurring crash (no crash report attached).

Mobile App Performance Overview dashboard

Use the **Mobile App Performance Overview** dashboard to analyze data generated by mobile applications. Access this dashboard via the **User Experience** tab in the **Start Center**.

The dashboard displays the response time, visits, errors, crashes, and user actions of mobile applications.



Mobile App Performance Overview Dashboard

Collected metrics

The following metrics are collected and available in various parts of UEM dashboards.

Metric	Description
Agent Version	Version of the Mobile Agent with which the application was instrumented (e.g. 6.5.0.2013).
Application Name	The mobile application name.
Application Version	iOS: `CFBundleShortVersionString` from `Info.plist` (e.g. 2.0.1). Android: `android:versionName` from `AndroidManifest.xml` (e.g. 2.0.1).
Application Build Version	iOS: `CFBundleVersion` from `Info.plist` (e.g. 2.0.1.4567). Android: `android:versionCode` from `AndroidManifest.xml` (e.g. 234).
Battery Status	Device battery strength (e.g. 70%).

Connection Type	Mobile, WiFi, LAN or Offline.
Network Protocol	Offline, 802.11x (for WiFi connections), LTE, etc.
Device Name	The make and model of the device. For example, SAMSUNG-SGH-I727, iPhone 5S, iOS Simulator, google_sdk (if emulator).
Device Manufacturer	For example, Apple, Samsung, unknown (if emulator).
Total Memory	Device RAM, in megabytes (e.g. 1024).
Free Memory	Percentage of free RAM (e.g. 39%).
Number of Running Processes	The number of active application processes running on the mobile device.
OS Family	The operating system: iOS, Linux.
OS Name	The name and version of the operating system, e.g. iOS 7.0, Android 4.2.
Screen Resolution	The screen resolution in pixels, e.g. 480 X 800
Device Orientation	The default orientation for the device, e.g. portrait for a mobile phone or landscape for a tablet.
Device Carrier	The carrier that provides the signal for the Android device, for example AT&T. *Android* displays if the carries is an emulator. Not available for iOS.
CPU Information	The device's CPU, e.g. ARMv7.
Rooted/Jailbroken	Whether the device's software is rooted (Android) or jailbroken (iOS). Displays *genuine* if device's software is not rooted or jailbroken.

Overhead

The overhead depends on the instrumentation that you put into your app. It consists of:

- Data traffic to the server
- Memory usage
- Execution time of the action in the app

All three factors mainly depend on the number of nodes that your instrumentation is about to produce.

For example, assume a user generates 20 user action per agent send interval (the default interval time is 2 minutes) and each user action consists of ten nodes: a root node and nine subnodes. The average node size is approximately 150 bytes. So ten notes add up to 1500 bytes (10 * 150) and 20 user action add up to ~30 KB (20 * 1500 bytes). Normal user actions consist of less than ten nodes per action, so overhead should be less than in this example. Also the agent uses gzip compression to reduce the data traffic to the server.

How to instrument an Android app

Related pages

Android Agent manual instrumentation API

Android Agent troubleshooting and FAQ

Auto-instrumentation for Android

Gradle plugin for Android

Migrating from previous Android Mobile Agent versions

How to instrument an iOS app

The recommended way to instrument an Android app is to use the Gradle plugin. You can also use either Android auto-instrumentation or the Android Agent manual instrumentation API if more detailed instrumentation is required.

Auto-instrumentation for Android

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Auto-instrumentation from the command line

Auto-Instrumentation features

Mobile agent start-up

Crash reporting

Lifecycle monitoring

User action monitoring

Web request monitoring

Additional customization

Full property list

Requirements

Limitations

Multi-dex applications

Related pages

Gradle plugin for Android

Android Agent manual instrumentation API

How to instrument a hybrid app

Android Agent troubleshooting and FAQ

Auto-instrumentation for iOS

Downloads

Latest update

External links

Android Multidex

ART and Dalvik

Android Plugin DSL Reference

ApkTool

OkHttp 3

Android auto-instrumentation lets anyone instrument Android applications for monitored using the mobile agent. The instrumentation process is an automated way to add the mobile agent to an application without manually modifying the source code. The mobile agent is added in the auto-instrumentation process to the application. Data collection is the same with an auto-instrumented application as it is with a manually instrumented application.

There are two options how you can start the auto-instrumentation process:

- Call `instrument.<cmd_or_sh> apk=<app-name>.apk prop=<prop-name>.properties` from the command line.
- Integrate the auto-instrumentation process into your Gradle build process with the Dynatrace Gradle plugin

Auto-instrumentation from the command line

See the Android system requirements before running the auto-instrumentor the first time.

1. Download the newest mobile agent. The file name is `dynatrace-mobile-agents-<version>.zip`. Unpack it to a directory of your choice.
2. Copy the `APK-Instr.properties` property template file from the `<MobileAgentUnzipDir>/Android/auto-instrumentor` directory to a directory of your choice, and rename it.

3. Edit the properties file to customize the auto-instrumentation process. The `DTXApplicationID` and `DTXAgentStartupPath` properties are mandatory.
4. Run the `instrument` command with the `apk` and `prop` parameters: `instrument. <cmd_or_sh> apk=<app-name>.apk prop=<prop-name>.properties`.
5. Sign the instrumented APK `<pathToYourApk>/<nameOfAPK>/dist/<nameOfAPK>.apk` with your debug or Google playstore certificate, or use the `<pathToYourApk>/<nameOfAPK>/dist/<nameOfAPK>-final.apk` APK, which is signed with the auto-instrumentor debug certificate.
6. Start AppMon, including the web server (master) agent and your instrumented web or app server.
7. Install and run your instrumented APK file.

Auto-Instrumentation features

This level of instrumentation provides visibility into:

- Automatic mobile agent startup. You must use instrumentation keys to configure the automatic startup.
- Lifecycle data.
- Crash detection and reporting.
- Real user experiences of the flow within the application, such as responsiveness on user actions like a button touch.
- Web requests – by tagging them.
- Web request events, such as reporting execution time and identifying third-party requests.

Mobile agent start-up

The auto-instrumentor inserts the mobile agent start-up into the beginning of the application start procedure. Therefore you can't control the start-up parameters with manual instrumentation, because the agent ignores multiple start-up calls.

The `DTXApplicationID` and `DTXAgentStartupPath` properties are mandatory, because they are needed for the `Dynatrace.startup()` call. For the `DTXApplicationID` property, specify a unique application ID. For the `DTXAgentStartupPath` property, specify the address of your instrumented web or application server. For example:

```
DTXApplicationID=YourUniqueAppId
DTXAgentStartupPath=http://<instrumentedWebOrAppServer>[:portIfNot80]
```

Specifying an HTTPS address in the `DTXAgentStartupPath` property lets the agent verify the server certificate and the host name. The agent communication with the instrumented web or application server fails when verification steps can't complete successfully.

If you don't have a root certificate for your instrumented web or application server, you must provide a BKS keystore file, which contains the certificate(s) for SSL communication. Add the keystore file to the `raw` directory of the APK and specify the BKS file name (without the file extension) in the `DTXBKSFileName` property. You also must specify the keystore file password in the `DTXBKSPassword` property. For example:

```
# for file raw/mykeystore.bks use
DTXBKSFileName=myKeyStore
DTXBKSPassword=myPassword
```

The keystore password is not secured and can be retrieved by decompiling the application.

Another option is to deactivate the certificate validation with the `DTXAllowAnyCert` property. Host name verification can't be deactivated.

Crash reporting

Crash reporting is enabled by default. The mobile agent captures all unhandled exceptions/errors and sends the crash report immediately to the server. The Android crash report includes the occurrence time and the full stack trace of the exception. The crash report is attached to the user action that has triggered the exception.

Use the `DTXCrashReportingEnabled` property to disable crash reporting. Then the auto-instrumentor does not insert the method call `Dynatrace.enableCrashReporting(boolean sendCrashData)` into your applications start procedure. If you deactivate crash reporting, the agent ignores the corresponding value from your user experience configuration.

Lifecycle monitoring

The auto-instrumentor automatically instrument all activities listed in the `AndroidManifest.xml` file that are not manually instrumented. Use the `DTXInstrumentLifecycleMonitoring` property to deactivate lifecycle instrumentation. You can find more information in the chapter Collected lifecycle data.

User action monitoring

Auto-instrumentation allows you to automatically detect user actions. Therefore the auto-instrumentor instruments the following listeners and methods:

- Android
  - method: `android.app.Activity.onOptionsItemSelected`
  - `android.view.MenuItem$OnMenuItemClickListener`
  - `android.view.View$OnClickListener`
  - `android.widget.AdapterView$OnItemClickListener`
  - `android.widget.AdapterView$OnItemSelectedListener`
- Android Support
  - `android.support.v4.view.ViewPager$OnPageChangeListener`
  - `android.support.v4.widget.SwipeRefreshLayout$OnRefreshListener`
- ActionBarSherlock
  - method: `com.actionbarsherlock.app.SherlockActivity.onOptionsItemSelected`
  - `com.actionbarsherlock.internal.widget.IcsAdapterView$OnItemSelected`
  - `com.actionbarsherlock.view.MenuItem$OnMenuItemClickListener`

While an user action is open, the mobile agent adds all triggered events such as web requests and crashes to this user action. But most user actions are generated in the main thread and their web requests are triggered in a background thread after the user actions are closed. Normally these web requests would not be added to the user action. Therefore the auto-instrumentor configures a time frame. In this time frame the mobile agent adds web requests to the closed user action. You can set this time duration in milliseconds with the `DTXAutoActionTimeoutMilliseconds` property. The default value for this property is `500 ms` and the valid range is `100 ms - 5,000 ms`.

You can also configure the maximum duration of the user action with the `DTXAutoActionMaxDurationMilliseconds` property. The property default value is `60,000 ms` (one minute) and the valid range is `100 ms - 540,000 ms`. After the first time frame expires the mobile agent keeps the user action until all outstanding web requests or child actions complete, or the user action maximum duration is reached. In this waiting time frame the mobile agent does not accept web requests. These web requests are added to other user actions. When the maximum duration is reached, then the mobile agent closes the user action and all outstanding web requests or child actions.

Note

You can only configure one value for the `DTXAutoActionTimeoutMilliseconds` and `DTXAutoActionMaxDurationMilliseconds` properties, and these values must fit all user actions on all devices.

By default, the mobile agent discards empty user actions to reduce the amount of data that is sent to the server. If you are interested in these user actions, you can change this behavior with the `DTXSendEmptyAutoAction` property. You can also deactivate user action monitoring with the `DTXInstrumentAutoUserAction` property.

Note

The auto-instrumentor only instruments your byte-code. It does not instrument listener methods defined in `xml` files. You must manually instrument these listener methods.

Also, the auto-instrumentor does not instrument controls placed in a `WebView`. See How to instrument a hybrid app for more information.

Web request monitoring

The auto-instrumentor activates web request tagging and timing for your web requests. The mobile agent automatically truncates the query from the captured URL and only reports the domain name and path of your web requests.

The following list contains the supported HTTP frameworks:

- Apache HttpClient
  - DefaultHttpClient
  - AndroidHttpClient
- HttpURLConnection
- OkHttp 3

Note

The auto-instrumentor does not support OkHttp 3 library obfuscation. You must add the following rule to your ProGuard rules file:

```
-keep class okhttp3.** { *; }
```

The mobile agent automatically adds your web requests to the user action that triggered these web requests. For Web requests that can't be assigned to a user action, mobile agent generates a new action and attach the web request to it. The name of this action contains the key word `WebRequest` and the domain name of the web requests. For example `WebRequest(www.domain.com)`. Use the `DTXShowFullWebRequestURL` property to replace the name of this wrapping action with the full URL. There is an action name length limit and the web request URL is truncated if it exceeds this limit. The `DTXShowFullWebRequestURL` property only changes the name of the wrapping action. The web request event is not affected by the property and does not show the full URL.

Deactivate web request tagging and timing with the `DTXInstrumentWebRequestTagging` and `DTXInstrumentWebRequestTiming` properties. Because web request tagging is a precondition for web request timing, you can't deactivate tagging without deactivating timing. The auto-instrumentor ignores the `DTXInstrumentWebRequestTagging` property value when the `DTXInstrumentWebRequestTiming` property is activated.

Note

When you use auto-instrumentation and activate web request and user action monitoring, is it recommended to renounce manual web request instrumentation for the supported HTTP frameworks. Otherwise, your implementation can have unexpected results.

Additional customization

Auto-instrumentation instruments all packages by default. To skip third party content, you can use the `DTXExcludePackages` property to exclude packages, classes, or methods. For example:

```
DTXExcludePackages=com.xyz.IncludedClass.excludeThisMethod,com.xyz.ExcludedClass
```

If you only want to instrument your application package, use the `DTXIncludePackages` property to select the packages and deactivate `DTXIncludeAllPackages`. You can't include packages using the `DTXIncludePackages` property, if they are already excluded by the `DTXExcludePackages` property.

```
DTXIncludeAllPackages=false
DTXIncludePackages=com.this.pkg,com.that.pkg
```

To refresh the uninstrumented application with the instrumented application on Google Play, you must change the version number. By default, the auto-instrumentator does not change the version code and name. Use the `DTXVersionCode` and `DTXVersionName` properties to change these two values. If you have build a new APK file with a new version name and code, then you do *not* have to change these values, because the auto-instrumentor uses your new version name and code.

Activate the `DTXInstrumentGPSLocation` property to monitor the user location. This lets the auto-instrumentor instrument all your location listeners, and appends the captured user position to the sent data. The agent captures only three fractional digits. Your application must request the location permission and implement the location listener. The auto-instrumentor supports the following location listeners:

- `android.location.LocationListener`
- `com.google.android.gms.location.LocationListener`

Set the `DTXLogLevel` property to the value `debug` to activate the agent and auto-instrumentor debug logs. The auto-instrumentor and mobile agent log additional information in this mode that helps support troubleshoot problems. It is recommended recommend to remove this property when building your playstore or production application, because the additional logging can slow down the application. The auto-instrumentor creates a log file `AIA_<timestamp>.log` in the `log` directory of your working directory, depending on the terminal where you execute the `instrument` command. The mobile agent logs are avialable through the `logcat` tool or view.

Full property list

The following list shows the properties used to fine-tune your instrumentation:

Key	Key Type	Description

`DTXApplicationID`	String	This key's value is used as your applic to identify it on the AppMon Server. If t not present, the auto-instrumentor thro error.
`DTXAgentStartupPath`	String	This key with a valid value lets the mob start automatically, using this value as `serverUrl`, and ignores any manua `Dynatrace.startup` calls in you The `serverUrl` needs to start with transport mechanism to use (`http:/` or `https://`). If it is not present, th instrumentor throws an error.
`DTXAllowAnyCert`	boolean	This key is equivalent to the `useAny` parameter in the `Dynatrace.sta` call. If set to `true`, all certificates are accepted. If set to `false`, valid certifi from known certificate authorities are a Self-signed certificates (see `DTXBKSFileName` and `DTXBKSPassword`) are checked fo The default value is `false`.
`DTXBKSFileName`	String	This key (no file extension) defines the a BKS keystore file (must be in the APK directory), which contains a (self-signe certificate and which is used as an add anchor to validate HTTPS communicati key is needed if `DTXAllowAnyCer` `false` and a self-signed certificate is the server.
`DTXBKSPassword`	String	The password for the BKS keystore file `DTXBKSFileName`).
`DTXCrashReportingEnabled`	boolean	The default value is `true`. Set to `fal` deactivate crash reporting.
`DTXInstrumentLifecycleMonitoring`	boolean	The default value is `true`. Set to `fal` disable automatic lifecycle monitoring.
`DTXInstrumentWebRequestTagging`	boolean	The default value is `true`. Enables we request tagging for all supported HTTP frameworks. Set to `false` to disable automatic web request tagging.
`DTXInstrumentWebRequestTiming`	boolean	The default value `true`. Enables web timing for all supported HTTP framewo to `false` to disable automatic web re timing. If web request timing is enabled value from `DTXInstrumentWebRequestTa` is ignored, because web request taggi precondition for web request timing.

`DTXInstrumentAutoUserAction`	boolean	The default value `true`. If enabled, th agent automatically creates user actior user interactions such as button press to `false` to disable automatic user ac creation.
`DTXAutoActionTimeoutMilliseconds`	int	The default value is `500 ms`. Sets the for how long a particular automatic use is active. The purpose of this setting is all web requests, that happen when an automatic user action is active. If the a user action has completed web reques mobile agent leaves the action at the e this time. The minimum allowed value is `ms`, the maximum allowed value is `500` (five seconds).
`DTXAutoActionMaxDurationMilliseconds`	int	The default value is `60000 ms` (60 s Sets the value for how long a particular automatic user action is retained befor deleted. The purpose of this setting is all web requests, that happen when an automatic user action is active. If the a user action has pending web requests (because they are taking a long time to complete), the mobile agent waits for th amount of time for the web requests to complete before leaving the user actio minimum allowed value is `100 ms`, th maximum allowed value is `540000 m` minutes).
`DTXSendEmptyAutoAction`	boolean	The default value is `false`. If enable mobile agent sends automatic user act do not contain any child elements.
`DTXMultiDexKeep`	String	Defines a list of classes and packages should stay in the primary `.dex` file. T instrumentor also analyzes the depend of these classes and ensures these dependecies stay in the primary `.dex` Separate the list elements with a comm mark packages with an asterisk.
`DTXMultiDexKeepFile`	String	Defines the absolute path to a file cont list of classes that should stay in the pr `.dex` file. It is recommend to define th `maindexlist.txt` file, which is g by the Android Gradle plugin.
`DTXPrimaryDexLimit`	int	Defines the limit of referenced methods primary `.dex` file (before byte-code instrumentation). The default value is a 62K.
`DTXSecondaryDexLimit`	int	Defines the referenced methods limit fo secondary `.dex` files (before byte-co instrumentation). The default value is r 64K.
`DTXMonitorCookie`	String	The given value is automatically addec cookie to the http requests of the agen communication, so they can pass your infrastructure's requirements.

`DTXSetCookiesForDomain`	String	For hybrid applications using the JavaS agent or JavaScript bridge, cookies ne set for each instrumented domain or se application communicates with. You ca domains, host or IP addresses. Domair sub-domains must start with a dot. Sep the list elements with a comma.
`DTXAllowFileCookies`	boolean	The default value is `false`. Set to `tr` use file cookies.
`DTXNameUseValueOf`	String	For KonyOne based applications, use property to use the “serviceID” runtime the operation name, which in turn is us the wrapping action name.
`DTXIncludeAllPackages`	boolean	The auto-instrumentor instruments all Set the value to `false`, if you only wa instrument a part of your app (see `DTXIncludePackages`). The def value is `true`.
`DTXIncludePackages`	String	Select a list of packages which should instrumented. Separate the packages with a comma.
`DTXExcludePackages`	String	Select a list of packages, classes, or m which should not be instrumented. Sep the list elements with a comma.
`DTXVersionCode`	int	The default value is the version code v your APK. The auto-instrumentor chan instrumented APK file version code if y different value.
`DTXVersionName`	String	The default value is the version name your APK. The auto-instrumentor chan instrumented APK file version code if y different value.
`DTXInstrumentGPSLocation`	boolean	The default value is `false`. If set to `t` the auto-instrumentor instruments your `LocationListener` classes and the captured location as metric to the s
`DTXLogLevel`	String	Default logs of the mobile agent can’t b deactivated. Debug logs can be activat the value `debug`.

Requirements

- Supported OS versions (dependence on Google Android SDK system requirements):
  - Windows Vista to 10; other versions may work.
  - Ubuntu 12 and 14, 32 or 64 bits with the right ia32-libs installed; other versions and Linux flavors may work.
  - Mac OSX 10.8 to 10.9; other versions may work.
- Java SDK 1.8

- On Linux and OSX make sure that `<MobileAgentUnzipDir>/Android/auto-instrumentor/instrument.sh` has the execute permission `chmod +x instrument.sh`.
- Install ia32-libs on 64 bit Linux.

Expand for ia32-lib installation instructions

For Ubuntu later than 13.10:
```
sudo apt-get install ia32-libs
```

Later Ubuntu:
```
sudo dpkg --add-architecture i386
sudo apt-get update
sudo apt-get install libncurses5:i386 libstdc++6:i386 zlib1g:i386
```

If you want to use a different distribution you must check out installation commands.

Limitations

- The auto-instrumentor has to add the agent code into the primary `.dex` file. There are some limitations for multi-dex applications.
- If you obfuscate your application and you use both manual instrumentation and auto-instrumentation, you must exclude the `Dynatrace.jar` library from obfuscation. See the section ProGuard Settings for more information.
- The keystore password, if specified, is not secured. It is just as if you coded the password into the application. A possible attacker can retrieve the password by decompiling the application.
- Auto-instrumentation only provides limited information for hybrid applications. See How to instrument a hybrid app for more information.
- Avoid using special characters and spaces in file and path names (such as ( ) [ ] $ % *). If you are having trouble instrumenting your app, rename your file to use only alphanumeric characters (a-z, A-Z, 0-9) and the dot character (.) and try again.
- APK protection tools (like DexGuard) are not fully supported.

Multi-dex applications

There is a limit on the number of referenced methods in a given `.dex` file. The specification limit for this number is 65,536. Android refers to that limit as the 64K reference limit.

Android introduced the ART runtime with Android 5.0, which natively supports multidex applications. But for platforms prior to Android 5.0, Android uses the Davlik runtime, which only supports single-dex applications. With the multidex support library lets you use additional `.dex` files after the `MultiDex.install()` task completes. Therefore all classes loaded at the app start must be part of the primary `.dex` file (`classes.dex`).

Because the auto-instrumentor inserts the agent start-up routine into the application start procedure, the agent code has to be part of the primary `.dex` file. The auto-instrumentor moves some classes from the primary `.dex` file into another `.dex` file if there is not enough space in the primary `.dex` file, . For most applications, the default settings should work. But for some apps you have to fine-tune the settings.

> **Note**
>
> The auto-instrumentor analyzes class dependencies based on the byte code of for application start procedure. If you hide your dependencies, for example with reflection, the auto-instrumentor can't detect all your application start dependencies. If you use the Android Gradle properties `multiDexKeepFile` and/or `multiDexKeepProguard`, adding the specified classes also to the corresponding auto-instrumentor properties is recommended, or use the `DTXMultiDexKeepFile` property as descripted in the next section.
>
> You should test the application start on a device with Android 4.4 or lower. If the application crashes with a `java.lang.NoClassDefFoundError`, then you have to add the missing class to the corresponding auto-instrumentation properties.

Fine-tune the multidex preparation step

Following properties fine-tune auto-instrumentation for mutlidex applications.

- Use the `DTXPrimaryDexLimit` and `DTXSecondaryDexLimit` properties to configure the limit of referenced methods for the primary and the secondary `.dex` files.
- Use the `DTXMultiDexKeep` and `DTXMultiDexKeepFile` properties to configure the auto-instrumentor internal main dex list.

The instrumentation step increases the number of referenced methods (depending on your implementation) and fails if the number of referenced methods exceeds the 64K limit. The auto-instrumentor also adds the agent code into the primary `.dex` file. For every `.dex` file, the number of referenced methods is analyzed and adapted before the instrumentation step. The auto-instrumentator adapts the number of referenced methods by moving a precalculated number of classes from the `.dex` file to another `.dex` file. Use the `DTXPrimaryDexLimit` and `DTXSecondaryDexLimit` properties to configure the final limit of referenced methods for this step. Use the `DTXPrimaryDexLimit` property to set the limit for the primary `.dex` file (`classes.dex`) and the `DTXSecondaryDexLimit` property sets the limit for the secondary `.dex` files, such as `classes2.dex`.

Currently, the auto-instrumentor can't calculate the number of added methods and therefore uses default values for the limits. The default value for `DTXPrimaryDexLimit` is around 62K and for `DTXSecondaryDexLimit` it is nearly 64K. This configuration should fit most applications and also keeps the changes to a minimum. If the auto-instrumentation fails for your application, then you have to update the two properties.

For the primary `.dex` file the auto-instrumentor generates an internal main dex list and does not move the classes to another `.dex` file. If using the Android Gradle plugin to build you APK file, set the `DTXMultiDexKeepFile` property. The Android Gradle plugin generates a `maindexlist.txt` file. This file contains a list of classes, which have to be part of the primary `.dex` file. You should specify this file via the `DTXMultiDexKeepFile` property. For example:

```
DTXMultiDexKeepFile=<path_to_project>/<project>/<module>/build/intermediates/multi-dex/<b
```

You can also specify you own class list in a file, to be part of the primary `.dex` file. Only one class per line is accepted. The class name must end with the `.class` suffix. Separate the package parts with a slash (for example `com/example/Foo.class`). You can only specify one file for the `DTXMultiDexKeepFile` property. The auto-instrumentor does not analyze the dependecies of the specified classes. You have to track them manually.

Use the `DTXMultiDexKeep` property if you don't want to track dependencies manually. The auto-instrumentor analyzes the specified classes and keeps these classes with their dependencies in the primary `.dex` file. Use the `DTXMultiDexKeep` to create a list of packages and classes. For example:

```
DTXMultiDexKeep=com.example.*, com.example2.Foo
```

This property is very useful when for resolving a `java.lang.NoClassDefFoundError` message on application start-up for an Android 4.x device. Add the classes (mentioned in the error message) to the `DTXMultiDexKeep` property. The auto-instrumentor detects the class dependencies.

Build an ajusted APK file

If you want to reduce the execution time of the auto-instrumentor, you can also fine-tune the Android build process to generate an adjusted APK file. With this file, you can skip the multidex preparation step from the auto-instrumentor. Ensure there is enough space for the mobile agent in the primary `.dex` file. You can reach this goal with the `--minimal-main-dex` or `--set-max-idx-number` option from the `dx` command. You can set this option with using additionalParameters property from the Android Gradle plugin.

`--minimal-main-dex` option: Use this option to have Android store classes only needed for the application start procedure in the primary `.dex` file.

```
dexOptions {
        additionalParameters '--minimal-main-dex'
}
```

`--set-max-idx-number` option: Use this option to have Android reduce the referenced method limit to the specified value. This setting is used for all your APK `.dex` files.

```
dexOptions {
        additionalParameters '--set-max-idx-number=62000'
}
```

> **Note**
>
> Before you modify your application settings, you should aware of the main dex list and how you can add missing classes to the main dex list.

Android Agent manual instrumentation API

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Getting Started

Crash reporting

Lifecycle instrumentation

User action monitoring

Web request monitoring

Advanced settings

ProGuard settings

Limitations

Overhead

Related pages

How to instrument an Android app

Android Agent troubleshooting and FAQ

Auto-instrumentation for Android

Gradle plugin for Android

Migrating from previous Android Mobile Agent versions

iOS Agent manual installation

External links

Android permissions

ProGuard

Getting Started

Do the following to setup Android apps for instrumentation:

- Configure your project
- Update your permissions
- Update your build script
- Start the agent

Configure your project

1. Download the latest mobile agent. The file name is `dynatrace-mobile-agents-<version>.zip` and unpack it to a directory of your choice.
2. Add the library `Dynatrace.jar` from the folder `Android/agent` to your project. In Android Studio, place the `Dynatrace.jar` file in the default library folder `<project_name>/<module_name>/libs`. Your module's `build.gradle` configuration imports it with the dependency `compile fileTree(dir: 'libs', include: ['*.jar'])`.
3. If you have a hybrid application that targets API 17 or higher, then you also have to include the `com.dynatrace.android.ext.jsi.jar` library from the `Android/extensions` folder.

Update your permissions

The mobile app must have the `INTERNET` and `ACCESS_NETWORK_STATE` permissions to use the Android mobile agent, as shown below.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.example
        <uses-permission android:name="android.permission.INTERNET"/>
        <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
...
```

The following permission is optional. See details in Lifecycle Instrumentation and Limitations below.

```
<uses-permission android:name="android.permission.GET_TASKS"/>
```

To report network and/or WiFi signal state:

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
```

> **Note**
>
> If permissions change for an app such as for instrumentation, users must run a manual update on the Google Play store once and can then accept the new permission level.

Update your build script

If the `compileSdkVersion` setting in your `build.gradle` file is 23 (Android 6.0) or above, then you must add the compile-time dependency for the Apache HTTP client library as shown below.

```
android {
    useLibrary 'org.apache.http.legacy'
}
```

Start the agent

Call the `Dynatrace.startup` method to initialize and start the mobile agent as shown below. You must provide a unique application id and the address of your instrumented web or application server as parameter values. Invoke the `Dynatrace.startup` method as early as possible. Using the `onCreate` method from your application class is recommended.

```
public class MyApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();
        Dynatrace.startup(this, "MyAppId", "http://myhost.mydomain.com", false, null);
    }
}
```

If specifying an HTTPS address in the `agentPath` parameter, the agent verifies the server certificate and the host name. The agent communication with the instrumented web or application server fails, when the verification steps can't be successfully completed. If you don't have a root certificate for you instrumented web or application server, you must provide a `KeyStore` object. This object mast hold the certificate chain of the web server you want to connect to. You can also deactivate the certificate validation with the `useAnyCert` parameter. Host name verification can't be deactivated.

```
KeyStore trusted = KeyStore.getInstance("BKS");
InputStream in = getResources().openRawResource(R.raw.mykeystore);
try {
    trusted.load(in, "myverysecretpassword".toCharArray());
} finally {
    in.close();
}
Dynatrace.startup(this, "MyAppId", "https://myhost.mydomain.com:443", false, trusted);
```

> **Note**
>
> If using BouncyCastle as your keystore instance, please be sure that the keystore is generated with BouncyCastle 1.45. The BouncyCastle 1.46 file format is not compatible with older versions of Android, see this Android AOSP issue for more details.

Crash reporting

Crash reporting is disabled by default. To activate or deactivate crash reporting, call the `Dynatrace.enableCrashReporting` method. If you deactivate crash reporting, the agent ignores the corresponding value from your user experience configuration.

When crash reporting is activated, the mobile agent captures all unhandled exceptions and errors and sends the crash report immediately to the server. The Android crash report includes the occurrence time and the full stack trace of the exception. When the crash occurs, while a lifecycle action is active, the mobile agent attaches the crash report to the lifecycle action.

Lifecycle instrumentation

By default, lifecycle data collection is **on**. You can turn it off in the user experience configuration file.

Complete the following tasks to enable lifecycle data capturing:

- Update the AndroidManifest.xml file
- Update the application class
- Update the activity classes

You can also use Auto-Instrumentation for Android to automatically perform this instrumentation.

Update the AndroidManifest.xml file

1. If the application `name` element does **not** specify an application class, add `android:name="com.dynatrace.android.app.Application"` and skip the next step Update the application class.
2. Optionally add the `GET_TASKS` permission. See section Update your permissions for more information.

The `GET_TASKS` permission detects application lifecycle transitions from foreground to background. This method provides the most reliable detection of application state changes, if lifecycle monitoring is enabled. It does not collect, store, or transmit which applications are running on a user's device. The mobile agent can only reliably determine the foreground/background transition, if **all** activities in an application extend `Dynatrace` activities. The `GET_TASKS` permission is only needed when your application supports Android versions prior to Android 5.0.

Update the application class

If your `AndroidManifest.xml` specifies an application class (the `android:name` attribute of the `application` element), you must derive it from the Dynatrace `com.dynatrace.android.app.Application` class. For example, change the import statement from `import android.app.Application` to `import com.dynatrace.android.app.Application`.

> **Note**
>
> Your lifecycle methods (like `onCreate`, …) from the application and activity classes must call their `super` methods.

Update the activity classes

Similar to the application class, your activity classes must derive from the corresponding Dynatrace activity class. Therefore, change the import statement by prefixing **`com.dynatrace.`** to the package name. For example, change `import android.app.Activity` to `import com.dynatrace.android.app.Activity`.

The `Dynatrace.jar` library supports the following activity classes:

- `android.app.Activity`
- `android.app.AliasActivity`
- `android.app.ListActivity`
- `android.app.ExpandableListActivity`
- `android.app.LauncherActivity`

Your app may contain activity classes, that are not part of this list. Such activities are `android.support.v7.app.AppCompatActivity`, `com.google.android.maps.MapActivity` or other activity classes from third party libraries. In this case you must generate a wrapper class for these activity classes. Therefore you have to execute the following steps:

1. Generate a class with the name of activity class in the package `com.dynatrace.android.app`.
2. Copy the code snippet below into your new class file.
3. Replace the activity name placeholder with the activity name.
4. Derive your activity classes from the new wrapper class.

Wrapper class template

```
package com.dynatrace.android.app;

import android.os.Bundle;

public abstract class <activity_name> extends <package_name>.<activity_name> {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityCreate(this, savec
        }

        super.onCreate(savedInstanceState);
    }

    @Override
    protected void onPostCreate(Bundle savedInstanceState) {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityPostCreate(this, s
        }

        super.onPostCreate(savedInstanceState);
    }

    @Override
    protected void onRestart() {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityRestart(this);
        }

        super.onRestart();
    }

    @Override
    protected void onStart() {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityStart(this);
        }
```

```
        super.onStart();
    }

    @Override
    protected void onResume() {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityResume(this);
        }

        super.onResume();
    }

    @Override
    protected void onPostResume() {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityPostResume(this);
        }

        super.onPostResume();
    }

    @Override
    protected void onPause() {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityPause(this);
        }

        super.onPause();
    }

    @Override
    protected void onStop() {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityStop(this);
        }

        super.onStop();
    }
```

```
    @Override
    protected void onDestroy() {
        if (LcContext.getInstance().isLifecycleInEffect()) {
            LcContext.getInstance().getActivityLcCallbacks().onActivityDestroy(this);
        }

        super.onDestroy();
    }

}
```

Collected lifecycle data

With lifecycle instrumentation, the mobile agent automatically collects the following data and renders it as actions:

- **Application Start Event**: Represents the time from `Application.onCreate()` to `Activity.onPostResume()` of the first activity displayed. If the first activity does not extend a `Dynatrace` activity, the application-start event time is not recorded correctly.
- **Activity Display**: Represents the time from `Activity.onCreate()` to `Activity.onPostResume()`.
- **Activity Redisplay**: Represents the time from `Activity.onStart()` to `Activity.onPostResume()`.

These activities' display/redisplay actions can also contain web requests or crash events.

User action monitoring

Create user actions with the `Dynatrace.enterAction` method and close them with the `leaveAction` method. The resulting action is visible as a mobile user action PurePath in the AppMon client. For example, you can use the `enterAction` and `leaveAction` methods to determine how long it takes to pull content from a remote server. You get visibility to the corresponding backend transactions within the action scope. Other event types (`reportEvent`, `reportValue` and `reportError`) can be reported within the context of the action object, thus creating a series of events within an action scope.

The `reportError` event is different from the `reportValue` event in that it is specifically identified as an error type of event. The integer value most likely contains an error code. You can also use the `Dynatrace.reportError` method to create a stand-alone error action.

> **Note**
>
> When the parent action is closed, the mobile agent automatically closes all nested/child actions of the parent action. Keep in mind, that the mobile agent discards the captured action data (including child actions), when the action is not closed.

User action PurePath example

The code below shows a sample instrumentation of the fictional method `search`, which makes a web request to an instrumented application server and parses the received result.

```
public boolean search(String query) {
        // [1a] start outer/parent action
        DTXAction searchAction = Dynatrace.enterAction("search");

        // [2a] start first inner action
        DTXAction searchRequestAction = Dynatrace.enterAction("search request", searchAct

        try {
                // [3] tag a web request inside the inner action
                URL url = new URL("http://<host_name>/ajax?searchquery=" + query);
                HttpURLConnection connection = (HttpURLConnection) url.openConnection();
                searchRequestAction.tagRequest(connection);

                InputStream stream = connection.getInputStream(); // can throw an excepti

                // [4] add a named event
                searchRequestAction.reportEvent("read input stream");
                readStream(stream);

                // [2b] end first inner action
                searchRequestAction.leaveAction();
```

```
        } catch (Exception e) {
                // [7] add an error
                searchRequestAction.reportError("connection failed", e);

                // [2b] end first inner action
                searchRequestAction.leaveAction();

                // [1b] end outer action
                searchAction.leaveAction();

                return false;
        }

        // [5a] start second inner action
        DTXAction parseAction = Dynatrace.enterAction("parse result", searchAction);

        results = parseResult();

        // [6] add a custom value
        parseAction.reportValue("list size", results.size());
        // [5b] end second inner action
        parseAction.leaveAction();

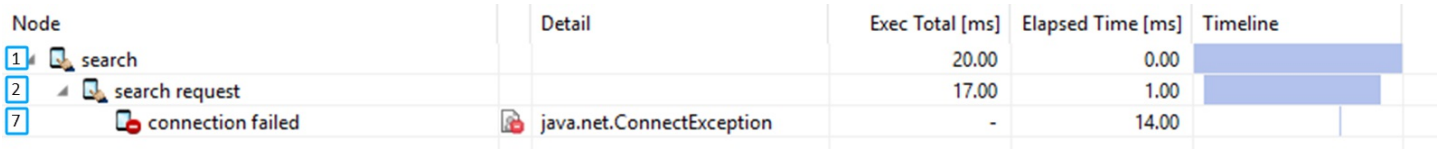
        // [1b] end outer action
        searchAction.leaveAction();

        return true;
}
```

The figures below show the two differnt user actions, that are generated by the provided sample instrumentation.

	Node	Detail	Exec Total [ms]	Elapsed Ti...	Timeline
1	search		24.00	0.00	
2	search request		12.00	1.00	
3	(200) ' host_name /ajax/'		3.00	9.00	
	doFilter(ServletRequest request, Serv	/ajax/	2.85	9.70	
4	read input stream		-	11.00	
5	parse result		8.00	15.00	
6	list size	5	-	22.00	

User action example with a successful web request



Node	Detail	Exec Total [ms]	Elapsed Time [ms]	Timeline
1 search		20.00	0.00	
2 search request		17.00	1.00	
7 connection failed	java.net.ConnectException	-	14.00	

User action example with an unsuccessful web request

Web request monitoring

The following list shows the supported HTTP frameworks:

- Apache HttpClient
- HttpURLConnection

For all other HTTP frameworks, ensure that the Dynatrace mobile tag is placed on the HTTP request.

Tagging

Web request tagging lets the AppMon server identify mobile traffic and correlates web request server-side PurePath data to a corresponding mobile user action. Therefore the mobile agent must add an additional HTTP header to your web request. The web server agent analyzes this HTTP header, and the AppMon server generates a corresponding web request PurePath. For web requests to unmonitored servers such as third party servers, no web request PurePath will be generated.

There are two ways to assign a web request to an action.

- Use the `DTXAction.tagRequest` methods of a specific action to assign a web request to this action.
- Use the `Dynatrace.tagRequest` methods to let the mobile agent assign the web requests to an appropriate user action.

The mobile agent will use the following rule set:

1. If the current thread has an open action, when the request is tagged, then the mobile agent assigns the web request to the open action.

2. If the current thread has no open action, the AppMon server assigns the web request to an action that calls its `leaveAction()` method in the same thread where the tagging is done. The tagging time must fall between the action enter and leave times.
3. If neither of the preceding cases applies, the web request is not assigned to any action. In this case, the server-side PurePath data is not linked to a mobile user action, but PurePath data can be identified as mobile traffic in the transaction flow.

The `tagRequest(java.net.HttpURLConnection)` and `tagRequest(org.apache.http.HttpRequest)` methods generate and add the correct HTTP header to your web request automatically.

```
DTXAction action = Dynatrace.enterAction("<action_name>");
URL url = new URL("http://myhost.mydomain.com/example");
HttpURLConnection connection = (HttpURLConnection) url.openConnection();
action.tagRequest(connection);

// handle web request

action.leaveAction();
```

```
DTXAction action = Dynatrace.enterAction("<action_name>");
HttpGet get = new HttpGet("http://myhost.mydomain.com/example");
action.tagRequest(get);

// handle web request

action.leaveAction();
```

For the Apache HttpClient library, you can also use the `Dynatrace.registerRequestTaggingInterceptor(org.apache.http.impl.client.` method. This method adds the additional HTTP header to all requests executed with the given `DefaultHttpClient`. The interceptor overwrites any existing tag placed on the request.

For third party libraries, you must use the `getRequestTag()` general method to generate the correct tag value. You must manually add the needed HTTP header to your web request. Use the `Dynatrace.getRequestTagHeader()` method to get the header key, as shown below for the OkHttp 3 library.

```
DTXAction action = Dynatrace.enterAction("<action_name>");
Request request = new Request.Builder()
                .url("http://myhost.mydomain.com/example")
                .addHeader(Dynatrace.getRequestTagHeader(), action.getRequestTag())
                .build();


// handle web request

action.leaveAction();
```

> **Note**
>
> Web request tagging does not measure network contribution time. The generated web request PurePath only contains the timing values from the server agent. You must use web request timing for full visibility.

Timing

Web request timing is an extension of web request tagging. It lets you monitor the web request on the mobile side. It also monitors web requests to unmonitored servers.

You must use web request tagging and create manual timings by using the `WebRequestTiming` class. The `Dynatrace.getWebRequestTiming(java.net.HttpURLConnection)` and `Dynatrace.getWebRequestTiming(org.apache.http.HttpRequest)` methods tag your web request (like the `Dynatrace.tagRequest` methods) and provide the corresponding `WebRequestTiming` object. For third party libraries, you must use the general method `Dynatrace.getWebRequestTiming(String)` and generate the parameter value with the `Dynatrace.getRequestTag()` method. You must manually add the needed HTTP header to your web request.

Call the `startWebRequestTiming` method before the web request is sent to the server. Call the `stopWebRequestTiming` method after receiving and reading the HTTP response. The mobile agent generates a web request action with measured timing values and the AppMon server appends the server PurePath data to this web request action. You can see network timings in the `Network Contribution` column of the User Actions PurePath dashlet and other dashlets.

Examples

Apache HttpClient example with a HttpGet request

```
// [1] First, create an action
DTXAction webAction = Dynatrace.enterAction("time service");
String url = "http://myhost.mydomain.com/TimeService?tz=CET";
HttpClient httpclient = new DefaultHttpClient();
HttpGet httpget = new HttpGet(url);

// [2] Once we have our HttpGet object, we can tag it automatically and receive
// a WebRequestTiming instance
WebRequestTiming timing = Dynatrace.getWebRequestTiming(httpget);

try {
        // [3] We try to wrap the timings as closely as we can to the actual network
        // communications.  We want to avoid any client-side processing in the timings.
        // Here, we call startWebRequestTiming() on the timing object, and then execute
        // the request
        timing.startWebRequestTiming();
        HttpResponse response = httpclient.execute(httpget);

        // [3a] If your response contains an HttpEntity, you will want to fetch that
        // within the timing block.  For small requests, this is usually not necessary.
        HttpEntity entity = response.getEntity();
        if (entity != null) {
                InputStream stream = entity.getContent();
                readStream(stream);
        }

        // [4] When we are done requesting the resource, call stopWebRequestTiming() on
        // your timing object.  This finalizes the timing and transmits it to the server
        // for correlation.  If you include the HttpResponse object, the HTTP response co
```

```
        // and status text will be reported as well.
        timing.stopWebRequestTiming(response);

} catch (Exception e) {
        // Attach the exception to the action
        webAction.reportError("Exception", e);

        // user-defined exception handling ...
} finally {
        // [5] It is also good practice to call stopWebRequestTiming() in the finally blo
        // so you still receive network timings in the event of any communications errors
        timing.stopWebRequestTiming();

        // [6] Lastly, leave the action
        webAction.leaveAction();
}
```

HttpURLConnection example

```
HttpURLConnection conn = null;
WebRequestTiming timing = null;

// [1] First, create an action
DTXAction webAction = Dynatrace.enterAction("time service");

try {
        URL url = new URL("http://myhost.mydomain.com/TimeService?tz=CET");
        conn = (HttpURLConnection) url.openConnection();

        // [2] Once we have our connection object, we can tag it automatically and receiv
        // a WebRequestTiming instance
        timing = Dynatrace.getWebRequestTiming(conn);

        // [3] Call startWebRequestTiming() to begin the timing, and then handle the inpu
        // stream from the connection
        timing.startWebRequestTiming();
        InputStream stream = new BufferedInputStream(conn.getInputStream());
        readStream(stream);

        // [4] Once we're done reading, we can stop the timing
```

```
        // [4] Once we're done reading, we can stop the timing
        timing.stopWebRequestTiming();

} catch (Exception e) {
        // Attach the exception to the action
        webAction.reportError("Exception", e);

        // user-defined exception handling ...
} finally {
        // [5] Lastly finalize the timing and the action in the finally block.
        // This will allow us to receive timings and close the action even if there is an
        // exception thrown above
        if (timing != null) {
                timing.stopWebRequestTiming();
        }
        webAction.leaveAction();

        if (conn != null) {
                conn.disconnect();
        }
}
```

synchronous OkHttp3 call example

```
OkHttpClient client = new OkHttpClient();

// [1] First, create an action
DTXAction webAction = Dynatrace.enterAction("time service");

// [2] Generate a new request tag
String requestTag = Dynatrace.getRequestTag();

// [3] Associate the timing with the tag
WebRequestTiming timing = Dynatrace.getWebRequestTiming(requestTag);

// [4] Fetch the request tag header name
String requestTagHeaderName = Dynatrace.getRequestTagHeader();

// [5] Define your okhttp3 request, this varies greatly depending on your implementation
Request request = new Request.Builder()
```

```
        .url("http://myhost.mydomain.com/TimeService?tz=CET")
        // [6] Define your headers for the okhttp3 request
        .addHeader(yourKey1, yourValue1)
        .addHeader(yourKey2, yourValue2)
        // [7] Add the dynatrace header key and the tag value to the request
        .addHeader(requestTagHeaderName, requestTag)
        .build();

// For web requests to unmonitored web servers:
// Add important values to the action (like url, status code, ...)
webAction.reportValue("URL", url);

try {
        // [8] Call startWebRequestTiming() to begin the timing, and then handle the
        // response body from the okhttp3 call
        timing.startWebRequestTiming();
        Response response = client.newCall(request).execute();

        // handle response ...
        String body = response.body().string();

        // [9] Once we're done reading, we can stop the timing
        timing.stopWebRequestTiming();

        // For web requests to unmonitored web servers:
        // Add important values to the action (like url, status code, ...)
        webAction.reportValue("HTTP status code", response.code());

} catch (IOException e) {
        // Attach the exception to the action
        webAction.reportError("Exception", e);

        // user-defined exception handling ...
}
finally {
        // [10] Lastly, finalize the timing and the action in the finally block.
        // This will allow us to receive timings and close the action even if there
        // is an exception thrown above.
        timing.stopWebRequestTiming();
        webAction.leaveAction();
```

```
}
```

asynchronous OkHttp3 call example

```
OkHttpClient client = new OkHttpClient();

// [1] First, create an action
final DTXAction webAction = Dynatrace.enterAction("time service");

// [2] Generate a new request tag
String requestTag = Dynatrace.getRequestTag();

// [3] Associate the timing with the tag
final WebRequestTiming timing = Dynatrace.getWebRequestTiming(requestTag);

// [4] Fetch the request tag header name
String requestTagHeaderName = Dynatrace.getRequestTagHeader();

// [5] Define your okhttp3 request, this varies greatly depending on your implementation
Request request = new Request.Builder()
        .url("http://myhost.mydomain.com/TimeService?tz=CET")
        // [6] Define your headers for the okhttp3 request
        .addHeader(yourKey1, yourValue1)
        .addHeader(yourKey2, yourValue2)
        // [7] Add the dynatrace header key and the tag value to the request
        .addHeader(requestTagHeaderName, requestTag)
        .build();

// For web requests to unmonitored web servers:
// Add important values to the action (like url, status code, ...)
webAction.reportValue("URL", url);

// [8] Call startWebRequestTiming() to begin the timing, and then handle the
// response body from the okhttp3 call
timing.startWebRequestTiming();
client.newCall(request).enqueue(new Callback() {
        @Override
        public void onFailure(Call call, IOException e) {
                // [9] We can stop the timing
                timing.stopWebRequestTiming();
```

```
                // Attach the exception to the action
                webAction.reportError("Exception", e);

                // user-defined exception handling ...

                // [10] Lastly, close the action
                webAction.leaveAction();
        }

        @Override
        public void onResponse(Call call, Response response) throws IOException {
                // handle response ...
                String body = response.body().string();

                // [9] Once we're done reading, we can stop the timing
                timing.stopWebRequestTiming();

                // For web requests to unmonitored web servers:
                // Add important values to the action (like url, status code, ...)
                webAction.reportValue("HTTP status code", response.code());

                // [10] Lastly, close the action
                webAction.leaveAction();
        }
});
```

SOAP call example

```
// [1] First, create an action
DTXAction webAction = Dynatrace.enterAction("time service");

// [2] Generate a new request tag
String requestTag = Dynatrace.getRequestTag();

// [3] Associate the timing with the tag
WebRequestTiming timing = Dynatrace.getWebRequestTiming(requestTag);

try {
        // [4] Define your SOAP request, this varies greatly depending on your
```

```
        // implementation
        SoapObject request = new SoapObject(namespace, methodName);
        SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(SoapEnvelope.V
        envelope.setOutputSoapObject(request);
        HttpTransportSE androidHttpTransport = new HttpTransportSE(url);

        // [5] Define the properties for the soap request
        List<HeaderProperty> list = new ArrayList<HeaderProperty>();
        list.add(new HeaderProperty(yourKey, yourValue));
        list.add(new HeaderProperty(yourKey2, yourValue2));

        // [6] Fetch the request tag header
        String requestTagHeaderName = Dynatrace.getRequestTagHeader();

        // [7] Add the header key and the tag value to the properties list
        list.add(new HeaderProperty(requestTagHeaderName, requestTag));

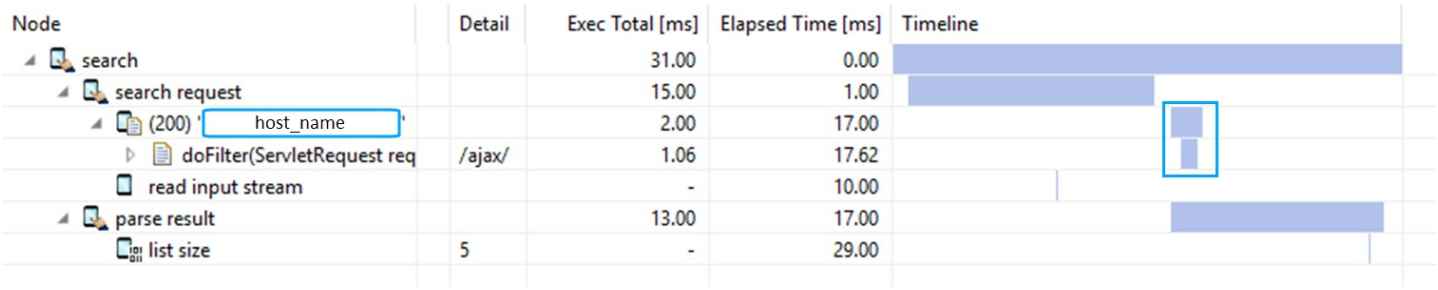
        // [8] Then start the timing, make the call to the SOAP service, and
        // end the timing
        timing.startWebRequestTiming();
        androidHttpTransport.call(action, envelope, list);
        timing.stopWebRequestTiming();
} catch (Exception e) {
        // Attach the exception to the action
        webAction.reportError("Exception", e);

        // user-defined exception handling ...
} finally {

        // [9] Lastly, finalize the timing and the action in the finally block.
        // This lets AppMon receive timings and close the action even if there
        // is an exception thrown above.
        if (timing != null) {
                timing.stopWebRequestTiming();
        }
        webAction.leaveAction();
}
```

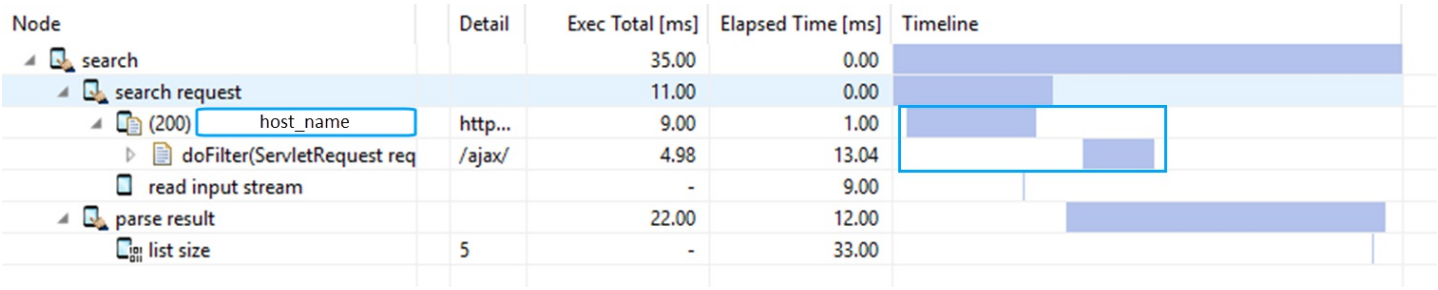
Difference between tagging and timing

When you use web request tagging, the AppMon server generates the corresponding mobile web request PurePath. The timing values are based on the server-side timing values. As shown below, the timing values of the mobile web request don't fit the mobile user action `search request`. This is because the mobile devices' system clocks differ from the server clock. The mobile agent tries to minimize this offset and provides timing values based on the server time.



Example with web request tagging

When you use web request timing, the mobile agent generates a mobile web request event and sends the data to the AppMon server. The AppMon Server links the PurePath data from the mobile and Server agent. This approach lets you measure the network contribution time and shows the correct time values for mobile nodes in the user action PurePath dashlet. Unfortunately the server node may differ from the mobile nodes because of the system clock difference, as shown below.



Example with web request timing

Advanced settings

Most mobile agent-specific settings can be set as parameter for the `Dynatrace.startup` method. Some advanced settings must be specified by adding a `Dynatrace.properties` file to the `assets` folder. The following table shows the available advanced configuration settings.

Key	Type	Default	Description

`DTXLogLevel`	String		Debug logs can be activated with the value `debug`.
`DTXAllowFileCookies`	boolean	false	Set the value to `true` to use file cookies.

Note

If you use a combination of manual and auto-instrumentation, the auto-instrumentor overrides the `Dynatrace.properties` file and your configuration is lost. In this case you must move your configuration settings to the auto-instrumentor properties file. The auto-instrumentor ensures that these properties are handled correctly and are part of the generated `Dynatrace.properties` file.

The `Dynatrace.properties` file does not support the other auto-instrumentation properties. You can't use these properties for manual instrumentation, because the corresponding instrumentation is missing.

ProGuard settings

ProGuard is a free Java class file shrinker, optimizer, obfuscator, and preverifier. It detects and removes unused classes, fields, methods, and attributes. It is recommended to exclude all mobile agent classes by adding the following lines to your `proguard.pro` file:

```
-keepattributes EnclosingMethod
-keep class com.dynatrace.** { *; }
```

Note

If you are using a combination of manual and auto-instrumentation, you must exclude all mobile agent classes from obfuscation. This step is necessary, because the auto-instrumentor inserts non-obfuscated code into your application.

If you only use manual instrumentation, you can use ProGuard to obfuscate the `Dynatrace.jar` library. For hybrid applications, you must exclude the class `JsNativeBridgeAPI17` from obfuscation. The class `JsNativeBridgeAPI17` is part of the `com.dynatrace.android.ext.jsi.jar` library .

```
-keep com.dynatrace.android.ext.jsi.JsNativeBridgeAPI17 { *; }
```

Limitations

- For Android 4.4 and earlier, the agent can't detect the correct foreground or background application state unless all activity classes are instrumented. You also have to add the `GET_TASKS` permission.
- The agent does not contain a specific web request tagging and timing API for third party web request frameworks. You have to use the general API methods (like `Dynatrace.getWebRequestTiming(String)`).
- There is limited support for multi-process applications, as each process starts its own agent and has its own session number.

Overhead

Adding the Android mobile agent (whether manually or via auto-instrumentation) to an application increases the size of the application by about 100 KB to 200 KB. The size increase depends on the implementation of your application and how effectly Android can store the mobile agent and the instrumentation code in your `.dex` files.

The impacts on app performance and mobile data usage are summarized in the general overhead section.

Gradle plugin for Android

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Supported Android Gradle plugin versions

Configure your build script

Signing

Optimizations

Limitations

Related pages

Android Agent manual instrumentation API

Auto-instrumentation for Android

How to instrument a hybrid app

Android Agent troubleshooting and FAQ

External links

Android Gradle Plugin

Release Notes for Android Gradle Plugin

Dynatrace Gradle plugin on JCenter

Dynatrace Gradle plugin on Maven Central

Android Plugin DSL Reference

Gradle DSL Reference

The Dynatrace Gradle plugin is based on the Android Gradle plugin. It inserts an `autoInstrument` task into the Android build task sequence. The Dynatrace Gradle plugin lets Android Studio users modify their `build.gradle` file in a simple way and activate auto-instrumentation with few steps. The Dynatrace Gradle Plugin is hosted on JCenter and Maven Central.

Supported Android Gradle plugin versions

Dynatrace Gradle plugin	Required Android Gradle plugin version
7.0.0.2327 - 7.0.0.2429	1.5 - 2.3.3

Note

Version 2.5 of the Android plugin for Gradle contains major changes that break the current Dynatrace Gradle plugin version. Please check this page for further instructions before you upgrade to the 2.5 version. Instructions will be provided once the plugin successfully adapts to the changes.

Configure your build script

Modify your module's `build.gradle` file to integrate the Dynatrace Gradle plugin into your build task sequence. First, add the remote repository and the Dynatrace Gradle plugin dependency. Then add the `dynatrace` block with your configuration values.

```
buildscript {
  repositories {
    jcenter()
  }
  dependencies {
    classpath 'com.dynatrace.tools:android:7.0.+'
  }
}

apply plugin: 'com.dynatrace.tools.android'
dynatrace {
  defaultConfig {
    applicationId 'MyExampleAppID'
    startupPath 'https://dev.example.com:443'
  }
}
```

The minimal configuration must contain the `applicationId` and the `startupPath` value. Specify a unique application ID with the `applicationId` property, and specify the instrumented web or application server address with the `startupPath` property.

Customize auto-instrumentation properties

Use the auto-instrumentation properties to customize your auto-instrumentation process. Set these auto-instrumentation properties via the `agentProperties` plugin property. The property syntax is a key-value list, where the elements are separated by a comma and the key-value pair is separated by a colon. For example:

```
dynatrace {
  defaultConfig {
    applicationId 'MyExampleAppID'
    startupPath 'https://dev.example.com:443'
    agentProperties 'DTXSendEmptyAutoAction': 'true', 'DTXLogLevel': 'debug'
  }
}
```

Combine with manual instrumentation

Add the Dynatrace OneAgent library via the Dynatrace Gradle plugin to use a combination of manual and auto-instrumentation to monitor your application. This allows you to automatically update the OneAgent library with the Gradle plugin, and ensures that the OneAgent library and the auto-instrumentation have the same version. To enable the manual instrumentation, add the `dynatrace.agent()` dependency. For example:

```
dependencies {
  compile dynatrace.agent()
}
```

Note

Do not obfuscate or optimize the Dynatrace agent library. See ProGuard configuration) for more information.

For multi-dex applications, ensure all agent classes are in the primary `.dex` file. Use the `multiDexKeepProguard` Android plugin property and add the line `-keep class com.dynatrace.android.** { *; }` to the provided file. For example:

```
buildTypes {
  release {
    multiDexKeepProguard file('<your_keep_file>')
  }
}
```

Use different product flavor configurations

If you use different product flavors, you can also specify different configurations for the auto-instrumentation task. For every product flavor the Gradle plugin checks if you've specified a configuration in the `productFlavors` block of the `dynatrace` block. If no configuration is available, the `defaultConfig` configuration is used.

```
dynatrace {
  defaultConfig {
    applicationId 'MyDefaultAppID'
    startupPath 'https://dev.example.com:443'
  }

  productFlavors {
    prod {
      applicationId 'MyProdAppID'
      startupPath 'https://prod.example.com:443'
    }
  }
}
```

Use a Gradle property to deactivate instrumentation

Currently there is no option to deactivate the plugin dynamically. Use a Gradle property to activate or deactivate the plugin. For example:

```
if(project.hasProperty('instrument')) {
  apply plugin: 'com.dynatrace.tools.android'
  dynatrace {
    defaultConfig {
      applicationId 'MyExampleAppID'
      startupPath 'https://dev.example.com:443'
    }
  }
}
```

You can activate the Dynatrace Gradle plugin by adding the `-Pinstrument` parameter to your build command. If you want to build an instrumented release version, execute the `gradlew assembleRelease -Pinstrument` command.

Signing

The Dynatrace Gradle plugin uses the signing configuration and logic from the Android Gradle plugin. You do not have to add any additional settings to the Dynatrace Gradle plugin.

```
android {
    ...
    defaultConfig { ... }
    signingConfigs {
        release {
            storeFile file("my-release-key.jks")
            storePassword "password"
            keyAlias "my-alias"
            keyPassword "password"
        }
    }
    buildTypes {
        release {
            signingConfig signingConfigs.release
            ...
        }
    }
}
```

The Dynatrace Gradle plugin stores the instrumented and signed app under the same name in the same location (`<project>/<module>/build/outputs/apk/<app_name>.apk`). The advantage of this approach is that the Run button from Android Studio can use the instrumented APK file and can install it on your device/emulator.

Optimizations

The auto-instrumentor must execute some preparation steps before instrumenting your APK file. You can reduce auto-instrumentation execution time by generating an adjusted APK file. This lets the auto-instrumentor skip some of the preparation steps and reduce the execution time. This optimization options are **optional**, because the auto-instrumentor has a built-in logic to execute similar steps.

You can set the jumbo mode option in the Android Gradle plugin to skip the internal auto-instrumentor step “string identifier adaption”. For example:

```
dexOptions {
  jumboMode true
}
```

This option reduces the auto-instrumentation execution time for multidex apps (and apps exceeding the 64K string identifier border).

For multi-dex application the auto-instrumentor has to ensure there is enough space for the agent in the primary `.dex` file. To do this, it moves some classes from the primary `.dex` file into another `.dex` file. You can skip this step, if you generate a primary `.dex` file that has enough space for the agent. You can archive this by using the `--minimal-main-dex` or `--set-max-idx-number` option of the Android Gradle plugin. See the Multi-dex applications section in Auto-instrumentation for Android for more information.

Limitations

- Do NOT use whitespaces in your project path.
- Do NOT use the Android Studio feature Instant Run.

See the Auto-instrumentation limitations section in Auto-instrumentation for Android for more information.

Instant Run

With Android Studio 2.0, Android introduced the new feature Instant Run. This feature significantly reduces the app update time. The Dynatrace Gradle plugin for Android does not support Instant Run APKs and the corresponding Gradle build process. You should disable the **Instant Run** setting. To do this, open the **Settings** and navigate to **Build, Execution, Deployment > Instant Run**. Then clear the **Enable Instant Run** check box.

Migrating from previous Android Mobile Agent versions

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Changes introduced with AppMon 2017 May

Changes introduced with AppMon 6.5

Changes introduced with AppMon 6.2

Related pages

Android Agent manual instrumentation API

Auto-instrumentation for Android

How to instrument a hybrid app

Android Agent troubleshooting and FAQ

Changes introduced with AppMon 2017 May

The migration steps for AppMon 2017 May are based on a 6.5 setup.

Manual instrumentation

- The `com.dynatrace.android.ext.jar` library has been removed. You must do your own lifecycle instrumentation for activity classes that are not part of the Android SDK. See Lifecycle instrumentation for more information.

Auto-instrumentation

- The `DTXApplicationID` property is mandatory. The auto-instrumentor does not use the application name as fallback. However, you can still use your application name as application ID value, if it is unique for your environment.
- The auto-instrumentor uses the APK Signature Scheme v2 with a debug certificate to sign your instrumented APK file
- The auto-instrumentor contains a new preparation step for multidex applications. See multi-dex applications for more information.
- The auto-instrumentor can monitor `OkHttp 3` web requests and `OkHttp 3`-based frameworks like `Retrofit2`.
- The Android auto-instrumentation wizard in the AppMon Client has been removed. You shouldn't use the wizard from a 6.5 AppMon client to instrument an application with an AppMon 2017 May mobile agent. This use case is not supported.

Dynatrace Gradle plugin

- Update the gradle plugin configuration to fetch an AppMon 2017 May mobile agent (use `classpath 'com.dynatrace.tools:android:7.0.+'`)

Changes introduced with AppMon 6.5

The migration steps for AppMon 6.5 are based on an AppMon 6.3 setup.

Replace DynatraceUEM.jar with dynatrace.jar

1. Add the `Dynatrace.jar` file to your `/libs` folder where the `DynatraceUEM.jar` file currently resides.
2. Update any classpath references from `DynatraceUEM.jar` to `Dynatrace.jar`.
3. Remove the old `DynatraceUEM.jar` file from your project.
4. Update any reference of `com.dynatrace.apm.uem.mobile.android.*` to `com.dynatrace.android.agent.*`.

Replace com.dynatrace.android.ext.jar if needed

- Replace the old the `com.dynatrace.android.ext.jar` file in the `/libs` folder with the new one.
- Sherlock Activity classes are no longer supported.

Search and replace

- Change status code return value prefix of manual instrumentation methods from `CPWR_` to `DTX_`.
- Update `CPWR_Uem/CPWR_UemOff` to `DTX_CaptureOn/DTX_CaptureOff`.
- Update any reference of `DynatraceUEM.uemCaptureStatus()` to `Dynatrace.getCaptureStatus()`.
- Update any reference of `DynatraceUEM` to `Dynatrace`.
- Update any reference of `UEMAction` to `DTXAction`.
- Update any reference of `UEMWebRequestTiming` to `DTXWebRequestTiming`.

Other changes

- The `APK-Instr.properties` file now only uses `DTX_<...>` keys. Migrate the settings from your previous `APK-Instr.properties` file.
- If you use undocumented keys, please contact support for the `DTX_<...>` replacement.
- Update your ProGuard settings.
- If you need additional logging, output see advanced settings on how to enable in AppMon 6.5.

JavaScript bridge

- Replace `CompuwareUEM.js` with `Dynatrace.js`.

Changes introduced with AppMon 6.2

AppMon 6.2 and later includes a renaming from Compuware back to the original Dynatrace. If you are currently using a version of the Android mobile agent prior to AppMon 6.2, replace the existing Compuware-branded library files with the new Dynatrace ones (`DynatraceUEM.jar` and `com.dynatrace.android.ext`) and adapt your code accordingly.

Replace `CompuwareUEM.jar`

1. Add the `DynatraceUEM.jar` file to your `/libs` folder where the `CompuwareUEM.jar` file currently resides.
2. Update any classpath references from `CompuwareUEM.jar` to `DynatraceUEM.jar`.
3. Remove the old `CompuwareUEM.jar` file from your project.

Replace `com.compuware.android.ext.jar` if needed

If using the Sherlock ext library, the upgrade process is nearly identical. This is not required if you don't use this library,.

1. Add the `com.dynatrace.android.ext.jar` file to the `/libs` folder where the `com.compuware.android.ext.jar` file currently resides.
2. Update any classpath references from `com.compuware.android.ext.jar` to `com.dynatrace.android.ext.jar`.
3. Remove the old `com.compuware.android.ext.jar` file from your project.

Handling deprecated classes

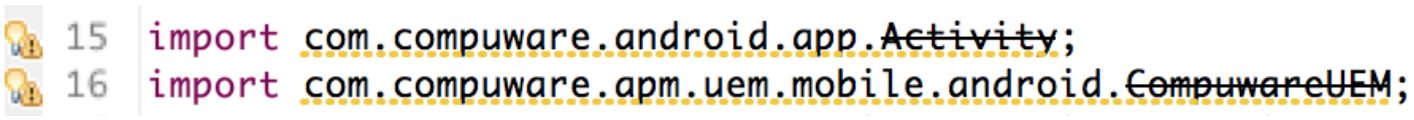
File names have changed to reflect rebranding. The change has a minimum impact. You can simply replace the old file with the new. However the compiler issues notices that CompuwareUEM classes have been deprecated. Your project continues to build and run as before. It is recommended that you eventually update your code to use the new class names, since the deprecated classes will eventually be removed.

1. Update any reference of `com.compuware.android.app.*` to `com.dynatrace.android.app.*`.
2. Update any reference of `com.compuware.apm.uem.mobile.android.*` to `com.dynatrace.apm.uem.mobile.android.*`.
3. Update any reference of `CompuwareUEM.method()` to `DynatraceUEM.method()`.

Common Build Warnings

The type activity is deprecated

If you see a warning like "The type Activity is deprecated":



Please replace the statements with

```
import com.dynatrace.android.app.Activity;
import com.dynatrace.apm.uem.mobile.android.Dynatrace.UEM;
```

The type CompuwareUEM is deprecated

When removing the CompuwareUEM import, compile errors may display anywhere you call any of the static methods in this class.

```
217             rv = CompuwareUEM.startup(this, "easyTravel mobile", getUEMServerString(), true, null);
218             CompuwareUEM.setMonitorCookie("App=easyTravel mobile; Platform=iOS; Vendor=Compuware");
219             CompuwareUEM.enableCrashReporting(true);
220         } catch (RuntimeException e) {
```

Simply update the class to DynatraceUEM:

```
217             rv = DynatraceUEM.startup(this, "easyTravel mobile", getUEMServerString(), true, null);
218             DynatraceUEM.setMonitorCookie("App=easyTravel mobile; Platform=iOS; Vendor=Compuware");
219             DynatraceUEM.enableCrashReporting(true);
220         } catch (RuntimeException e) {
```

The type application is deprecated

If you see the warning “The type Application is deprecated”:

```
10 import com.compuware.android.app.Application;
11 import com.compuware.apm.uem.mobile.android.CompuwareUEM;
12
13 /**
14  * @author
15  *
16  */
17 public class EasyTravelApplication extends Application {
```

Please replace with `import com.dynatrace.android.app.Application;`

Android Agent troubleshooting and FAQ

On this page

Mobile Agent

Manual instrumentation

Auto-Instrumentation

Dynatrace Gradle plugin

Related pages

How to instrument an Android app

Android Agent manual instrumentation API

Auto-instrumentation for Android

Gradle plugin for Android

Migrating from previous Android Mobile Agent versions

External links

Android Multidex

ApkTool

Mobile Agent

How do I verify if I have used the correct `agentPath` parameter (manual-instrumentation) or `DTXAgentStartupPath` property (auto-instrumentation)?

You can send a GET request to the web server agent and check if the web server agent delivers the correct result. The url can be compound from the agent path and application id value:

```
<your_agent_path>/dynaTraceMonitor?type=m&app=<your_app_id>
```

Example:

```
http://myhost.mydomain.com/dynaTraceMonitor?type=m&app=myAndroidApp
```

The response must start with `type=m` and should not contain the key-value pair `cp=0` (UEM is disabled). A typical result should look like the following example:

```
type=m&cp=1&si=120&cl=1&cr=1&er=1&bn=dynaTraceMonitor&id=1&bl=30&st=600
```

How do I verify that the mobile agent is connected and can send data to the AppMon server?

First, you should check if the web server agent is working properly, as shown in the question above. Then you should enable debug logging with the `DTXLogLevel` property. Compare the following log snippets with the debug log from your application.

1. Check if the agent has been started. You can identify the startup sequence from the mobile agent with the following log snippet:

   ```
   Dynatrace OneAgent (Android) startup parms: <startup_parameters>
   ```

2. Check if the agent has established a successful connection to the web server agent. The initial GET request can be found with the log snippet:

   ```
   Send <number>bytes ... <your_agent_path>/dynaTraceMonitor?type=m&<other_parameters>
   ```

   You can verify if the connection was successfully established, when the mobile agent logs the following two lines:

   ```
   Response code: 200 msg: OK
   Result: type=m&cp=1&<other_config_parameters> Success: true
   ```

3. Check if the agent has sent data to the web server agent. The POST request can be found with the log snippet:

```
aPOST url[<your_agent_path>/dynaTraceMonitor?type=m&<other_parameters>] data[<data>]
```

You can verify if the connection was successfully established, when the mobile agent logs the following two lines:

```
Response code: 200 msg: OK
Result: type=m&cp=1&<other_config_parameters> Success: true
```

Can the agent monitor the application in offline mode and sends the data later?

No. Because of efficiency reasons, the AppMon server will not accept monitoring data, that is older than 10 minutes.

Why is my reported value, action name and/or web request url truncated?

The mobile agent truncates most (user-related) values, if they exceed the border limit value of 250.

Where do I find the logs from the mobile agent?

The mobile agent uses the default log framework from Android. You can use the logcat monitor from Android Studio or the command-line tool logcat to view these log lines.

Manual instrumentation

Why do I get the error message "package org.apache.http does not exist"?

The `Dynatrace.jar` library contains an dependency to the Apache HTTP client library. You must add the compile-time dependency for this library to your `build.gradle` file.

```
android {
    useLibrary 'org.apache.http.legacy'
}
```

How can I customize the action for web request timing?

The default behavior of the mobile agent is, that it will assigned the `WebRequestTiming` object to the last action. If you want to use other action, then you have to tag the web request *before* you create the `WebRequestTiming` object. The following example (see point `2a` and `2b`) shows how you can assigne the `WebRequestTiming` object to the `webAction` variable instead of the `otherAction` variable.

```
HttpURLConnection conn = null;
WebRequestTiming timing = null;

// [1] First, create an action
DTXAction webAction = Dynatrace.enterAction("time service");
DTXAction otherAction = Dynatrace.enterAction("dummy action");

try {
        URL url = new URL("http://myhost.mydomain.com/TimeService?tz=CET");
        conn = (HttpURLConnection) url.openConnection();

        // [2a] Select your own action
        webAction.tagRequest(connection);
        // [2b] Use the existing tag and return the WebRequestTiming object
        timing = Dynatrace.getWebRequestTiming(conn);

        // [3] Call startWebRequestTiming() to begin the timing, and then handle the inpu
        // stream from the connection
        timing.startWebRequestTiming();
        InputStream stream = new BufferedInputStream(conn.getInputStream());
        readStream(stream);

        // [4] Once we're done reading, we can stop the timing
        timing.stopWebRequestTiming();

} catch (Exception e) {
        // Attach the exception to the action
        webAction.reportError("Exception", e);

        // user-defined exception handling ...
} finally {
        // [5] Lastly finalize the timing and the action in the finally block.
        // This will allow us to receive timings and close the action even if there is ar
        // exception thrown above
```

```
        if (timing != null) {
                timing.stopWebRequestTiming();
        }
        webAction.leaveAction();

        if (conn != null) {
                conn.disconnect();
        }
}
```

Auto-Instrumentation

Is the retrofit2 framework supported?

Yes, all frameworks based on `OkHttp 3`, are supported.

Is auto-instrumentation supported for APKs that are protected by DexGuard?

No. Using DexGuard with the default configuration generates a protected APK file that the auto-instrumentor can't modify. The auto instrumentation process fails because the Unicode characters introduced by DexGuard are not supported. A possible workaround for this issue is to use `-classobfuscationdictionary empty.txt`, where `empty.txt` is an empty file. This option instructs DexGuard to assign obfuscated class names such as `a`, `b`, or `c`, avoiding the Unicode class name issue.

You may have to deactivate additional protection and optimization features from DexGuard to generate an APK file that is compatible with the auto-instrumentor. If you do not want to modify you DexGuard configuration, then use manual instrumentation. Manual instrumentation is fully compatible with DexGuard or other protection tools.

Can I re-run auto-instrumentation on an application that I have already auto-instrumented?

No, the auto-instrumentor cancels the auto-instrumentation process, if the APK file is already auto-instrumented. You must use an uninstrumented APK file. You can run the auto-instrumentation process on a manually instrumented APK file.

Why do I get the message "This APK already underwent auto-instrumentation."?

The provided APK file is already auto-instrumented and can't be auto-instrumented again. You must use an uninstrumented APK file.

Can I use auto-instrumentation on an application that is manually instrumented with an earlier release of the mobile agent?

No, this is not supported. Generally, manual and auto instrumentation must use the same mobile agent version.

Why do I get the message "The APK contains an incompatible Dynatrace OneAgent (Android) ... expected 7.0"?

The provided APK file contains an old mobile agent. When you use a combination of manual and auto-instrumentation, you can't mix agent versions.

Why aren't all my activities instrumented?

Only activities defined and enabled in `AndroidManifest.xml` are instrumented.

Why don't I see web requests?

Only web requests from a supported HTTP framework are captured. Check if your HTTP framework is in this list. If not, you must instrument your web requests manually.

Why aren't web requests associated with a given user action?

The mobile agent only attaches web requests to a user action triggered within a specific time frame. The time frame starts when the action opens, and ends `500 ms` after the action closes. You can change this value with the `DTXAutoActionTimeoutMilliseconds` property.

Which attributes are used to create the action name?

MenuItems:

- getTitle()

Other views:

- `view.getContentDescription()` if not null.
- `getText()` for a TextView (or subclass of).

- For `LinearLayout` the name is determined by looking at the children.
- The class name is used as backup.

Why doesn't my UI component generate a user action?

There can be multiple reasons why the mobile agent can't create a user action:

- The UI component is defined in a WebView.
- The listener method is defined in a resource XML file.
- The UI component uses an unsupported listener. The User action monitoring section in Auto-instrumentation for Android lists all supported listeners.

Enable the `DTXSendEmptyAutoAction` property if your user interaction does not trigger a web request or lifecycle change. The mobile agent discards empty user actions by default.

Why does google maps stop working on an instrumented application?

You may be running your application in *development* or *testing* mode (not signed and deployed to Google Play) with a Google Maps *release* API key.

Can I adjust JVM heap settings?

Yes. The `instrument.sh` and `instrument.cmd` files have a default maximum heap (`-Xmx`) setting. If you experience an out-of-memory exception, you can increase the memory by updating that setting.

Why do I get the message "Error: Resource is not public"?

You are using a resource from Android that is not part of the `public.xml` file. Instead of referring to a non-public resource, you should copy the resource into your application and refer it directly, as recommended here.

The auto-instrumentor uses the tool `Apktool` and this tool blocks non-public resources for safety reasons.

Why does the auto-instrumentation fails with a brut.androlib.AndrolibException and/or a brut.common.BrutException?

These exceptions are trigger by the tool `Apktool` (and `aapt`) and indicate that the auto-instrumentor was not able to read/modify the binary files of your APK file.

These tools are shipped within the mobile agent zip file and are regularly updated through public updates. Therefore, you should always use the latest mobile agent version.

Why does auto-instrumentation fails with the error message "compile android manifest failed rc = 1"?

The auto-instrumentor uses tools like `aapt` and `Apktool` to extract infromation from the `AndroidManifest.xml` file and to modify it. In this case the tools are not able to generate a compiled version of the modified `AndroidManifest.xml` file.

These tools are shipped within the mobile agent zip file and are regularly updated through public updates. Therefore, you should always use the latest mobile agent version.

Why does auto-instrumentation fails with the error message "extracting the apk failed rc = 1"?

The auto-instrumentor uses tools like `aapt` and `Apktool` to unpack and decode the provided APK file. In this case these tools are not able to extract the information from the `resources.arsc` file.

These tools are shipped within the mobile agent zip file and are regularly updated through public updates. Therefore, you should always use the latest mobile agent version.

Why does my app crash after auto-instrumentation?

When the crash (like `java.lang.NoClassDefFoundError` or `java.lang.ExceptionInInitializerError`) happens only on devices with Android 4.4 or lower, then you have to adjust the multidex preparation step of the auto-instrumentor. See the multi-dex applications section for more details.

When you get a `java.lang.VerifyError`, then you should enable the jumbo mode in your `build.gradle` script. Maybe this workaround solves your problem, because it reduces the number of classes, that are affected by the auto-instrumentor.

```
dexOptions {
    jumboMode true
}
```

If this workaround does not solve the problem, then you must exclude the class (mentioned in the error) from auto-instrumentation.

Why does my app crash with a java.lang.NoClassDefFoundError?

When the crash happens at the app startup phase on devices with Android 4.4 or lower, the mentioned class is not part of the primary `.dex` file. Update the `DTXMultiDexKeep` and/or `DTXMultiDexKeepFile` property to resolve the issue. See the Multi-dex applications section in Auto-instrumentation for Android for details.

Why do I get the message "Instrumented primary DEX is too large #methods=<number>"?

You have to adjust the default value of the `DTXPrimaryDexLimit` property. See the Multi-dex applications section in Auto-instrumentation for Android for details.

Dynatrace Gradle plugin

Why does my build fail with "java.lang.IllegalArgumentException: Invalid instrumentation properties!"?

There can be multiple reasons for this error message:

- Not specifying the `applicationId` and/or the `startupPath` property.
- Not adding a configuration block (`dynatrace`) to your module's `build.gradle` file.
- Not defining a configuration for your product flavor and also not adding a default configuration.

Why does my build fail with "Plugin with id 'com.dynatrace.tools.android' not found"?

Check if you have added the `buildscript` block with the `classpath 'com.dynatrace.tools:android:7.0.+'` dependency to your module's `build.gradle` file.

Why does my build fail with "Could not find com.dynatrace.tools:android:<version_number>"?

Gradle was not able to find the specified version. Check if you have specified the correct version number. You can check the available version on JCenter and Maven Central. You also must add `jcenter()` or `mavenCentral()` to your `buildscript` configuration.

Why does my build fail with "Could not find any version that matches com.dynatrace.tools:android:<version_number>.+."?

Gradle was not able to find a version that matches your version settings. You can check the available version on JCenter and Maven Central. You also have to add `jcenter()` or `mavenCentral()` to your `buildscript` configuration.

Why does my build fail with "Could not get unknown property 'dynatrace' for object of type org.gradle.api.internal.artifacts.dsl.dependencies.DefaultDependencyHandler"?

Check if you have applied the Dynatrace Gradle plugin before the `dynatrace` block and the `dynatrace.agent()` dependency

```
apply plugin: 'com.dynatrace.tools.android'

dynatrace {
...
```

Why does my build fail with "Error:Execution failed for task ':app:autoInstrumentDebug'" and "auto-instrumentation failed"?

The Dynatrace Gradle plugin started the auto-instrumentation, but it failed. Please look at the error message from the auto-instrumentor and check the auto-instrumentation FAQ section for further help.

Why does my build fail with "org.gradle.api.GradleException: Failed to sign apk"?

The Dynatrace Gradle plugin auto-instrumented your APK file, but was not able to successfully extract your signing information and sign the instrumented APK file with your certificate. Check if your Dynatrace plugin version fits the used Android plugin version. See the section Supported versions on the Gradle Plugin for Android page.

Because auto-instrumentation was successful, you can use the generated instrumented APK in the `<project>/<module>/build/outputs/apk/<app_name>/dist` folder. You must sign the `<app_name>.apk` file manually with your Google playstore certificate.

Why does my build fail with "com.android.dex.DexException: Multiple dex files define Lcom/dynatrace/android/<agent_class>;"?

You have added the Android agent library multiple times. Check if you have added the `Dynatrace.jar` to your libs directory and remove it. See the section Combine with manual instrumentation on the Gradle Plugin for Android page for information on how to integrate manual instrumentation into your build process.

Where do I find the auto-instrumentor log file?

The log file is stored in the `<project>/<module>/build/tmp/autoInstrument<build_variant>/logs` directory.

Do I need the file <app_name>_uninstrumented.apk from the outputs folder?

For version 2.2 and later of the Android Gradle plugin, the Dynatrace Gradle plugin generates an uninstrumented APK file with a file suffix of `_uninstrumented`. In some cases it can be useful to have an uninstrumented version of the application, for example if you notice that the agent interferes with another library. If you use manual instrumentation, then the manual instrumentation is still part of the `_uninstrumented` APK file.

The Google Play Console does not accept my instrumented APK file, because it contains a wrong certificate.

Check if you have uploaded the correct file: `<project>/<module>/build/outputs/apk/<app_name>.apk`. Do not use the APKs in the `<project>/<module>/build/outputs/apk/<app_name>/dist` directory, because they are unsigned or signed with a debug certificate.

How to instrument an iOS app

Related pages

Auto-instrumentation for iOS

iOS Agent troubleshooting and FAQ

iOS manual instrumenation

Migrating from previous iOS agent versions

How to instrument an Android app

External links

Dynatrace CocoaPods

There are two ways to set up the iOS Mobile Agent.

- For AppMon 6.5 and later, you can install the Mobile Agent using the Dynatrace CocoaPod.
- Manually add the Mobile Agent framework. See iOS Agent Setup.

For a more detailed instrumentation of special code parts, use the manual instrumentation API.

Xcode Plugin

As of Xcode 8 the AppMon Xcode Plugin is no longer supported. For security reasons, Apple discontinued Plugin support in Xcode and moved to Source Code Editor Extensions. However, Source Code Editor Extensions can only modify source code and cannot implement new functionality that is required for the AppMon plugin.

Auto-instrumentation for iOS

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Auto-Instrumentation Features

Auto-Instrumentation Overview

Instrumentation settings

Limitations

Related pages

iOS Agent manual installation

Auto-Instrumentation for iOS instruments iOS applications to be monitored using the Mobile Agent. The instrumentation process is an automated way to add the Mobile Agent to an application without manually modifying the source code. The auto-instrumented application is equivalent to an application manually instrumented for basic data collection. This level of instrumentation provides visibility into real user experiences of the flow within the application, crash detection, and performance monitoring of application startup and web request response times.

Auto-Instrumentation Features

The following features are automatically instrumented:

- Automatic Mobile Agent startup – You must use instrumentation keys to configure the automatic startup.
- Lifecycle data
- Crash reporting
- Web request tagging
- Web request events (reporting execution time and identifying 3rd party requests)

- Web views
- Automatic user action detection (the detection and timing of certain user actions such as button presses and other view actions and controls)

All these functions are enabled by default. You can disable or refine the Auto-Instrumentation functions by adding configuration keys to your application's `Info.plist` file.

You can combine manual instrumentation with Auto-Instrumentation. For example, you may want to manually instrument certain user-defined actions (and report values and events) at development time, and then use Auto-Instrumentation to add the features listed above to your application.

Auto-Instrumentation Overview

1. Link the iOS Mobile Mobile Agent static library or framework to your application and build it in Xcode. See iOS Agent Setup and Instrumentation for details.
2. Add keys to your application's `Info.plist` file to configure the automatic instrumentation.
3. Auto-instrumentation happens during runtime. The resulting application is instrumented to the levels configured in the application's `Info.plist` file.

Instrumentation settings

The keys (properties) in the following table are the configuration options for automatic instrumentation. Add the keys to your application's `Info.plist` file as needed to fine-tune the Auto-Instrumentation. If you want to keep your `Info.plist` file clean you can move all mobile agent related `DTX` keys to a `Dynatrace.plist` file and add the `Dynatrace.plist` to the *Copy Bundle Resources* build phase. The `Dynatrace.plist` file needs to be at the root of your resources bundle, so it should be created in the same location as the `Info.plist` file.

Many of the properties reflect the method parameters of the instrumentation API.

Key	Key Type	Description
`DTXLogLevel`	String	If this key is present with a valid value `FINEST`, `FINER`, `FINE`, `CONFIG` `WARNING`, `SEVERE`, `OFF`) Mobile A logging is automatically enabled with If the key is not present or does not h valid value, automatic logging is **off** a must turn logging on manually in the

`DTXAgentStartupPath`	String	If this key is present with a valid value Mobile Agent starts automatically and value as the `serverUrl` name. It i any manual `startupWithApplicationNa` in your app. `serverUrl` must start transport mechanism to use (`http:` or `https://`). If it is not present, y make a manual call to `startupWithApplicationNa` start the Mobile Agent.
`DTXMonitorCookie`	String	If this key is present, the value is auto added as a cookie for HTTP requests `serverUrl` (see `DTXAgentStartupPath` in table) so that they can pass your infrastructure's requirements.
`DTXApplicationID`	String	This key's value is used as your appl to identify it on the AppMon Server. A generated if it is not present.
`DTXAllowAnyCert`	BOOL	Equivalent to the `allowAnyCert` in the startup call. If set to `true`, all are accepted. If set to `false`, valid from known certificate authorities are The default value is `false`.
`DTXAgentCertificatePath`	String	This key defines the path to a (self-si certificate in `DER` format, which is use additional anchor to validate HTTPS communication. This key is needed if DTXAllowAnyCert is `false` and a signed certificate is used on the serv default value is null.
`DTXInstrumentWebViewTiming`	BOOL	Detects and times web requests origi from web views in your hybrid app. Se value to `false` to disable automatic and timing of web view web requests. default value is `true`.
`DTXInstrumentWebViewTagging`	BOOL	The default value is `false`. It autom tags web requests originating from we your hybrid app. Set the value to `tr` enable automatic tagging to link webv request to their server-side PurePath available.

`DTXExcludedControls`	Array	This key defines an array of items wh item contains a type of view or contro exclude from automatic creation of us actions. Each item in the array is a ca insensitive string. The possible value `Button`, `DatePicker`, `Slider` `Stepper`, `Switch`, `RefreshCo` `toolBar`, `segmentedControl` `tableView`, `tabBar`, `alertVi` `alertAction`, `pageView`, `navigationController`, `collectionView`, `gesture` a `actionSheet`.
`DTXExcludedLifecycleClasses`	Array	This is an array of items where each contains the name of a class to exclu automatic lifecycle instrumentation. E in the array is a case-sensitive string **exactly** match the name of the class excluded.
`DTXCrashReportingEnabled`	BOOL	Set to `false` if you do not want cras reporting. If you want to report crashe server other than the AppMon Server enable sending email notification of c add the appropriate `enableCrashReportingWitl` API call in your code to override the c The default value is `true`.
`DTXSendCrashReports`	BOOL	Corresponds to the `sendCrashRe` parameter for `enableCrashReportingwitl` to send crash reports to the AppMon Set the key to `false` if you do not w send crash reports. The default value
`DTXInstrumentLifecycleMonitoring`	BOOL	Automatic lifecycle detection is enabl the need to override your view contro classes with the iOS Mobile Agent life classes. Set the value to `false` to d automatic lifecycle monitoring. The de value is `true`.
`DTXInstrumentWebRequestTagging`	BOOL	Set to `true` (default), which turns on automatic web request tagging. `webRequestTagging` is enable `webRequestTiming` is enabled, it is a dependency. You must set **both** `webRequestTiming` and `webRequestTagging` to `false` off automatic web request tagging.
`DTXInstrumentWebRequestTiming`	BOOL	The default value is `true`, which tur automatic web request timing. Set the `false` to disable automatic web req timing. Note that `webRequestTag` enabled if `webRequestTiming` i because it is a dependency.

`DTXInstrumentWebRequestActionWrapping`	BOOL	The default value is `true`. It automa wraps web requests in user actions if user action or automatic user action i available. Set the value to `false` t automatic wrapping of web request a
`DTXInstrumentAutoUserAction`	BOOL	The default value is `true`. It turns o ability to automatically create user ac user interactions with the app, such a presses. Set the value to `false` to automatic creation of user actions.
`DTXAutoActionTimeoutMilliseconds`	Number	The default value is `500 ms`. It sets for how long a particular automatic us is active. The purpose is to catch all requests that happen when an autom action is active. If the automatic user completed web requests, the Mobile A leaves the action at the end of this tir minimum allowed value is `100 ms`, t maximum allowed value is `5000 ms` seconds).
`DTXAutoActionMaxDurationMilliseconds`	Number	The default value is `60000 ms` (60 It sets the value to retain an automati action before deletion. The purpose i all web requests that happen when a automatic user action is active. If the user action has pending web request taking a long time to complete, the M Agent waits for this amount of time fo requests to complete before leaving t action. The minimum allowed value is `ms`, the maximum allowed value is `54 ms` (nine minutes).
`DTXSendEmptyAutoAction`	BOOL	Determines whether to send automat actions that don't contain any web re lifecycle actions. The default value is
`DTXInstrumentGPSLocation`	BOOL	The location is captured only if the ap `CLLocationManager` and send captured location as a metric to the s iOS agent does no GPS location capt its own. Set to `false` to disable iOS location capturing. The default value

Limitations

- An auto-instrumented application cannot carry out functions like `Dynatrace.shutdown()` or Dynatrace`.flushEvents()`. You can manually insert these methods and other user-defined actions and events before performing Auto-Instrumentation.

- There are two controls that do not yet create auto user actions:
  - Gestures.
  - Certain `UIBarButton` items, including custom `UIBarButton` items added to the navigation bar by storyboard (such as `info`) that use segues to transition to another view.

iOS Agent manual installation

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Manual iOS agent setup

Instrument a manual setup

Related pages

Auto-instrumentation for iOS

How to instrument an iOS app

iOS agent setup with CocoaPods

iOS Agent troubleshooting and FAQ

Migrating from previous iOS agent versions

AppMon UEM's Mobile Application support lets you trace visitors, actions, and action PurePaths triggered by your iOS application. To trace activity that was not auto-instrumented, you can add Mobile Agent method calls to your code.

Existing users: If you used the Mobile Agent prior to AppMon 6.3, see Migrating from previous iOS Mobile Agent versions. (library files and classes)

Manual iOS agent setup

1. Go to the Dynatrace Application Monitoring downloads page. Under the ADK (Agent Development Kit) section, download the UEM Mobile ADK zip file, which contains `dynatrace-mobile-agents-<version>.zip`. Extract this file to any folder.

2. In Xcode, add the following from the `\iOS\Agent\` folder of the extracted `.zip` to your Xcode Project and all targets that use the agent. Select **Copy items if needed**.

   - If your deployment target is iOS 8+, drag the `Dynatrace.framework` folder to **Embedded Binaries**.
   - If your deployment target is iOS 7 and earlier, drag the `libDynatrace.a` file to **Linked Frameworks and Libraries**.
   - For all deployment targets, add `Dynatrace.h`.

   > **Note**
   >
   > Do not add both `Dynatrace.framework` and `libDynatrace.a` to the same project. Use only one of them depending on your deployment target.

3. Add the required frameworks and libraries to your Xcode Project. To do this:

   - In the project navigator, select your project and target.
   - Select the **Build Phases** tab.
   - Expand **Link Binaries with Libraries**.
   - Click the **+** button, and select the following frameworks and libraries: `libsqlite3.dylib` `CoreLocation.framework` `CoreTelephony.framework` `Security.framework` `SystemConfiguration.framework` `libz.dylib` `libc++.dylib` `MessageUI.framework`

4. Add flags to **Other Linker Flags**.

   - In the project navigator, select your project.
   - Select your target, then select **Build Settings**.
   - Add the `-ObjC` option to the **Other Linker Flags** setting.

5. Set the build setting **Strip Style** to **Debugging Symbols**.
   This increases the size of your binary slightly, but leaves intact the information needed for on-device crash symbolication. This is not necessary if you intend to use Mobile Symbolication Service to symbolicate your crash reports.

6. Add the key `DTXAgentStartupPath` to your `Info.plist` file. Its value is your instrumented web server. For example `http(s)://<myWebServer>:<my port>`.

7. Add the key `DTXApplicationID` to your `Info.plist` file. This value is used to identifier for your App on the AppMon Server. For example `easyTravel mobile`.

Instrument a manual setup

1. Use the suitable keys from Auto-Instrumentation for iOS to enable and control auto instrumentation.

2. Perform this step only if using `Dynatrace.framework` with a deployment target iOS 8 or later.

   Add a Run Script Phase as last Build Phase of your application target and select **Run script only when installing**. This removes the iOS simulator architecture from your release binary used for iTunes Connect upload. Add the following script:

```
APP_PATH="${TARGET_BUILD_DIR}/${WRAPPER_NAME}"
find "$APP_PATH" -name '*.framework' -type d | while read -r FRAMEWORK
do
          FRAMEWORK_EXECUTABLE_NAME=$(defaults read "$FRAMEWORK/Info.plist" CFBundleEx
          FRAMEWORK_EXECUTABLE_PATH="$FRAMEWORK/$FRAMEWORK_EXECUTABLE_NAME"
          EXTRACTED_ARCHS=()
          for ARCH in $ARCHS
          do
                    lipo -extract "$ARCH" "$FRAMEWORK_EXECUTABLE_PATH" -o "$FRAMEWORK_
                    EXTRACTED_ARCHS+=("$FRAMEWORK_EXECUTABLE_PATH-$ARCH")
          done
          lipo -o "$FRAMEWORK_EXECUTABLE_PATH-merged" -create "${EXTRACTED_ARCHS[@]}"
        rm "${EXTRACTED_ARCHS[@]}"
        rm "$FRAMEWORK_EXECUTABLE_PATH"
        mv "$FRAMEWORK_EXECUTABLE_PATH-merged" "$FRAMEWORK_EXECUTABLE_PATH"
done
```

iOS agent setup with CocoaPods

On this page

Related links

How to instrument an iOS app

iOS Agent manual installation

Auto-instrumentation for iOS

iOS Agent troubleshooting and FAQ

External links

CocoaPods

Dynatrace CocoaPods

CocoaPods Getting Started

Using CocoaPods

You can also install the mobile agent using CocoaPods with the Dynatrace iOS agent pod. If using CocoaPods for the first time, read the Install Instructions and how to prepare your Xcode project using CocoaPods.

Include the following in your Podfile to include the mobile agent as Framework.

```
use_frameworks!
pod 'Dynatrace', '~> 7.0'
```

Include the following in your Podfile to use the static library version of the mobile agent.

```
pod 'Dynatrace/lib', '~> 7.0'
```

1. Add the `DTXAgentStartupPath` key to your `Info.plist` file. Use instrumented web server URL for the value. For example `http(s)://<myWebServer>:<my port>`.
2. Add the `DTXApplicationID` key to your `Info.plist` file. Use the ID your App should be identified by on the AppMon Server for the value. For example, `easyTravel mobile`.
3. Use the suitable keys from Auto-Instrumentation for iOS to enable and control auto instrumentation.

iOS manual instrumenation

On this page

Swift apps manual instrumentation

Dynatrace

DTXAction

Logging

Lifecycle instrumentation

Overhead

Advanced authentication

Related pages

How to instrument an iOS app

iOS Agent manual installation

Auto-instrumentation for iOS

iOS agent setup with CocoaPods

iOS Agent troubleshooting and FAQ

Migrating from previous iOS agent versions

Two classes defined in `Dynatrace.h` instrument your iOS app.

- The `Dynatrace` class manages the Mobile Agent's operation.
- The `DTXAction` class creates actions and report against them.

Swift apps manual instrumentation

Create a bridging header that contains the Mobile Agent header. For example:

```
#ifndef myApp_Bridging_Header_h
#define myApp_Bridging_Header_h

#import "Dynatrace.h"

#endif
```

- In the project navigator, select your project and your target.
- Select **Build Settings**.
- Add `$(SRCROOT)/myApp/Bridging-Header.h` to the **Objective-C Bridging Header** setting.

Dynatrace

setMonitorCookie

```
+ (void) setMonitorCookie:(NSString *)cookieString
```

Parameter	Meaning
`cookieString`	A cookie string such as `MIGRATION_FLAG=3` or `n1=v1; n2=v2`. Pass nil to remove the Cookie header from requests.

Set the value of the HTTP Cookie header to be sent along with every HTTP GET/POST request made by the Mobile Agent when establishing contact with the backend or sending data to it. This method is only needed if your network infrastructure requires the cookie. The method must be invoked before `startupWithApplicationName:serverURL:allowAnyCert:certificatePath:` to ensure that the earliest requests include the cookie. It can be invoked again to change the cookie string at any time.

startupWithApplicationName:serverURL:allowAnyCert:certificatePa

```
+ (DTX_StatusCode) startupWithApplicationName:(NSString *)applicationId
                                    serverURL:(NSString *)serverURL
                                 allowAnyCert:(BOOL)allowAnyCert
                              certificatePath:(NSString *)pathToCertificateAsDER
```

Auto-instrumentation should be preferred over manual call of startup method

Use the settings keys from Auto-Instrumentation for automatic mobile agent startup instead of the manual startup call.

Parameter	Meaning
`applicationName`	A user-defined application identifier, such as `@"EasyTravel"`. Do not use underscores (_) in the application name.
`serverURL`	The path to the placed UEM agent, for example `http://myhost.mydomain.com:8080/myApplication]`. In the AppMon Client, you can find the path in **System Profile Preferences > User Experience > JavaScript Agent > Agent Location**. Be sure to include the transport mechanism (`http` or `https`) in the path.
`allowAnyCert`	Allow any certificate for `https` communication. This parameter is only evaluated if you specified the `https` transport mechanism in the server name.
`pathToCertificateAsDER`	Path to a (self-signed) certificate in DER format; or null.

Call this method to initialize the Mobile Agent. **This method must be invoked before any other methods are called**. Multiple calls to this method are ignored if the Mobile Agent was not shut down before. A certificate in DER format that is used as an additional anchor to validate HTTPS communication is needed if `allowAnyCert` is `false` and a self-signed certificate is used.
For example:

```
NSString *pathToCertificateAsDER = [[NSBundle mainBundle] pathForResource:@"easyTravelSer
```

startupWithApplicationName:serverURL:authenticationClassDeleg

```
+ (DTX_StatusCode) startupWithApplicationName:(NSString *)applicationId
                                    serverURL:(NSString *)serverURL
                    authenticationClassDelegate:(Class)callbackDelegate
```

Parameter	Meaning

`applicationName`	A user-defined application identifier, for example `@"EasyTrave`
`serverURL`	The path to the placed UEM agent, for example `http://myhost.mydomain.com:8080/myApplica` find the path in **System Profile Preferences > User Experienc Location**. Be sure to include the transport mechanism (`http` or
`authenticationClassDelegate`	The name of the class that implements a class method with the fo `+ (void)requestAuthenticationChallenge:(NSURLAuther` Add <DTXAuthenticationClassDelegate> to your class interface.

Call this method to initialize the Mobile Agent. **This method must be invoked before any other methods are called**. Multiple calls to this method are ignored if the Mobile Agent was not previously shut down. The `authenticationClassDelegate` allows for advance authentication not provided by the previous `startWithApplicationName` method. See the Advanced authentication section for more information.

shutdown

```
+ (DTX_StatusCode) shutdown
```

Call this method to shut down the Mobile Agent. Collected data is flushed to the AppMon Server. This function does not return until the attempt to send data either succeeds or fails. Call this method on the main thread makes your UI unresponsive. If the attempt fails, the data is retained and sent later if the Mobile Agent restarts before the data becomes too old.

enableCrashReportingWithReport

Depending on the syntax used, the `enableCrashReportingWithReport` method can send crash reports to the AppMon Server, HockeyApp server, Quincy server, or Victory server, or email crash reports to specified recipients.

AppMon Server

```
+ (DTX_StatusCode) enableCrashReportingWithReport:(BOOL)sendCrashReport
```

Parameter	Meaning

`sendCrashReport`	`true` to send a complete crash report. `false` to send only a crash event without full details.

This methods activates the `KSCrash` framework within the Mobile Agent to capture signals and unhandled exceptions. When a crash occurs, a report is saved on the device. The next time the app is started, the report is processed. If the parameter is `true`, then the complete iOS crash report is sent to the AppMon Server. Otherwise a crash event containing some information about the crashed thread is sent instead of the complete report. If the same exception reoccurs within seven days, the crash data is not reported again but a crash event is reported that references the previously reported crash data.

HockeyApp

```
+ (DTX_StatusCode) enableCrashReportingWithReport:(BOOL)sendCrashReport
                                  hockeyAppIdentifier:(NSString *)appIdentifier
                                hockeyWaitUntilReachable:(BOOL)waitUntilReachable
                                        hockeyUserId:(NSString *)userId
                                   hockeyContactEmail:(NSString *)contactEmail
                                hockeyCrashDescription:(NSString *)description
```

Parameter	Meaning
`sendCrashReport`	`true` to send a complete crash report. `false` to send only a crash event without full details.
`appIdentifier`	The application identifier assigned by HockeyApp to your application.
`waitUntilReachable`	`true` to wait until HockeyApp is reachable before sending. `false` to attempt sending whether available or not; simply fails if not available.
`userId`	Optional HockeyApp user ID. Set to nil if not needed.
`contactEmail`	Optional HockeyApp user contact email address. Set to nil if not needed.
`description`	Optional description to include with the crash report for HockeyApp.

Activates the `KSCrash` framework within the Mobile Agent to capture signals and unhandled exceptions. When a crash occurs, a report is saved on the device. When a crash occurs, a report is saved on the device. The report processes the next time the app starts. Setting the `sendCrashReport` parameter is `true` sends the complete iOS crash report to the AppMon Server. Otherwise a crash event containing some information about the crashed thread is sent instead of the complete report. If the same exception reoccurs within seven days, the crash data is not reported again but a crash event is reported that references the previously reported crash data. The crash report is also sent to HockeyApp using the parameters provided.

Quincy

```
+ (DTX_StatusCode) enableCrashReportingWithReport:(BOOL)sendCrashReport
                                         quincyURL:(NSString *)serverURL
                                quincyWaitUntilReachable:(BOOL)waitUntilReachable
                                      quincyUserId:(NSString *)userId
                                 quincyContactEmail:(NSString *)contactEmail
                               quincyCrashDescription:(NSString *)description
```

Parameter	Meaning
`sendCrashReport`	`true` to send a complete crash report. `false` to send only a crash event without full details.
`serverURL`	The Quincy server URL.
`waitUntilReachable`	`true` to wait until the Quincy server is reachable before sending. `false` to attempt sending whether available or not; simply fails if not available.
`userId`	Optional Quincy user ID. Set to nil if not needed.
`contactEmail`	Optional Quincy user contact email address. Set to nil if not needed.
`description`	Optional description to include with the crash report for Quincy.

Activates the `KSCrash` framework within the Mobile Agent to capture signals and unhandled exceptions. When a crash occurs, a report is saved on the device. The report processes the next time the app starts. Setting the `sendCrashReport` parameter is `true` sends the complete iOS crash report to the AppMon Server. Otherwise a crash event containing some information about the crashed thread is sent instead of the complete report. If the same exception reoccurs within seven days, the crash data is not reported again but a crash event is reported that references the previously reported crash data. The crash report is also sent to the Quincy server using the parameters provided.

Victory

```
+ (DTX_StatusCode) enableCrashReportingWithReport:(BOOL)sendCrashReport
                                        victoryURL:(NSString *)serverURL
                                  victoryUserName:(NSString *)userName
                                 victoryUserEmail:(NSString *)userEmail
```

Parameter	Meaning
`sendCrashReport`	`true` to send a complete crash report. `false` to send only a crash event without full details.
`serverURL`	The Victory server URL.
`userName`	Optional Victory user name. If not set, the value of `UIDevice.currentDevice.name` is used.
`userEmail`	Optional Victory user contact email address.

Activates the `KSCrash` framework within the Mobile Agent to capture signals and unhandled exceptions. When a crash occurs, a report is saved on the device. The report processes the next time the app starts. Setting the `sendCrashReport` parameter is `true` sends the complete iOS crash report to the AppMon Server. Otherwise a crash event containing some information about the crashed thread is sent instead of the complete report. If the same exception reoccurs within seven days, the crash data is not reported again but a crash event is reported that references the previously reported crash data. The crash report is also sent to the Victory server using the parameters provided.

Email crash report

```
+ (DTX_StatusCode) enableCrashReportingWithReport:(BOOL)sendCrashReport
                                  emailRecipients:(NSArray *)recipients
                                     emailSubject:(NSString *)subject
                                     emailMessage:(NSString *)message
                                    emailFilename:(NSString *)filename
                                sendAppleStyleReport:(BOOL)sendAppleStyleReport
                                       alertTitle:(NSString *)alertTitle
                                     alertMessage:(NSString *)alertMessage
                                    yesButtonText:(NSString *)yesButtonText
                                     noButtonText:(NSString *)noButtonText
```

Parameter	Meaning
`sendCrashReport`	`true` to send a complete crash report. `false` to send only a crash event without full details.
`recipients`	An array of `NSString`s, each containing one email address.
`subject`	The email subject.
`message`	The email body text.
`filename`	The name to use for the file attachment that contains the crash report. Set to nil to use the default name.
`sendAppleStyleReport`	`true` to send an Apple-style crash report. `false` to send a JSON-style crash report.
`alertTitle`	Title for the alert.
`alertMessage`	Message for the alert.
`yesButtonText`	Label for the yes button in the alert. If the user taps this button, then the email UI appears.
`noButtonText`	Label for the no button in the alert. If the user taps this button, then the crash report is not emailed. It is still sent to AppMon.

Activates the `KSCrash` framework within the Mobile Agent to capture signals and unhandled exceptions. When a crash occurs, a report is saved on the device. The report processes the next time the app starts. Setting the `sendCrashReport` parameter is `true` sends the complete iOS crash report to the AppMon Server. Otherwise a crash event containing some information about the crashed thread is sent instead of the complete report. If the same exception reoccurs within seven days, the crash data is not reported again but a crash event is reported that references the previously reported crash data.

You can also email the report using the settings you provide. First, an alert dialog box appears. If you press the **yes** button, the email UI appears, pre-populated with the values specified. Press **Send** to send the crash report.

setGpsLocation

```
+ (DTX_StatusCode) setGpsLocation:(id)gpsLocation
```

Parameter	Meaning
`gpsLocation`	`CLLocation` object with GPS coordinates acquired by the customer's application.

Use this method to record the current GPS location of the user. The Mobile Agent does not automatically collect any location information. The GPS location is currently not evaluated by the AppMon Server, but the data might be available in a future AppMon release.

lastErrorCode

```
+ (DTX_StatusCode) lastErrorCode
```

Return the error code associated with the most recent internal Mobile Agent error. Zero (0) is returned if there is no error.

lastErrorMsg

```
+ (NSString *) lastErrorMsg
```

Return the error message associated with the most recent internal Mobile Agent error. It returns nothing if there is no error.

flushEvents

```
+ (DTX_StatusCode) flushEvents
```

Send all collected events immediately. To reduce network traffic/usage, the collected events are usually sent in packages where the oldest event has an age of up to nine minutes. Use this method to force sending of all collected events regardless of their age.

DTXAction

enterActionWithName

```
+ (DTXAction *) enterActionWithName:(NSString *)actionName
+ (DTXAction *) enterActionWithName:(NSString *)actionName
                        parentAction:(DTXAction *)parentAction
```

Parameter	Meaning
`actionName`	The name of the action.
`parentAction`	The DTX`Action` object for the parent of this action.

Error Value	Meaning
`DTX_Error_InvalidParameter`	The `parentAction` is nil, already ended, or the action name is nil or empty.
`DTX_CaptureOff`	The Mobile Agent is not initialized.
`DTX_TruncatedEventName`	The action name was truncated to the maximum length. This is a warning; the action is still created.
`DTX_Error_InternalError`	An Mobile Agent internal error occurred.

Start an action, which results in a mobile action PurePath in AppMon. Call this method at the beginning of the code that you want to time. You must set the action end via a call to `leaveAction`. A value is returned if the action is created successfully, but not if an error occurs. Call `lastErrorCode` or `lastErrorMsg` to find out which error occurred.

The returned `DTXAction` is retained, therefore it has to be released if no longer needed (only valid for non-ARC environments).

leaveAction

```
- (DTX_StatusCode) leaveAction
```

End a previously started action. All reported events, values, or tagged web requests between the start and end of an action are part of the action, and nested in the mobile action PurePath. Call this method at the end of the code that you want to time.

endVisit

```
- (DTX_StatusCode) endVisit
+ (DTX_StatusCode) endVisit
```

Ends the current visit, closes all current actions, flushes pending data to the server, and then begins a new visit.

reportEventWithName

```
- (DTX_StatusCode) reportEventWithName:(NSString *)eventName
```

Parameter	Meaning
`eventName`	The name of the event.

Send an event to AppMon, which results in a node of a mobile action PurePath.

reportValueWithName

```
- (DTX_StatusCode) reportValueWithName:(NSString *)valueName
                               intValue:(NSInteger)value
- (DTX_StatusCode) reportValueWithName:(NSString *)valueName
                            doubleValue:(double)doubleValue
- (DTX_StatusCode) reportValueWithName:(NSString *)valueName
                            stringValue:(NSString *)stringValue
```

Parameter	Meaning
`valueName`	The name of the value.
`intValue`	An integer value in the range of 0–2147483.
`doubleValue`	Double value.
`stringValue`	String value.

Send a key/value pair to AppMon, which results in a node of a mobile action PurePath. The value can be processed by a measure and be charted.

reportErrorWithName

```
- (DTX_StatusCode) reportErrorWithName:(NSString *)errorName
                             errorValue:(NSInteger)errorValue
- (DTX_StatusCode) reportErrorWithName:(NSString *)errorName
                              exception:(NSException *)exception
- (DTX_StatusCode) reportErrorWithName:(NSString *)errorName
                                  error:(NSError *)error
+ (DTX_StatusCode) reportErrorWithName:(NSString *)errorName
                             errorValue:(NSInteger)errorValue
+ (DTX_StatusCode) reportErrorWithName:(NSString *)errorName
                              exception:(NSException *)exception
+ (DTX_StatusCode) reportErrorWithName:(NSString *)errorName
                                  error:(NSError *)error
```

Parameter	Meaning

`errorName`	The name of the value.
`errorValue`	The integer error value.
`exception`	The description of this exception is passed to the Server.
`error`	The localizedDescription of this exception is passed to the Server.

Send an named error to AppMon, which results in a node of a mobile action PurePath. There are instance and class versions of these methods.

getRequestTagHeader

```
- (NSString *) getRequestTagHeader
```

Return the name of the HTTP header that you must add to a Web request for the AppMon Server to link it to the `DTXAction` in the PurePath. You should not normally need this method, because the automatic request tagging should catch all web requests.

getRequestTagValue

```
- (NSString *) getRequestTagValue
```

Return the value to assign to the HTTP header returned by `getRequestTagHeader`, for the AppMon Server to link it to the `DTXAction` in the PurePath. You should not normally need this because the automatic request tagging should catch all web requests.

getDTXWebRequestTiming

```
+ (NULLABLE DTXWebRequestTiming *)getDTXWebRequestTiming:(NONNULL NSString *)requestTagSt
                                              requestUrl:(NULLABLE NSURL *)requestUrl
```

Creates an instance of the DTXWebRequestTiming object. Parameter requestTagString is the value of the HTTP header that has to was to the web request. This obtained from the `getRequestTagValueForURL` method.

startWebRequestTiming

```
- (DTX_StatusCode)startWebRequestTiming
```

The Dynatrace OneAgent automatically times web requests made using NSURLRequest, NSURLConnection, NSURLProtocol, NSString, UIWebView, and ASIHTTPRequest. If you use an alternate technology to make web requests and want to time them, use the `getRequestTagHeader` method, adding that information to your request, and then this method to start the timing.

stopWebRequestTiming

```
- (DTX_StatusCode)stopWebRequestTiming:(NULLABLE NSString *)errorCode
```

The Dynatrace OneAgent automatically times web requests made using NSURLRequest, NSURLConnection, NSURLProtocol, NSString, UIWebView, and ASIHTTPRequest. If you use an alternate technology to make web requests and want to time them, use the `getRequestTagHeader` method, adding that information to your request, and then this method to stop the timing and send the information to the mobile action PurePath.

Instrumentation example for manual web request tagging and timing (in Swift)

For 3rd party web request frameworks that are not automatically tagged this can be done manually. As AppMon does not display parentless web request events in the User Action PurePaths dashlet this requires 3 steps:

- create a manual action
- create client side (mobile App) part of web request event (web request timing)
- add tag to web request

```
/**
 *  simplified manual tagging demo
 */
<!-- *force italic end in Atom* -->
```

```
import UIKit
import WebKit
import Dynatrace

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        let wkWebView = WKWebView(frame: self.view.frame)
        self.view.addSubview(wkWebView)

        manualTaggingDemo(wkWebView: wkWebView)
    }

    func manualTaggingDemo(wkWebView: WKWebView) {
        let parentAction = DTXAction.enter(withName: #function)

        let url = URL(string: "https://www.dynatrace.com")
        downloadRequest(url: url!, wkWebView: wkWebView, parentAction: parentAction)  //a
    }

    func downloadRequest(url: URL, wkWebView: WKWebView, parentAction: DTXAction?) {
        let childAction = DTXAction.enter(withName: #function, parentAction: parentActior
        let session = URLSession.shared
        let request = NSMutableURLRequest(url: url)
        request.httpMethod = "GET"
        request.cachePolicy = .reloadIgnoringCacheData

        var webrequestTiming: DTXWebRequestTiming?
        if let dynatraceHeaderValue = Dynatrace.getRequestTagValue(for: url) {
            let dynatraceHeaderKey = Dynatrace.getRequestTagHeader() //this could be cach
            request.addValue(dynatraceHeaderValue, forHTTPHeaderField: dynatraceHeaderKey
            webrequestTiming = DTXWebRequestTiming.getDTXWebRequestTiming(dynatraceHeader
        }

        let task = session.downloadTask(with: request as URLRequest) {
            (location, response, error) in

            defer {
                parentAction?.leave()    //leave parent action when request finished - ali
```

```
            }

            guard let _:URL = location, let _:URLResponse = response, error == nil else {
                webrequestTiming?.stop("failed") //stop webrequest timing in error case
                return
            }
<!-- *force italic end in Atom_ -->
            let urlContents = try! String(contentsOf: location!, encoding: .utf8)

            guard !urlContents.isEmpty else {
                webrequestTiming?.stop("empty content") //stop webrequest timing in erro
                return
            }

            webrequestTiming?.stop(nil) //stop webrequest when request finished

            wkWebView.loadHTMLString(urlContents, baseURL: nil)
        }

        webrequestTiming?.start()    //start webrequest timing
        task.resume()
    }

}
```

Return code definitions

Codes	Value	Description
`DTX_CaptureOff`	1	Data capturing is off.
`DTX_CaptureOn`	2	Data capturing is on, or the method call was successful.
`DTX_CrashReportingUnavailable`	4	The `KSCrash` framework could not be initialized.
`DTX_CrashReportingAvailable`	5	The `KSCrash` framework initialized successfully.

`DTX_Error_NotInitialized`	-1	The Mobile Agent failed to initialize properly.
`DTX_Error_InvalidRange`	-2	The integer value is outside of the valid range.
`DTX_Error_InternalError`	-3	An error occurred within the Mobile Agent. Check the log for further details.
`DTX_Error_ActionNotFound`	-4	When using the JavaScript bridge, this means the action name was not found.
`DTX_Error_InvalidParameter`	-5	An invalid parameter was passed to an Mobile Agent method.
`DTX_Error_ActionEnded`	-6	An operation is being performed on an action already ended by the `leaveAction` method.
`DTXR_ReportErrorOff`	-8	The AppMon Server has disabled error reporting.
`DTX_TruncatedEventName`	-9	An action, error, or value name is too long and has been truncated.

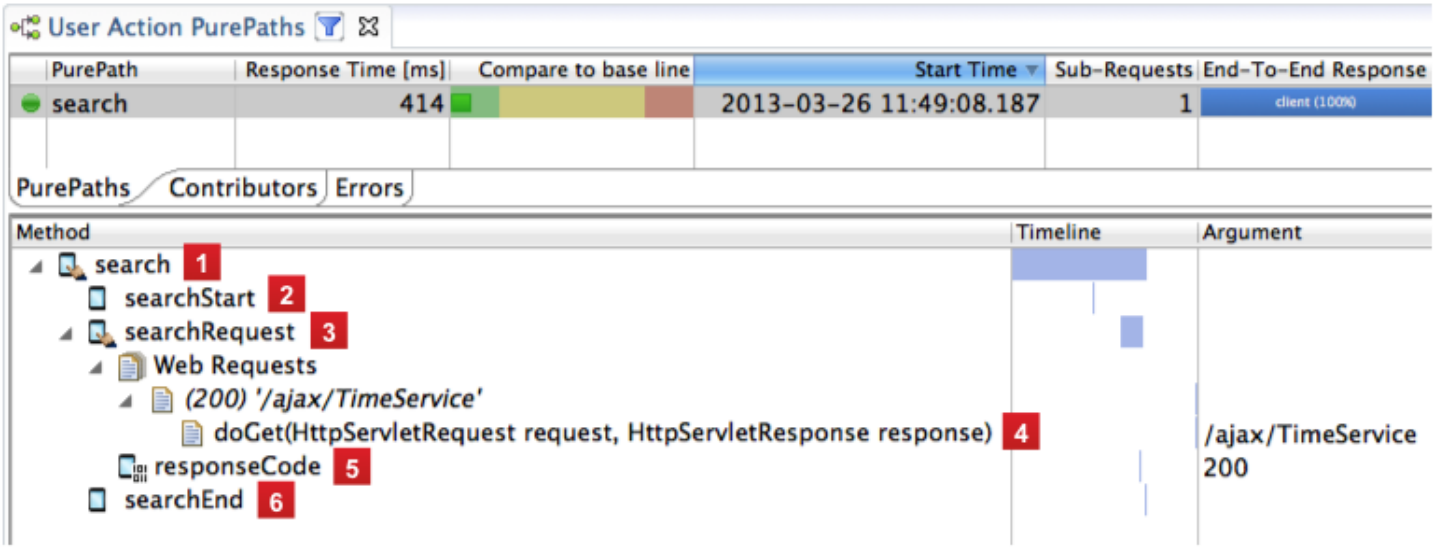
Logging

third-party

You can enable logging and set the logging level by adding the DTXLogLevel key to the application's `info.plist` file. See Auto-Instrumentation for iOS.

Instrumentation Example

The example code below creates a User Action PurePath that looks like this:



Example PurePath

```
#import "Dynatrace.h"

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictior
{
    ...
    // initialize Dynatrace here or anywhere else (mind that actions and events that occu
    [Dynatrace startupWithApplicationName:@"easyTravel"
                                serverURL:@"http://easytravel.example.com:8080/eT/"
                            allowAnyCert:NO
                          certificatePath:nil];
    ...
}
```

```
- (IBAction)performSearch:(id)sender
{
    ...
    // [1a] start action "search"
    DTXAction *action = [DTXAction enterActionWithName:@"search"];

    // [2] named event inside an action
    [action reportEventWithName:@"searchStart"];

    // [3a] starting sub-action
    DTXAction *child = [DTXAction enterActionWithName:@"searchRequest"
```

```
                                        parentAction:action];

    // [4] NSURLRequest and NSMutableURLReqest are tagged automatically
    NSError* error;
    NSHTTPURLResponse* response;
    NSURL *searchURL = [[NSURL URLWithString:
                         [NSString stringWithFormat:@"http://127.0.0.1:8080/ajax/TimeServ
                          [sender query]]] retain];
    NSData *data = [NSURLConnection sendSynchronousRequest:[NSURLRequest requestWithURL:s
                                        returningResponse:&response error:&error];
    [searchURL release];
    if(data == nil){
        // [x] report an error if communication fails
        [action reportErrorWithName:@"CommunicationError" errorValue:[error code]];
    }

    // [5] custom value inside an action
    [child reportValueWithName:@"responseCode" intValue:[response statusCode]];

    // [3b] end action "searchRequest"
    [child leaveAction];

    // [6] named event on successful completion
    [action reportEventWithName:@"searchEnd"];

    // [1b] end action "search"
    [action leaveAction];
    ...
}
```

Lifecycle instrumentation

Lifecycle Instrumentation is performed by Auto-Instrumentation.

Use the auto-startup feature or initialize the mobile agent very early in your application to capture the initial view appearance. Calling `startupWithApplicationName` from `main()` is acceptable.

Overhead

Adding the iOS Mobile Agent to an application increases the size of the application binary as follows:

Architecture	Size
armv7	801 KB
armv7s	801 KB
arm64	976 KB
All three	2576 KB

Actions, events, and errors created automatically or via the Mobile Agent API are stored in a SQLite database in the application's `Document` directory. On average, each of these items takes up about 150 bytes in the database. This size can vary depending on the length of the strings included in items such as names and error messages.

By default, the Mobile Agent stores items in the database up to two minutes before sending them to the Server and deleting them. You can manage this send interval in the AppMon Client.

Advanced authentication

Configuring advanced authentication disables the Mobile Agent internal handling of authentication. For example, the `DTXAllowAnyCert` setting is ignored and must be handled in the callback.

There are two ways to enable advanced authentication. Both ways involve creating or modifying an existing class to implement the **DTXAuthenticationClassDelegate** protocol and contains a method called **requestAuthenticationChallenge**.

```
...
@interface AuthenticationManager : NSObject <DTXAuthenticationClassDelegate>
...
@end
```

```
#import "AuthenticationManager.h"

@implementation AuthenticationManager
...
+ (void)requestAuthenticationChallenge:(NONNULL NSURLAuthenticationChallenge *)challenge
    if ([challenge.protectionSpace.authenticationMethod isEqualToString:NSURLAuthenticati
        // Handle server side authentication
        [challenge.sender useCredential:credential forAuthenticationChallenge:challenge];
        return;
    } else if ([[challenge protectionSpace] authenticationMethod] == NSURLAuthenticationM
                // Handle client side authentication

        [challenge.sender useCredential:credential forAuthenticationChallenge:challenge];
        return;
    }

    // If everything fails, we cancel the challenge.
    [[challenge sender] cancelAuthenticationChallenge:challenge];
}
...
@end
```

If using auto-instrumentation, add the following to your applications `info.plist`.

```
DTXAuthenticationClassDelegate = AuthenticationManager
```

Otherwise, you can call `startupWithApplicationName:serverURL:authenticationClassDelegate:` to configure and start the Mobile Agent.

```
#import "Dynatrace.h"
#import "AuthenticationManager.h"

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictior
{
    ...
    // initialize Dynatrace here or anywhere else (mind that actions and events that occu
    [Dynatrace startupWithApplicationName:@"easyTravel"
                                serverURL:@"http://easytravel.example.com:8080/eT/"
                authenticationClassDelegate:[AuthenticationManager class]];
    ...
}
```

Migrating from previous iOS agent versions

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Changes introduced with AppMon 2017 May

Changes introduced with AppMon 6.5

Changes introduced with AppMon 6.2

Versions earlier than AppMon 5.6

Related pages

How to instrument an iOS app

iOS Agent manual installation

Auto-instrumentation for iOS

iOS agent setup with CocoaPods

iOS Agent troubleshooting and FAQ

iOS manual instrumenation

Changes introduced with AppMon 2017 May

- Update your Podfile if you are using CocoaPods:

If you use the Dynatrace Framework:

```
use_frameworks!
pod 'Dynatrace', '~> 7.0'
```

If you use the Dynatrace static library:

```
pod 'Dynatrace/lib', '~> 7.0'
```

- Setting `DTXApplicationID` has become mandatory.

Changes introduced with AppMon 6.5

The migration steps for AppMon 6.5 are based on an AppMon 6.2 or 6.3 setup.

Replace libDynatraceUEM.a with libDynatrace.a or dynatrace.framework

1. Add the `libDynatrace.a` or `Dynatrace.framework` file to your project where the `libDynatraceUEM.a` file currently resides.
2. Update any library or framework path references from libDynatraceUEM.a to `libDynatrace.a` or `Dynatrace.framework`.
3. If you use the framework add it to the embedded frameworks of your Xcode project.
4. Remove the old libDynatraceUEM.a file from your project.

Search and replace

- Prefix of status code return value of manual instrumentation methods changed from `CPWR_` to `DTX_`
- Update `CPWR_Uem/CPWR_UemOff` to `DTX_CaptureOn/DTX_CaptureOff`
- Update any reference of `DynatraceUEM` to `Dynatrace`
- Update any reference of `UEMAction` to `DTXAction`
- Update any reference of `UEMWebRequestTiming` to `DTXWebRequestTiming`

Manual instrumentation

Already deprecated support for manual lifecycle Instrumentation was removed. If manual lifecycle instrumentation classes are still used as base class, replace them with their `UI<...>Controller` equivalent:

- Update `CompuwareUEMViewController` to `UIViewController`
- Update `CompuwareUEMNavigationController` to `UINavigationController`

- Update `CompuwareUEMPageViewController` to `UIPageViewController`
- Update `CompuwareUEMReferenceLibraryViewController` to `UIReferenceLibraryViewController`
- Update `CompuwareUEMSplitViewController` to `UISplitViewController`
- Update `CompuwareUEMTabBarController` to `UITabBarController`
- Update `CompuwareUEMTableViewController` to `UITableViewController`

Auto-instrumentation

The auto-instrumentation settings keys used in the `Info.plist` file have changed. The new keys can be found Auto-Instrumentation for iOS.

JavaScript bridge

- Replace `CompuwareUEM.js` with now `Dynatrace.js`

Changes introduced with AppMon 6.2

AppMon 6.2 and later includes a renaming from Compuware back to the original Dynatrace. If you are currently using a version of the iOS ADK prior to AppMon 6.2, replace the existing library and header files with the new ones.

Replace libCompuware.a with libDynatraceUEM.a

1. Add the `libDynatraceUEM.a` file to your project where the `libCompuwareUEM.a` file currently resides.
2. Update any library path references from libCompuwareUEM.a to libDynatraceUEM.a.
3. Remove the old libCompuwareUEM.a file from your project.

Handling deprecated classes

File names have changed to reflect rebranding. The change has minimum impact and you can simply replace the old file with the new. However the compiler issues notices that CompuwareUEM classes have been deprecated. Your project continues to build and run as it did before. It is recommended that you eventually update your code to use the new class names, since the deprecated classes will eventually be removed.

1. Update any reference of `CompuwareUEM.h` to `DynatraceUEM.h`.
2. Update any reference of `[CompuwareUEM method]` to `[DynatraceUEM method]`.

Versions earlier than AppMon 5.6

In releases earlier than AppMon 5.6, special characters such as spaces in System Profile - Error Detection conditions were automatically replaced with underscores by the Mobile Agent. Be sure you have the right comparison strings, because the Mobile Agent for AppMon 5.6 and later does not replace the strings.

iOS Agent troubleshooting and FAQ

Related pages

How to instrument an iOS app

iOS Agent manual installation

Auto-instrumentation for iOS

iOS agent setup with CocoaPods

iOS manual instrumenation

Migrating from previous iOS agent versions

How do I verifying that the mobile agent is connected

To verify that you are connected to your mobile application and can send data to the AppMon Server, enable logging and set the log level to ALL.

If the `serverURL` from the `startupWithApplicationName` call is correct and the Server is responding, a log message similar to this displays:

```
server response: type=m&cp=1&si=120&cl=1&cr=1&er=1&bn=dynaTraceMonitor&id=1&bl=30&st=600
```

However, if the `serverURL` is not correct or the Server is not functioning properly, a log message similar to this displays:

```
Error sending data to dynaTrace - comm in progress switch: 1
```

Why is my button captured as "Touch on button"?

Auto-instrumentation captures the title from `UIButton` by calling the following in order and stopping when getting valid text: `titleLabel.attributedText`, `attributedTitleForState:UIControlStateNormal`, and `accessibilityLabel`. If none of these produce usable text the default value is set to “button”.

Can I use auto-instrumentation on an application that is instrumented with an earlier release of the iOS mobile agent?

You can recompile your application with a new iOS Mobile Agent release. It combines the Auto-Instrumentation with any manual instrumentation present in the application. Use the instrumentation settings keys to control the level of Auto-Instrumentation.

Using -ObjC creates linker errors

Some 3rd-party libraries have problems using -ObjC as a linker flag. The solution is to use -force_load instead. Follow these steps to use -force_load:

1. Remove the Mobile Agent library from the **Link Binary With Libraries** setting in **Build Phases**.
2. Remove -ObjC from **Other Linker Flags** and add -force_load <path of libDynatrace.a>

```
-force_load $(PROJECT_DIR)/Dynatrace/libDynatrace.a
```

SQLite Databases

The iOS Mobile Agent uses a SQLite database to temporarily store performance metrics that it collects. The agent issues the `sqlite3.shutdown()` command to properly configure this database to be serialized. If your application also uses a SQLite database, this shutdown command also shuts down your application’s SQLite database. Therefore, it is recommend to not open your application’s SQLite database until after you call the iOS Mobile Agent `[startupWithApplicationName](/support/doc/appmon/user-experience-management/mobile-uem/how-to-instrument-an-ios-app/auto-instrumentation-for-ios/ios-manual-installation/#anchor_instrumentation)` command. If this is not possible, check whether your SQLite database is open before using it. If it is not, reopen it.

How to instrument a hybrid app

On this page

Instrumentation

Debugging

Kony One

Known issues

Related pages

Android Agent manual instrumentation API

Auto-instrumentation for Android

iOS Agent manual installation

Auto-instrumentation for iOS

AppMon supports instrumenting Cordova based applications or applications that use WebViews to display HTML content. There are three components which can be used for hybrid application instrumentation: the mobile agent, the JavaScript agent and the JavaScript bridge. When instrumented with all above, resulting visits can consist of user actions from the mobile agent (including the JavaScript bridge) and/or the JavaScript agent.

The mobile agent delivers device information and OS lifecycle events. The JavaScript agent gathers network actions. You can extend this behavior through the JavaScript bridge. The bridge lets you manually instrument your JavaScript code by calling mobile agent native functions.

Instrumentation

To instrument your Hybrid Application you can:

- Use the Dynatrace-Cordova-Plugin, which supports Cordova based applications only, such as Ionic.
- Include the files manually.

Use the Dynatrace-Cordova-Plugin

Install the Dynatrace-Cordova-Plugin in your Cordova based application. It automates each step you have to perform to successfully instrument your hybrid application. Click here to download the plugin and for installation instructions.

Include files manually

If you use hybrid platforms other than Cordova such as KonyOne, or don't want to instrument your Cordova based application using the plugin, you can instrument manually.

Not all of the steps for including files manually apply to other platforms. Some steps are not feasible if your hybrid application is not WebView based. For instance, Xamarin can only be instrumented by the Mobile Agent. The manual procedure lets you decide which components you want to include, but you may miss important actions or events if you leave a component out.

1. **Mobile Agent:** Use the mobile agent to instrument the native parts of your application. See Android agent setup and instrumentation or iOS agent setup and instrumentation for more Android and iOS Mobile Agent instrumentation information.

2. **JavaScript Agent:** Manually inject the JavaScript agent into your HTML5 content for example into `index.html`. The JavaScript agent detects the Ajax (XHR) calls.

3. **JavaScript Bridge:** Use the JavaScript bridge (`Dynatrace.js`) to manually create user actions in your JavaScript code. If you are using this component, be sure to also use the mobile agent. This is mandatory because the bridge calls the mobile agent to create user actions. The bridge does not work without the mobile agent.

The following sections detail each of these steps.

Step 1 - Mobile Agent

Important Android Auto Instrumentation Properties

Android Auto-Instrumentation is using a configuration file which contains different properties. There are several properties which are important for hybrid applications:

Key	Key Type	Description

DTXSetCookiesForDomain	String	For hybrid applications using the JavaScript agent or JavaScript bridge, cookies need to be set for each instrumented domain or server the application communicates with. You can specify domains, host or IP addresses. Domains (or sub domains) have to start with a dot. Separate the list elements with a comma.
DTXAllowFileCookies	boolean	The default value is false. Set to true to use file cookies.

Android WebView registration and AppMon startup

> **Note**
>
> Manual WebView registration and startup are only needed for manual instrumentation. Android Auto-Instrumentation performs these tasks automatically.

To instrument the JavaScript code in a WebView, register the WebView within the mobile agent native code by calling `Dynatrace.registerWebView(WebView)`. If you have more than one WebView, you must register each single WebView instance. You must also include the `Dynatrace.startup` mobile agent call within the native code, and the `Dynatrace.shutdown` call if desired.

In Cordova-based applications, the WebView is created in the `MainActivity` which is automatically created by Cordova. The `MainActivity` is a `CordovaActivity` subclass and contains the calls to `super.onCreate(savedInstanceState);` and `loadUrl(url)` in the `onCreate(savedInstanceState)`. After those two calls the WebView is initialized. Once initialized, instrument it by calling the `Dynatrace.registerWebView(WebView)` method.

The following example shows how the instrumentation may look:

```
public class MainActivity extends CordovaActivity {
        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                ..
                loadUrl(this.launchUrl);
        }

        @Override
        public void init() {
                super.init();
                Dynatrace.registerWebView((WebView)super.appView.getView());
                // For older version of Cordova use:
                // Dynatrace.registerWebView(super.appView);
        }
}
```

The `loadUrl(url)` function internally calls the `init()` method.

iOS WebView registration and AppMon startup

For iOS, the web view registration is automatic if the App is instrumented with the the mobile agent.

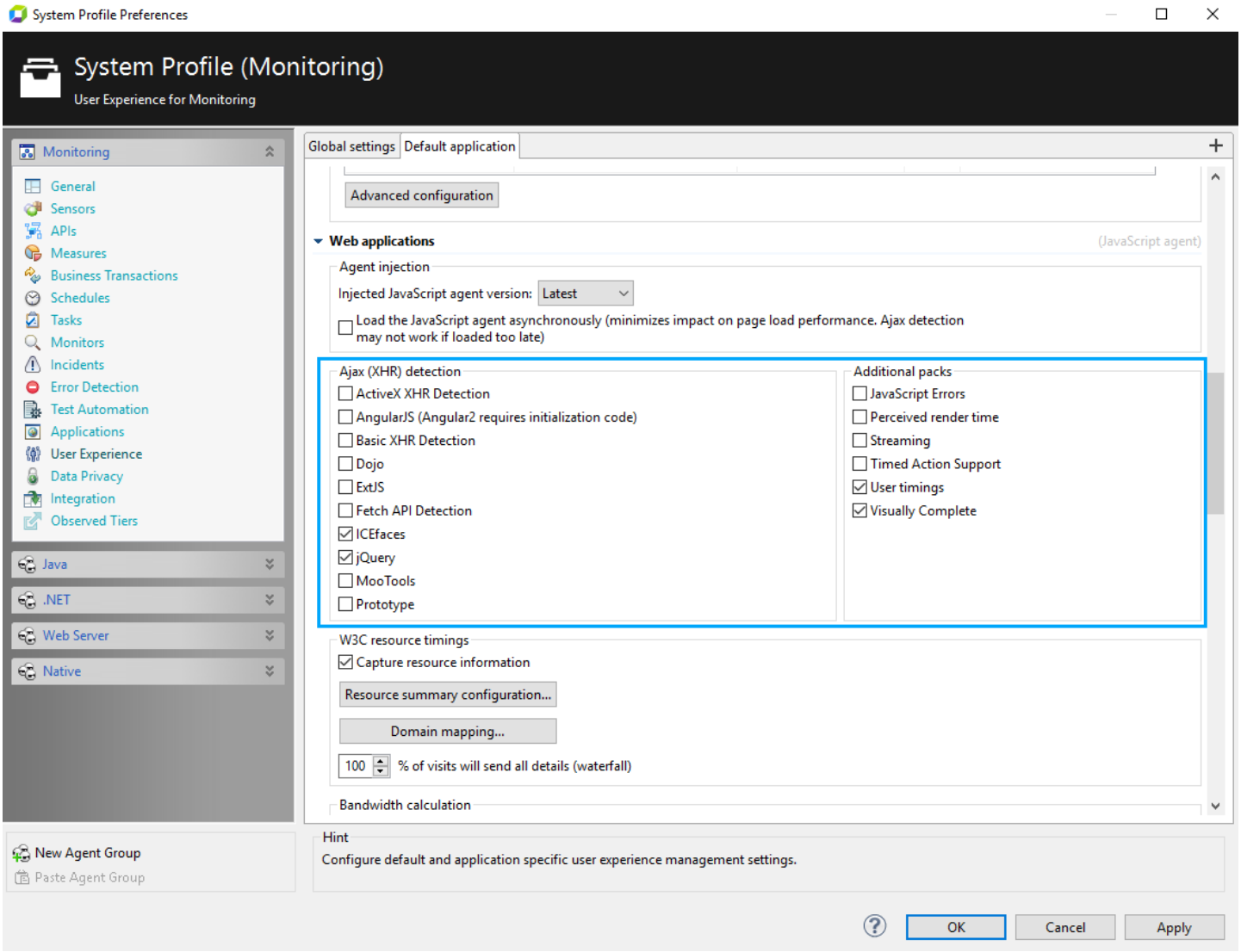
Step 2 - JavaScript Agent

There are several ways to manually inject the Agent. For hybrid applications, including the JavaScript agent code within the app and the WebViews is recommended and described in this section. See the Manual injection details section in JavaScript Agent for alternate ways to manually inject the JavaScript Agent, such as using the bootstrapped agent. The following steps describe how you configure the JavaScript Agent:

1. In the AppMon Client, Use the **System Profile Preferences** to edit System Profile settings for the app. Choose **System Profile > <System Profile> > User Experience > <application>**.

2. Scroll to the Web Applications section.

3. Under **Ajax (XHR) detection** and **Additional packs**, select the check boxes for each technology to be supported. For example, to support network calls within Ionic 2, select the **AngularJS (Angular 2 requires initialization code)** check box.



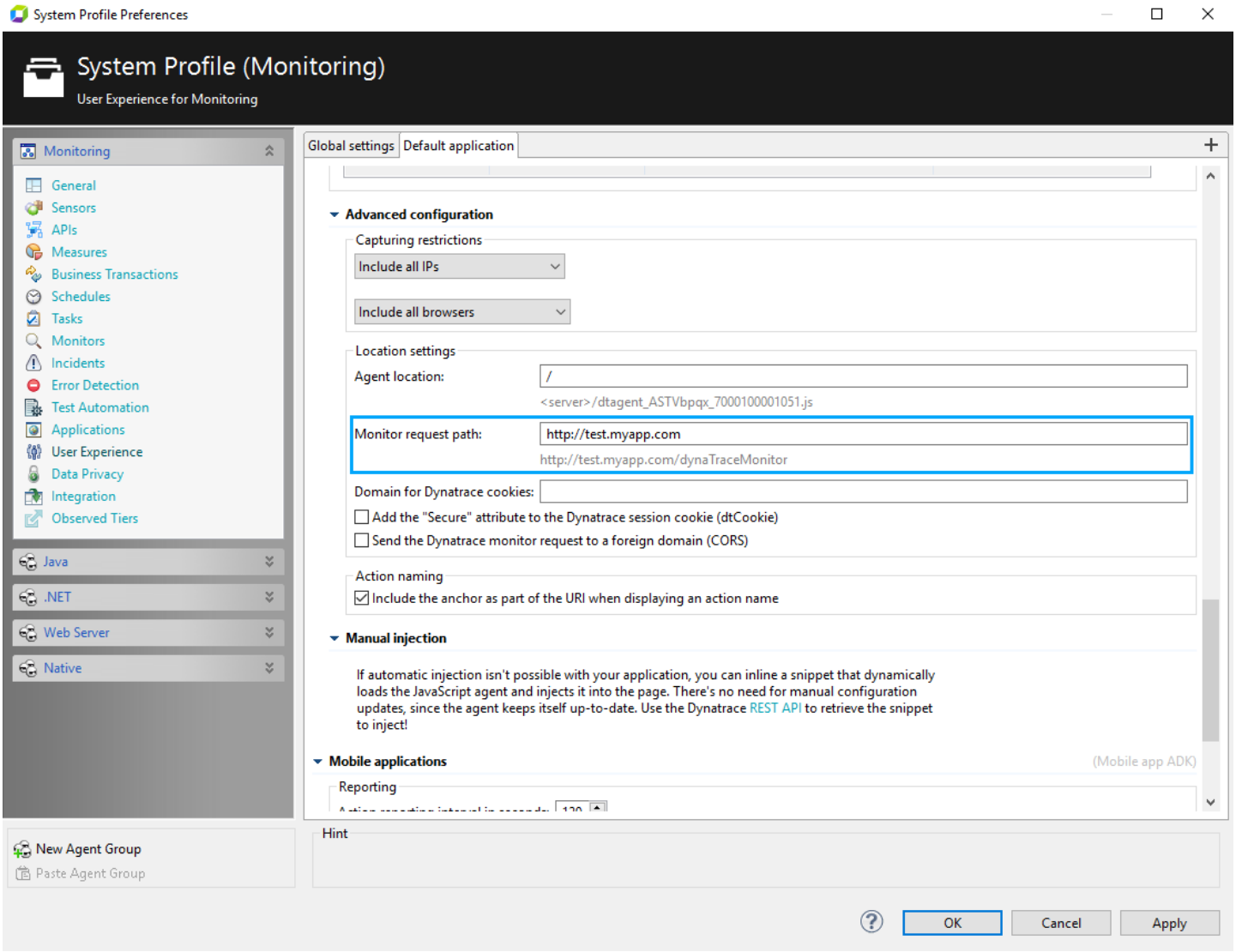
Note

Ionic 2 uses the Angular 2 framework supported by the JavaScript agent. However, you must insert the following initialization code everywhere you use Angular HTTP. This initialization code is not needed for Angular 1.

```
if(typeof dT_!='undefined' && dT_.initAngularNg){
        dT_.initAngularNg(http,Headers);
}
```

See Manual XHR framework instrumentation for more initialization code details.

4. In the **Advanced Configuration** section of the <application> tab, specify the path used by the JavaScript agent to communicate with the instrumented server in the **Monitor Request Path** field, as shown in the following.



Monitor Request Path (must include protocol; see resulting path below the entry field)

5. Select the **Send the Dynatrace monitor request to a foreign domain (CORS)** check box, if the AppMon monitor request is going to a different domain.

6. If the instrumented web server delivering the Agent is at a separate address, specify it in the **Agent Location** field.

You can retrieve the initialization tag and it's contained code using the Server REST Interfaces API with `https://AppMon-Server:8021/api/v1/profiles/systemprofile/applications/application/javas`
The initcode dynamically generates the agent at application runtime and serves it based on the settings mentioned above.

Use `https://AppMon-Server:8021/api/v1/profiles/systemprofile/applications/application/javas` to include the entire JavaScript agent code. When changing the JavaScript Agent version or configuration, ensure you download the agent again and update it if using the full agent code. Settings are based on agent filename, so don't change the agent name to prevent unintended behavior.

Step 3 - JavaScript Bridge

The mobile agent includes the JavaScript bridge file `Javascript/Dynatrace.js`, which can be loaded where applicable. The JavaScript functions mirror the native `Dynatrace` and `DTXAction` methods. See the corresponding native methods in Android Agent Setup and Instrumentation and iOS Agent Setup and Instrumentation for more information.

Return code definition

Return codes are available in addition to those defined for native methods. See the Android or iOS native method return codes for details.

Value	Meaning
-3	Not supported on the running platform.

Basic instrumentation methods

Use the `Dynatrace.js` functions for JavaScript instrumentation, as described below.

`enterAction(actionName)`

Parameter	Meaning
`actionName`	The name assigned to the action to be created.

Return Value	Meaning
Action identifier	The identifier needed for subsequent operations on the newly created action.

Create a new action with the given name.

`enterActionParentId(actionName, parentId)`

Parameter	Meaning
`actionName`	The name assigned to the child action to be created.
`parentId`	The identifier of the parent action.

Return Value	Meaning
Action identifier	The identifier needed for subsequent operations on the newly created child action.

Create a child action with the given name, using the specified parent action.
`leaveAction(actionId)`

Parameter	Meaning
`actionId`	The identifier of the action to be closed

Return Value	Meaning

`DTX_CaptureOn`
`DTX_CaptureOff`
`DTX_ActionEnded`
`DTX_Error_NotInitialized`
	Refer to the Android or iOS native method return codes for details.

Close the action that has the given identifier. **`reportEvent(eventName, actionId)`**

Parameter	Meaning
`eventName`	The name assigned to the event to be reported.
`actionId`	The identifier of the action to contain the event.

Return Value	Meaning

`DTX CaptureOn`
`DTX CaptureOff`
`DTX InvalidParameter`
`DTX ActionEnded`
`DTX_Error_NotInitialized`
	Refer to the Android or iOS native method return codes for details.

Report an event in the context of the action that has the given identifier.

```
reportValueInt(valueName, value, actionId)
reportValueDouble(valueName, value, actionId)
reportValueString(valueName, value, actionId)
```

Parameter	Meaning
`valueName`	The name assigned to the value to be reported.
`value`	The integer, double, or string value.
`actionId`	The identifier of the action to contain the value.

Return Value	Meaning

`DTX_CaptureOn`
`DTX_CaptureOff`
`DTX_Error_InvalidRange`
`DTX_ActionEnded`
`DTX_Error_NotInitialized`
`DTX_Error_InvalidParameter`
	Refer to the Android or iOS native method return codes for details.

Report an integer, double, or string value in the context of the action that has the given identifier.

```
reportErrorIntegerWithAction(errorName, errorValue, actionId)
reportErrorStringWithAction(errorName, errorValue, actionId)
```

Parameter	Meaning
`errorName`	The name assigned to the error to be reported.
`errorValue`	The integer or string (not supported on Android) value.
`actionId`	The identifier of the action to contain the error.

Return Value	Meaning

`DTX CaptureOn`
`DTX CaptureOff`
`DTX Error InvalidParameter`
`DTX ActionEnded`
`DTX Error NotInitialized`
`DTX_ReportErrorOff`
	Refer to the Android or iOS native method return codes for details.

-3	Not supported.

Report an integer or string error in the context of the action that has the given identifier.

```
reportErrorInteger(errorName, errorValue, actionId)
reportErrorString(errorName, errorValue, actionId)
```

Parameter	Meaning
`errorName`	The name assigned to the error to be reported.
`errorValue`	The integer or string (not supported on Android) value.
`actionId`	The identifier of the action to contain the value.

Return Value	Meaning

`DTX_CaptureOn`
`DTX_CaptureOff`
`DTX_Error_InvalidParameter`
`DTX_ActionEnded`
`DTX_Error_NotInitialized`
`DTX_ReportErrorOff`
	Refer to the Android or iOS native method return codes for details.

-3	Not supported.

Report an integer or string error not in the context of an action.

```
setGpsLocation(longitude, latitude)
```

Parameter	Meaning
`longitude`	The location longitude, for example -82.937.
`latitude`	The location latitude, for example 42.710.

Return Value	Meaning
`DTX_CaptureOn`	Successful.

Save the given GPS location for reporting along with the captured data.

`flushEvents()`

Send the collected data immediately.

`startTaggingRequests(actionId)`

Parameter	Meaning
`actionId`	The identifier of the action that will contain web requests.

Signal that any web requests created while an action is open should be assigned to that action.

Return Value	Meaning

`DTX_CaptureOn`
`DTX_CaptureOff`
`DTX_Error_InvalidParameter`
`DTX_ActionEnded`
`DTX_Error_NotInitialized`
	Refer to the Android or iOS native method return codes for details.

Debugging

Use the `Dynatrace` `debugOn` and `debugOff` JavaScript functions to turn debugging on or off.

Kony One

You must use Auto-Instrumentation for Android for KonyOne apps. Be sure to define the necessary instrumentation properties (for example, `DTXNameUseValueOf=serviceID`). If reporting the specific values, events, and actions are necessary, manually instrument the events by using the Kony Foreign Function Interface, and then apply Auto-Instrumentation on the resulting APK.

For example, to report a check-out amount:

```
// Android native code to report an event/value:

// DTXAction action = DTXAction.enterAction("Transaction Amount");
// action.reportValue("Total", 10.99);
// action.leaveAction();

// In Kony development studio create an FFI function and equivalent code:

package com.sample;
import com.dynatrace.android.agent.DTXAction;


public class DtIntegration {
        public DtIntergration() {}

        public static void reportTransAmount(String actionName, String eventName, double
                DTXAction action = Dynatrace.enterAction(actionName);
                action.reportValue(eventName, amount);
                action.leaveAction();
        }
}

// Invocation from javascript: DtIntegration.reportTransAmount("Transaction Amount", "Tot
```

Run the resulting APK through Auto-Instrumentation for Android to implement this code.

Known issues

- Android 2.3.*x* has a critical problem with the JavaScript interface. The issue is reported here: https://code.google.com/p/android/issues/detail?id=12987.
- JavaScript Agent injection does not capture load resources for this release. Only XHRs made to your instrumented server are captured.

Mobile symbolication service

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Enable the symbolication service and set storage

Symbolication service REST API

DSS client

Integration with build systems

Other modes

Overview

The Symbolication Service (also known as DSS, or De-obfuscation and Symbolication Service) allows you to symbolicate (iOS) or de-obfuscate (Android) mobile application crash reports or handled exceptions. This allows you to view the classes and methods in the stack trace in plain text.

Enable the symbolication service and set storage

By default the Symbolication Service is not enabled at the time the AppMon Server is installed. To enable the service:

1. In the AppMon Client, select the **Settings menu > Dynatrace Server > Services item > Symbolication Service tab**. Select the **Enable Deobfuscation and Symbolication Service** checkbox and click **OK**.
2. Return to the **Symbolication Service** tab and set the File Storage Location and Limit as needed for crash reports and unhandled exceptions. The default value of 0 indicates there is no storage limit for these files.

Symbolication service REST API

The symbolication service provides a set of REST calls to manage the ProGuard mapping and iOS symbol extract files needed to interpret the mobile stack traces when they arrive at the AppMon Server.

By default the REST server service is available at `https://<DynatraceServer>:8021/rest/dss` (SSL) or `http://<DynatraceServer>:8020/rest/dss` or ay the port, configured for HTTP(S). You can check these at **Settings menu > Dynatrace Server > Services item > Management tab**.

Use a REST client to execute the calls or the DSS Client (wrapper for REST) described below for additional command line functionality.

Path

`rest/dss/status`
Gets DSS status. The `dss-*` header items tells whether the ProGuard libraries or the Symbol File Manager are enabled / running.

`rest/dss/info`
Tells the DSS status plus the version, the number of mapping files, disk space available and used and disk quota

`rest/dss/symbolfiles`
Lists all iOS symbol extract files and ProGuard mapping files

`rest/dss/symbolfiles/ios`
Lists all symbol extract files `rest/dss/symbolfiles/ios/apps/{AppName}`
Lists all symbol extract files for {`AppName`} `rest/dss/symbolfiles/ios/apps/{AppName}/ver`
Lists all symbol extract files for {`AppName`} version {`AppVersion`}

`rest/dss/symbolfiles/android`
Lists all mapping files `rest/dss/symbolfiles/android/apps/{AppName}`
Lists all mapping files for {`AppName`} `rest/dss/symbolfiles/android/apps/{AppName}/ve`
Lists all mapping files for {`AppName`} version {`AppVersion`}

`rest/dss/symbolfiles/ios/apps/{AppName}/versions/{AppVersion}/builds/{u`
Uploads a symbol extract file for {`AppName`} version {`AppVersion`} with {`uuid`} and build time {`buildtir`} form of "`architecture.uuid`", e.g., "`arm64.020a290c987237f6a7d5dbb08d11ed5d`".
Be sure to specify the content-type of "`application/zip`" if the content is in PKZip format.

`rest/dss/symbolfiles/android/apps/{AppName}/versions/{AppVersion}`
Uploads a mapping file for {`AppName`} version {`AppVersion`}. Be sure to specify the content-type of "`appl` in PKZip format.

`rest/dss/symbolfiles/ios/apps/{AppName}/versions/{AppVersion}`
`rest/dss/symbolfiles/android/apps/{AppName}/versions/{AppVersion}`
Deletes symbol extract/mapping files for {`AppName`} version {`AppVersion`}

`rest/dss/symbolfiles/ios/apps/{AppName}?versionsOlderThan=1.2.3`
`rest/dss/symbolfiles/android/apps/{AppName}?versionsOlderThan=1.2.3`
Deletes symbol extract/mapping files for {`AppName`} versions older than `1.2.3`

DSS client

Beyond a GET in the browser address bar or a REST client GUI, the DSS client provides command line functionality for automation.

As of AppMon 6.3 you can find DSS clients in the Mobile ADK zip.
The `dss-client.bat` and `dss-client` shell script to start the Java-based utility are in the `<MobileADKUnzipDir>/Android/dss/bin`.
The `<MobileADKUnzipDir>/Android/dss/lib/dss-client-<version>.jar` referenced in the OS-specific shell scripts can also be directly used as shown further below.
The Mac-native `DTXDssClient` utility is in `<MobileADKUnzipDir>/iOS/dss`.
Please mind that you must make the Linux and Mac versions executable first with `chmod +x <filename>`, as this attribute is not preserved by the zip container.

Generic syntax

```
<dss-client-for-platform-file> server=<protocol>://<DynatraceServer>:<port> username=<use
```

Most basic Linux example to test if the command works / the file is executable (No connection to AppMon Server necessary, which could fail due to firewalls, etc.):

```
cd <locationOfDssClient>
dss-client -help
```

Windows SSL local example to see if the ProGuard files have been copied correctly and the DSS has been enabled in the Server settings, version, status, etc.:

```
cd <locationOfDssClient>
dss-client.bat server=https://localhost:8021 username=admin password=admin -info
```

Required parameters

Option	Value
server	The full URL, including protocol and port of the AppMon Server running the DSS service. E.g. `server=https://localhost:8021`

username	A username to connect to this Server, as used in the AppMon Client.
password	If you do not specify a password in the command line parameters, you will be prompted to enter one interactively.

Flags

The `-info`, `-status`, `-list`, `-delete` and `-file` flags are equivalent to the REST `/<subPathOrMethod>`. Only one flag is allowed for each DSS Client operation.

Options

Options below can be used in combinations with flags above to upload files, filter list output, delete single or multiple files.

Option	Value
os	Operating system. Must be either **iOS** or **Android**.
version	An integer representing the version number of your application. On Android, this corresponds to `android:versionCode` in `AndroidManifest.xml`. On iOS, this corresponds to `CFBundleVersion`.
appname	The name of your application.
uuid	iOS only: The UUID for the build of a particular architecture. Must be used in conjunction with `buildime`.
buildtime	iOS only: The build time for a particular architecture. Must be used in conjunction with `uuid`.
versionsOlderThan	Used in conjunction with the `-delete` flag: Delete all versions older than this integer. Example: `versionsOlderThan=6` would delete versions 1-5.

Examples

Using the `.jar` directly:

List all Android symbol files:

```
java -jar dss-client.jar server=http://server.com username=user password=pass -list os=ar
```

Upload a mapping file for Android app MyApp version 1:

```
java -jar dss-client.jar server=http://server.com username=user password=pass -file=myfil
```

Delete a mapping file for iOS app MyApp version 5:

```
java -jar dss-client.jar server=http://server.com username=user password=pass -delete os=
```

Integration with build systems

Apache ant

The `dss-client.jar` file is built to be directly compatible with Apache Ant builds. This can be configured automatically with the included `dss.xml` file, or you can define your own taskdef.

Getting started with ant - automatically detecting versionCode and app name

First, copy the `dss.properties` and `dss-client.jar` file into your project where your Android `build.xml` file is located. Modify the `dss.properties` file and set the `dss.username`, `dss.password`, `dss.server`, and `dss.jarfile` properties:

```
dss.username=admin
dss.password=admin
dss.server=http://myserver.local:8020
dss.jarfile=dss-client.jar
dss.mapping.file=bin/proguard/mapping.txt
```

dss.properties

Next, edit your Android `build.xml` file. It is usually located at the root of your Android project. We will create a new task that depends on the Android 'release' build, imports our taskdef, and invokes the dssUpload task. Importing the `dss.xml` script defines the properties `${dss.server}`, `${dss.username}`, `${dss.password}`, `${dss.appname}`, `${dss.manifest.versionCode}`, and `${dss.mapping.file}` for you.

```
<import file="dss.xml" />
<target name="uploadToDss" description="Build, obfuscate, and upload mapping file to DSS"
        <dssUpload server="${dss.server}" username="${dss.username}" password="${dss.pass
</target>
```

build.xml

Now we can simply execute this task to build your Android release, obfuscate the build with ProGuard and then upload the mapping file to the DSS server automatically. A successful run looks like this:

```
release:
uploadToDss:
[dssUpload] Beginning file upload.................................upload complete.
[dssUpload] ================================================================================
[dssUpload] DSS Operation Status: OK    Bytes Uploaded: 239017
[dssUpload] Upload (ANDROID) symbol file for application 'ProguardTestApplication' versic
================================================================================================
BUILD SUCCESSFUL
Total time: 17 seconds
```

The version number is read directly from the `AndroidManifest.xml` versionCode attribute. Running build against the same version will update the existing mapping file on the server. Incrementing the versionCode attribute will upload a new a mapping file linked to the new versionCode.

Advanced ant - defining your own taskdef and settings

If you don't want to import the `dss.xml` file and automatically, set the required dss properties, you can manually define the taskdef and then call dssUpload. First, define the taskdef in your build file:

```
<taskdef name="dssUpload" classname="com.dynatrace.diagnostics.dss.client.DssUploaderTask
    <classpath>
        <pathelement location="dss-client.jar" />
    </classpath>
</taskdef>
```

build.xml

And then call the task, populating the parameters manually:

```
<dssUpload server="http://server:port" username="user" password="password" appname="MyApp
```

build.xml

Other modes

Since the DSS runs as REST web service, it can be accessed in a variety of ways in addition to the DSS Client, wget, cURL, and web browsers provide a means to upload, view, and manage your mapping files.

Access via cURL examples

Get information about the DSS server, which is returned in the response headers:

```
curl --user <user>:<pass> -I https://server.com:8021/rest/dss/info
```

List all iOS symbol files for app MyApp version 5 (response in JSON):

```
curl --user <user>:<pass> https://server.com:8021/rest/dss/symbolfiles/ios/app/MyApp/vers
```

User experience management - FAQ

Related pages

User experience use cases and tasks

Monitor user visits by geographic region

Monitor user satisfaction

Trace user activity through all channels

Monitor top landing and exit pages

Find specific user visits

Drilldown to the AppMon Client

Can I capture individual values of the JSON body?

No. You have to capture them somewhere on the server-side PurePath, e.g: a method argument or return value.

Can I exclude certain URLs from tracing?

Yes. You can do it for the UEM-only info, or for the full stack info, i.e. PurePaths are not captured either. See the How does UEM works page to learn the difference between UEM-only and full stack.

To exclude the UEM-only info:

1. Navigate to the required Agent/Agent Group in the **System Profile Preferences** dialog box, and select the **Sensor Configuration** item.
2. Double click the **User Experience** sensor, to open its **Configure Sensor Properties** dialog box.
3. Click **+** to the right of the **URI-specific injection behavior** list.
4. In the table, specify the URIs and/or URI patterns to be excluded, and the rule for matching. Wildcards like * or ? are not supported, but for matching any URI you could use *contains /*.
5. Select the condition rule from the **Condition** list.

6. From the **Injection rule** list, select one of the following options:
   - **do not inject on this agent**: Agents matching other agent groups may take over and perform an injection themselves.
   - **do not inject on this and subsequent agents**: Same as **do not inject on this agent**, plus this rule prevents sensor properties set on other (outermore) agents from overriding this choice and performing the injection there.
7. Save and apply your changes.

To exclude full stack info:

1. Navigate to the required Agent/Agent Group in the **System Profile Preferences** dialog box, and select the **Sensor Configuration** item.
2. Double click the sensor, responsible for web requests, to open its **Configure Sensor Properties** dialog box.
3. In the **URI Filter** panel, select the **Exclude the following URIs** radiobutton.
4. In the table, specify the URIs and/or URI patterns to be excluded, and the rule for matching.
5. Save and apply your changes.

Can I restrict tracing to certain URIs?

Yes, you can do it for the full stack info. It is not possible to restrict tracing for the UEM-only info.
See the How does UEM works page to learn the difference between UEM-only and full stack.

To restrict full stack tracing:

1. Navigate to the required Agent/Agent Group in the **System Profile Preferences** dialog box, and select the **Sensor Configuration** item.
2. Double click the sensor, responsible for web requests, to open its **Configure Sensor Properties** dialog box.
3. In the **URI Filter** panel, select the **Only include the following URIs** radiobutton.
4. In the table, specify the required URIs and/or URI patterns, and the rule for matching.
5. Save and apply your changes.

Can AppMon monitor single page application

Yes. Check out the System Requirements page, if your technologies are based on what's listed here, then it will be captured by our sensors. Please note that you may need to explicitly enable some capturing options in the **User Experience** section of your System Profile, as they may not be enabled by default.

Can I disable third party transactions?

Yes, you can set specific third parties as ignored. You can set this for any application in your System Profile, including default one.

1. Right click your System Profile, and select **Edit System Profile > User Experience > tab of the required application**.
2. Expand the Web application section, and click **Domain mapping** in the **W3C resource timings** panel.
3. In the **Map detected domains** dialog box, click **+** to add the record to the list.
4. Specify the domain, or domain pattern, and then select **Ignored** from the **Type** list.
5. Save and apply your changes.

Can I see regions of a country on the World Map in AppMon Web

Yes. To do so:

1. Right click your System Profile, and select **Edit System Profile > Measures**.
2. Find the **Country of Visits** measure, and open its properties dialog box.
3. In the **Geographical granularity** list, select **Custom Regions**.
4. Click **+** to add a record to the **Regions enabled for** list, and select the required country from the list.
5. Save and apply your changes.

Can I forcibly end a user visit?

Generally the visit ends when user was inactive or browser was closed during a certain timeout. You can configure this timeout in the **User Experience** section of your System Profile. If the user opens the browser, or starts the activity again before the timeout, the visit will continue.

To end a visit forcibly you can use the ADK and code exactly where you want the visit to end. Refer to the JavaScript ADK and JavaScript Ajax ADK page to learn how.

I'm getting "No SSL certificate provided" and/or "Security has blocked a cleartext HTTP resource load since it is insecure" log messages

Your SSL certificate is missing, or isn't installed properly. Please see Certificate management to learn how to install a new one.

Some web requests are missing in user actions PurePath

Web requests should be performed within a certain timeframe from the user action to be linked to it. Web requests, performed outside of this timeframe are captured as separate PurePaths, but not linked to a user action PurePaths.

Network contribution data is missing in UEM

To see the **Network Contribution** metric, the two following options must be enabled:

- Capture of the request/response size in sensor properties.
- Bandwidth calculation in the UEM properties.

To enable those:

1. Enable capture of the request/response size.
    1. Open the System Profile editor, expand the required Agent Group/Tier, and select the sensor, responsible for web requests: **Servlet**, **Java Web Services**, or **Web Server**.
    2. Select the **Measure request/response sizes** checkbox.
    3. Save your changes.
2. Enable bandwidth calculation.
    1. In the System Profile editor, select **User experience > <your application>**.
    2. Scroll down to the **Web applications** section and expand it.
    3. Select the **Calculate bandwidth and refresh every** checkbox.
    4. Set the time period for bandwidth evaluation.
    5. Save your changes.
3. Wait some time for the new traffic to come, and see the results.

Continuous delivery and test automation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Analyzing test automation data

Automation and integration

Capture performance data from tests

Test automation explained

Test automation FAQ

Test automation walkthroughs

Watch and learn

Continuous delivery with Dynatrace Webinar

Test automation with Dynatrace

Metrics-driven continuous delivery with Dynatrace test automation

Blog posts

An introduction to performance metrics

Database access quality metrics

Logging quality metrics

Doing performance-focused DevOps

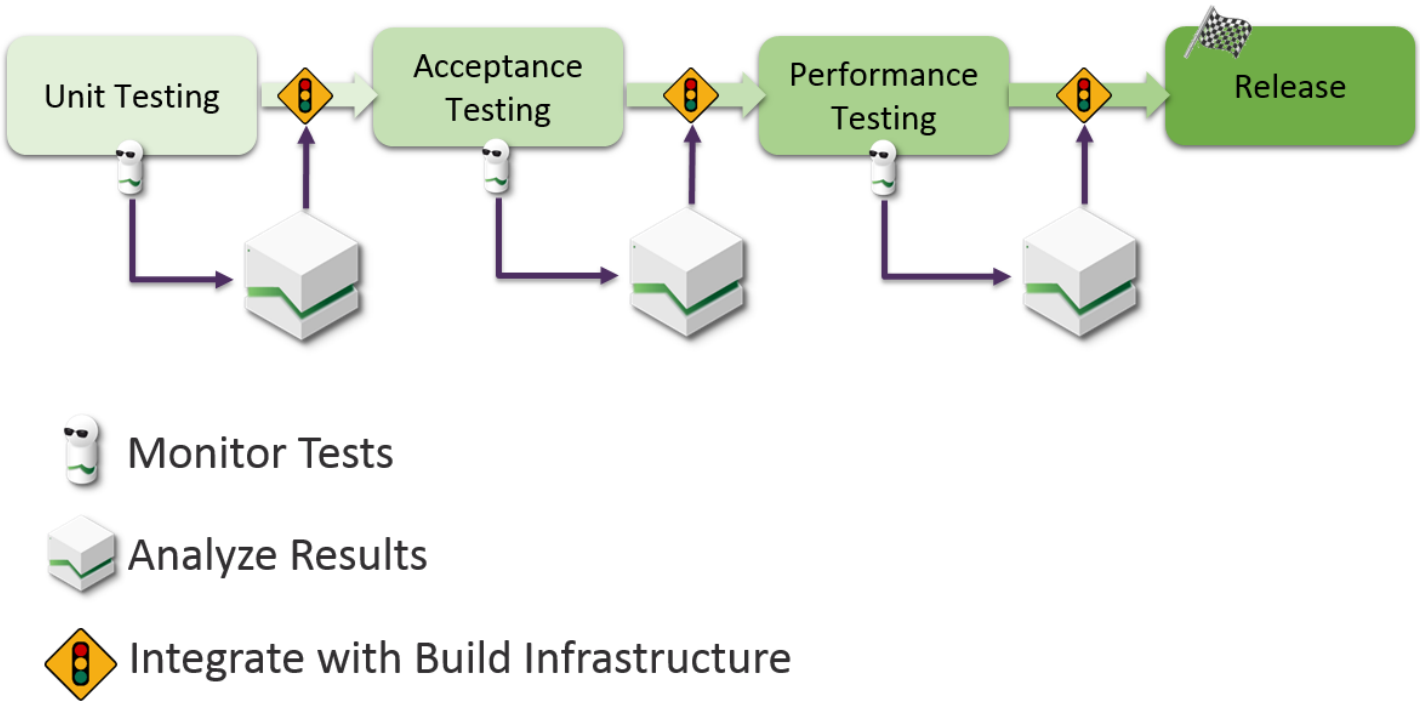
Continuous Delivery 101: automated deployments

AppMon supports continuous delivery processes by:

- Promoting performance data sharing across development, test, and operations.
- Providing reliable metrics along your delivery pipeline through automation interfaces for CI/CD systems, such as Jenkins, Bamboo, and TeamCity.
- Delivering more stable applications in production faster through automatic analysis of regressions and the comparison of response times, structural differences, code executions, errors, exceptions and database performance.

AppMon provides the following support for test automation:

- Automatic collection of performance metrics
- Calculation of upper and lower bounds for expected values.
- Notification of test violations.
- Drilldown and comparison for root cause analysis.



Test automation explained

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Step-by-Step Process to use Test Automation in AppMon

Managing test runs

Related pages

Continuous delivery and test automation

UI-driven browser test

System Profile - Test Automation

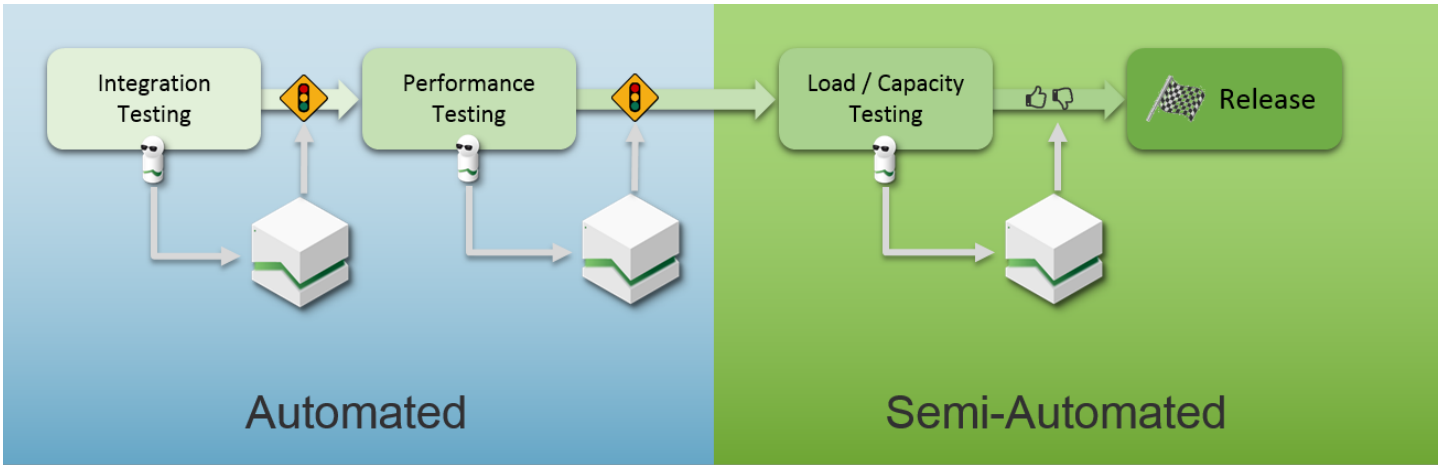
Watch and learn

Test automation with Dynatrace

Overview

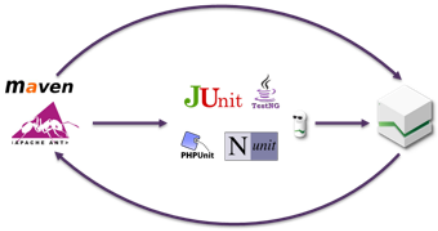
Application performance testing is often not done during software development or is only done at the end of a development cycle. This is because it is a time-consuming task, tests are long-running, and the results can often only be checked manually. However, the later in development a performance defect is detected, the higher the cost of fixing it. AppMon solves this problem by proactively monitoring tests executed in a *Continuous Integration (CI)* environment.

This page describes **Integration and Performance Testing**, which can be fully automated. See Load Tests for Load Testing information.



Every test run captures various measures such as duration, number of executed database statements, and used CPU time. Appmon uses these measures to calculate a *corridor* for the expected minimum and maximum value of a measure. Values outside of this corridor indicate a performance change. Designated developers are automatically notified about these changes.

All performance data is available to different stakeholders within the organization. Developers and architects can use PurePath data to continuously monitor performance and track of architectural metrics to identify regressions early in the development stage. The automated baseline helps this task, and identifies tests that exhibit performance problems. In AppMon, the Test Automation Dashlet provides an entry point to analyzing performance problems in tests. The metrics collected by AppMon and displayed there are exposed by REST interfaces that allow integration with other tools.



PurePath

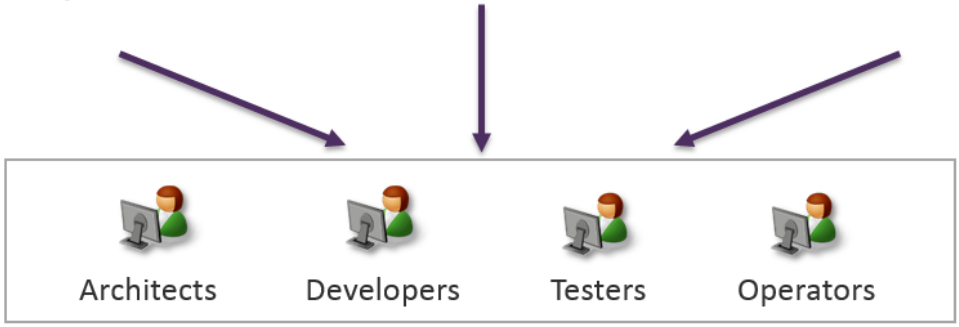
- Architecural validation
  - # of DB calls
  - # Exceptions
- Response time of tests
- Method hotspots

Automatic Baseline

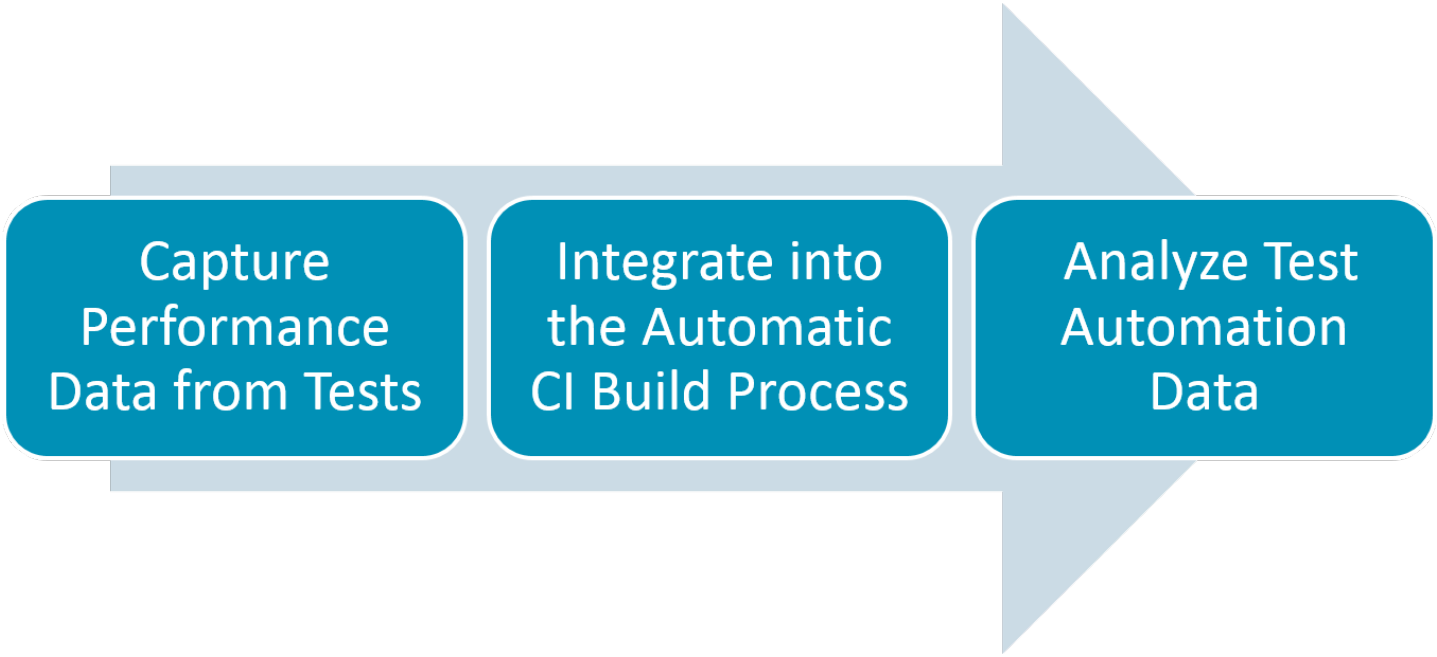
- No need to define thresholds manually
- Identify tests that are not normal

Test Automation Dashlet

- Analyze tests
- Compare tests
- Configure alerting



Step-by-Step Process to use Test Automation in AppMon



Capture performance related data from automatic tests

- See Unit/Integration and Performance Tests
- Web API Tests: Web API Tests
- Test Automation Configuration: System Profile - Test Automation

Integrate into the automatic CI build process

In addition to data captured by AppMon, you can include information such as the version number, a unique version identifier (such as the revision number in Subversion) and other custom data for a test run. Start session recording using tools listed below to gather in-depth data for all performance test runs.

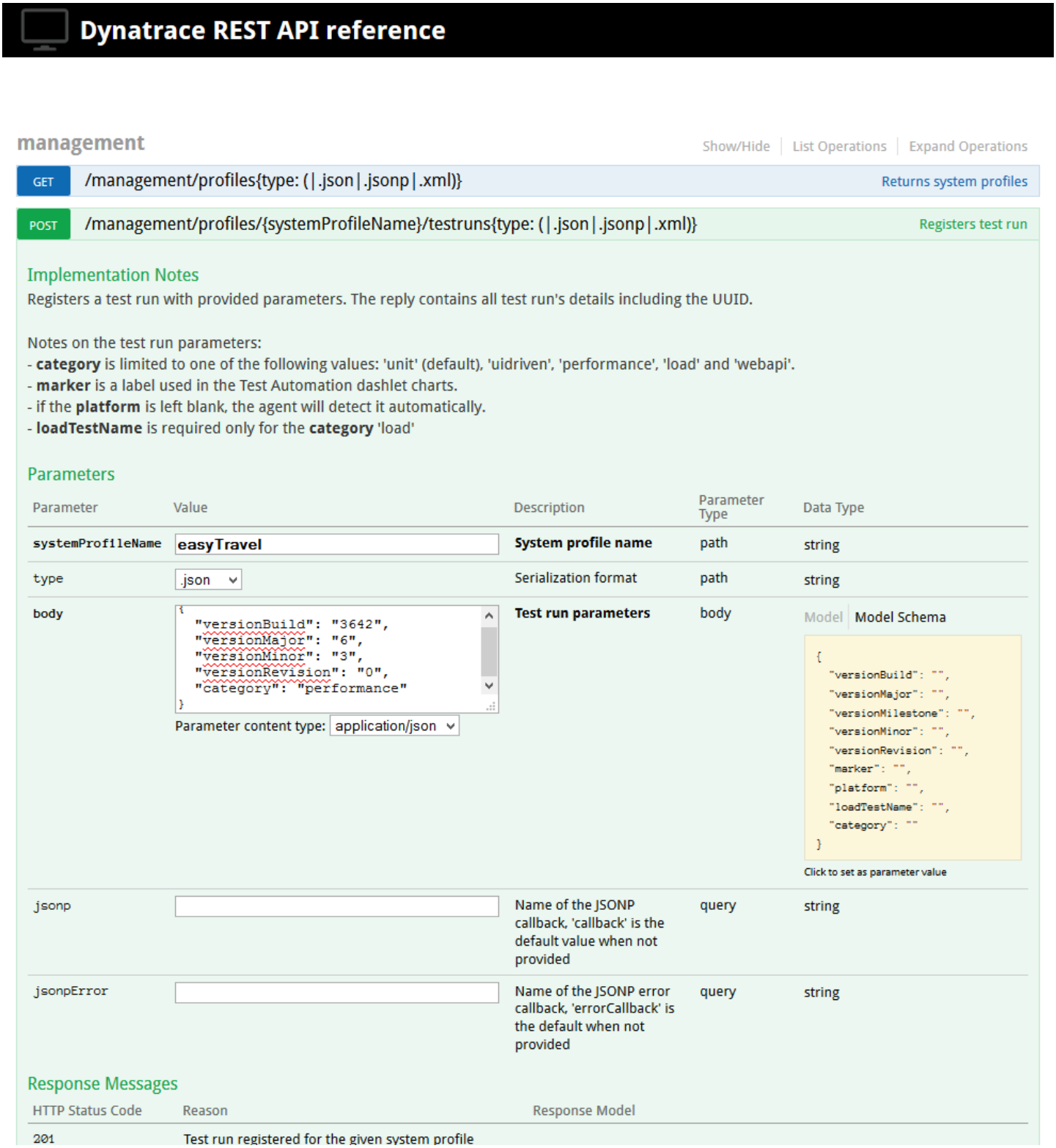
- ANT and Maven – See Automation Library.
- REST – See REST Interfaces for Test Automation.

You can manually start and stop session recording.

> **Note**
>
> Make sure to use trigger session storage from your test automation environment. These provide drilldown and comparison availability.

Setting test information manually

If you need to register a new test run manually, go to `https://<DynatraceServer>:8021/api-docs/index.html` for a web interface using the REST API for test automation. Create a new test run using the `POST /management/profiles/{systemProfileName}/testruns` endpoint.

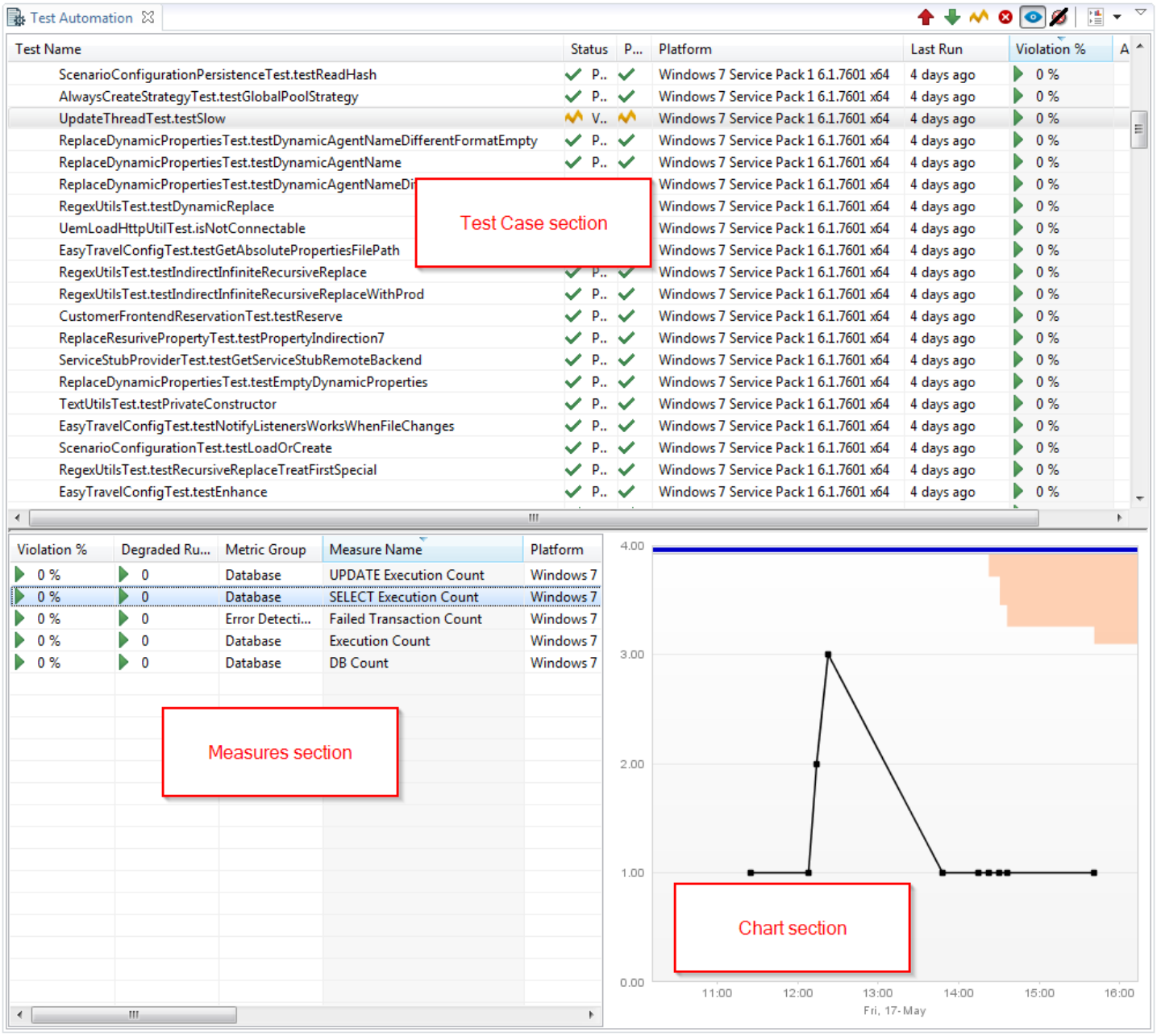


Example form filled with data

Analyze test automation data

Test Results dashlet

The **Test Results** dashlet visualizes the results of various tests in your environment.



This dashlet is only available in the AppMon Test Center edition.

Test case section

The upper section of the dashlet lists the test cases, organized by defined test category and package. Each test case entry shows an aggregated view of the KPIs.

Test status

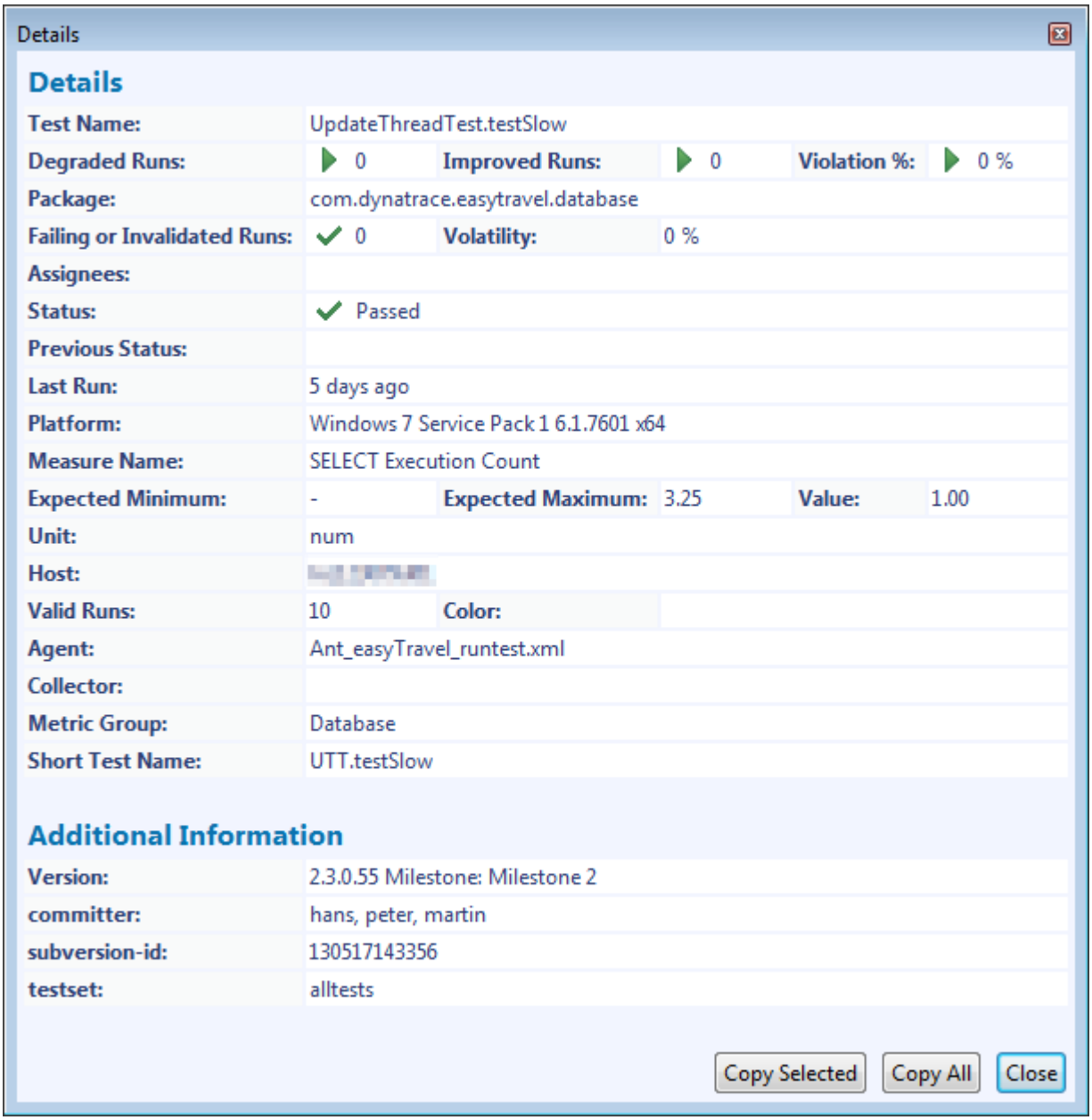
Every test case has a status:

- 
  OK: Test case executed correctly.
- 
  Degrading: Test case runs have become slower.
- 
  Improving: Test case runs have become faster.
- 
  Volatile: Test case has a very volatile outcome — sometimes faster, sometimes slower — so changes in performance might not be recognized correctly. This especially happens with very short-running tests. Try to increase the duration of the test (e.g. by executing the same operation multiple times).
- 
  Failing: Test case has a functional problem, so no performance data could be recorded.
- 
  Invalidated: The last test run of this test case was manually invalidated by the user.
  When you click a test status icon in the toolbar in the upper right corner of the **Test Results** dashlet, related test status columns display in the measure section. For example, click the **Degraded** icon to include the **Degraded Runs** column or the **Volatile** icon to include the **Volatility** column.

Grouping by package can be toggled by clicking the **Group by packages** icon.

Viewing details

Right-click the item and select **Details** to view details for a test case, measure, or test run. You can copy information from the **Details** dialog box.



Measures section

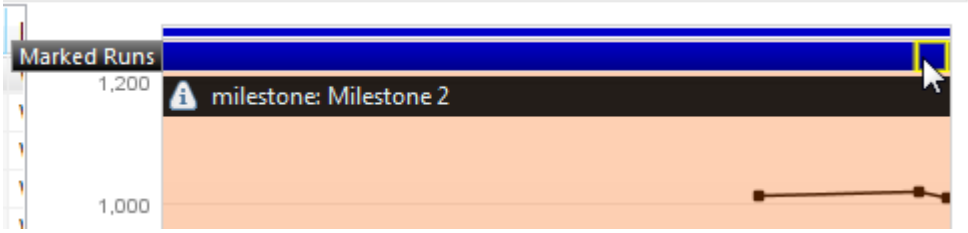
The measure section in the lower left of the dashlet shows the latest values of the KPIs associated with a test case, and indicates whether those values are within, above, or below the corridor.

Chart section

The chart section shows the historical values of a KPI for a test case, including the calculated performance corridor.

Viewing markers

Any marker set in the test metadata displays in the chart's heat field.



See the Test Automation FAQ for more information.

Assigning test cases

Test Cases can be assigned to a specific user in the system. Depending on the configuration, this user will receive email notifications for failing, degrading, improving, or volatile tests.

To assign a test, right-click a test case and select **Assign Test** to display the dialog box shown below. A test case can be assigned to one or more users.

Managing test runs

Comparing test runs

Comparing test runs and drilldown are only available if session recording was enabled during execution of the test case.

Do one of the following to compare two test runs.

- Double-click a test run. A new dashboard comparing the selected test run with the last test run appears.
- Select two test runs and click **Compare**. A new dashboard comparing the selected test runs appears.

Accepting test run changes

Sometimes a change in performance is unavoidable as functionality increases. In such a case, the performance change has to be manually accepted. Right-click a test run in the chart and select **Accept Change from here**. When the dialog box appears, select the appropriate option:

- **All Violated**: All consecutive test runs beginning with the selected run are used to calculate a new corridor. Use this option when there are multiple changes in performance.
- **Only Selected**: Use this option to use specific test runs to calculate the new corridor.

Invalidating test runs

Often, changes in performance are not caused by changes in the system, but by some random factors that affect only the given test run. For example, a database cleanup job was running at the same time as a test case. To exclude an outlier from the regular corridor calculation, invalidate it by right-clicking the test run and selecting **Invalidate Test Run**.

Capture performance data from tests

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Test category

Related pages

External tests

Load tests

UI-driven browser test

Unit/integration and performance tests

Web API tests

Tests are organized into five categories. A category principally defines the key performance indicators (KPIs) that AppMon evaluates for your test. For every test category, only KPIs of interest are shown. The configuration required for capturing performance data depends on the test category and on the test framework used.



Load Testing is handled differently. See Load Test Overview dashboard) for more information.

Test category

Unit / integration test

- **Test Framework supported**: JUnit, TestN, Nunit, Visual Studio Unit Testing Framework, PHPUnit.
- **Usage**: Integration test where time based metrics are not relevant, such as when run on a developer machine. Only architectural metrics are tracked.

See Unit/Integration and Performance Tests for more information.

Performance test

- **Test Framework supported**: JUnit, TestN, Nunit, Visual Studio Unit Testing Framework, PHPUnit.
- **Usage**: Integration test where time based metrics are relevant. Architectural metrics and time based metrics are tracked.

See Unit/Integration and Performance Tests for more information.

Web API test

- **Test Framework supported**: SoapUI, JMeter, any web request/service testing tool.
- **Usage**: Single web request / service validation. Do not use for load testing.

See Web API Tests for more information.

UI driven / browser test

- **Test Framework supported**: Selenium, QuickTest Progessional, SilkTest Metrics, any browser automation tool.
- **Usage**: Browser and Mobile Browser testing.

See UI-Driven Browser Test for more information.

External test

- **Test Framework supported**: n/a

- **Usage**: External test category allows users to provide test measures from any other testing tool via REST API and aggregate them in Dynatrace AppMon Test Automation.

See External Tests for more information.

Unit/integration and performance tests

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Java test framework

.NET test framework

PHP test framework

Measures

Related pages

UI-driven browser test

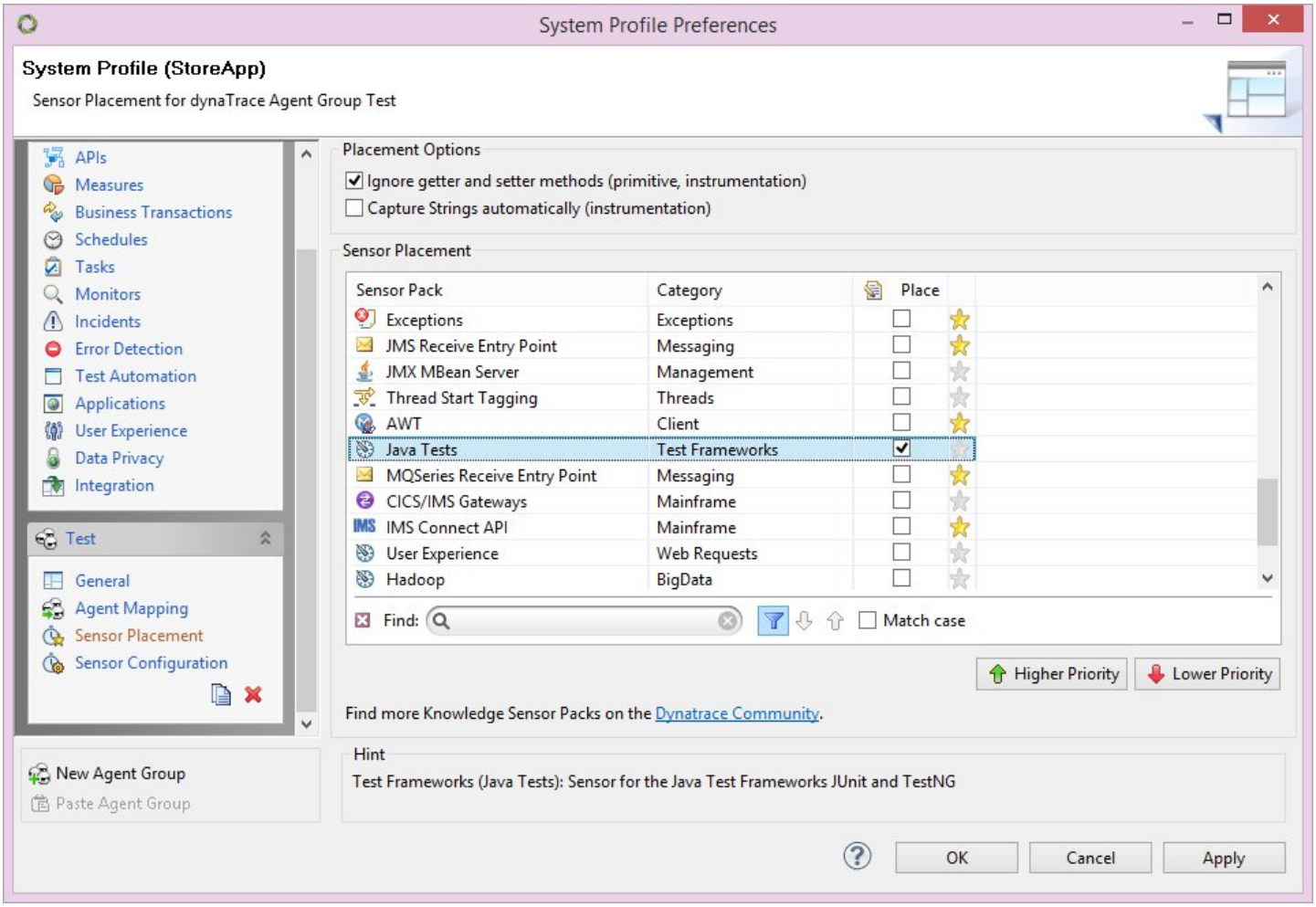
Web API tests

Load tests

.NET Agent configuration

Java test framework

To capture data, enable the Java Tests Sensor Packs in the **System Profile Preferences**. The Java Tests pack captures data from JUnit and TestNG.



Java tests sensor pack

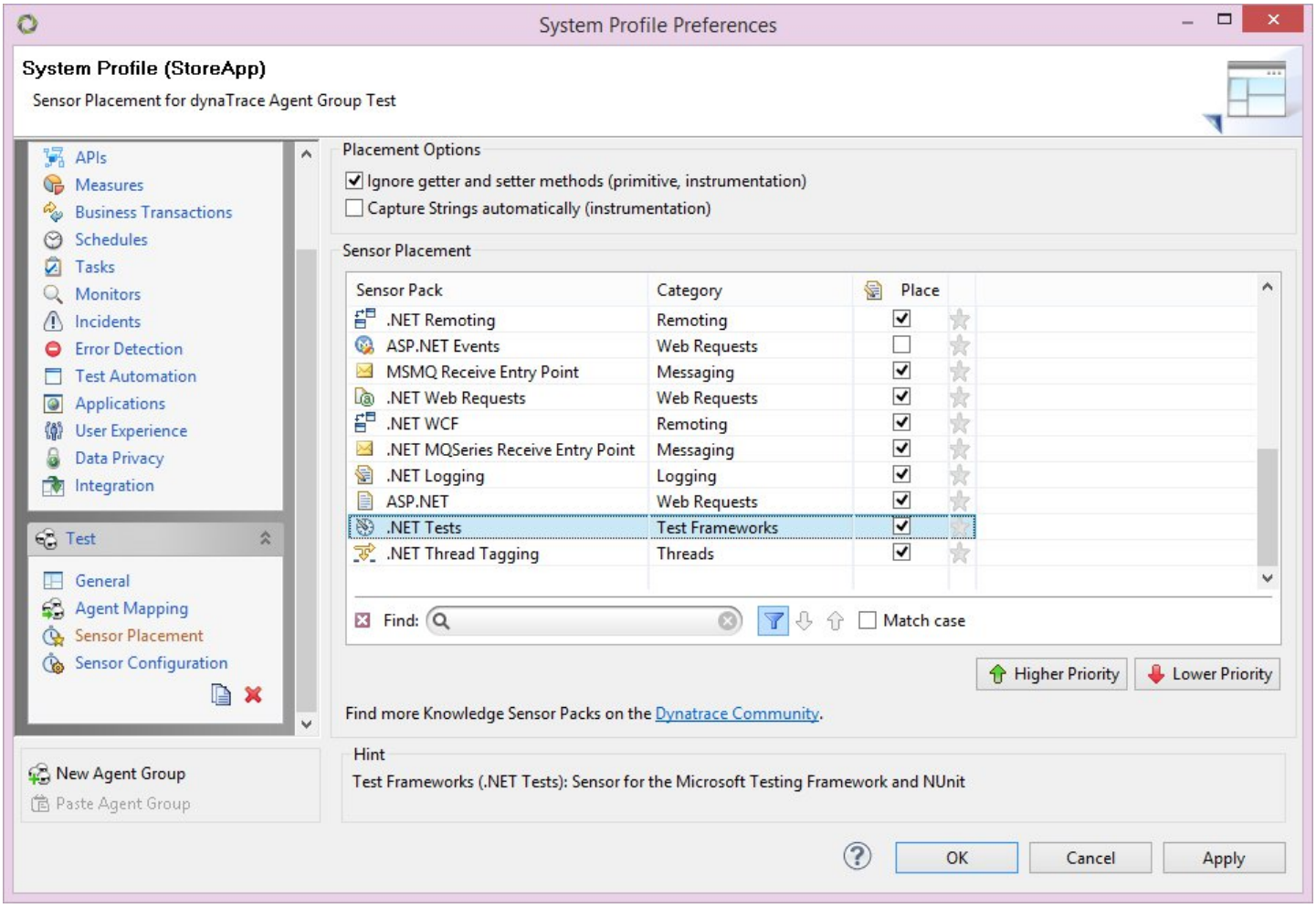
Placing the Java Tests Sensor Pack instruments the following methods.

Framework	Instrumentation Details
JUnit	All methods annotated with `org.junit.Test` are instrumented unless the annotation uses the `expected` property for expected Exceptions; this feature is not supported by the AppMon Sensor. Methods annotated with `org.junit.Ignore` are not instrumented. Any method with a name starting with `test` and `testclass` is considered a subclass of `junit.framework.TestCase`, to maintain compatibility with JUnit 3.*x*.
Test NG	All methods annotated with `org.testng.annotations.Test` are instrumented, unless the annotation property `expectedExceptions` is used or the `enabled` property is set to `false`.

Some other methods are also instrumented to gain additional information on, for example, parameterized tests: constructor of classed annotated with `org.junit.runner.RunWith` and the internal
method `org.junit.runners.Parameterized$TestClassRunnerForParameters.compu`

.NET test framework

To capture data, enable the .NET Tests Sensor Packs in the System Profile Preferences. The .NET Tests pack captures data from NUnit and Visual Studio Unit Testing Framework.



.NET tests sensor pack

Placing the .NET Tests Sensor instruments the following methods.

Framework	Instrumentation	Comment

Visual Studio Unit Testing Framework	Classes with the attribute `Microsoft.VisualStudio.TestTools.UnitTesting.TestClassAttribu` methods with the attribute `Microsoft.VisualStudio.TestTools.UnitTesting.TestMethodAttrib` instrumented, unless the method has the attribute `Microsoft.VisualStudio.TestTools.UnitTesting.ExpectedExceptio`
NUnit	Classes with the attribute `NUnit.Framework.TestFixtureAttribute` and methods attribute `NUnit.Framework.TestAttribute` or `NUnit.Framework.TestCase` are instrumented; also all `test()` methods (for backward compatibility), unless the method has `NUnit.Framework.ExpectedExceptionAttribute`.

Instrumentation

Depending on the technology and test framework, you must instrument processes for automated testing.

Sensorpack	Framework	Process to instrument
.NET Tests	NUnit	`nunit.exe`
.NET Tests	Visual Studio Unit Testing Framework (tests run directly from Visual Studio)	`vstest.executionengine.x86.exe`
.NET Tests	Visual Studio Unit Testing Framework (tests run from command line with `vstest.console.exe`)	`vstest.console.exe`, `QTAgent.exe`, `QTAgent32.` `QTAgent_35.exe`, `QTAgent32_35.exe`, `QTAgent_40.exe`, `QTAgent32_40.exe`
.NET Tests	MSTest (tests run with the console application `MSTest.exe`)	`QTAgent.exe`, `QTAgent32.exe`, `QTAgent_35.exe`, `QTAgent32_35.exe`, `QTAgent_40.exe`, `QTAgent32_40.exe`
.NET Tests	MSTest (tests run with reSharper)	`JetBrains.ReSharper.TaskRunner.CLR4.MSIL`

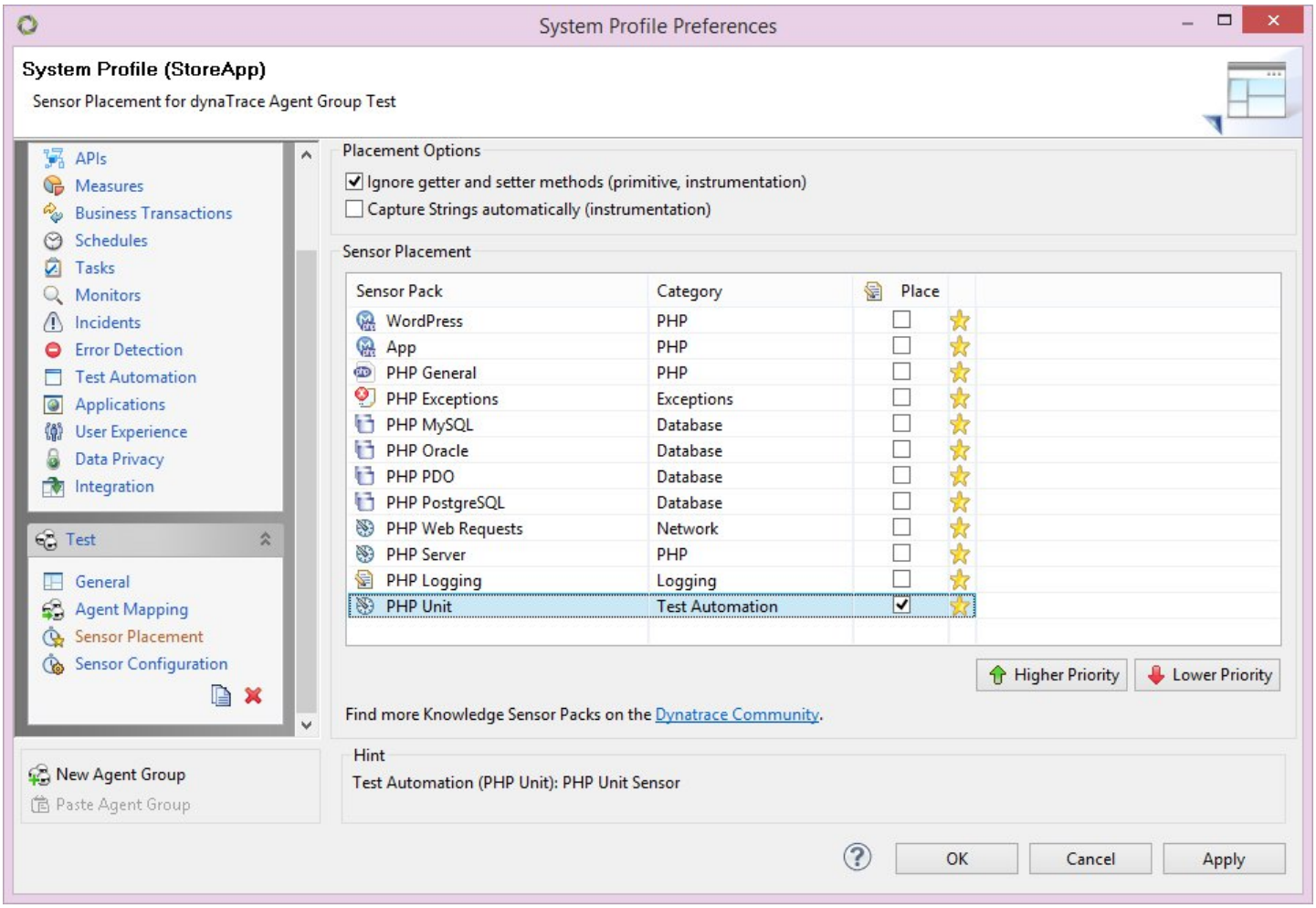
See .NET Agent Configuration for information on how to instrument .NET processes.

Known limitations

When using `vstest.console.exe` with the command-line option `/EnableCodeCoverage`, data capturing does not work. This usually applies if you analyze code coverage on the build service. See Analyzing code coverage in the build service for information on how to enable/disable this option from the Visual Studio Team Explorer.

PHP test framework

To capture data, enable the PHP Tests Sensor Packs in the System Profile Preferences.



Measures

Measures based on following Metric / Metric Groups are captured during test executions:

Unit Tests

Category	Metric Name	Notes
Remoting	Count	
Database	Execution Count	

Methods	Invocation
Exceptions	Count
Logging	Count
Web Requests	Number of Requests
Web Services	Count

Performance Tests

Category	Metric Name	Notes
Remoting	Count	
Database	Execution Count	
Methods	Invocation	
Exceptions	Count	
Logging	Count	
Web Requests	Number of Requests	
Web Services	Count	
PurePaths	PurePath response time	
PurePaths	PurePath CPU duration	

PurePaths	PurePath duration
PurePaths	PurePath duration without suspension

UI-driven browser test

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Requirements

Configuration

Mark a test as failed

Integration

Measures

Samples

Related pages

Test automation explained

Unit/integration and performance tests

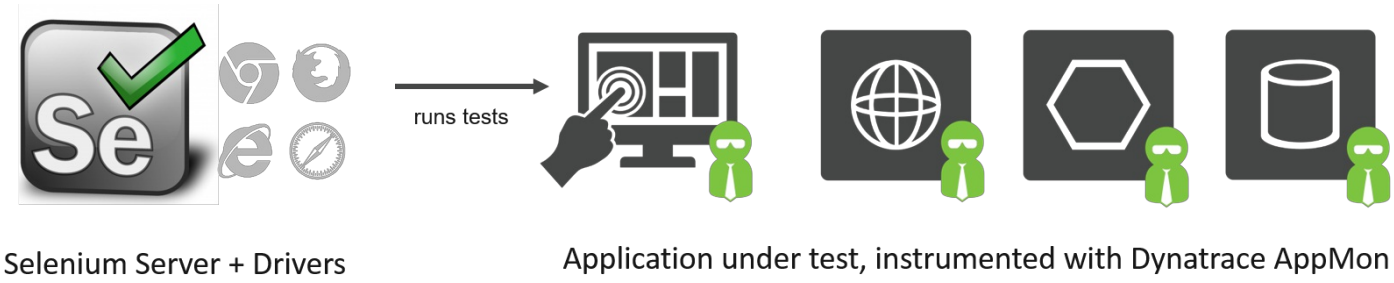
Web API tests

Watch and learn

Online perf clinic - UEM test automation

Functional tests from software testing frameworks such as Selenium, QuickTest Professional, or SilkTest can be easily turned into Performance Tests. You can track Architectural metrics such as the number of images, JavaScript files, third party requests, database and web services calls for each test executions.

The Test Automation Dashlet tracks UEM visits tagged as tests.

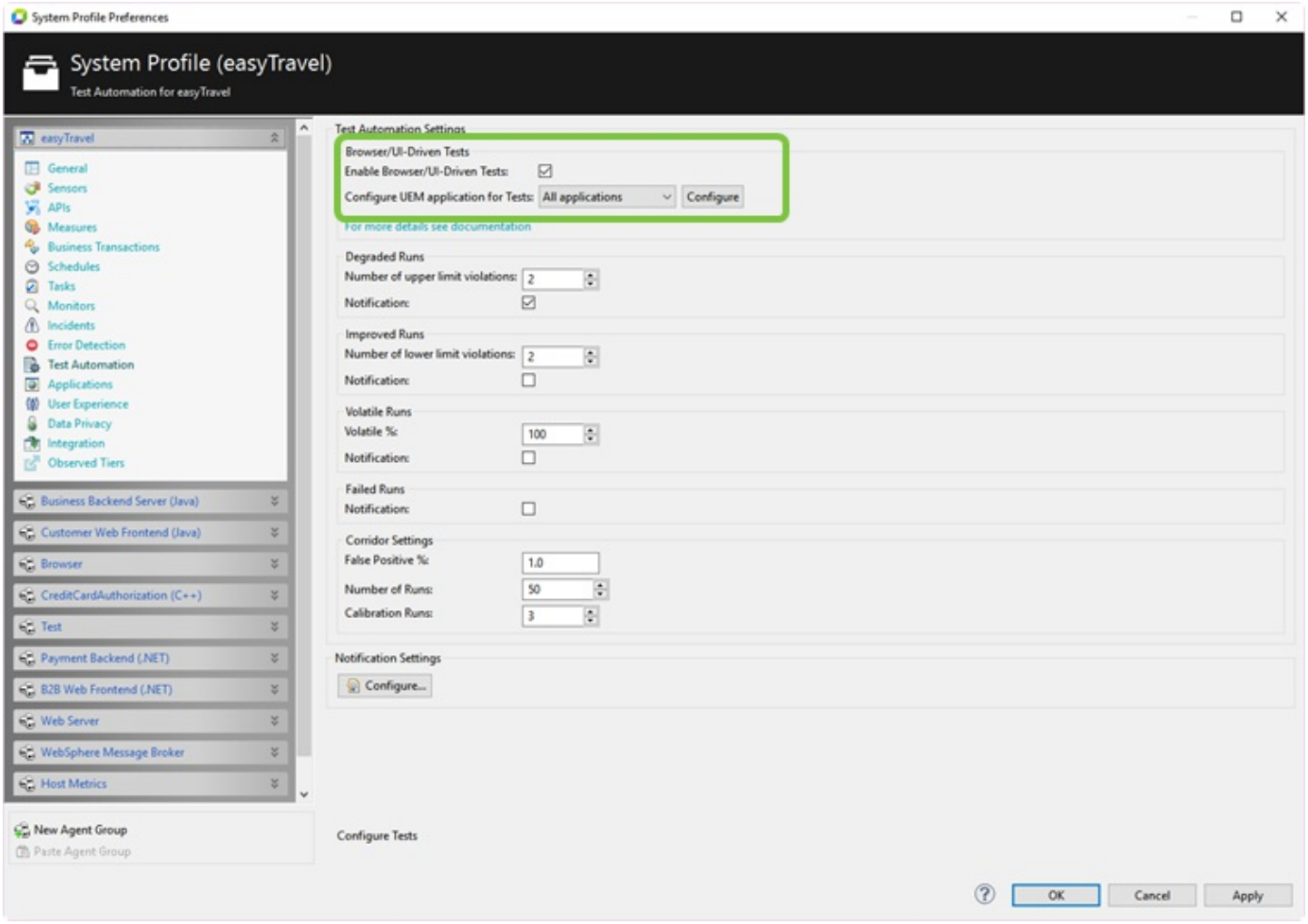


For AppMon 6.5 and later, Browser Tests use the UEM JavaScript Agent to capture the relevant metrics. In earlier versions, the now deprecated Browser Agent was used to capture the data. See AppMon 6.3 documentation for more information about Browser Agent Integration.

Requirements

- The application under test must be instrumented with AppMon agents.
- UEM must be configured to capture visits.

Configuration



1. Enable Browser/UI-Driven Tests in the Test Automation Settings.
2. Select the test application and click **Configure** to automatically configure all required UEM settings. The following settings are automatically configured but can be reverted/adapted manually:
   - UEM Meta Data capturing enabled to capture testname and testrun id. See UEM Web Settings - Metadata Capturing.
   - W3C metrics. See W3C Resource Timing Metrics for more information.
   - Visits time-out (5 minutes).

Mark a test as failed

All tests are marked as passing by default. Do the following to manually control the test result.

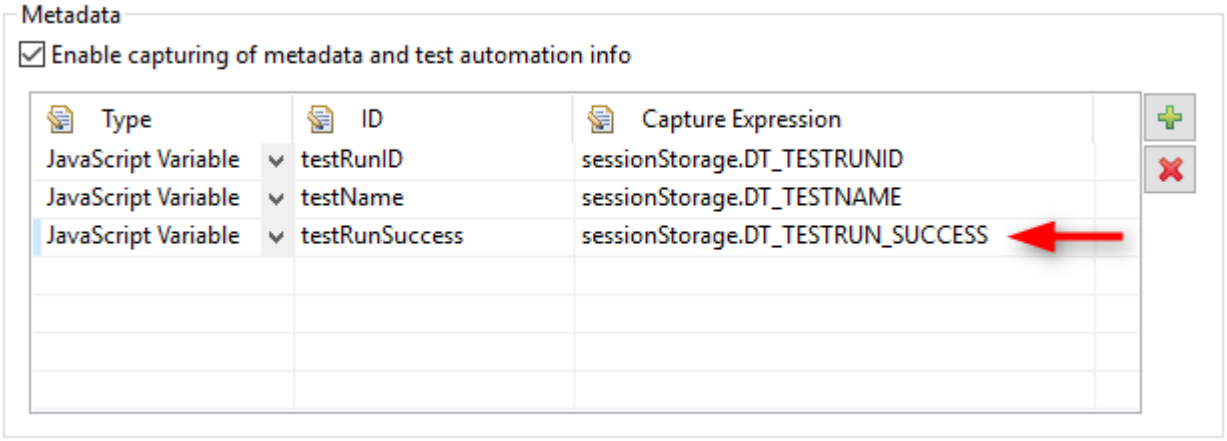
1. Configure an additional JavaScript metadata variable to be captured with the id `testRunSuccess`. If a metadata variable with id `testRunSuccess` exists, tests are marked as failing by default. This metadata value is not configured when configuring UEM Applications for tests.

2. With `testRunSuccess` defined, a test is marked as passing if at any point during the visit an Action arrives with the `testRunSuccess` expression value of `true`. For example, a test may set a JavaScript variable `testRunSuccess` to `true`, and navigate to another page to propagate the metadata to the server via an action.

Setting `testRunSuccess` to any value other than `true` has no impact on the test result. You cannot reset the valuee once the test is marked as passing using this expression. For example, assigning a false value has no impact.

Please make sure the action following `testRunSuccess` completes normally. You can use the *wait.until(ExpectedConditions.elementToBeClickable(By.id("XXX"))* Selenium construct to do this. If the action is not complete, it may not be picked up by the JavaScript agent. This this occurs, the test result is not forwarded to AppMon.

The `sessionStorage` prefix is used because the value set is likely not picked up by the first action in which the value is set, and must survive until the next action.



See UEM Web Settings - Metadata Capturing for more information.

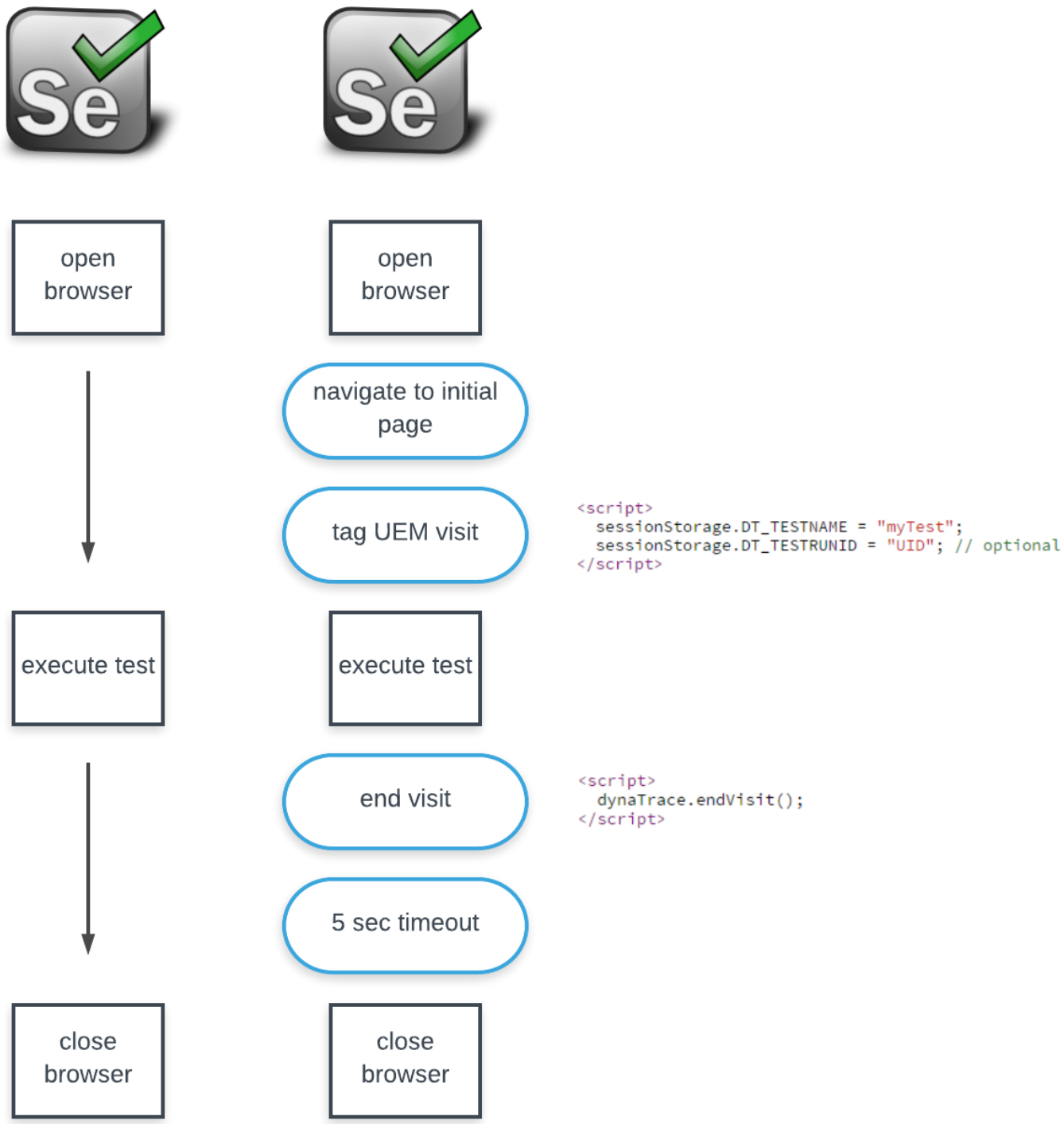
Integration

The typical sequence for a browser test is:

- Open the Browser
- Run the Test
- Close the Browser

Only the following changes are needed in your script are needed to track performance metrics for your tests using AppMon.

- Tag the UEM visit with a test name and (optional) with a test run id.
- End the UEM visit before closing the browser (optional, but recommended for best performance).



Tag the UEM visit as a test

Using the metadata capturing feature, the JavaScript agent picks up the test name and test run id. Ensure that `sessionStorage` is set after the browser has navigated to the domain in which the test executes. Session storage scopeds according to the same-origin-policy, within a browser window/tab.

```
sessionStorage.DT_TESTNAME = "myTest";
// test run id is optional
sessionStorage.DT_TESTRUNID = "12345...";
```

Java Example

```
JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("sessionStorage.DT_TESTNAME = \"myTest\";");
// test run id is optional
js.executeScript("sessionStorage.DT_TESTRUNID = \"12345...\";");
```

End the visit before closing the browser (optional)

Test automation data calculates after the visit ends. Because of this, end the visit before closing the browser to speed up test results, otherwise the visit times out at the configured visit timeout. Please allow up to 10 seconds before closing the window after `endVisit` invokes to allow invocation propagation from `endVisit()` to the AppMon Server.

```
dynaTrace.endVisit();
```

Java Example

```
js.executeScript("dynaTrace.endVisit();");
```

Measures

Measures based on following Metric / Metric Groups are captured during the test executions:

Category	Metric Name	Notes
Remoting	Count	

Database	Execution Count	
Methods	Invocation	
Exceptions	Count	
Logging	Count	
Web Requests	Number of Requests	
Web Requests	Bytes Received	
Web Requests	Bytes Sent	
Web Services	Count	
W3C Timings	W3C Resource Count	See W3C Resource Timing Metrics for more information.
W3C Timings	W3C Resource Timing	See W3C Resource Timing Metrics for more information.
Client Errors	Count	
Visits	Action Count	
Visits	Failed Actions Count	

Samples

See https://github.com/Dynatrace/Dynatrace-Test-Automation-Samples/tree/master/selenium.

Web API tests

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Usage

Related pages

Unit/integration and performance tests

UI-driven browser test

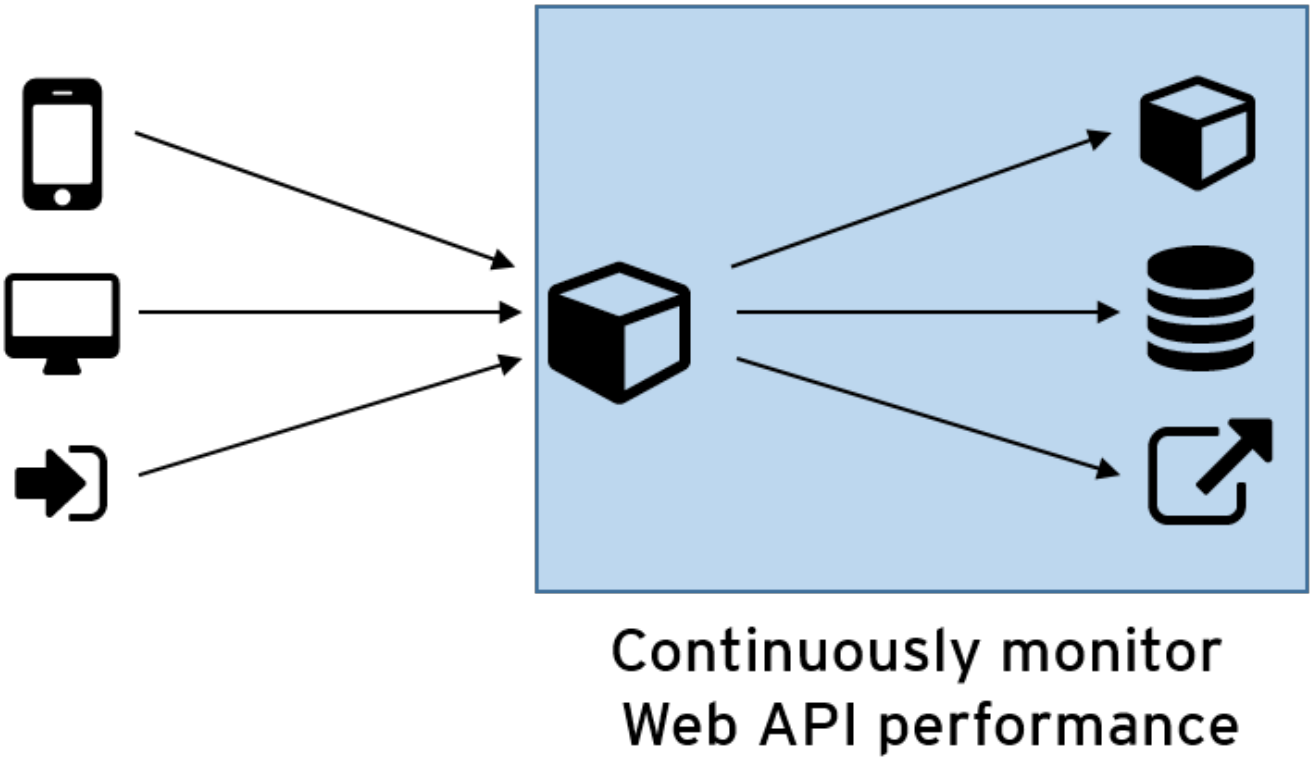
Load tests

External tests

Integrate web API performance monitoring in JMeter

Integrate web API performance monitoring in SoapUI

Web APIs are interfaces provided by a web server that can be used by a variety of clients like smartphone apps, rich-client applications, or other web applications. It is crucial to continuously ensure API performance, and with the Web API test category it is now possible to continuously monitor performance metrics for HTTP-based web service APIs. See Capture Performance Data from Tests for a list of metrics available for the Web API category.



Usage

Perform the following steps to execute a Web API test.

1. Register a new test run through the REST interface using the **webapi** category.
2. Send one or more HTTP requests to the URI page you want to test. These requests have to include the X-dynaTrace header where the test run ID returned by the AppMon Server REST Interface is specified to let AppMon know that these requests belong to a test. See below for a description of fields in the X-dynaTrace header.
3. Analyze the test results in the Test Automation dashlet, or retrieve them using REST API.

In case you don't want to provide specific test metadata, you only need to provide the test name using the **TN** field in X-dynaTrace header (see below). That way test run will be registered automatically.

X-Dynatrace header possible key-value pairs

The table lists possible key-value pairs. They can be provided in any order, separated with a semicolon.

Note

The **TN** field should be used to provide test name. The **TR** and **RC** fields can still be used if you want to provide test run id or the expected response code.

Key	Description	Usage
TN	Test Name (TN) should be provided using this field. Metrics for requests with the same name are aggregated. You can execute as many tests as desired within the same test run, but give them different names if they are not testing the same thing.	Required. You can still use **NA** and **TR** fields instead.
NA	Old way of passing Test **Na** me.	**Deprecated** Requires **TR** field to be provided
TR	TestRunId (TR), from a registered test run. See REST API POST testruns for more information.	Optional when **TN** field us used. If not provided, a test registers automatically. Required with **NA** field.
RC	Expected Response Code (RC). If the test results in any other response code, the test is marked as failed.	Optional. The default is 200.

X-Dynatrace header sample

Using test run id registered using Test Automation REST API, expecting 200 response code:

```
X-dynaTrace: TN=SampleTestName;TR=7a0520fc-eda9-4f78-a8fa-4cb7d04ac8b8;RC=200
```

Relying on automatic test run metadata registration (no test metadata provided), expecting default response code (200):

```
X-dynaTrace: TN=SampleTestName
```

Measures

Category	Metric Name	Notes
Webrequests	Bytes Sent	

Webrequests	Bytes Received
PurePaths	PurePath Duration
PurePaths	PurePath Response Time
Methods	Invocation
Logging	Count
Exceptions	Count
Database	Execution Count

Walkthroughs

See Integrate Web API Performance Monitoring in JMeter or Integrate Web API Performance Monitoring in SoapUI for more information how to use the Web API test category.

Load tests

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Unit/integration and performance tests

UI-driven browser test

Web API tests

Load tests can be analyzed using a separate Load Test Overview dashboard. You must record sessions for performing load tests.

Perform the following steps to record a session with a load test.

1. Start the session recording, with your session name specified. You can do that manually or using the REST API.
2. Run your load test.
3. Stop recording.

To analyze the session containing your load test, use the **Analyze Load Test Results** link from the Test Overview dashlet or choose the **Start Center > Analyze Performance > Analyze Load Test Results** link for more information.

External tests

On this page

Overview

Related pages

Web API tests

Server REST interfaces

Overview

The external tests category analyzes and aggregates measures from outside sources with the AppMon Test Automation module. Use it to put all test-related data in a single place, for incident tracking and corridor calculation.

Posting data

First, register a test run with the **external** category to get its unique ID (TestRunID). Using REST endpoint matching your test run, make a POST requests to send data for your tests. Each test may have multiple measures posted in a single request. You can adapt measure unit and color to your needs. It will be updated (if changed) when new data is posted. You can also provide expected min and max value for each measure to track the corridor on chart.

See the AppMon REST API documentation available under [https://<AppMon_server_host>:8021/api-docs/current/index.html#!/Test_Automation/postExternalTestData] and AppMon Server REST API documentation for more information.

Example

Register external TestRun:

```
curl -X POST -H "Content-Type: application/json" -d '{
  "category": "external"
}' "https://localhost:8021/api/v2/profiles/MyProfile/testruns"
```

Post external measures using the TestRun ID:

```
curl -X POST -H "Content-Type: application/json" -d '{
    "testName": "package.Class.testName",
    "measures": [
      {
        "name": "SomeMeasure",
        "value": "123.4",
        "timestamp": "2017-05-01T11:12:45",
        "unit": "s",
        "color": "#FF0000"
      },
      {
        "name": "OtherMeasure",
        "minValue": "5.0",
        "value": "123.4",
        "maxValue": "150.0",
        "timestamp": "2017-05-01T11:12:50",
        "unit": "B",
        "color": "#00FF00"
      }
    ]
}' "https://localhost:8021/api/v2/profiles/MyProfile/testruns/<testRunID>"
```

Analyzing test automation data

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Test Overview dashlet

Test Results dashlet

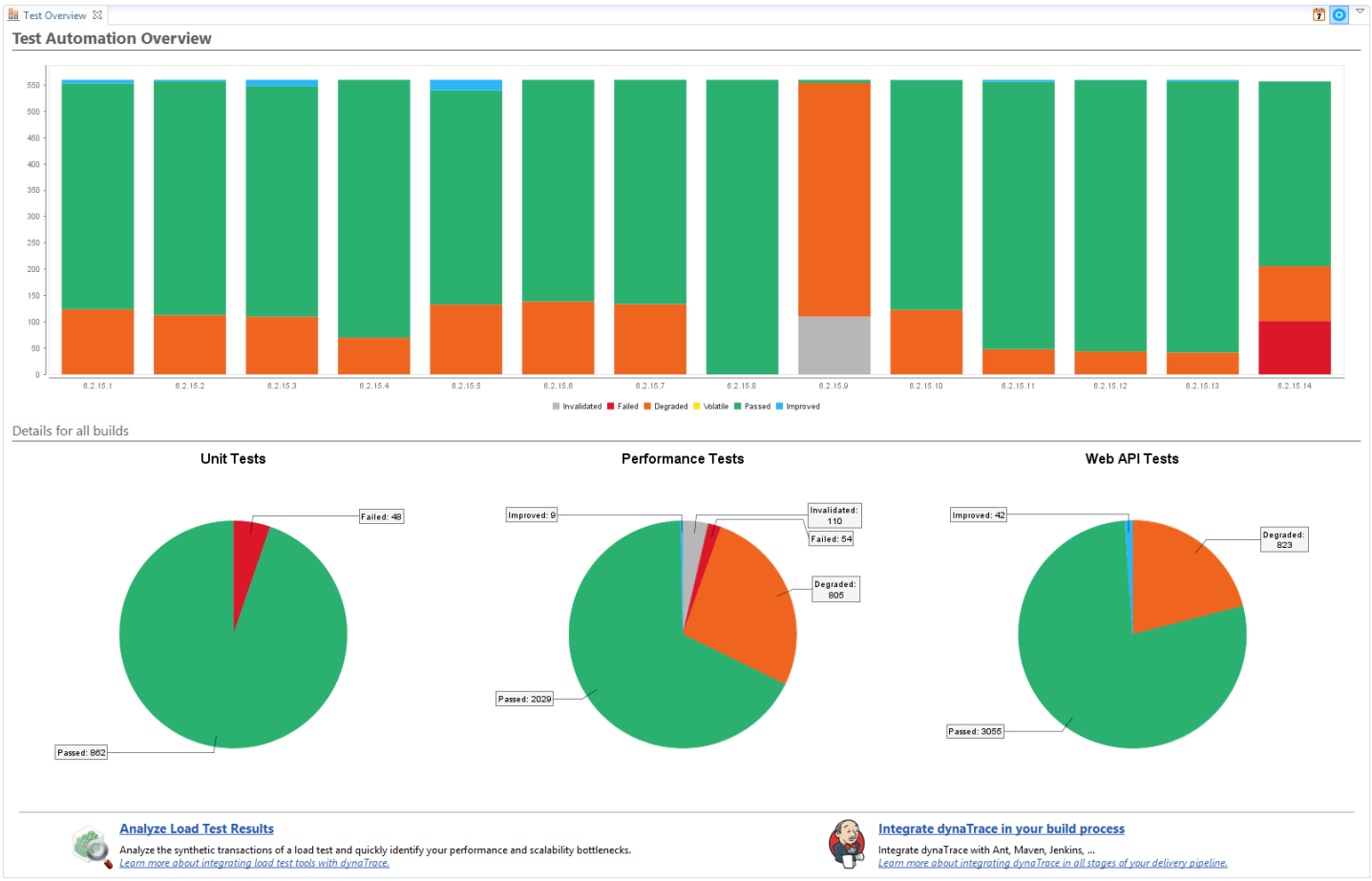
Load Test Overview dashboard

Jenkins plugin

Refer to the options below to analyze test automation data in AppMon or in your continuous integration server environment.

Test Overview dashlet

The **Test Overview** dashlet displays the general status of tests in your environment.



Note

This dashlet is only available in the Pre-Production (formerly Test Center) Edition.

Bar chart section

In the upper section of the dashlet, a bar chart displays information about the overall number of test executions with each status. The X-axis contains the aggregation specific keys and the Y-axis shows the number of matching test executions. You can choose the aggregation type in the dashlet header.

Aggregation types

The following aggregation types are available:

- **By build**: Test executions with the same build version parameters (major.minor.revision.build) are aggregated.

- **By day**: Test executions done in the same day are aggregated.

Pie charts section

Below the **Bar Chart** section, pie charts show the number of test executions with each status, as well as the specified test category such as Unit Tests, Performance Tests, Web API Tests, Browser Tests. There is one pie chart per each test category. Only pie charts with a test category that have at least one test execution are shown. As a result, one to four pie charts can be visible in this section.

By default, the pie charts filter test executions by matching test category and show a consolidated view of the details for all your builds for the selected timeframe.

Selecting any bar in the **Bar Chart** section by clicking it once also filters the data shown in **Pie Charts** section by aggregation specific key, as shown in the bar's X-axis. Clear the selection by clicking the empty surface of the chart above the bar or between the bars. The current selection has a thin border around the bar and the header of the **Pie Charts** section.

Drilldowns to Test Results dashlet

You can drill down from the Test Results Dashlet dashlet to see more detailed, filtered data by doing one of the following:

- **Double click** on any bar in the **Bar Chart** section. This filters data by an aggregation specific key, as shown in the bar's X-axis.
- **Single click** on the chart or chart title in **Pie Charts** section. This filters the data by a related test category to the chart and the aggregation specific key, as shown in the bar's X-axis if any bar is selected. You can also check it on the header of **Pie Charts** section.

A **Test Results** dashlet opened by drilldown inherits all filters applied to the source **Test Overview** dashlet. For example, the selected timeframe.

Footer section

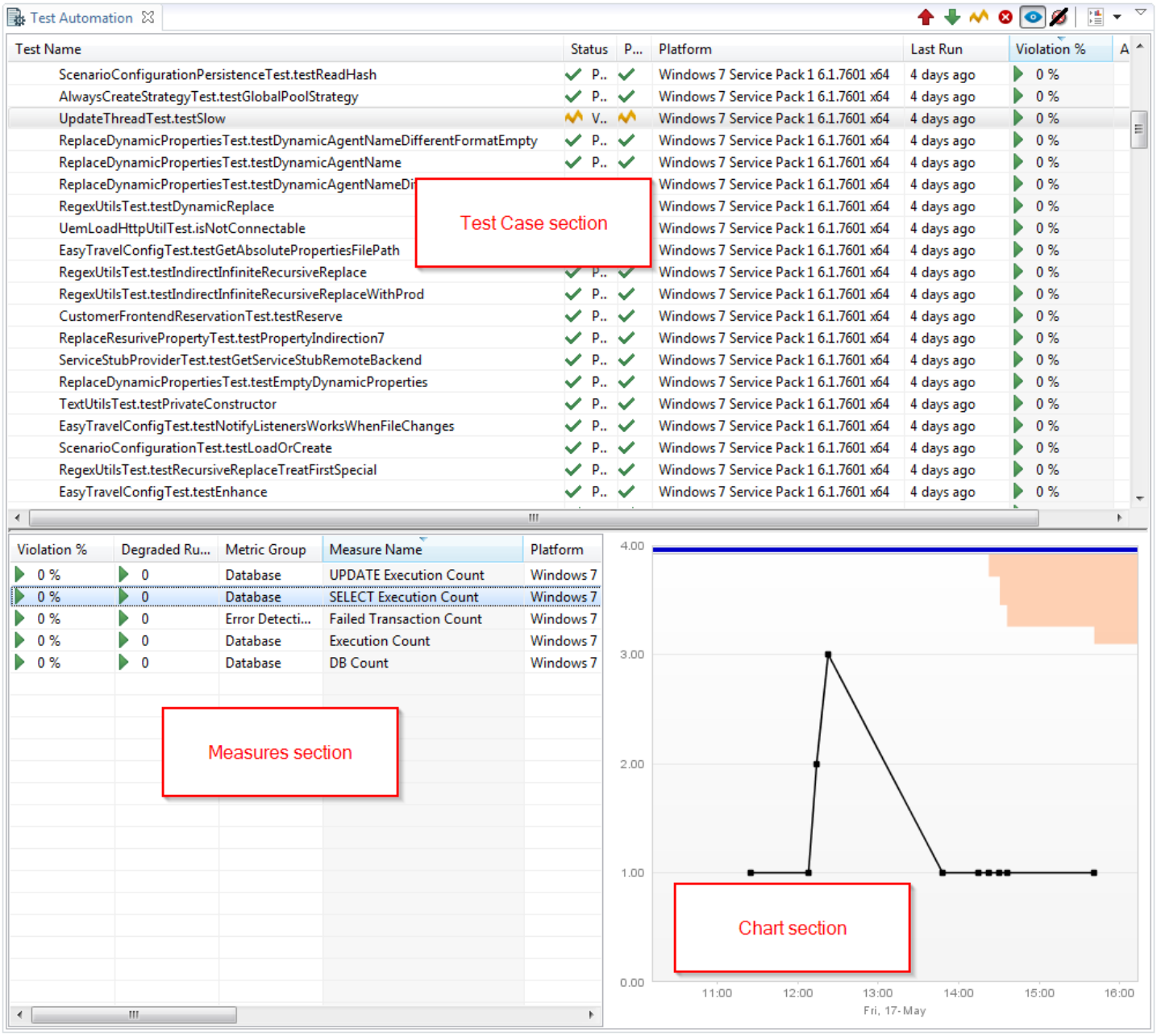
Load tests results

Load tests can be analyzed using a separate Load Test Overview dashboard. You can open it by clicking the **Analyze Load Test Results** link in **Test Overview** dashlet footer. See Load Tests for more information.

The **Footer** section also includes useful documentation links about tests.

Test Results dashlet

The **Test Results** dashlet visualizes the results of various tests in your environment.



This dashlet is only available in the AppMon Test Center edition.

Test case section

The upper section of the dashlet lists the test cases, organized by defined test category and package. Each test case entry shows an aggregated view of the KPIs.

Test status

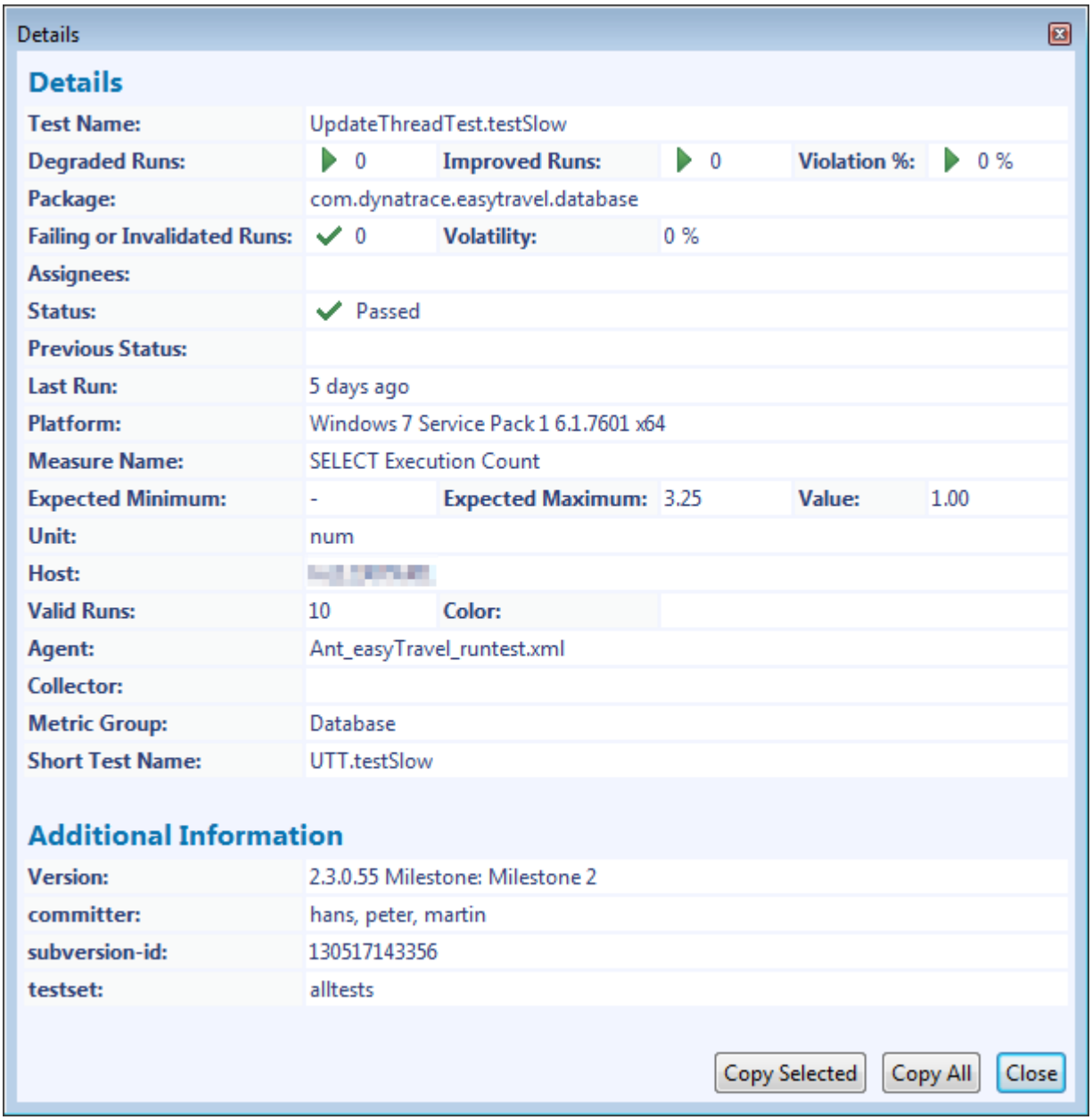
Every test case has a status:

-  **OK**: Test case executed correctly.
-  **Degrading**: Test case runs have become slower.
-  **Improving**: Test case runs have become faster.
-  **Volatile**: Test case has a very volatile outcome — sometimes faster, sometimes slower — so changes in performance might not be recognized correctly. This especially happens with very short-running tests. Try to increase the duration of the test (e.g. by executing the same operation multiple times).
-  **Failing**: Test case has a functional problem, so no performance data could be recorded.
-  **Invalidated**: The last test run of this test case was manually invalidated by the user.
  When you click a test status icon in the toolbar in the upper right corner of the **Test Results** dashlet, related test status columns display in the measure section. For example, click the **Degraded** icon to include the **Degraded Runs** column or the **Volatile** icon to include the **Volatility** column.

Grouping by package can be toggled by clicking the **Group by packages** icon.

Viewing details

Right-click the item and select **Details** to view details for a test case, measure, or test run. You can copy information from the **Details** dialog box.



Measures section

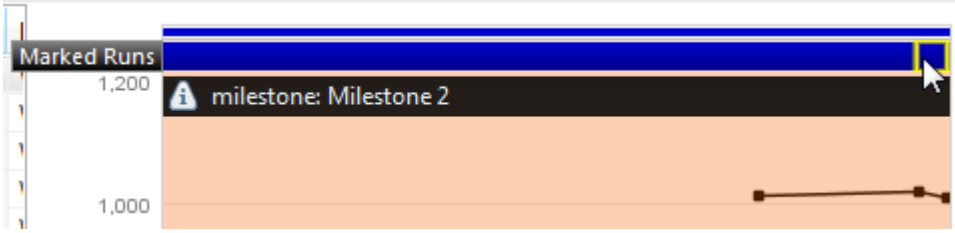
The measure section in the lower left of the dashlet shows the latest values of the KPIs associated with a test case, and indicates whether those values are within, above, or below the corridor.

Chart section

The chart section shows the historical values of a KPI for a test case, including the calculated performance corridor.

Viewing markers

Any marker set in the test metadata displays in the chart's heat field.

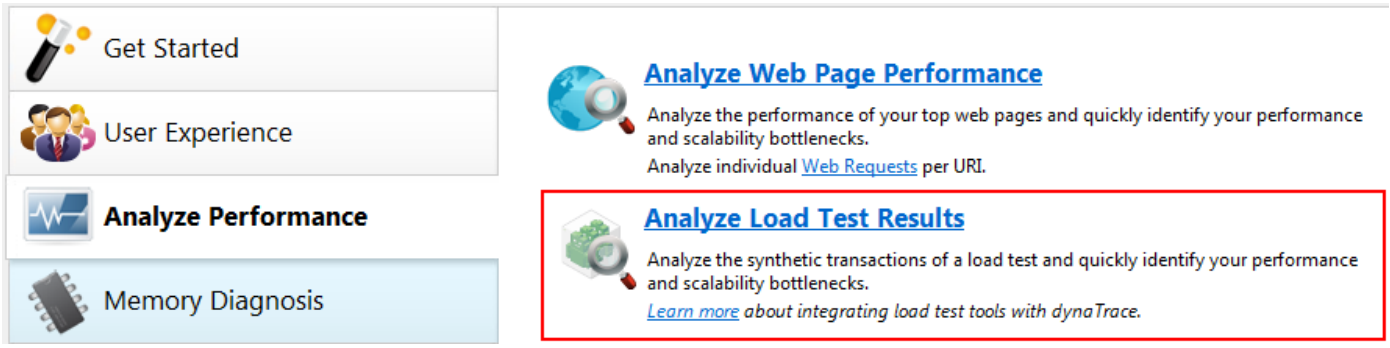


See the Test Automation FAQ for more information.

Load Test Overview dashboard

When you use the Load Testing Integration to run load tests, you can analyze the results that start in the **Tagged Web Requests** dashlet or use the **Load Test Overview** dashboard.

1. Launch the dashboard from the Start Center.



2. Select a stored session or System Profile. Typically, you have a stored session for a previously run load test. The new dashboard that opens up prompts you for the data source. Select a System Profile if you want to analyze the live session or select a stored session.

3. Analyze the load test result.
   The top-left of the dashboard shows the **Tagged Web Requests** dashlet. All other dashlets on that dashboard are linked with that dashlet. You must select a test name (you can also multi-select). Data from the name or names you select displays in the charts and the dashlets such as Transaction Flow Dashlet, Database Dashlet, and Response Time Hotspots Dashlet. Use this to analyze the root cause of tests that run slow.

Jenkins plugin

The AppMon Jenkins plugin retrieves test automation data from the AppMon Server and displays it through charts and tables on the project and build level. See Test Automation Plugin for Jenkins for installation, configuration, and usage instructions.
AppMon 6.3 and later includes a new default setting to accept authentication data only via HTTPS. See Breaking Changes and Potential Pitfalls if you need more information on this.

Automation and integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Build server

Continuous integration builds

REST interfaces for test automation

AppMon integrates seamlessly in unit and integration testing. Use a REST call to pass additional test information. Plugins allow Appmon Test Automation to integrate within your build automation server and execute the required call. You must run tests with the an agent inject into the test application. See the related pages links for more information.

Build server

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Jenkins plugin

Community provided plugins

Watch and learn

Eclipse and Jenkins integration

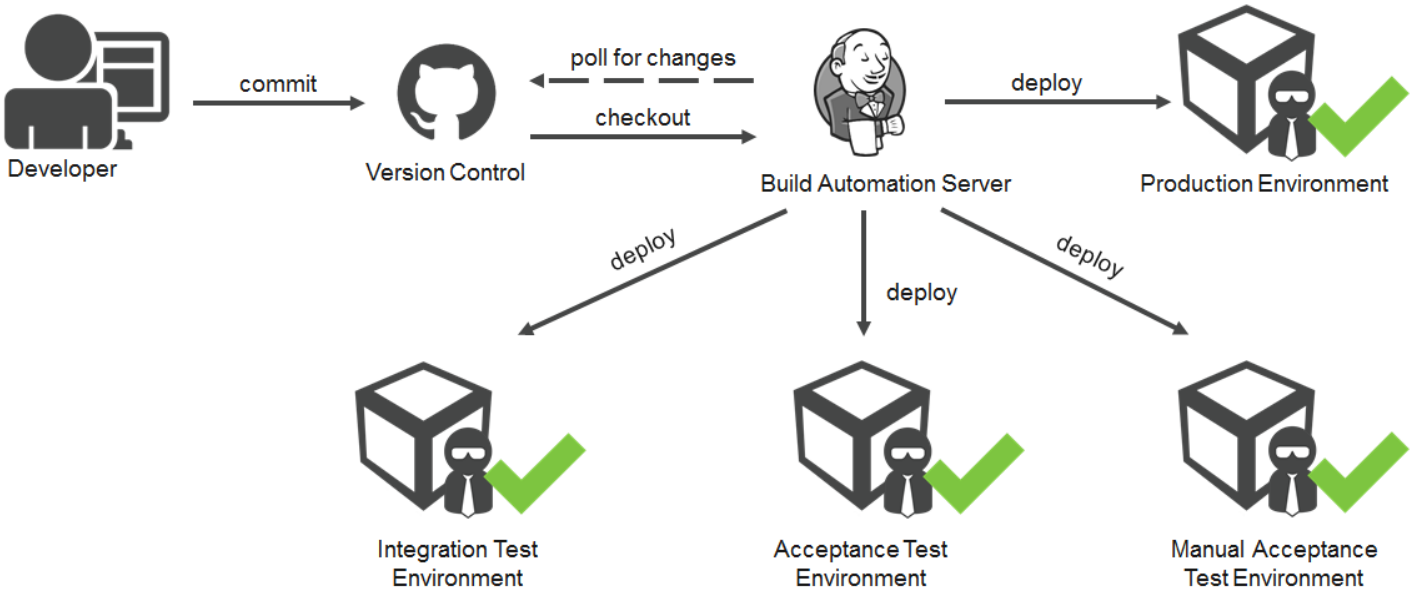
Downloads

Jenkins plugin

Bamboo plugin

AppMon Test Automation can easily be integrated within your Build Automation Server using the following plugins:

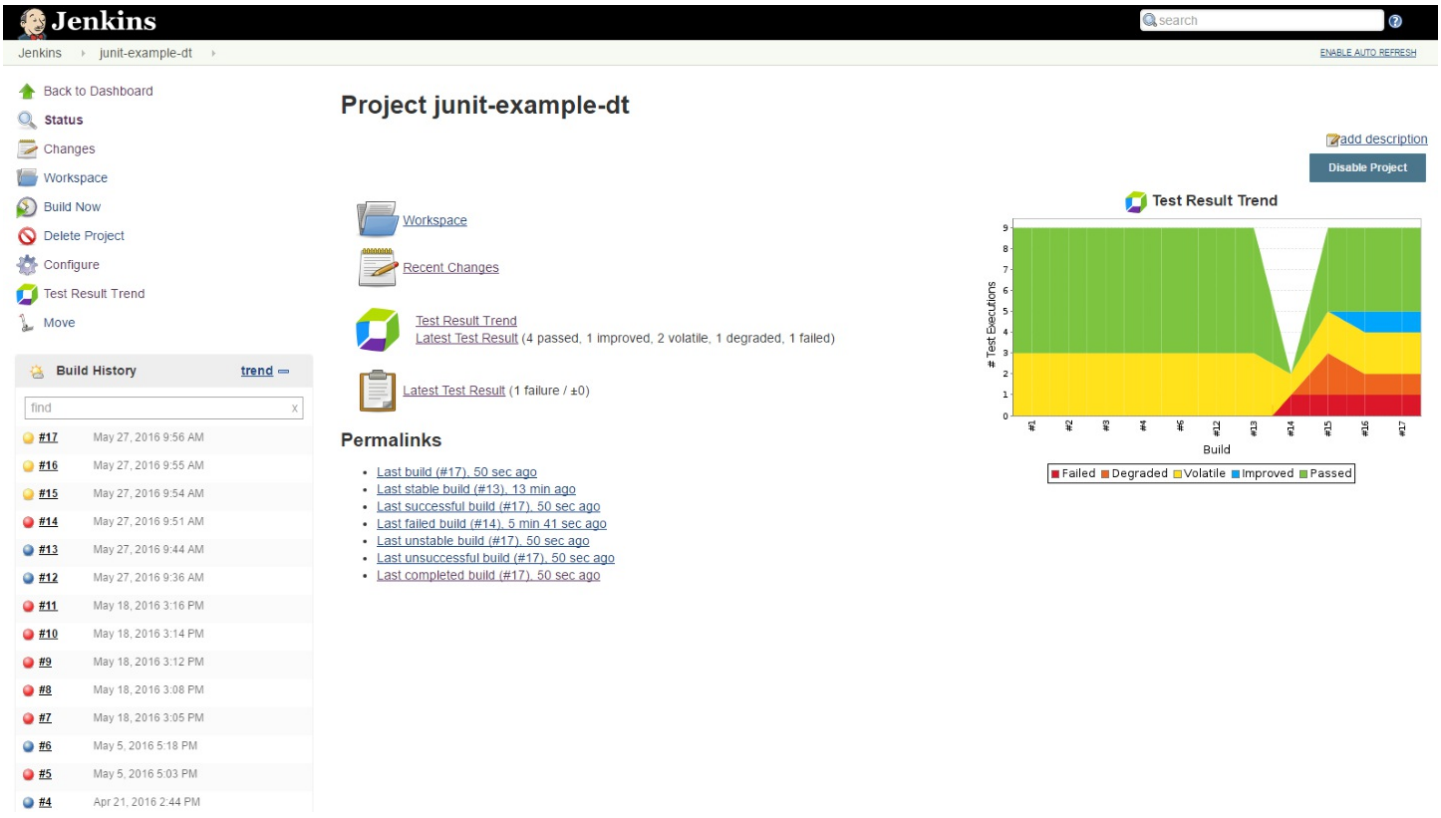
- Jenkins Plugin
- The Community provided Plugin for Bamboo, the Test Automation Libraries for Ant, Maven, Gradle, MSBuild & NAnt or simply by using the Test Automation REST API.

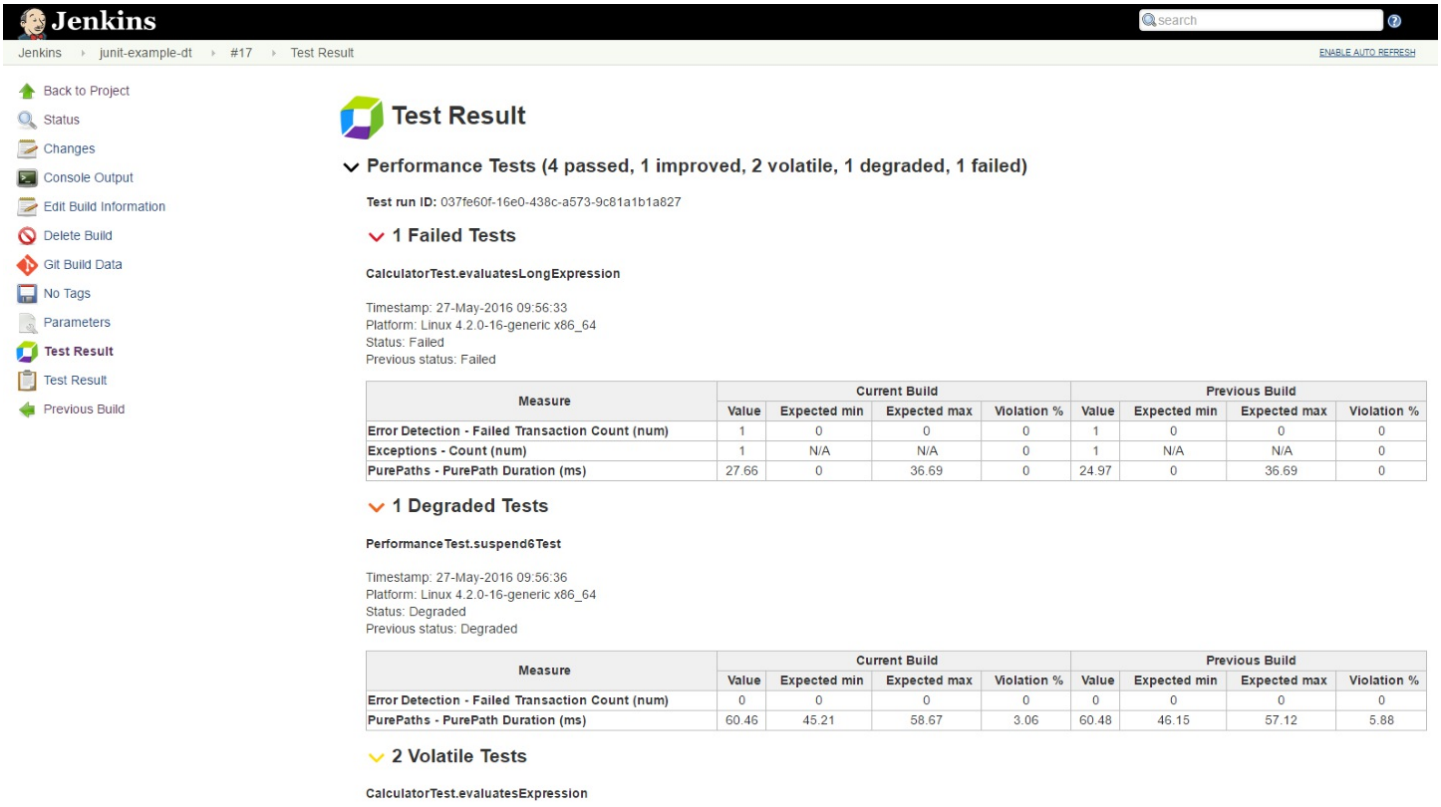


Jenkins plugin

AppMon provides an official Plugin for Jenkins:

- Download & Documentation
- Online Perf Clinic – Eclipse and Jenkins Integration





Community provided plugins

The Community has also provided plugins, including the Test Automation Plugin for Atlassian Bamboo and the Performance Signature Plugin for Jenkins / Hudson. You can find other community plugins on the Community Fastpacks and Plugins page.

Continuous integration builds

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Integration process

Downloads

Ant tasks library

Maven Plug-in

Gradle Plugin

MSBuild tasks library

NANT task library









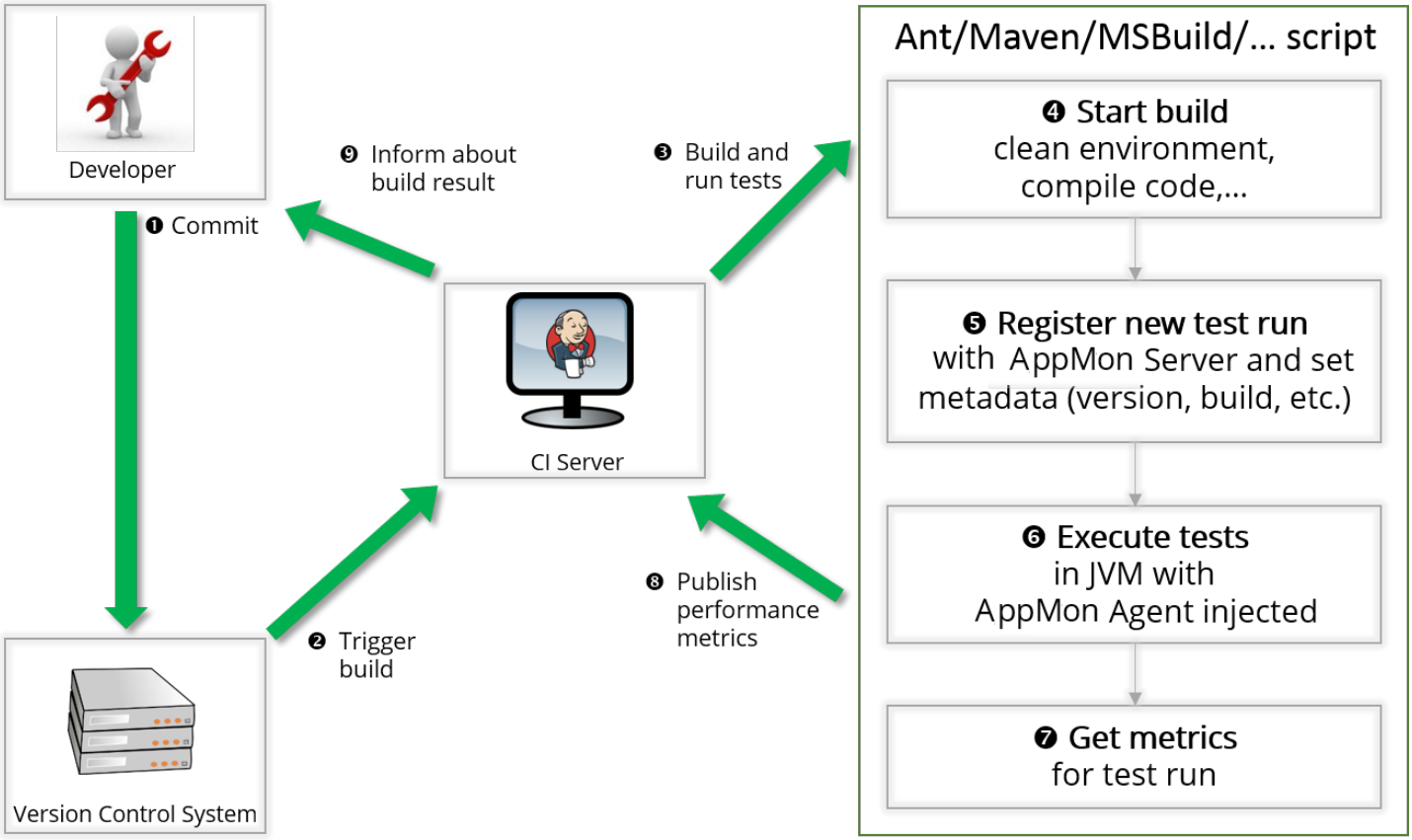


Integrating AppMon in unit and integration tests to use the Test Automation features includes the following:

1. Run your tests with the AppMon Agent injected. Doing this provides end-to-end PurePaths starting in your tests and going all the way through your backend.
2. For the Test Automation dashlet, AppMon allows you to **pass in some additional meta information** like the build version. This information is passed via a REST call to the AppMon Server, and you can either use plug-ins for Ant, Maven, Gradle, MSBuild or NAnt, or execute the required call directly.

Integration process

The full process looks as described in this diagram:



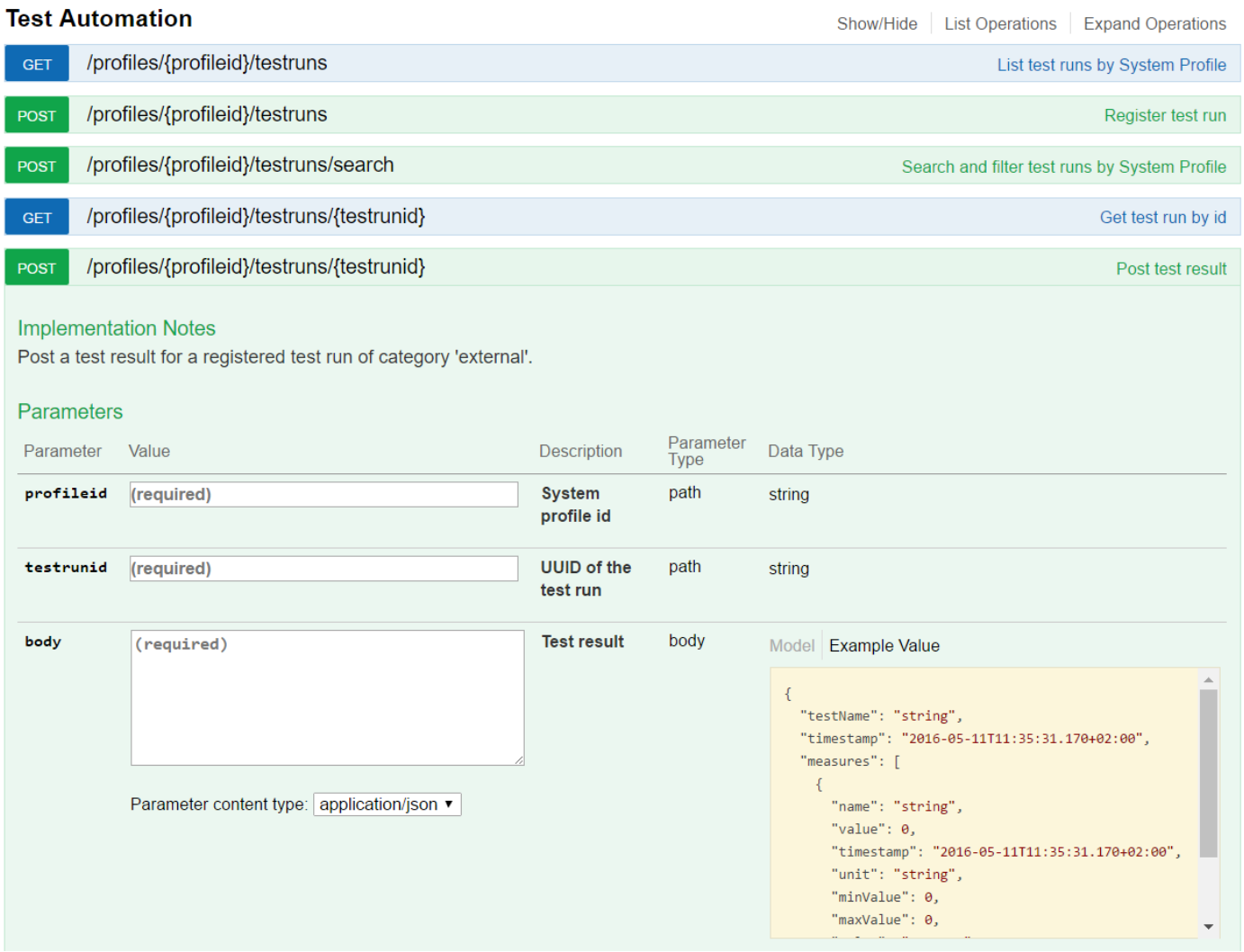
- A build is triggered, for example by a developer checking in code into the repository, which in turn triggers a new build on the CI server.
- The CI server executes the build based on an `Ant/Maven/...` script, and then runs the included tests with an AppMon Agent injected. It also sets test metadata which can include information like the version, build ID, and subversion committers.
- The AppMon Agent collects PurePath data. At the end of the build, the CI server can pull the compiled performance metrics from the server and display them either on a dashboard such as with the Test Automation Plugin for Jenkins), or by storing the HTML or PDF report provided.
- You can also export a session file for a specific test run, which lets you store test suite-executed transactions for later comparison and analysis.

See the following pages for information on download, configuration and usage details for each supported build tool.

- Ant Tasks Library
- Maven Plug-in
- Gradle Plugin
- MSBuild Tasks Library
- NANT Task Library

The automation plugins / libraries provide an easier way to use the relevant REST interfaces on the AppMon Server. It does not perform any injection of agents.

REST interfaces for test automation




```
https://<server>:8021/api-docs/current/index.html#/Test_Automation
```

Detailed REST API descriptions, examples, and interactive testing capability can be found on your local AppMon Server at the following location:

- Create a test run.
- Read a test run with its test results. Various filters can be applied.
- Update a test run state, such as finish, invalidate, or mark as failed.
- Add test results to an external test run.

Test Automation REST API endpoints allow the following operations on test automation data:

Test automation walkthroughs

Integrate Web API Performance Monitoring in JMeter
Integrate Web API Performance Monitoring in SoapUI

Integrate web API performance monitoring in JMeter

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Requirements

Goal

Steps

Download

Related pages

Integrate web API performance monitoring in SoapUI

Overview

Web-based APIs in server-side applications allow other clients such as mobile or AppMon Clients to use their functionality. Tools like JMeter or SoapUI typically perform functional testing of those APIs. You can continuously monitor API performance with AppMon's Web API support.

Requirements

- A suppported version of AppMon with a license that allows using the Test Automation Dashlet (Test Center Edition) - Get the free trial if you're not yet a customer.
- easyTravel
- JMeter. This walkthrough uses JMeter 2.11.

Goal

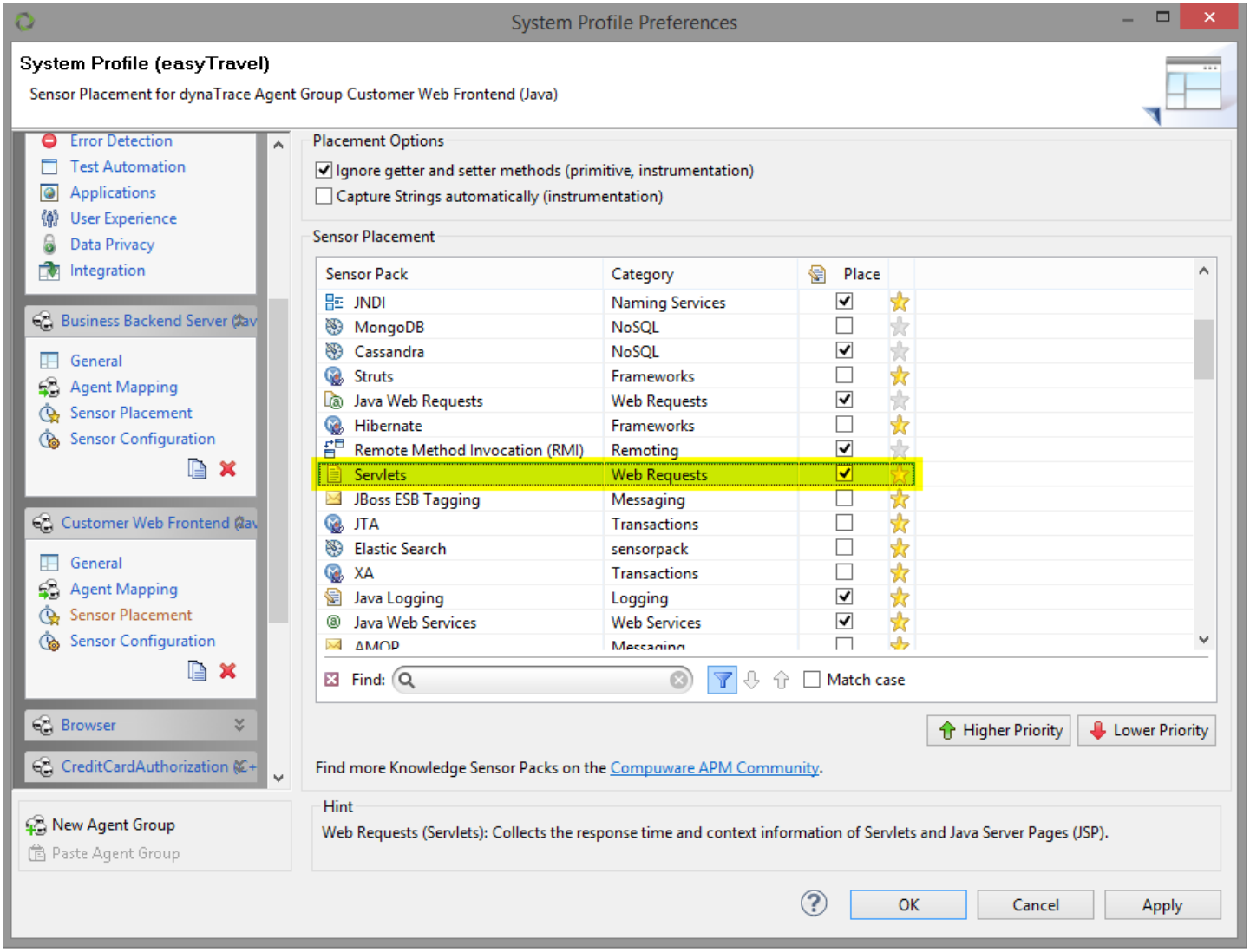
This walkthrough shows you how to:

- Create a JMeter script that tests a web service.
- Analyze the web service performance.

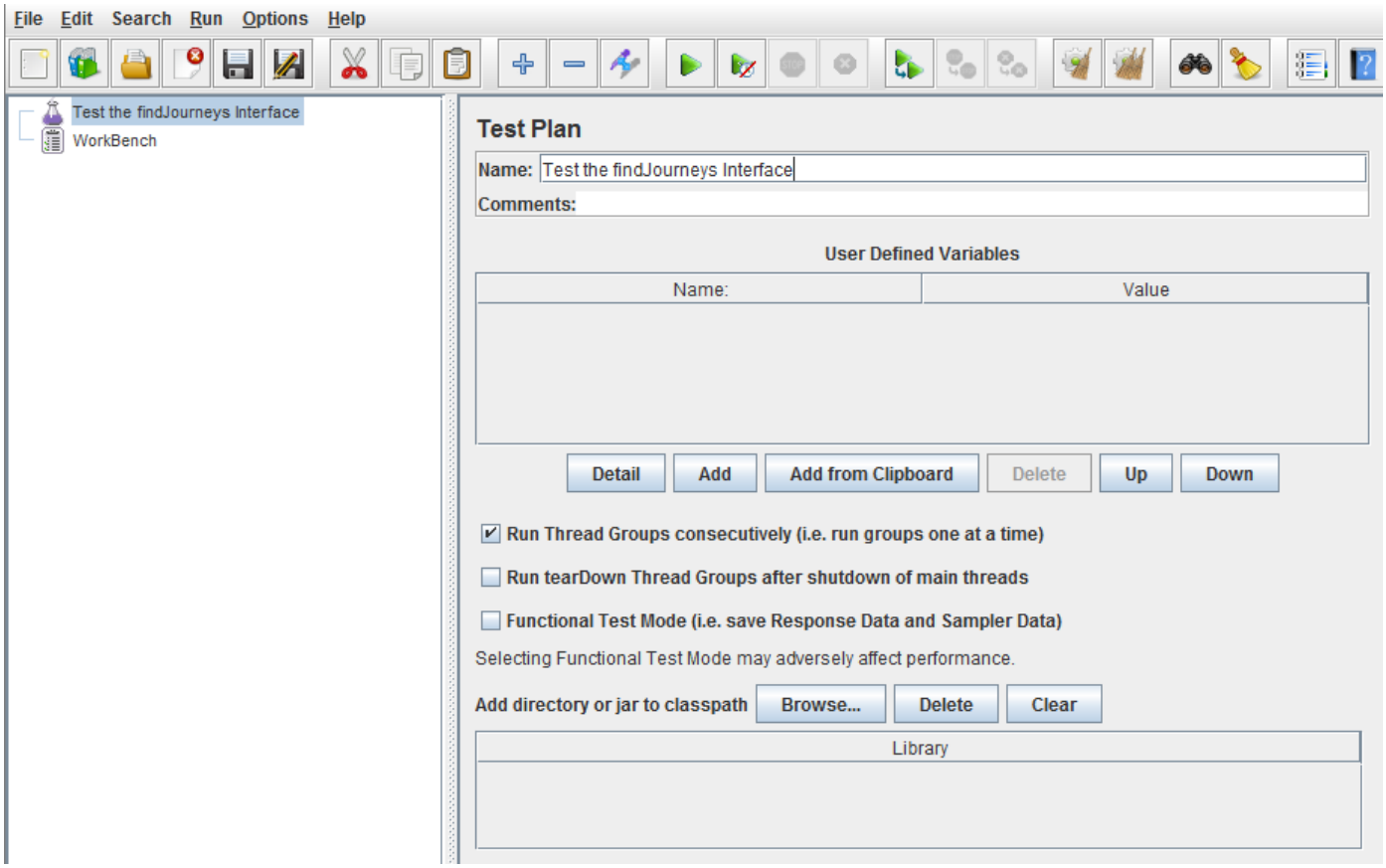
The walkthrough accesses an easyTravel URI (`/services/JourneyService/findJourneys`) where you can find journeys filtered by start date, end date, and location, and get a list of journeys in XML. If you have easyTravel running on your machine you can try to access http://localhost:8080/services/JourneyService/findJourneys?destination=New%20York&fromDate=1406282400000&toDate=1437818400000 for a list of journeys to New York. If this call fails, make sure you have easyTravel started and running on port 8080.

Steps

This example uses the system profile supplied with easyTravel. If you prefer to use your own system profile, make sure you have the Servlet sensor enabled as shown in the following:



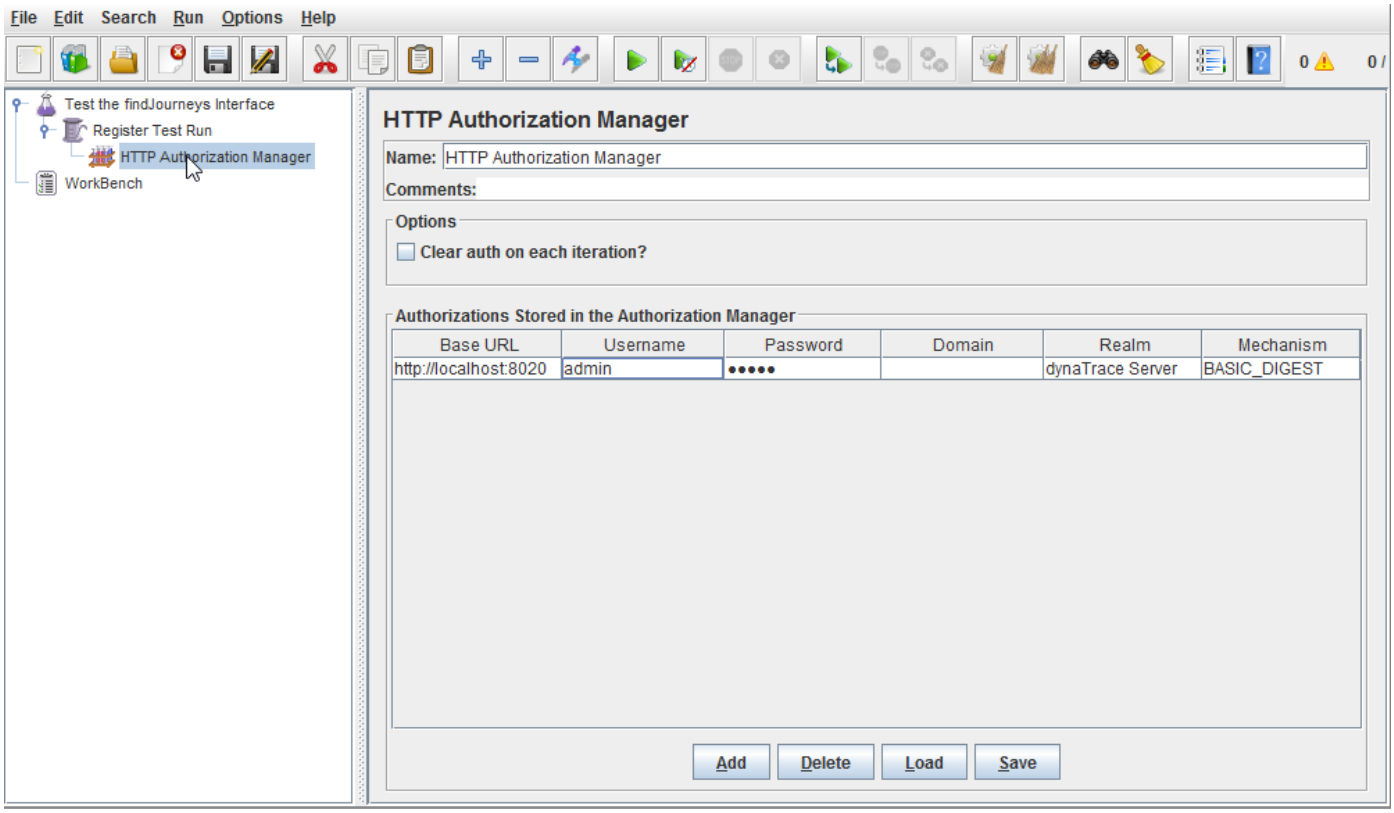
1. Create the JMeter test plan, and name it **Test the findJourneys Interface**.



2. Make sure the **Run Thread Groups consecutively** check box is selected in Test Plan properties. This ensures test run registration always executes before running the test.

3. Register a new test run of the Web API category, This is done by a REST call to the AppMon server. The REST call executes a POST request which registers a test run.

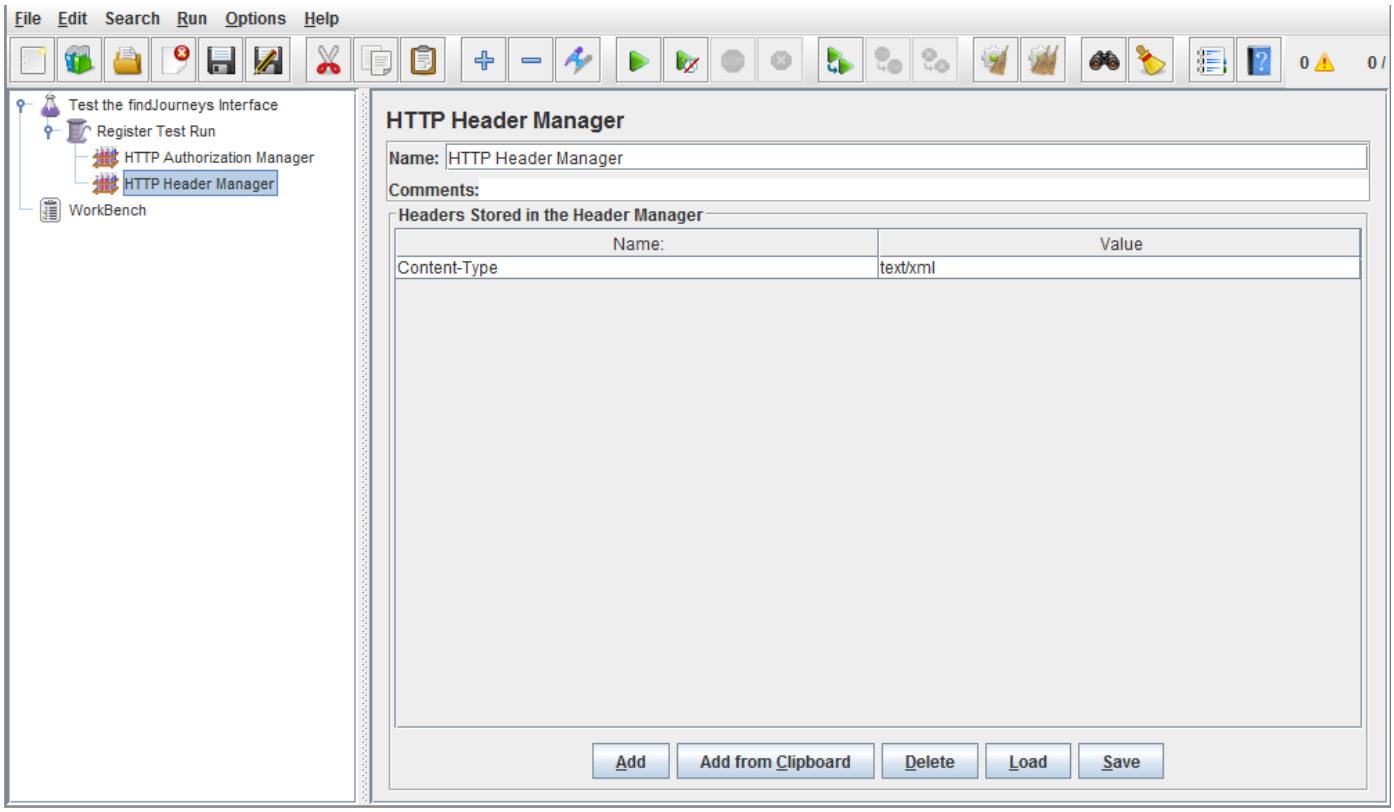
   In JMeter, create a new ThreadGroup and name it **Register Test Run**. Add an instance of `HTTP Authorization Manager`. This is done because server REST interface access is protected with basic authentication:



Tip

You can pass additional metadata during registering. See REST Interfaces for Test Automation for more information.

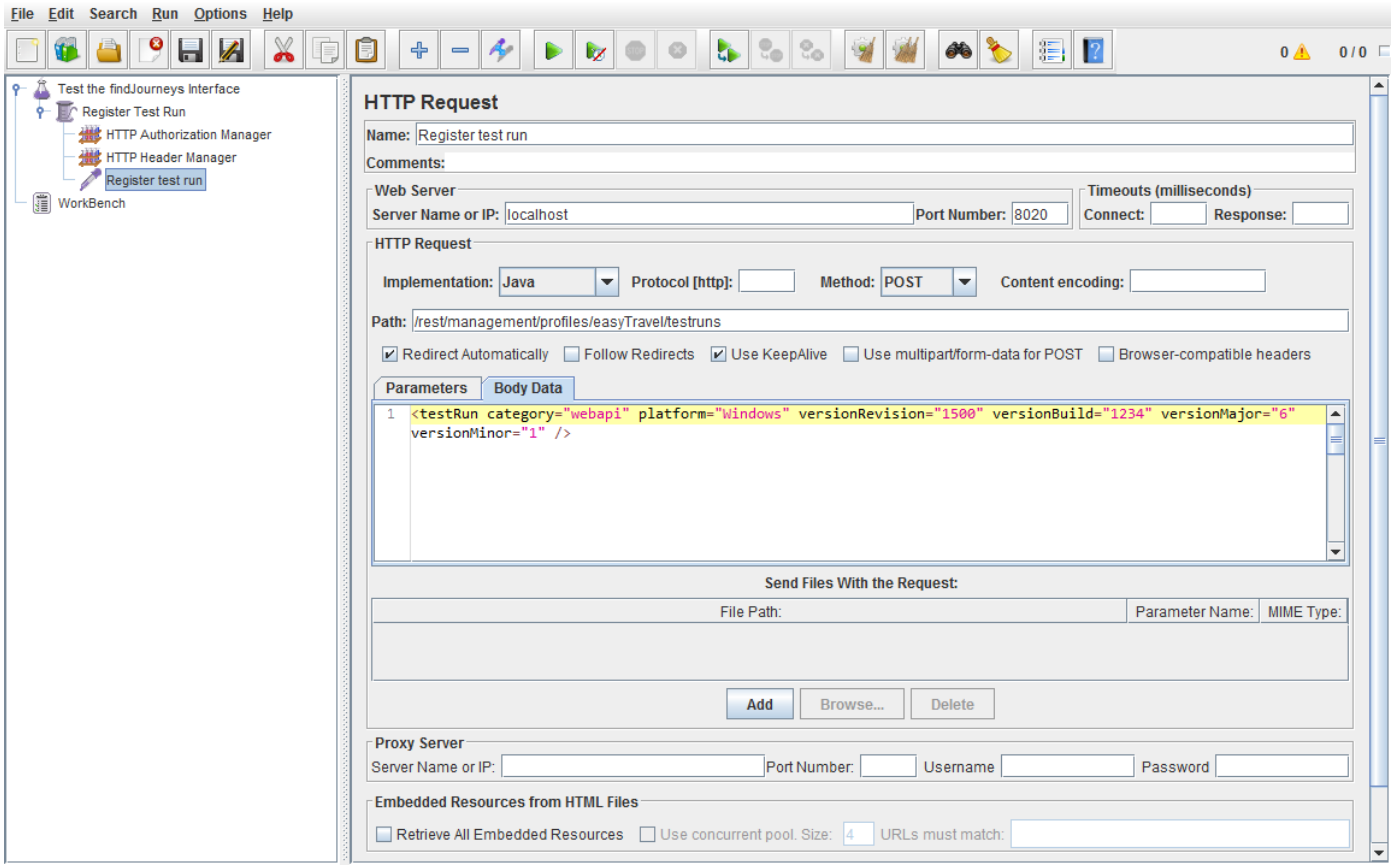
4. Add a HTTP Header Manager to set the default Content-Type to text/xml:



5. Add an HTTP Request Sampler that POSTs to the testrun URI to register a new test run. To do this, right click on the thread group and select **Add > Sampler > HTTP Request**. You can specify the body of this request either as XML or JSO.

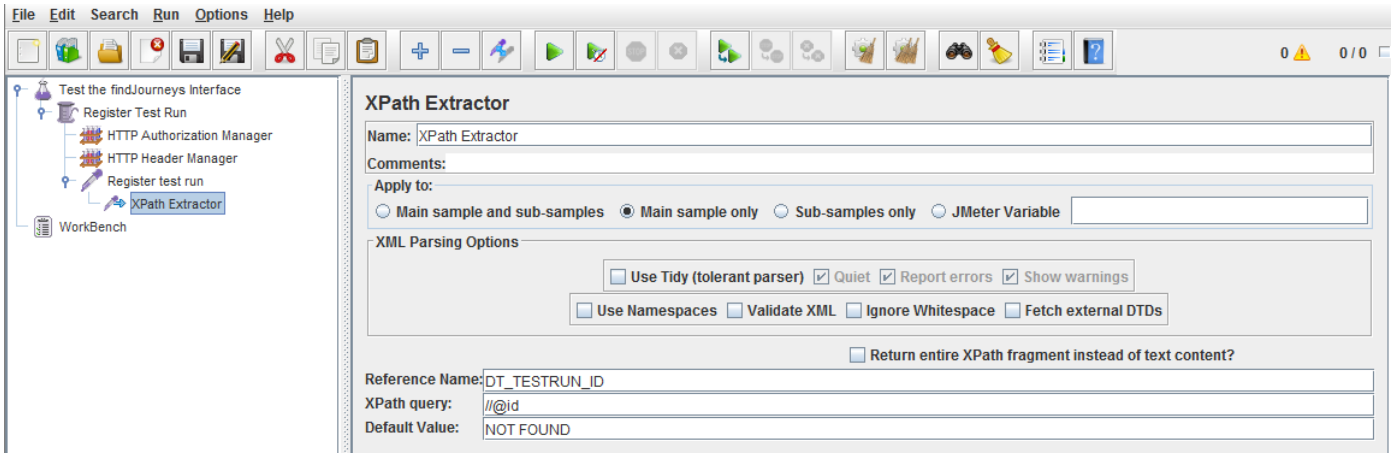
   This example uses XML to register a test run for Web API tests, running on platform Windows and for version 5.1.1500.1234:

```
<testRun category="webapi" platform="Windows" versionMajor="5" versionMinor="1" versio
```

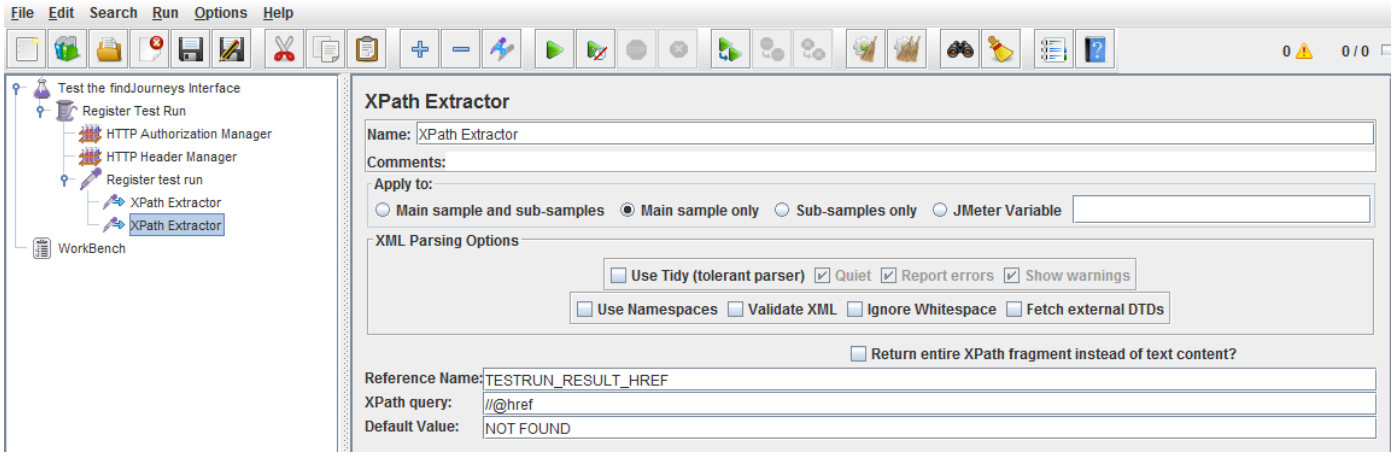


The XML response returned for this call contains the test run ID. Use this as the `id` attribute value. This tags all requests belonging to this test run. It also contains an attribute called `href` that contains a URI that fetches the results for this test run.

6. Extract the test run ID to tag all the HTTP requests to the findJourneys service. Use an XPath Extractor to get the id attribute and save it in the `DT_TESTRUN_ID` variable.



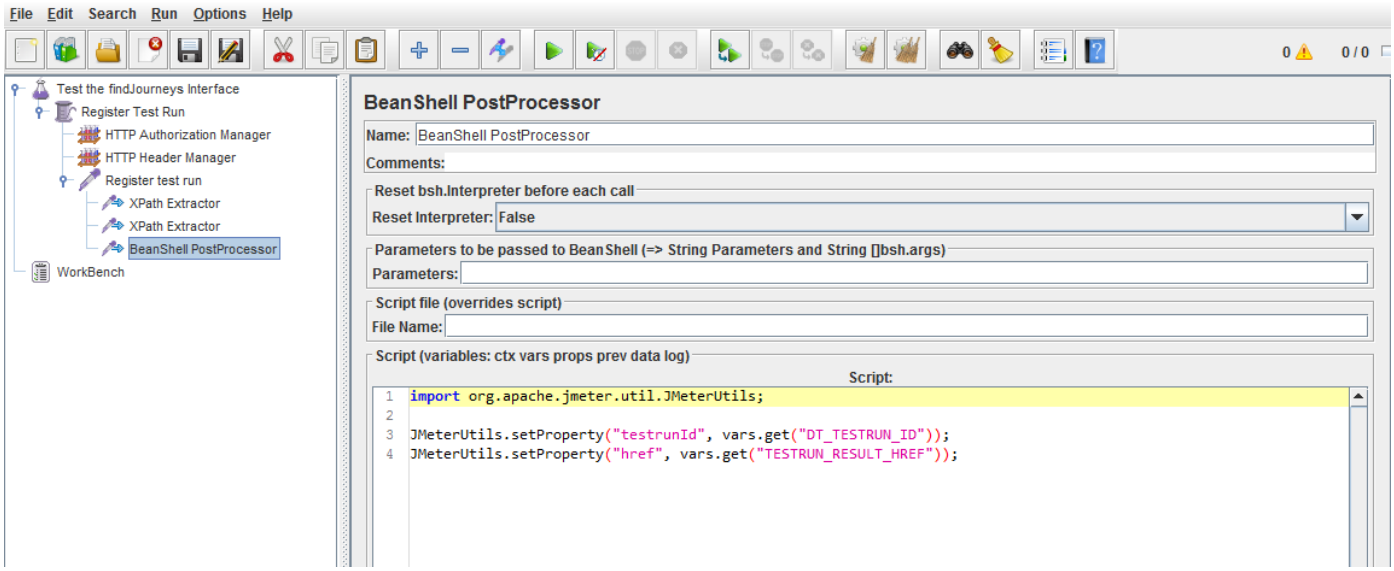
To make fetching the results easier, add a second XPath extract for the `href` attribute:



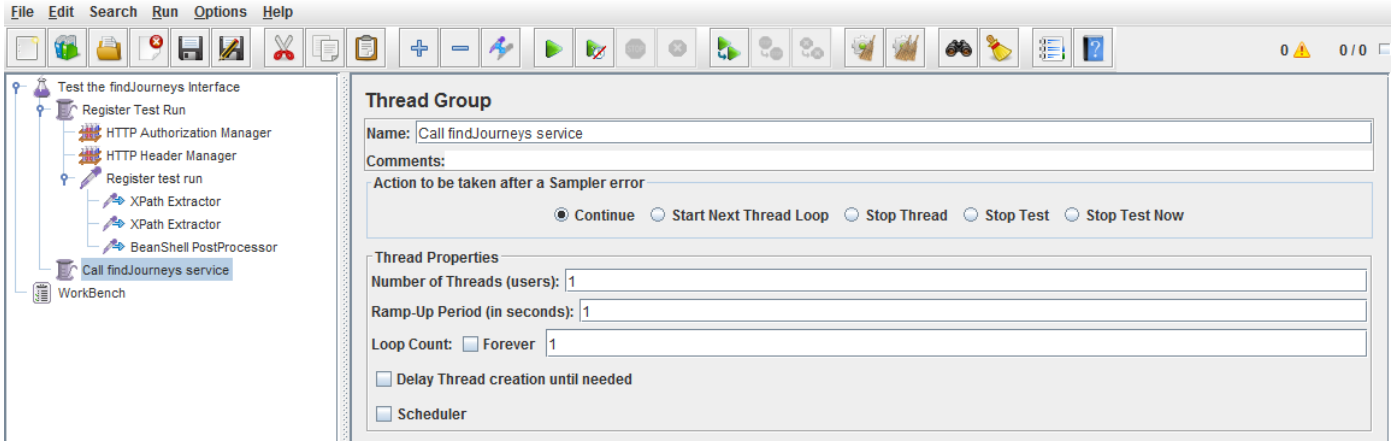
> **Tip**
>
> These variables are only available within this sample.

These variables are used in subsequent steps of the test plan. Use JMeter properties to pass them along. To do this, add an instance of the BeanShell PostProcessor with the following code. The code uses the variables `DT_TESTRUN_ID` and `TESTRUN_RESULT_HREF` as properties in other steps of the test plan.

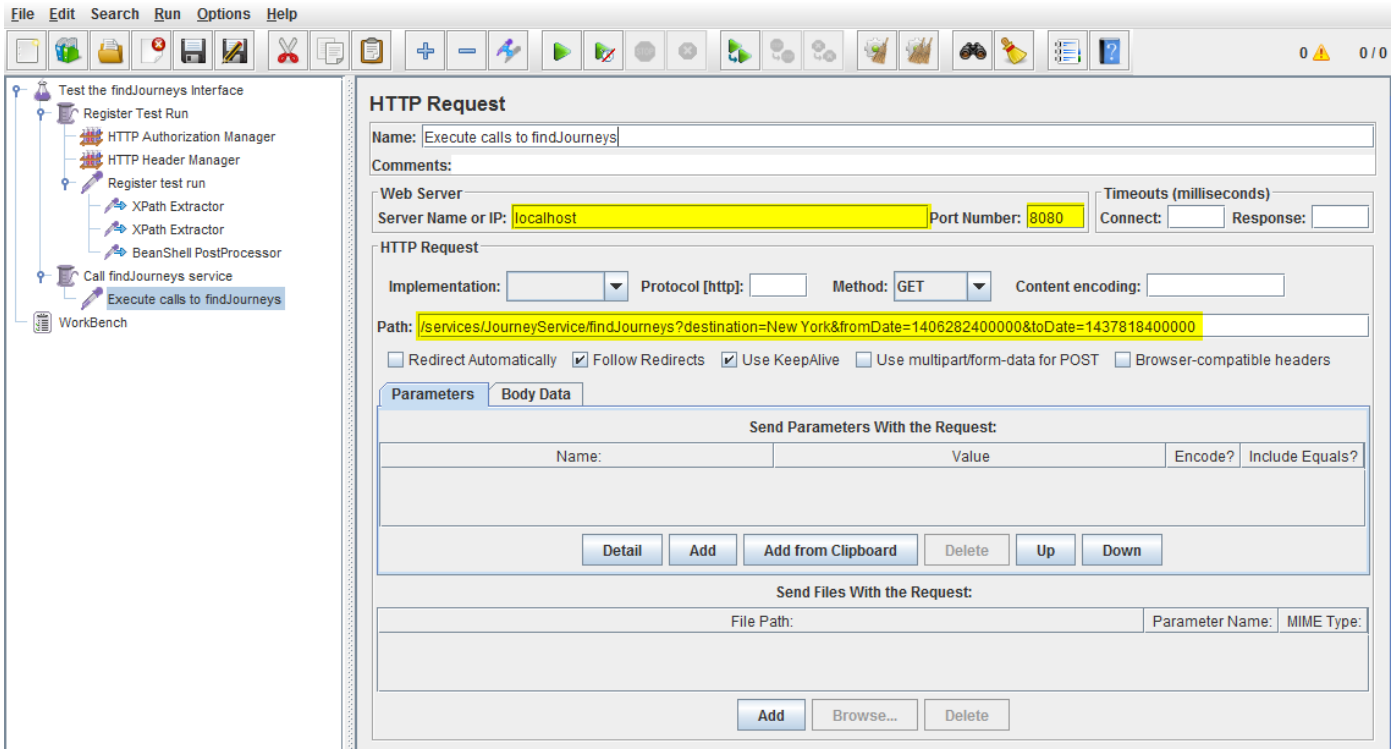
```
import org.apache.jmeter.util.JMeterUtils;
JMeterUtils.setProperty("testrunId", vars.get("DT_TESTRUN_ID"));
JMeterUtils.setProperty("href", vars.get("TESTRUN_RESULT_HREF"));
```



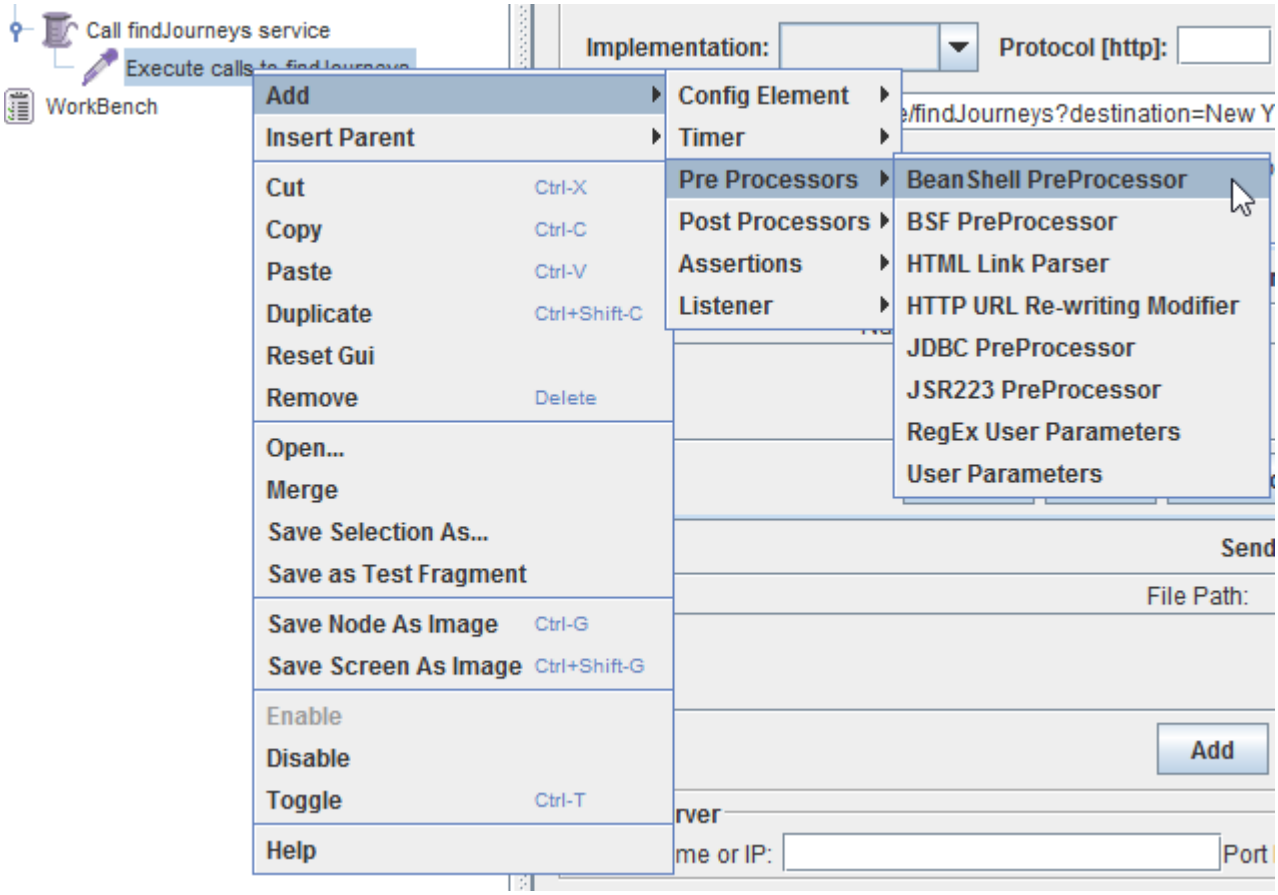
7. Create the actual test request to the web service. First, add a new ThreadGroup to the test plan and call it **Call findJourneys Service**.



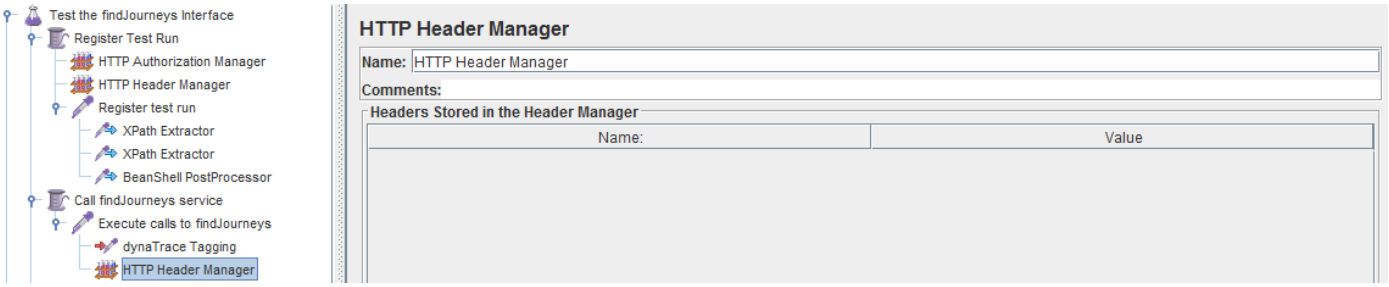
Add an HTTP Request sampler and name it **Execute request to findJourneys**. Configure it as shown in the following, which assumes the easyTravel customer frontend is running at `localhost:8080`. Adapt the host/port as needed.


File Edit Search Run Options Help
0 0 / 0
Test the findJourneys Interface
Register Test Run
HTTP Authorization Manager
HTTP Header Manager
Register test run
XPath Extractor
XPath Extractor
BeanShell PostProcessor
Call findJourneys service
Execute calls to findJourneys
WorkBench
HTTP Request
Name: Execute calls to findJourneys
Comments:
Web Server
Server Name or IP: localhost
Port Number: 8080
Timeouts (milliseconds)
Connect:
Response:
HTTP Request
Implementation:
Protocol [http]:
Method: GET
Content encoding:
Path: /services/JourneyService/findJourneys?destination=New York&fromDate=1406282400000&toDate=1437818400000
Redirect Automatically
Follow Redirects
Use KeepAlive
Use multipart/form-data for POST
Browser-compatible headers
Parameters
Body Data
Send Parameters With the Request:
Name:
Value
Encode?
Include Equals?
Detail
Add
Add from Clipboard
Delete
Up
Down
Send Files With the Request:
File Path:
Parameter Name:
MIME Type:
Add
Browse...
Delete

8. To tag the request with the test run ID, add a BeanShell PreProcessor to the HTTP Request and call it **AppMonTagging**. This Preprocessor reads the set properties and sets the test run ID as part of the `X-dynatrace` header tag:



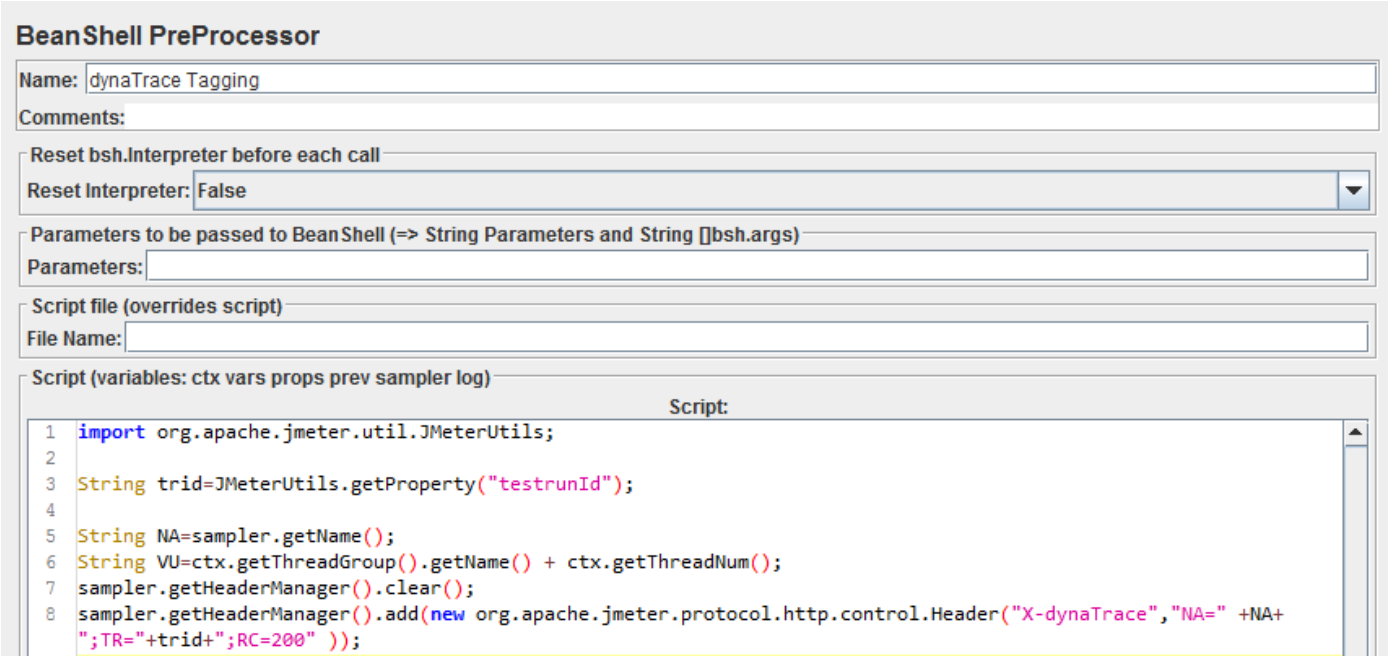
An HTTP Header Manager is needed for this step. Just having the header manager is enough, so no further configuration is needed:



Use the following script to set the right header tag.

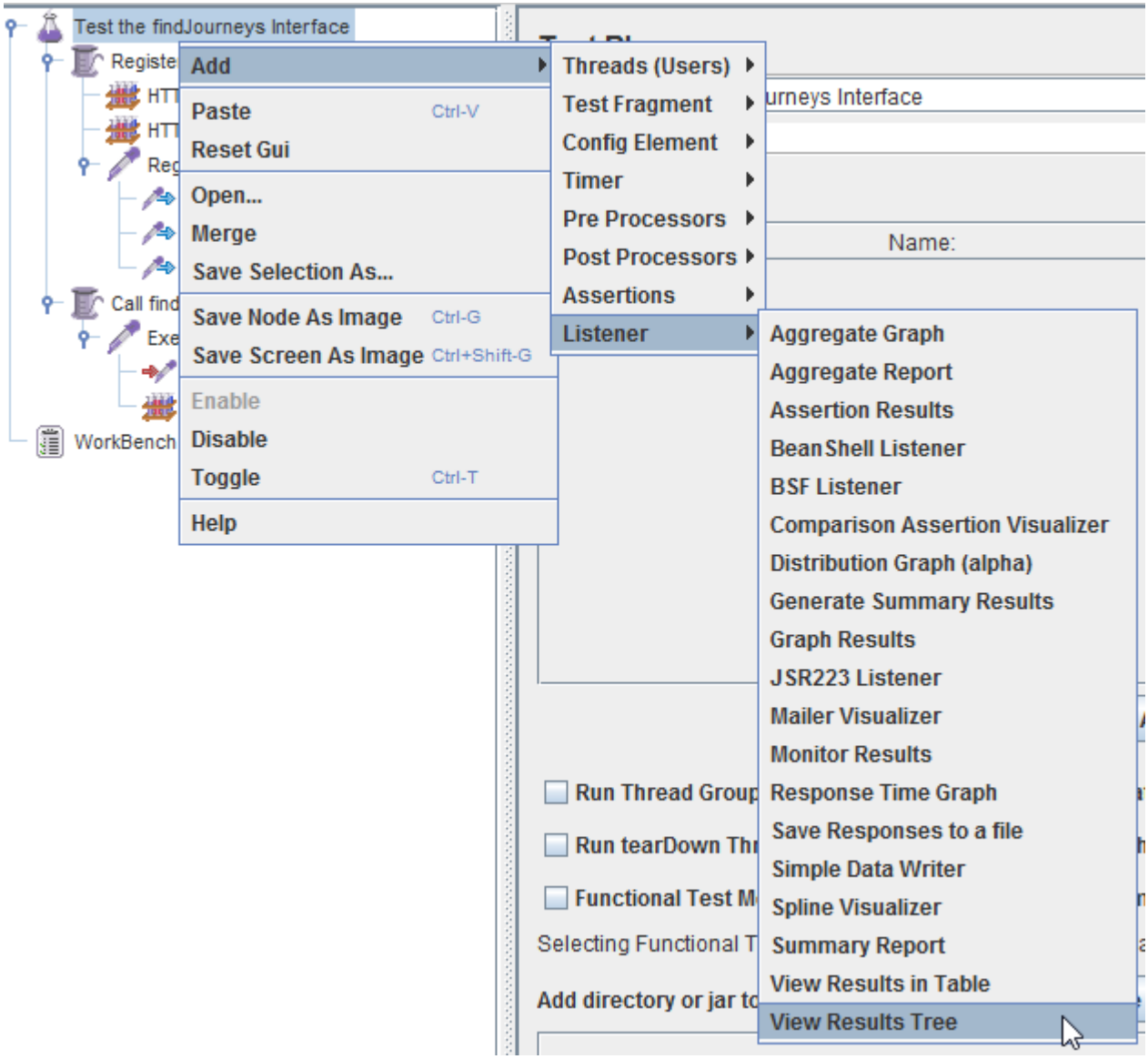
```
import org.apache.jmeter.util.JMeterUtils;
String trid=JMeterUtils.getProperty("testrunId");
String NA=sampler.getName();
String VU=ctx.getThreadGroup().getName() + ctx.getThreadNum();
sampler.getHeaderManager().clear();
sampler.getHeaderManager().add(new org.apache.jmeter.protocol.http.control.Header("X-c
```

The final preprocessor configuration should look as follows.



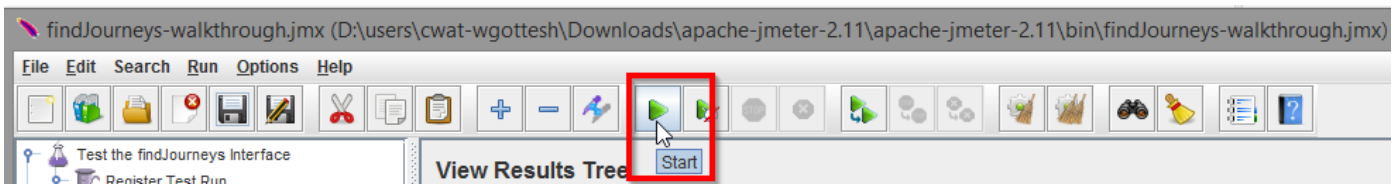
This executes the web service call using the right `X-dynatrace` tag to have this request show up as a Web API test in the Test Automation Dashlet. The request uses NA field sampler name, which is the name of the API shown in the dashlet. the TR field contains the test run ID obtaining by registering the test run the RC field contains the expected response code.

9. You can now looks at the results on JMeterTo do this, add a Results Tree to the Test Plan.

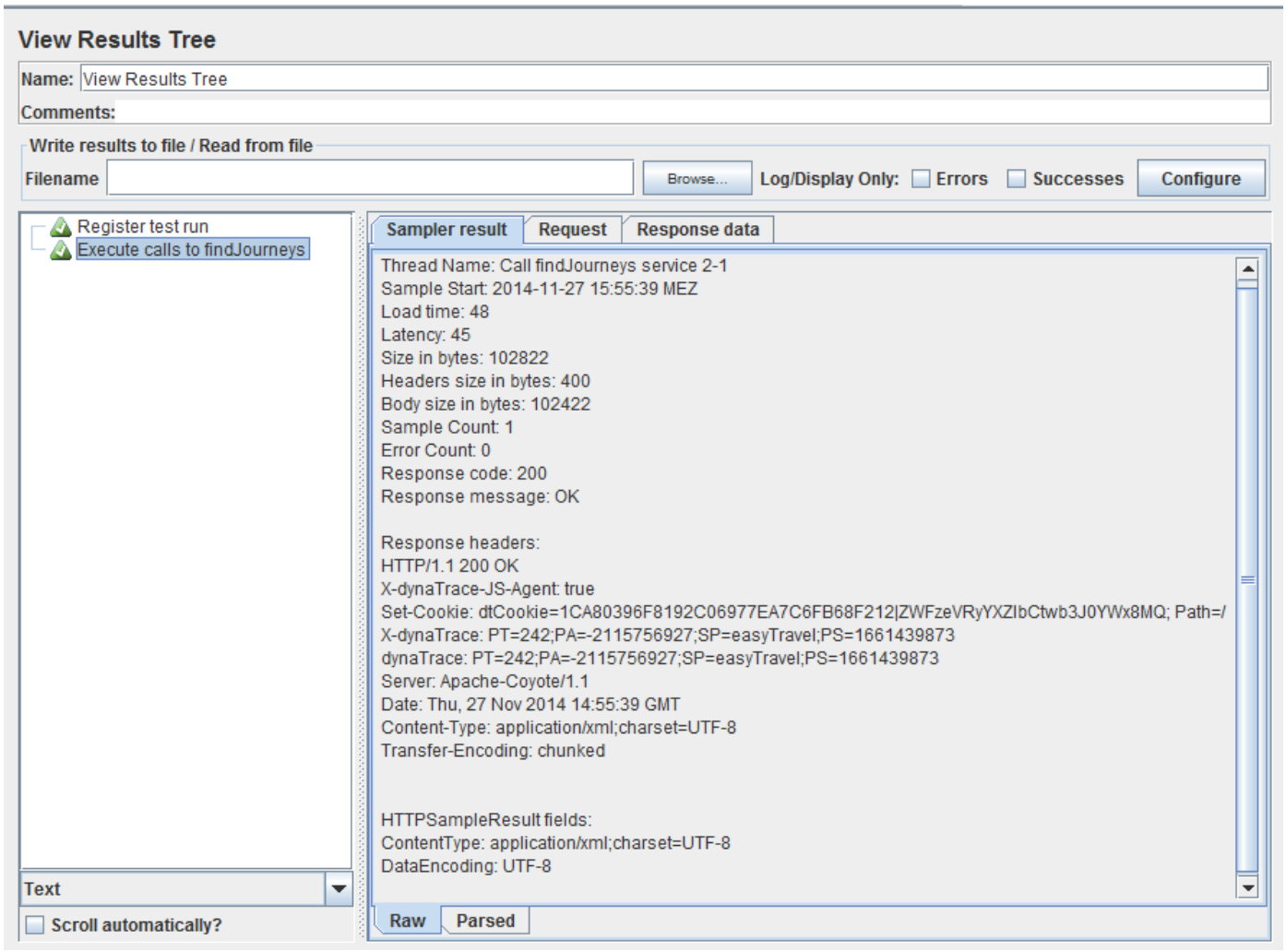


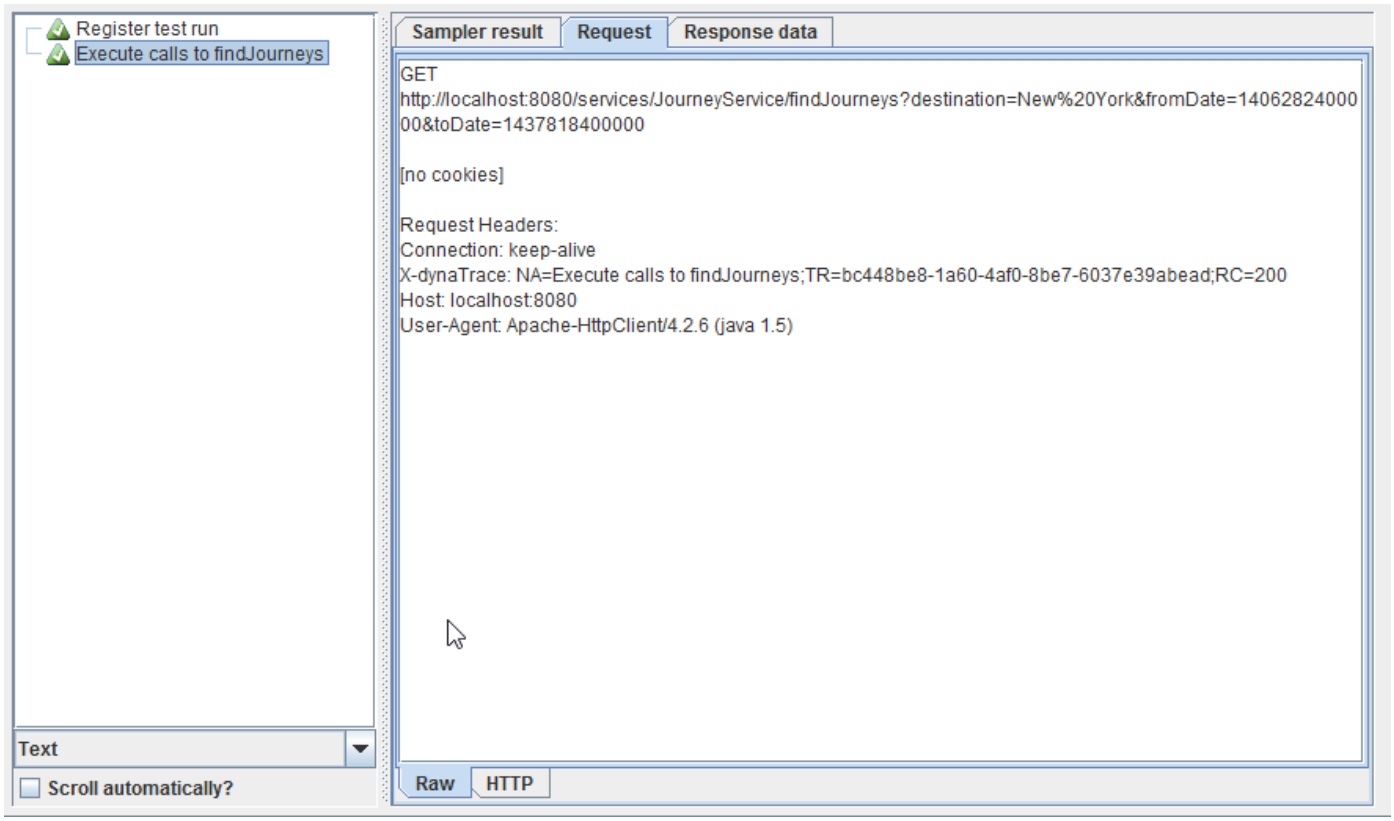
0. You can now run the test plan. Use the **Play** button in the toolbar to run the test plan.

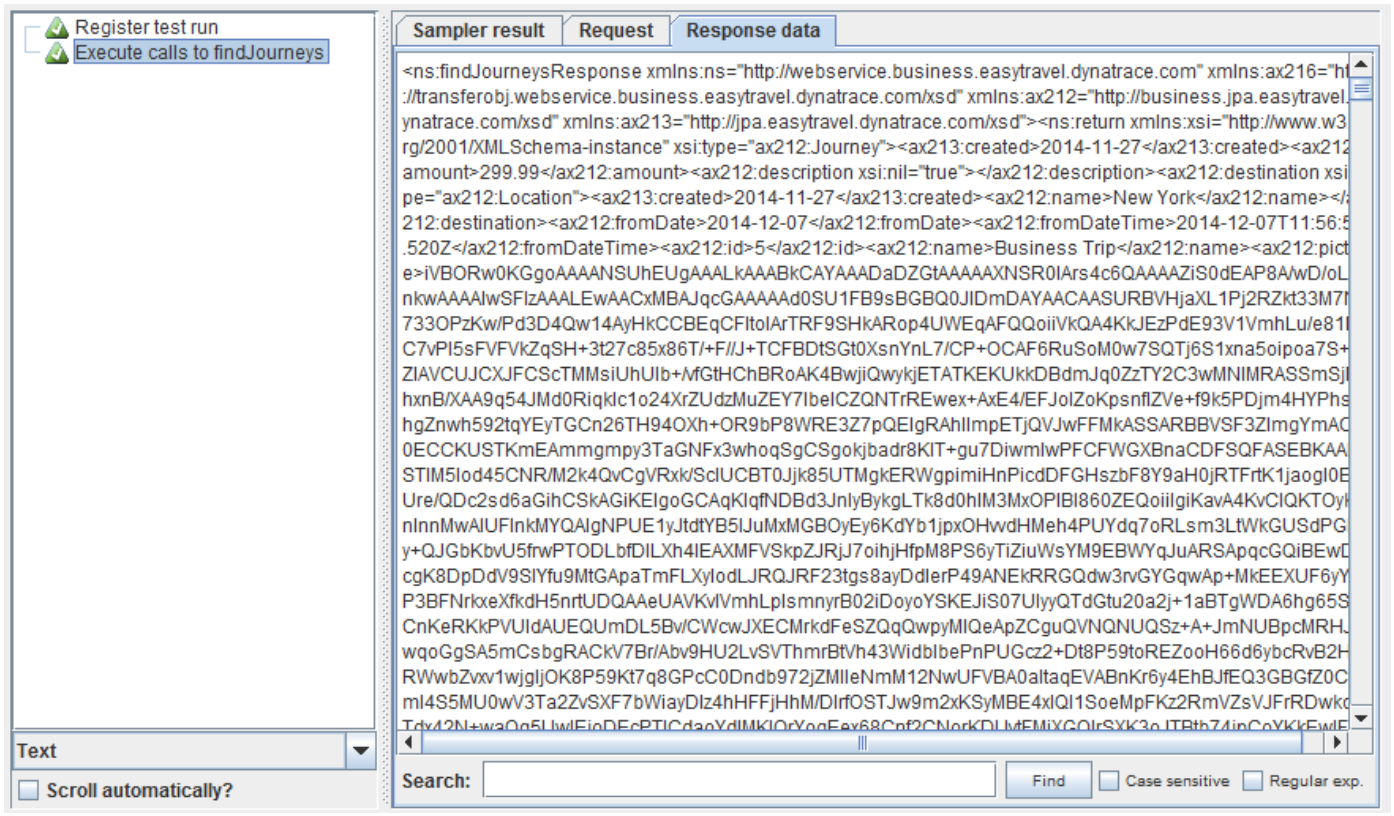
> **Tip**
>
> Remember that easyTravel must be running.



The View Results tree shows individual test steps, the sampler, and the request and response data, as shown in the following.

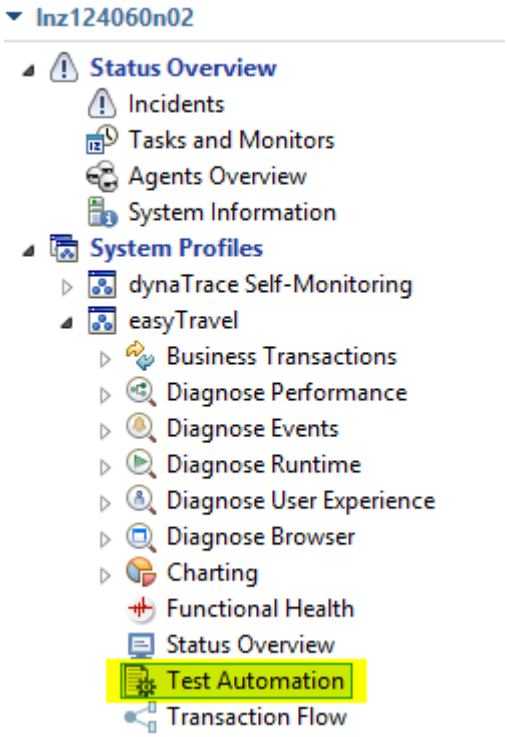




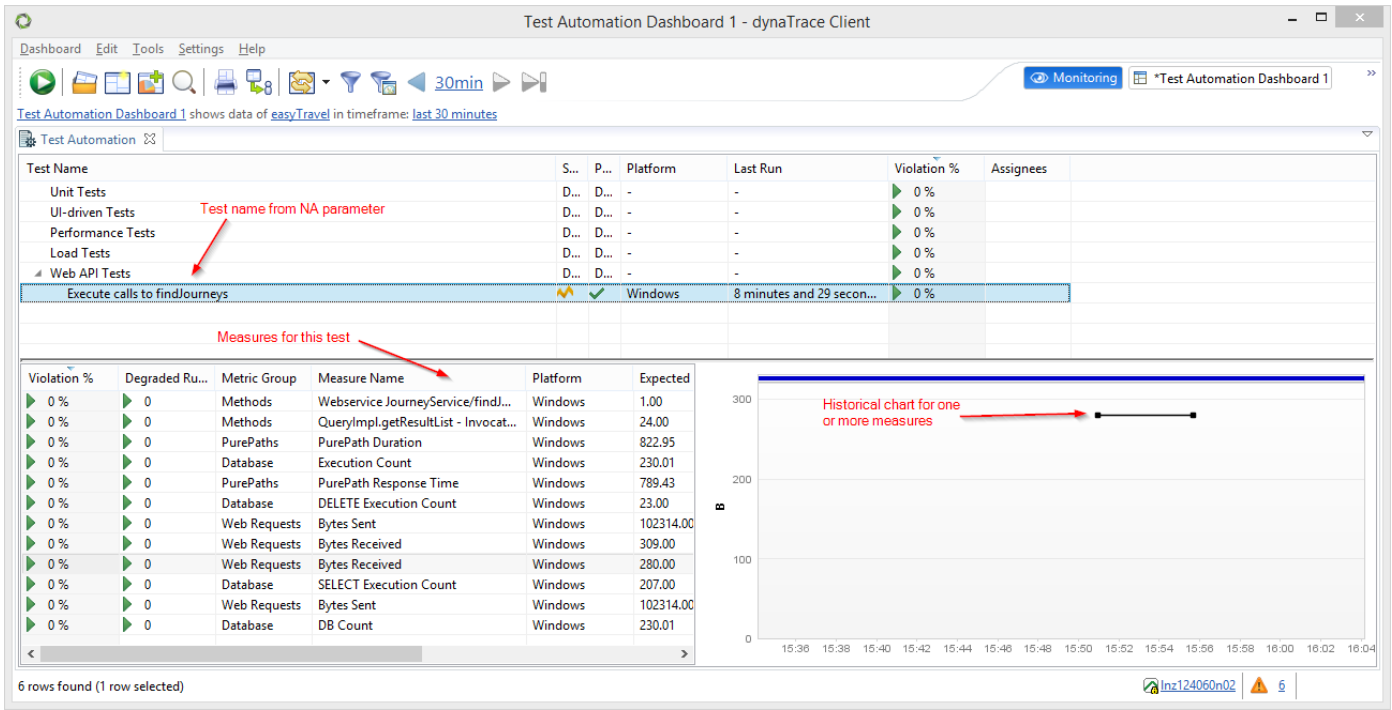


Run the test a few more times to have more data in the Test Automation Dashlet.

1. Once the request executes, you can see request results in the Test Automation Dashlet. In the AppMon Client, open the Test Automation Dashlet from the Cockpit.



2. With the dashlet open, adjust the time filter to the last 30 minutes, to make it easier to find the requests you just executed. This screen shot shows the request made, a list of metrics as well as a historic chart of metric values:



Download

Sample script

Integrate web API performance monitoring in SoapUI

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Requirements

Goal

Create a SoapUI project that tests the functionality of a web service

Monitor the functional test with AppMon

Download

Related pages

Integrate web API performance monitoring in JMeter

Overview

Server-side applications provide web-based APIs that allow other clients (like mobile or AppMon Clients) to use their functionality. Functional testing of those APIs is typically performed using tools like JMeter or SoapUI. With the Web API Test Category, the performance of those API can also be monitored continuously.

Requirements

- A currently supported AppMon version with a license that allows using the Test Automation Dashlet (Test Center Edition) - Get the free trial if you're not yet a customer.
- easyTravel

- SoapUI. This walkthrough uses SoapUI 5.0.0.

Goal

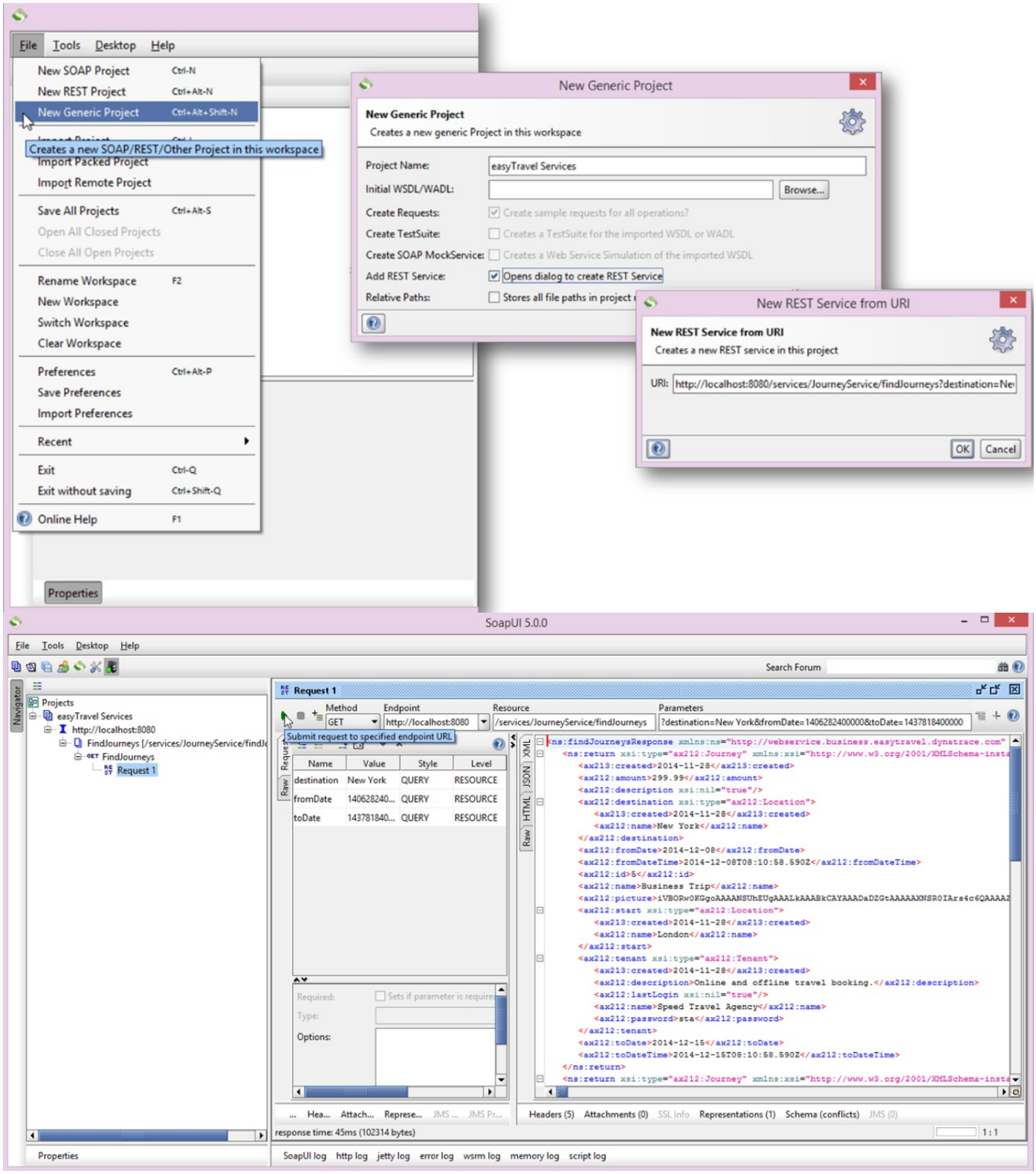
This walkthrough shows how to:

- Create a SoapUI project that tests a web service.
- Register the functional test with AppMon to retrieve performance metrics.

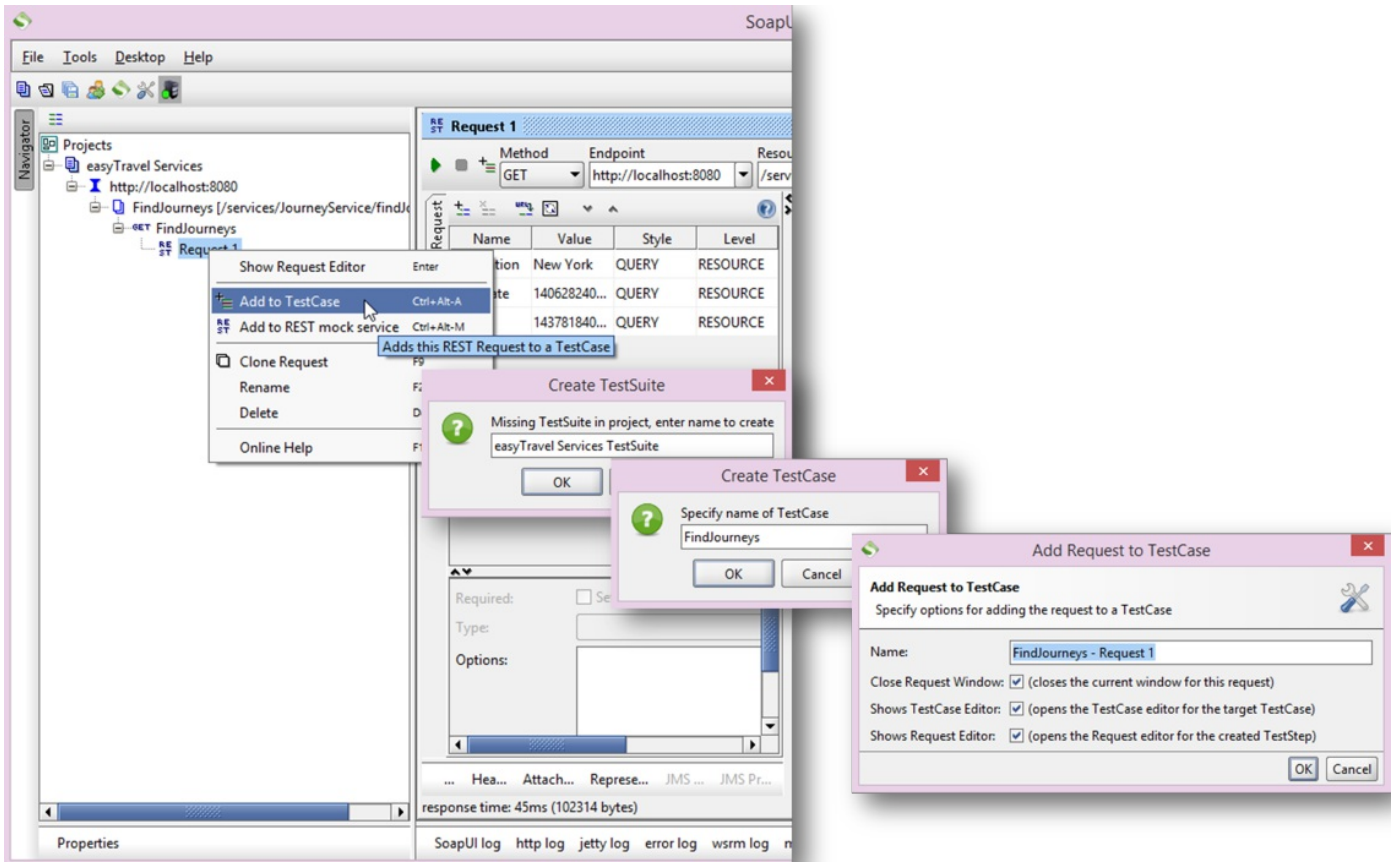
The walkthrough accesses an easyTravel URI (`/services/JourneyService/findJourneys`) where you can find journeys filtered by start date, end date, and location, and get a list of journeys in XML. If you have easyTravel running on your machine you can try to access http://localhost:8080/services/JourneyService/findJourneys?destination=New%20York&fromDate=1406282400000&toDate=1437818400000 for a list of journeys to New York. If this call fails, make sure you have easyTravel started and running on port 8080.

Create a SoapUI project that tests the functionality of a web service

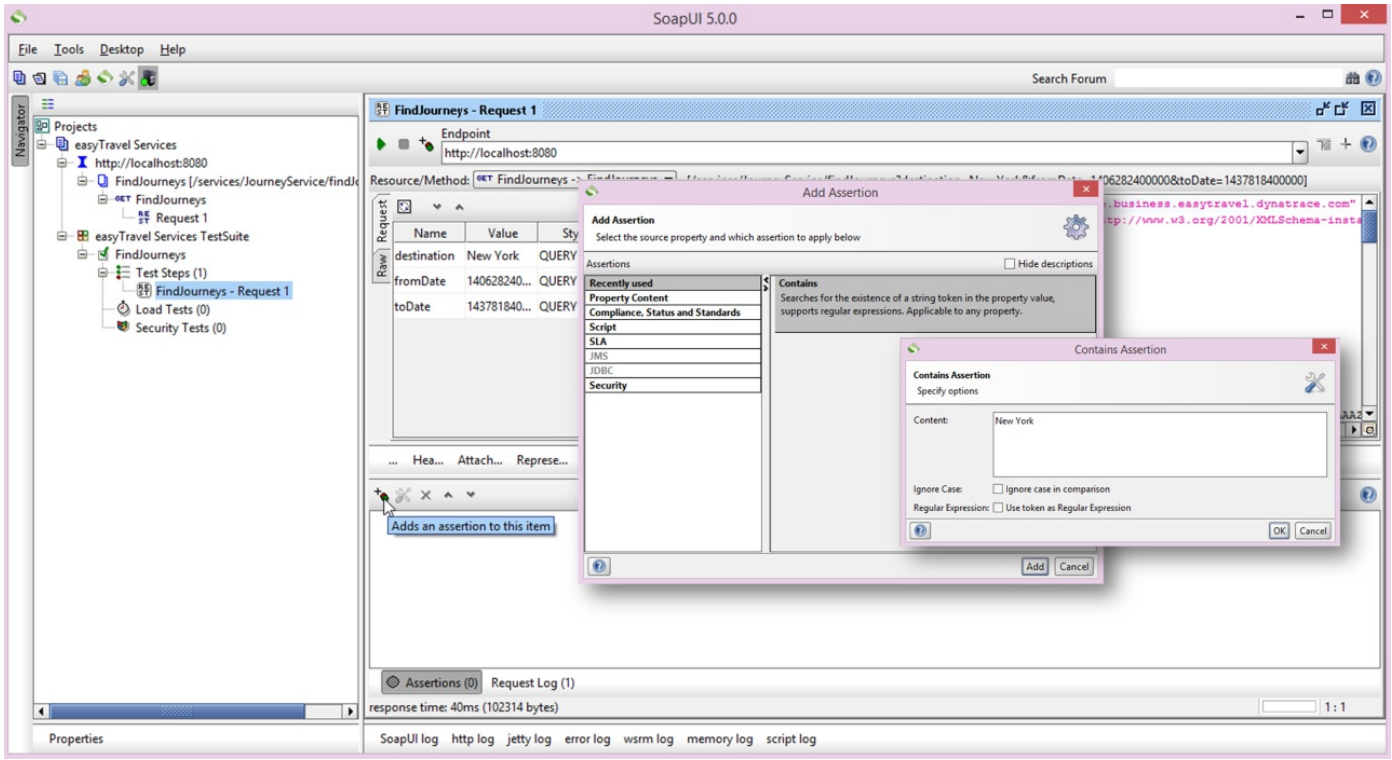
1. Create a SoapUI project for the easyTravel `findJourneys` Service. Once this is done, click **Submit request to specified endpoint URL** to verify that the request and the response are correct.



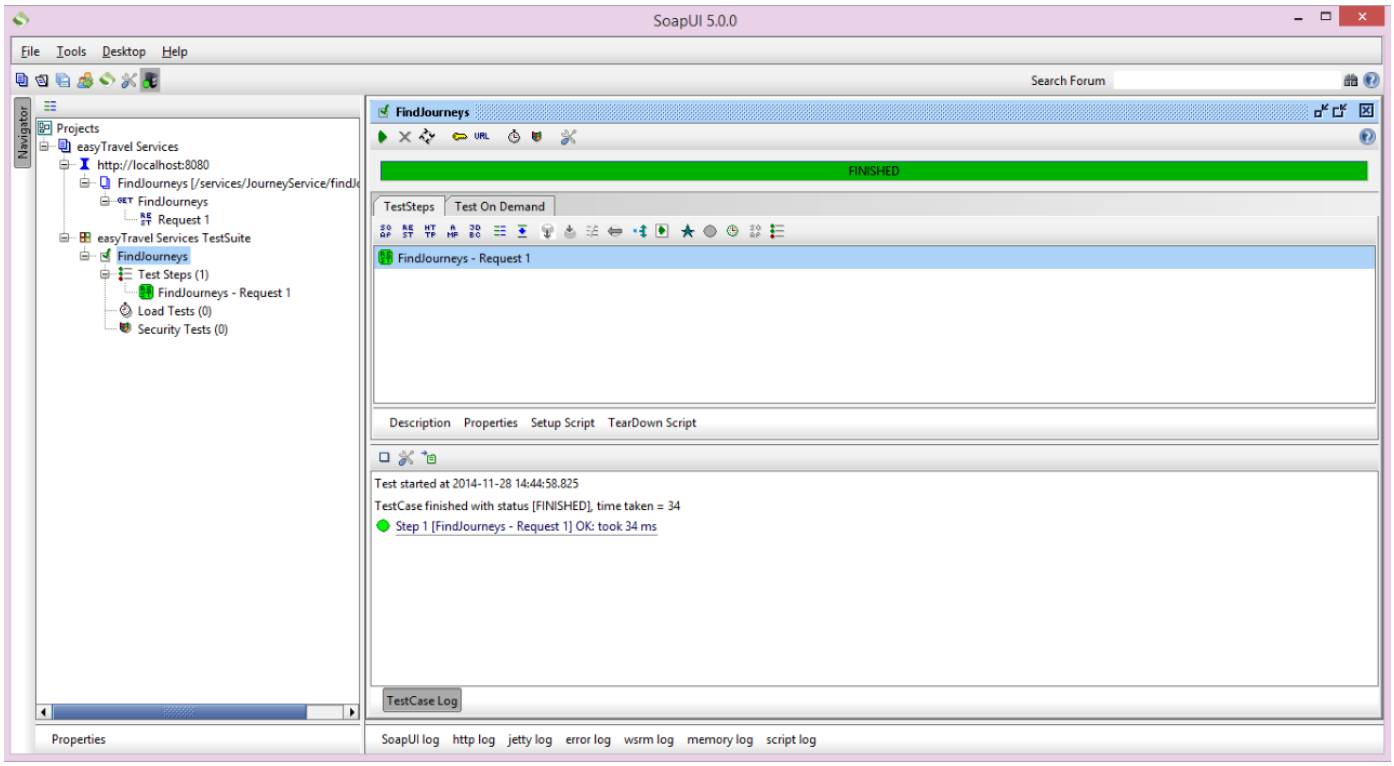
2. Add a request to a new Test suite and a Test case, as shown in the following:



3. Add an assertion for this test case, such as one where the response contains the string “New York”.

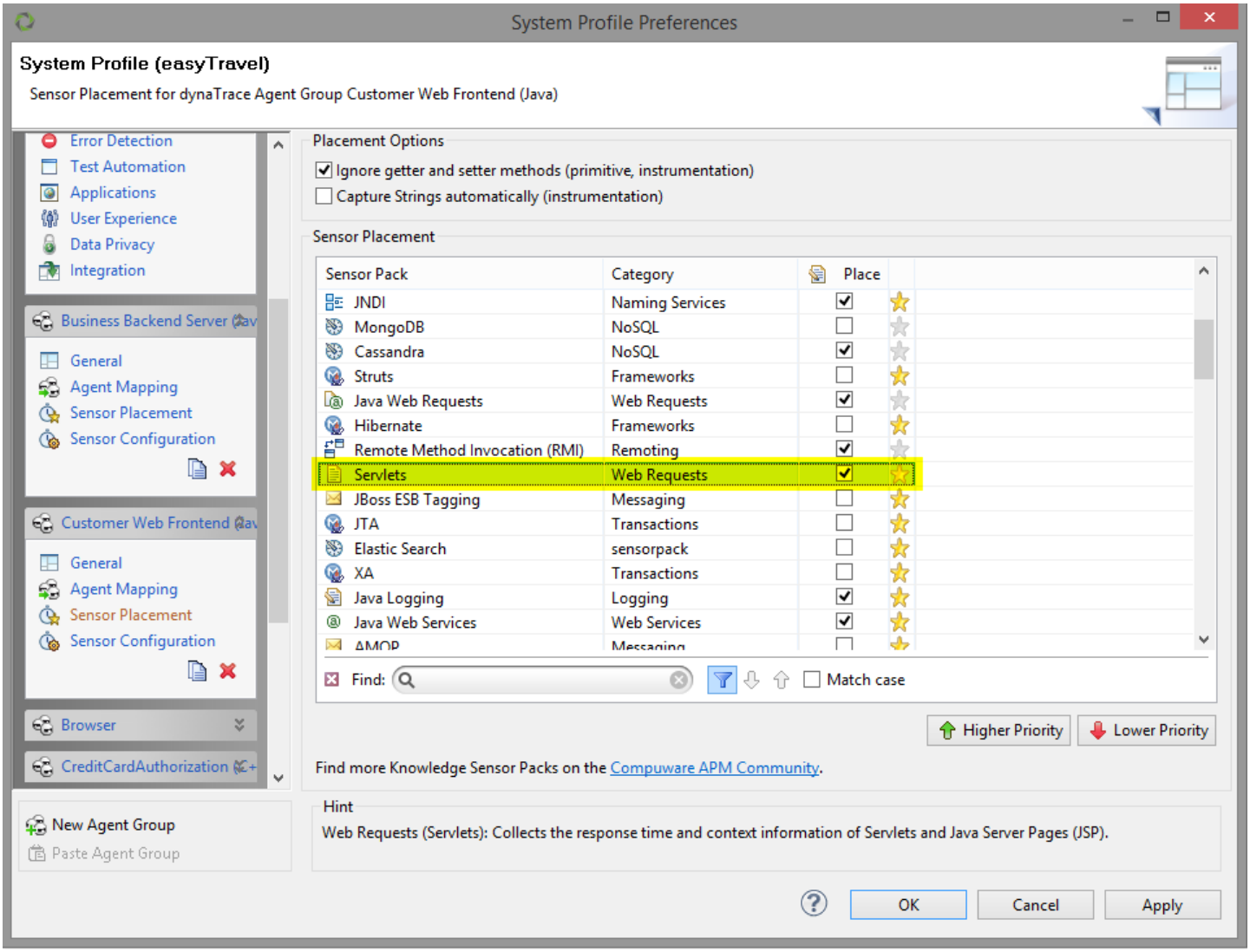


4. Run the test and verify the assertion works.

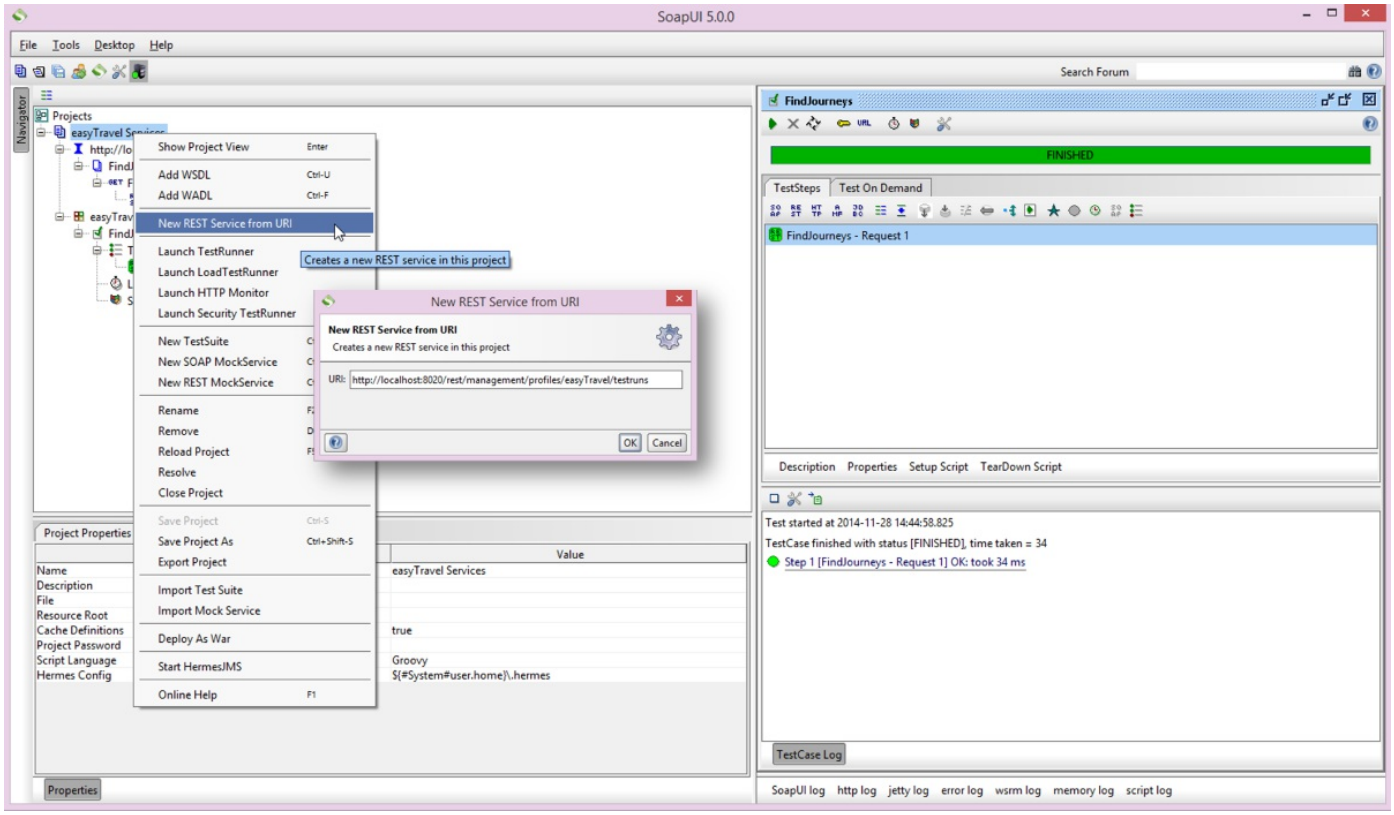


Monitor the functional test with AppMon

This example uses the system profile supplied with easyTravel. If you prefer to use your own system profile, make sure you have the Servlet sensor enabled:



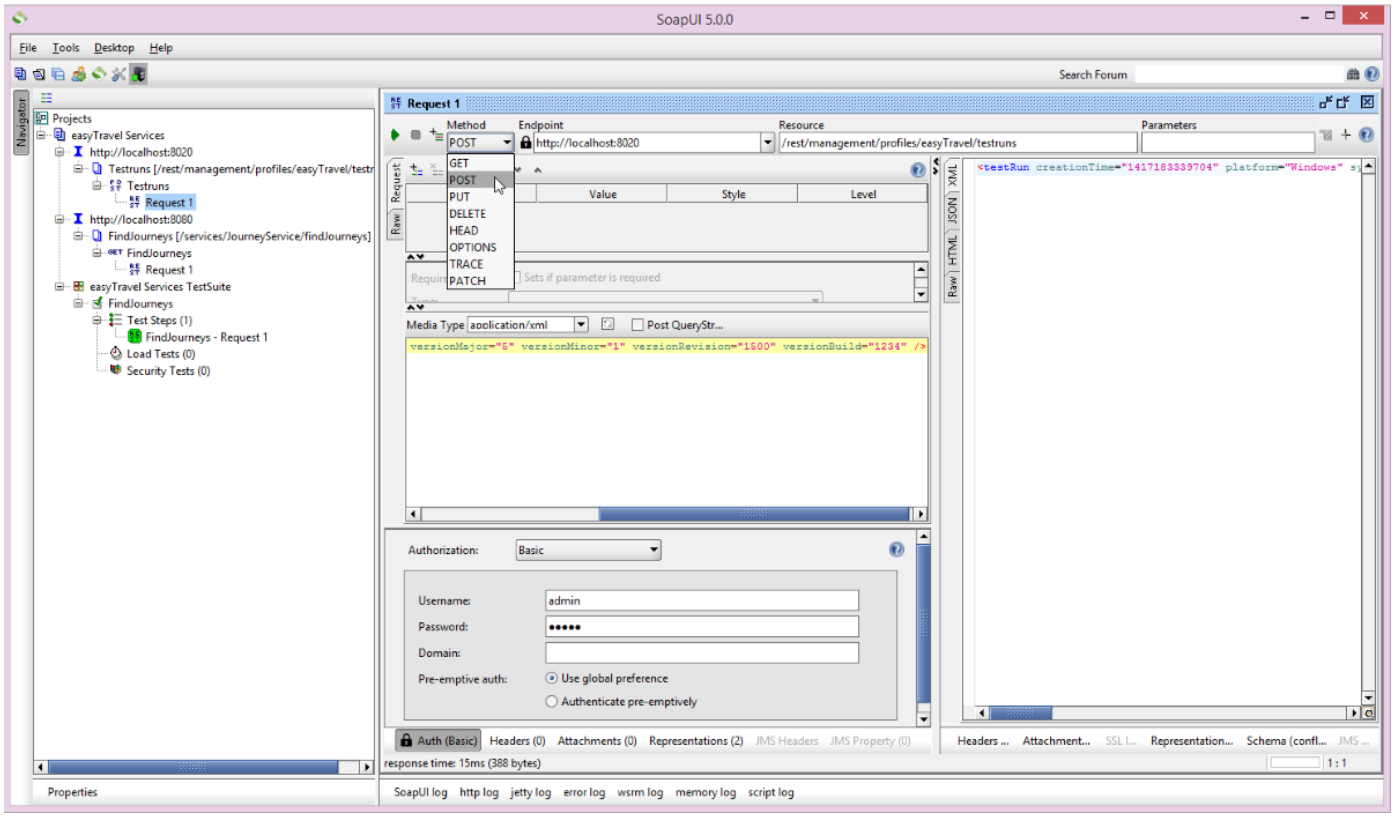
1. Add a service end point for the AppMon REST service testrun http://<server>:8020/rest/management/profiles/<profilename>/testruns into the project. When registering the test run, you can pass in additional metadata such as build number. See REST Interfaces for Test Automation for more information.



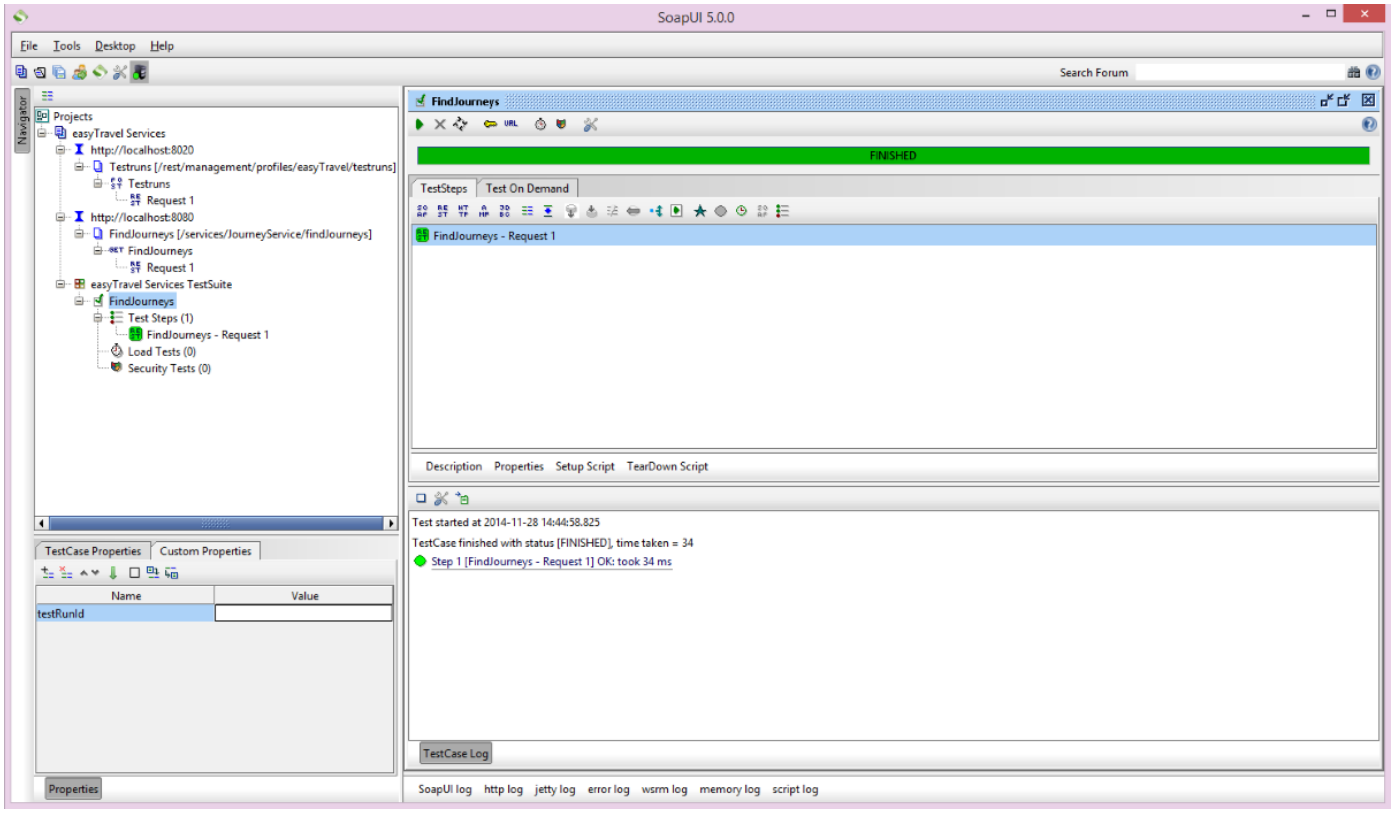
2. Change the method used to POST to register a new test run. You can specify the body of this request either as XML or JSON, This example shows XML used to register a test run for Web API tests, running on platform Linux 64 bits and for version 5.1.1500.1234:

```
<testRun category="webapi" platform="Windows" versionMajor="5" versionMinor="1" versic
```

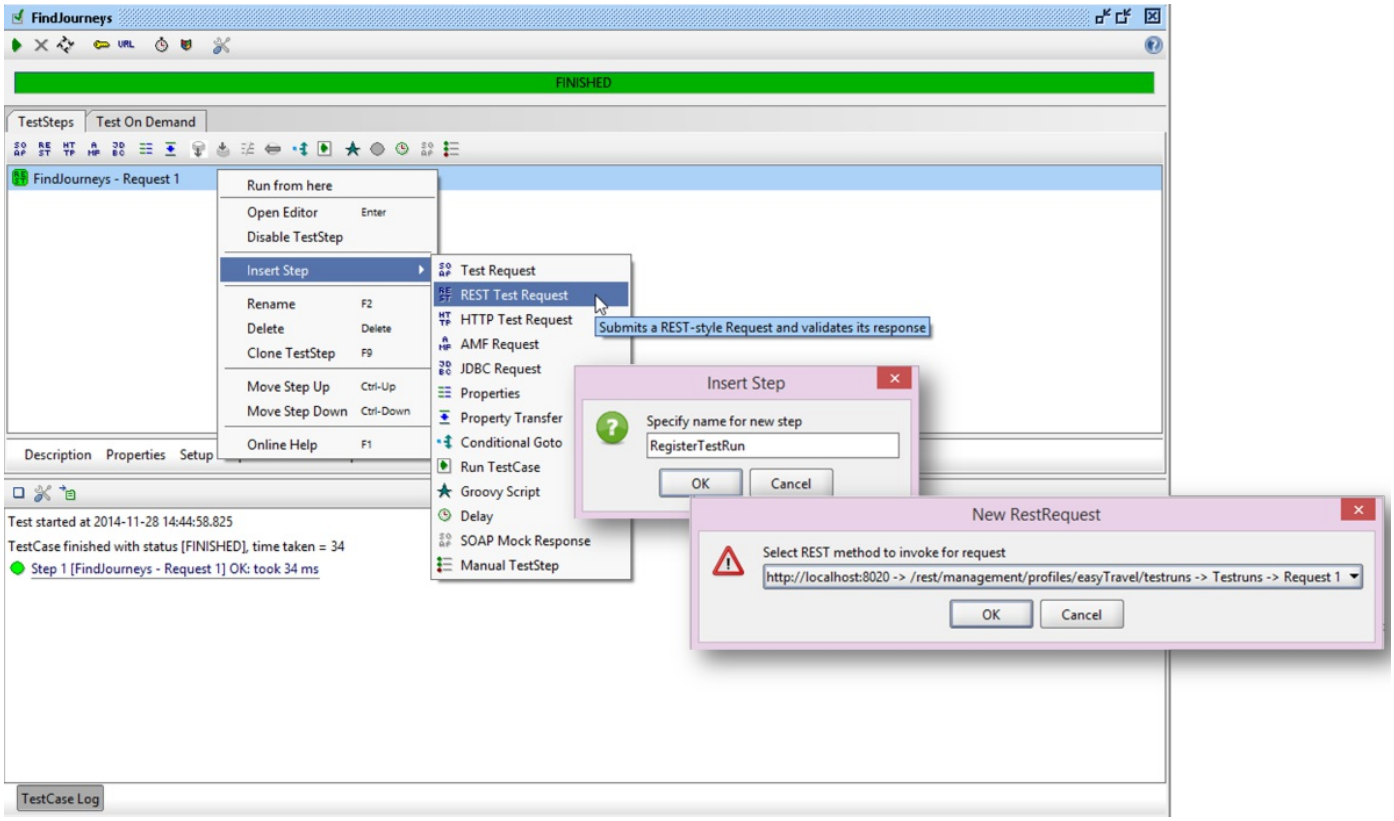
Do not forget to set up the authentication details (AppMon Server user - per default admin/admin). Click **Submit request to specified endpoint URL** to verify that you are getting a correct response from the AppMon Server.



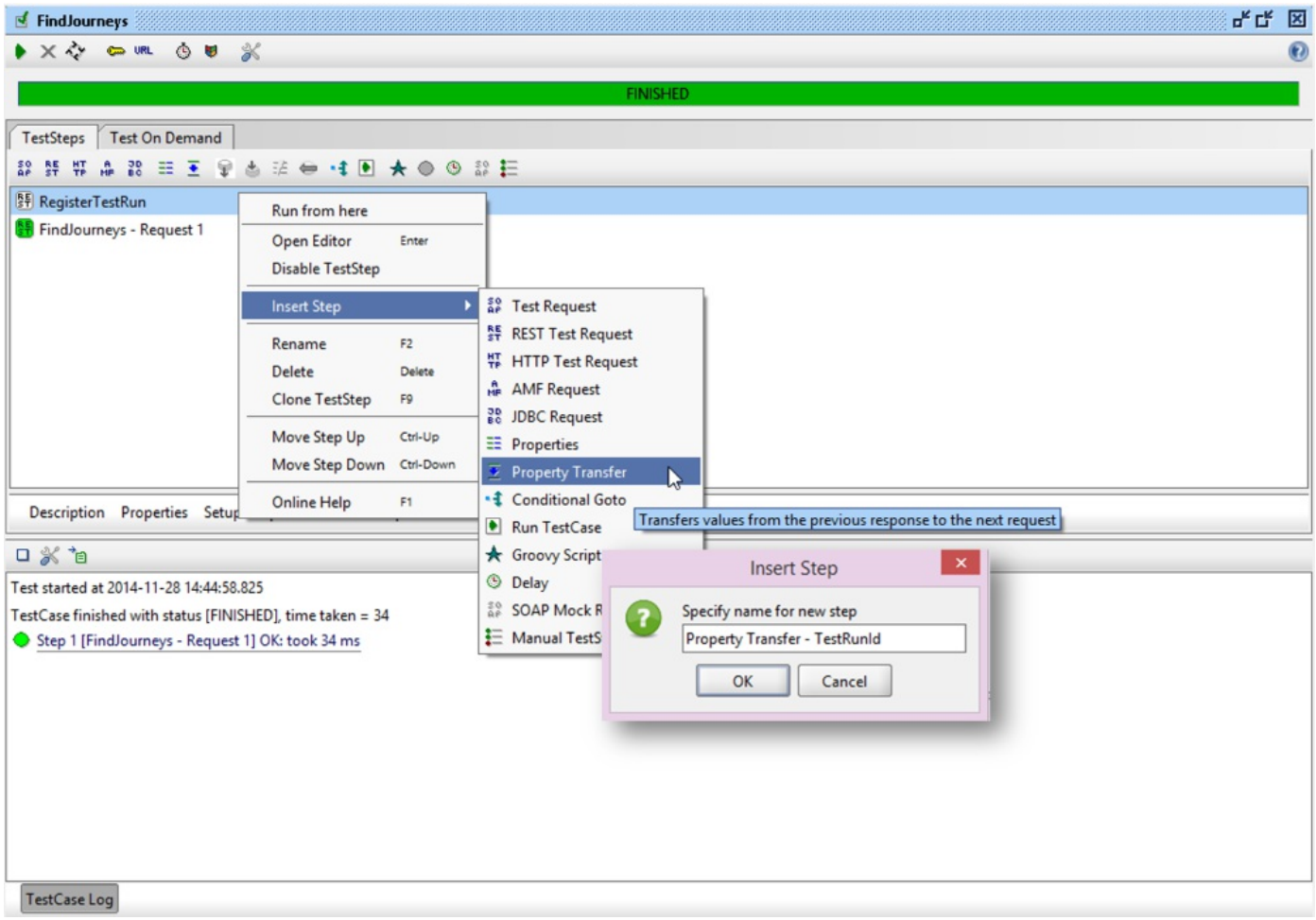
3. Create a new Custom Property into the Test Case FindJourneys to store the unique test run ID sent by the AppMon Server.

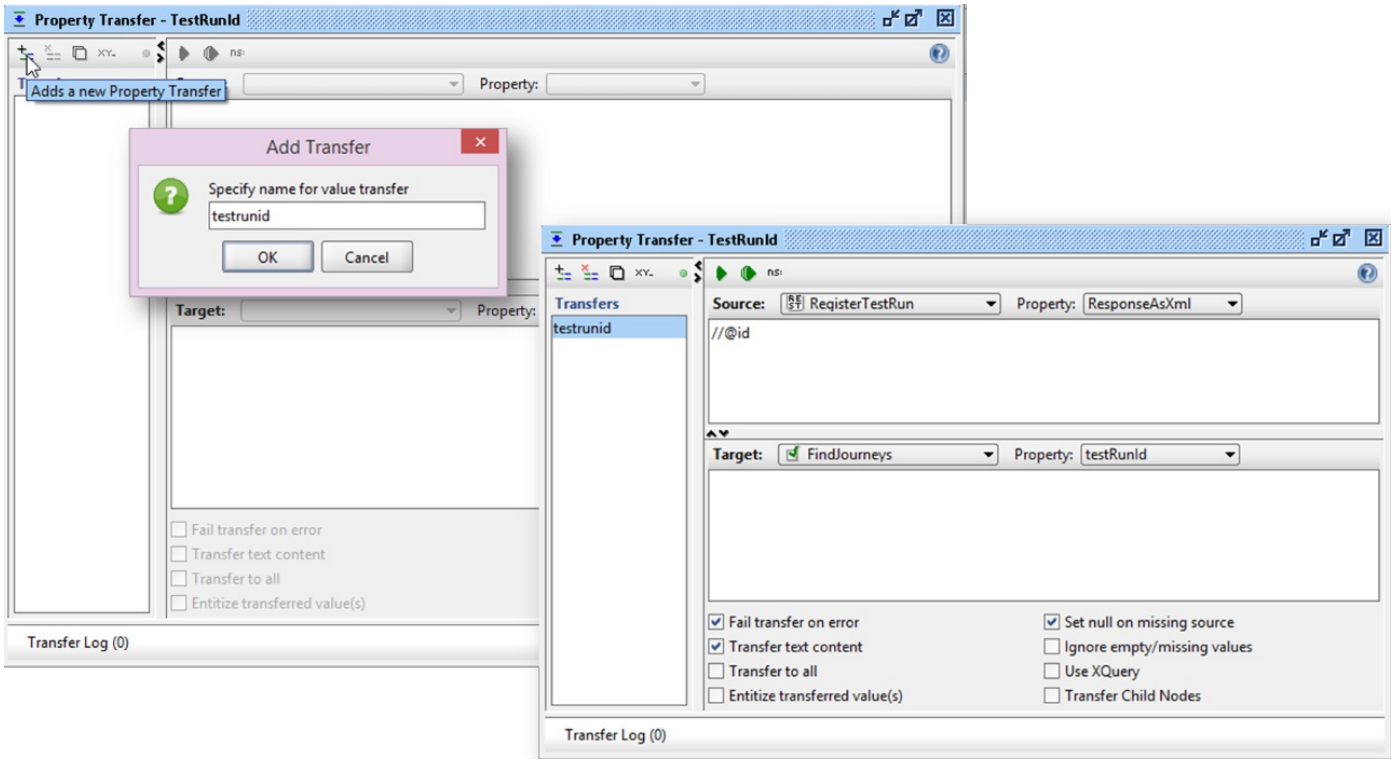


4. In the test case, insert a new step (REST Test Request) at the beginning to register the test run to the AppMon Server.

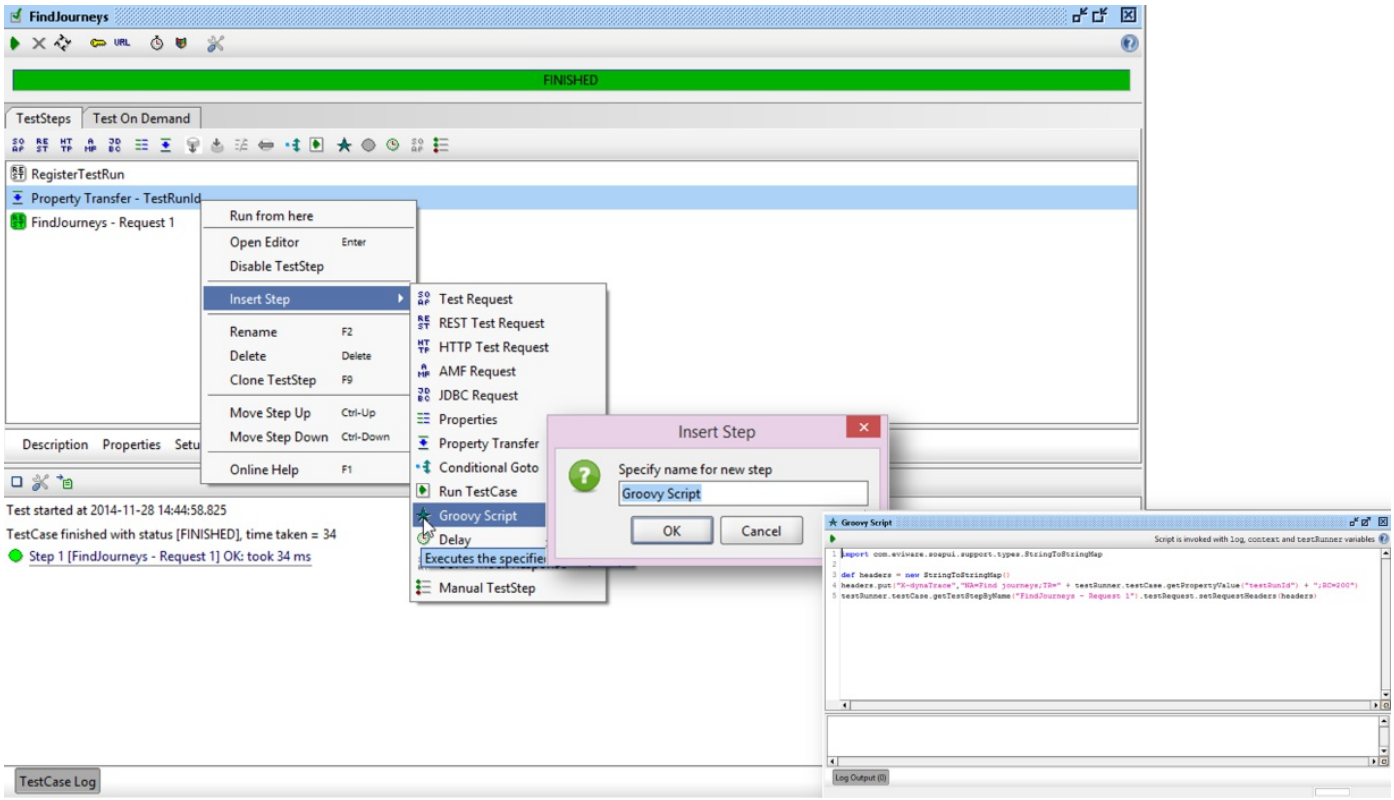


5. Add a step **Property Transfer** to extract the test run ID (xml attribute) from the request's response and save it into the previously defined property `testRunId`.



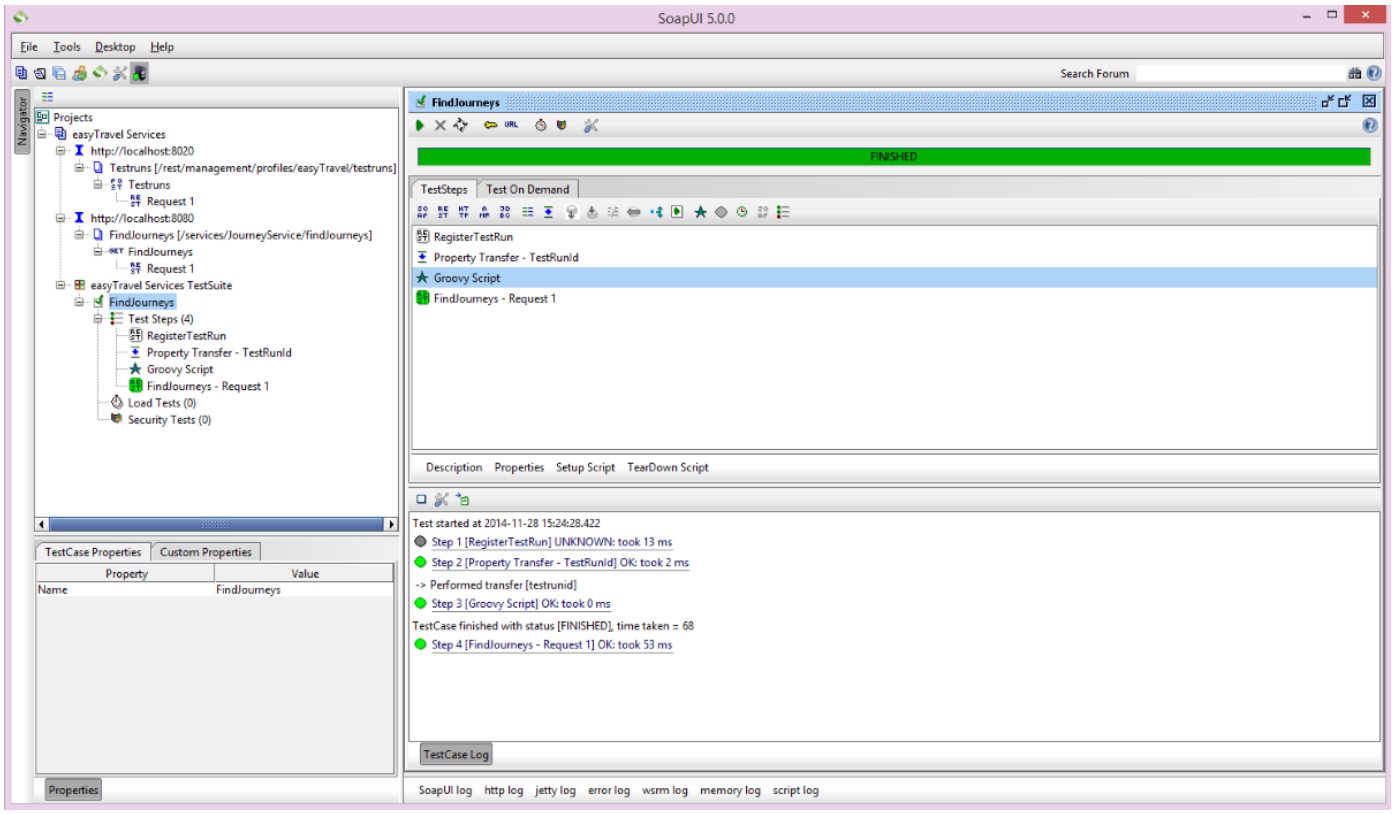


6. Finally, add a last step (Groovy Script) to set the right header tag for AppMon:

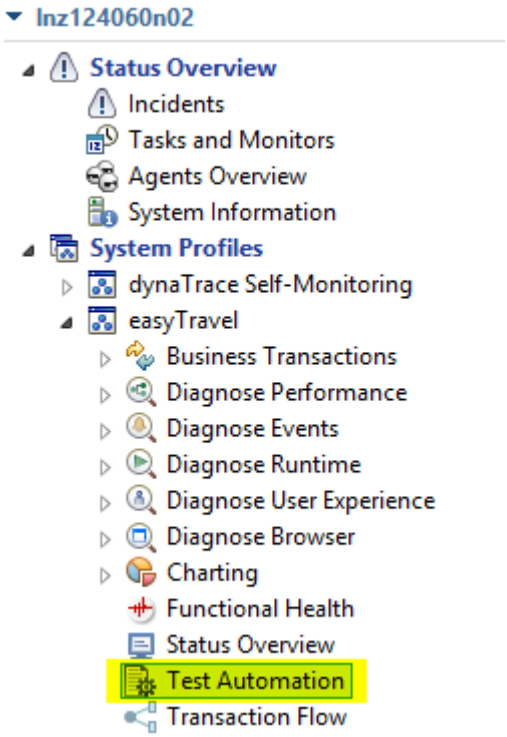



```
import com.eviware.soapui.support.types.StringToStringMap
def headers = new StringToStringMap()
headers.put("X-dynaTrace","NA=Find journeys;TR=" + testRunner.testCase.getPropertyValu
testRunner.testCase.getTestStepByName("FindJourneys - Request 1").testRequest.setReque
```

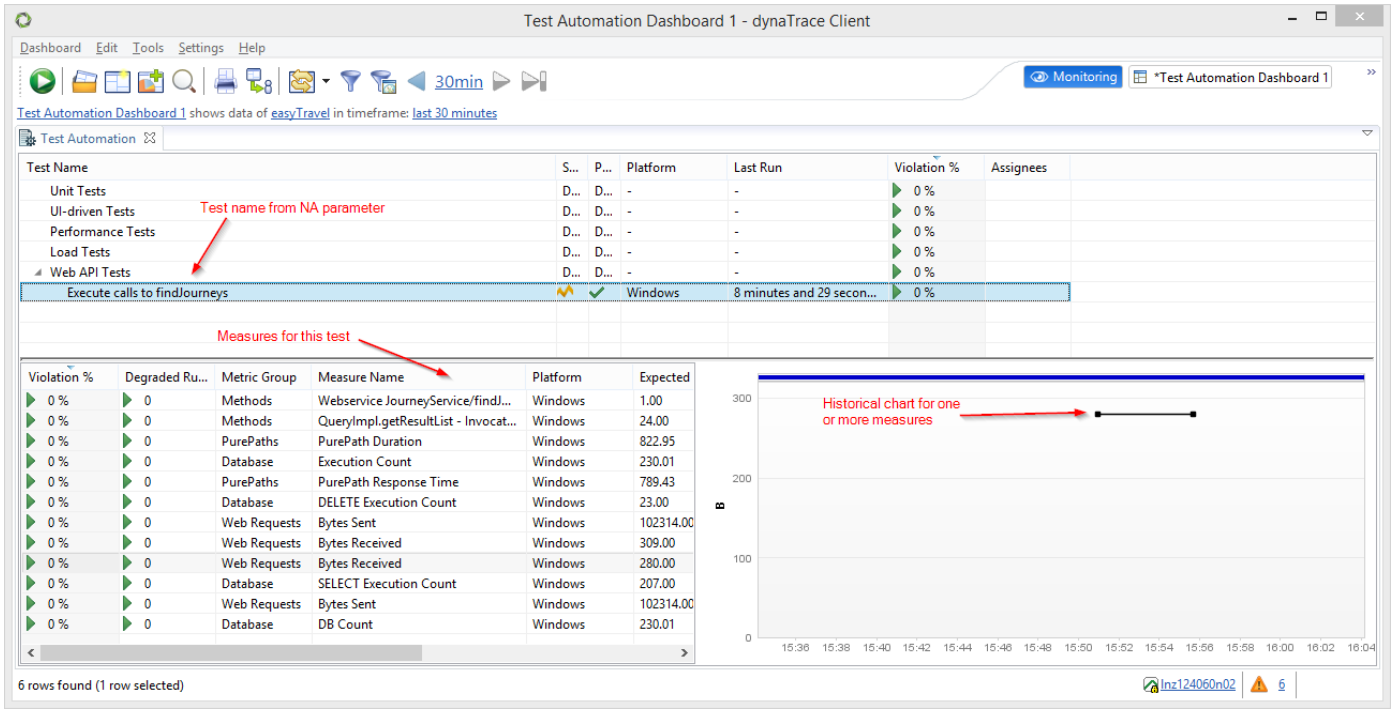
7. Run the test a few times to have more data in the Test Automation Dashlet.



8. You can see request results in the Test Automation Dashlet. Open the AppMon Client and open the Test Automation Dashlet from the Cockpit.



9. Adjust the time filter to the last 30 minutes to make it easier to find the requests you just executed. Your requests display with a list of metrics and a historic chart of metric values:



Download

Sample SoapUI Project

Test automation FAQ

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

FAQs

Test automation terms

Related pages

Continuous delivery and test automation

Test automation explained

System Profile - Test Automation

FAQs

Why is no data added to the Test Automation dashlet?

To troubleshoot and resolve this issue:

- Make sure the Performance Warehouse is connected.
- For unit tests, make sure the Java Tests or .NET Tests Sensor Pack is enabled for your Agent Group.
- Review the Agents overview to make sure the Agent is connected to the Server and it is in the correct Agent Group.
- Make sure the measures listed on the Capture Performance Data from Tests page have the **Create a measure for each agent** option selected. Also see below: Why are some measures missing in the Metrics section of the Test Results dashlet?
- Make sure you are using a Pre-Production License. Test Automation is not available in Production Edition.

Why aren't comparison and drilldowns available for the selected test runs?

To troubleshoot this issue:

- Check whether session recording was enabled for the test run.
- Check whether the session recording for this test run has already been deleted.

Why are some measures missing in the metrics section of the Test Results dashlet?

The Metrics section contains a limited set of measures. Each test category has its own list of measures, matching its characteristics. A list of measures assigned to each test category is on the Capture Performance Data from Tests page.

A measure might not appear in the Metrics section if it's not configured to be split by Agent. Check the **Create a measure for each agent** option in measure properties (**Details** panel, **Measure Spliting** section) to enable that bahavior.

Why are some measures duplicated in the metrics section of the Test Results dashlet?

Measures are considered different if they are coming from Agents running on different hosts. Such measures are reported in separate rows (you can display the Host column in the metric table to verify). In the case of larger CI environments that make use of many hosts running the builds and executing tests (build farm), that approach may lead to unwanted measure duplication. To solve the problem, use the `overridehostname` Agent option that would cause the Agent to report a given host name instead of the detected one. See Java Agent Configuration page for details.

If machines executing the builds have different performance capabilities and you force all of them to be reported as a single host, you may see shifts in performance related metric values as the consequent test executions are reported from different hosts. That may cause unwanted alerts on measure volatility or baseline violation.

Test automation terms

Test Case: AppMon identifies test methods of unit tests from testing frameworks such as JUnit, NUnit, and MSUnit as a test case. Every test method is listed as an individual test.

Corridor: The corridor is the expected range of values for a measure in a test case.

- The corridor is the `((100 - False Positive %) - Confidence Interval %)` of the Student's t-Distribution of a measure. By default, **False Positive %** is 1, so the default **Confident Interval %** is 99.
- The **False Positive %** is an AppMon term used for setting the confidence interval.
- Volatility is the Coefficient of Variation. By changing the **Volatile %** in the Test Automation settings, you define how high this Coefficient of Variation has to be for a test to be considered volatile.
- The calibration runs are applied every time changes are accepted.
- When accepting changes, the system behaves exactly as if the selected measurement was the first one ever observed. All existing values are discarded and are only shown in the chart. Selecting **Accept Change** instead of **Accept Changes from Here** uses the first measurement outside the corridor for accepting changes.

Test Case Assignee: The test case assignee is the person responsible for the test performance. This person receives notification emails when a measure for a test case is outside the corridor for the number of tests configured in the System Profile - Test Automation settings.

Test Run: One execution of a test case, such as a unit test.

Integrations and extensions

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Development kits

IDE integration

Integration with web load testing and monitoring tools

Load testing and monitoring tool integration

Plugin architecture

Realtime streaming

Expand the capabilities of AppMon by integrating with the following development environments, load testing, and monitoring tools:* **Eclipse** – launch applications and tests with agent injected, retrieve test results and quickly navigate from the AppMon Client to the exact location within your Eclipse development environment where you need to make changes.

- **IntelliJ IDEA Plugin** - launch applications and tests with agent injected, retrieve test results and quickly navigate from the AppMon Client to the exact location within your IntelliJ IDEA development environment where you need to make changes.
- **Microsoft Visual Studio** – launch applications and tests with agent injected, quickly navigate from the AppMon Client to the exact location within your Visual Studio development environment where you need to make changes.
- **Synthetic Monitoring Integration** – enables the AppMon Server to send common application-related UEM data as well as data regarding your synthetic tests to be displayed in the Synthetic Portal.
- **DC RUM** – correlate the data collected by DC RUM to AppMon PurePaths.
- **LoadRunner** – thoroughly evaluate HTTP Requests generated by LoadRunner´s Virtual User Generator.

Extend the default AppMon features and product functionality by using the API, plugins, and ADKs:* **REST interface** – expose AppMon Server and AppMon Client management functions through the available REST interface.

- **AppMon plugins** – extend AppMon with new features and functions for system monitoring, application monitoring, notifications, custom actions, and lifecycle automation.
- **ADKs** – use the AppMon Agent Development Kit, JavaScript ADK, and Mobile App ADK to extend the default AppMon Agent monitoring capabilities by integrating AppMon with technologies such as native C or C++ applications, custom communication protocols, custom JavaScript frameworks, and native mobile applications.
- **Mobile Symbolication Service** – upload iOS dsym files and Proguard mapping files to the AppMon Server to get readable stack traces for mobile crashes.

IDE integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Eclipse integration plugin

IntelliJ IDEA plugin

Source lookup feature

Visual Studio extension

AppMon integrates with the Eclipse IDE, IntelliJ IDEA and Visual Studio Extension through the AppMon CodeLink component, which provides for a convenient way to perform lookups of source files and methods from applications under diagnosis.







Eclipse integration plugin

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Launcher

Test result

Source code look-up

Watch and learn

Downloads

Dynatrace Eclipse Integration Plugin



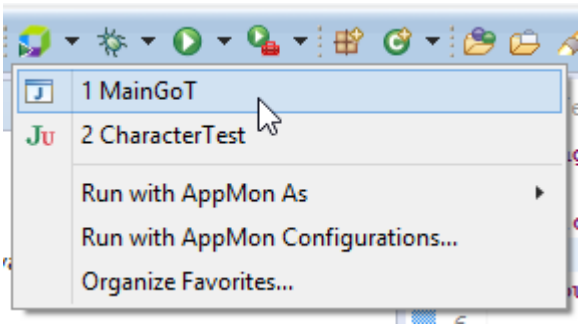
To simplify problem resolution, AppMon integrates with Eclipse through the AppMon Eclipse Integration Plugin. The AppMon Eclipse Integration Plugin enable you to:

- Launch applications with an injected AppMon Agent directly from Eclipse.
- Retrieve & display the key architectural metrics (such as number of SQL queries, external API calls, exceptions and log messages) from your tests.
- Perform look-ups of sources files and methods from applications under diagnosis in AppMon.

Download, installation and configuration instructions are available on the Eclipse Integration Plugin page.

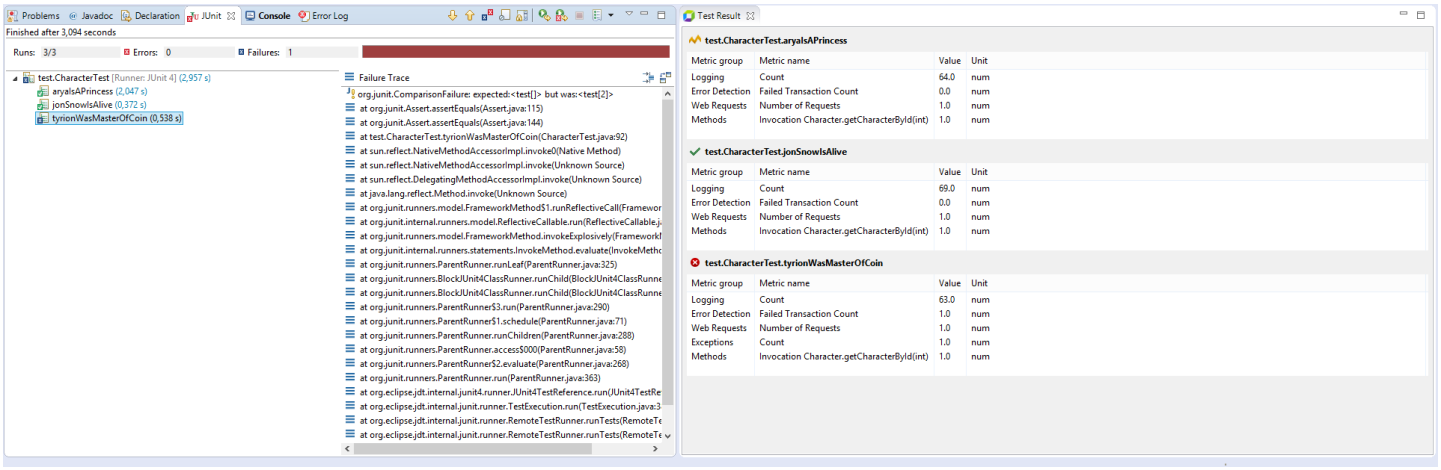
Launcher

With the AppMon Launcher, Auto instrument your Java applications from Eclipse:

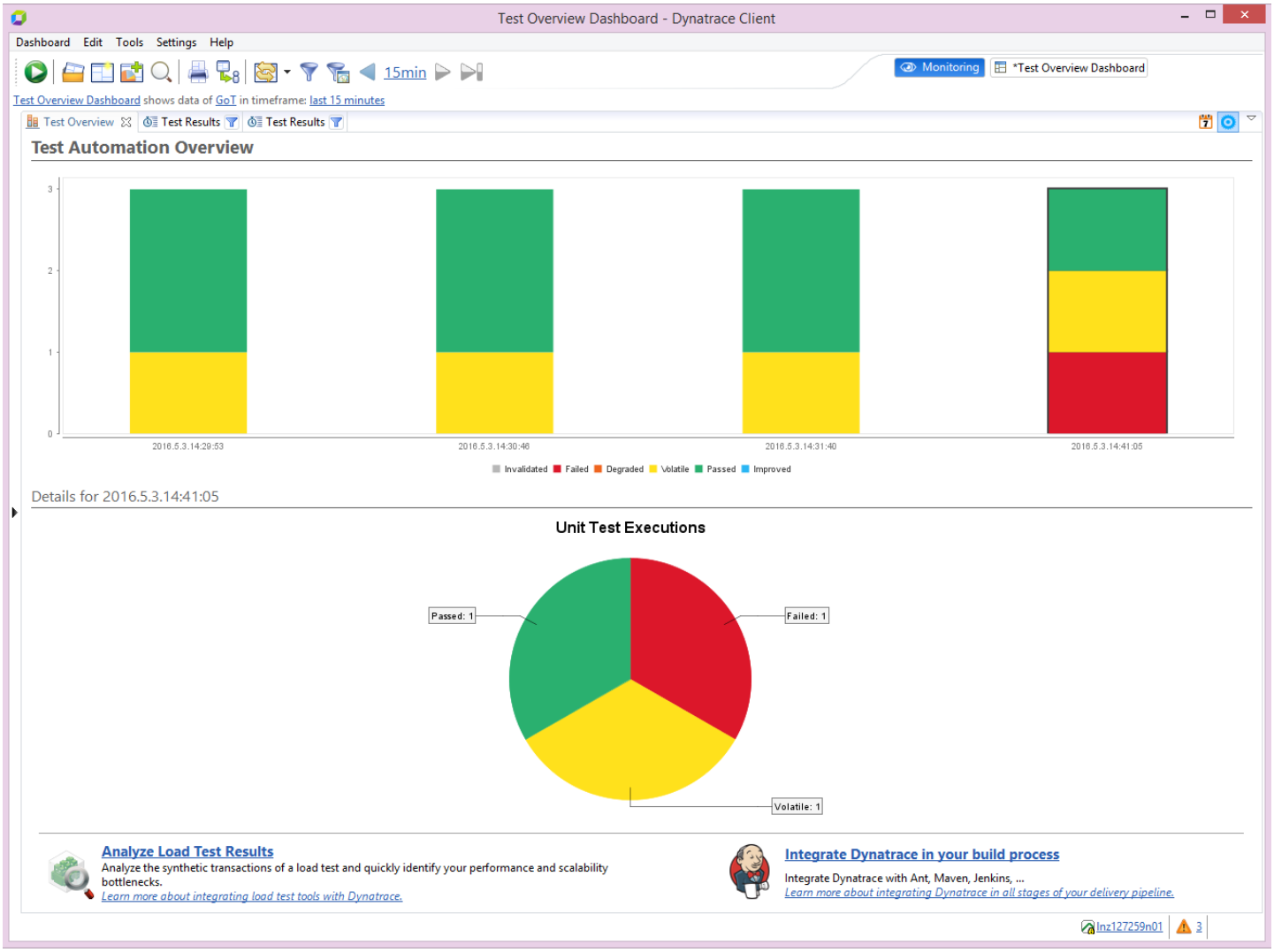


Test result

When lauching JUnit tests, the plug-in will automatically register a test run to the AppMon Server. The results and key architectural metrics are automatically retrieved and displayed in the Test Result tab.



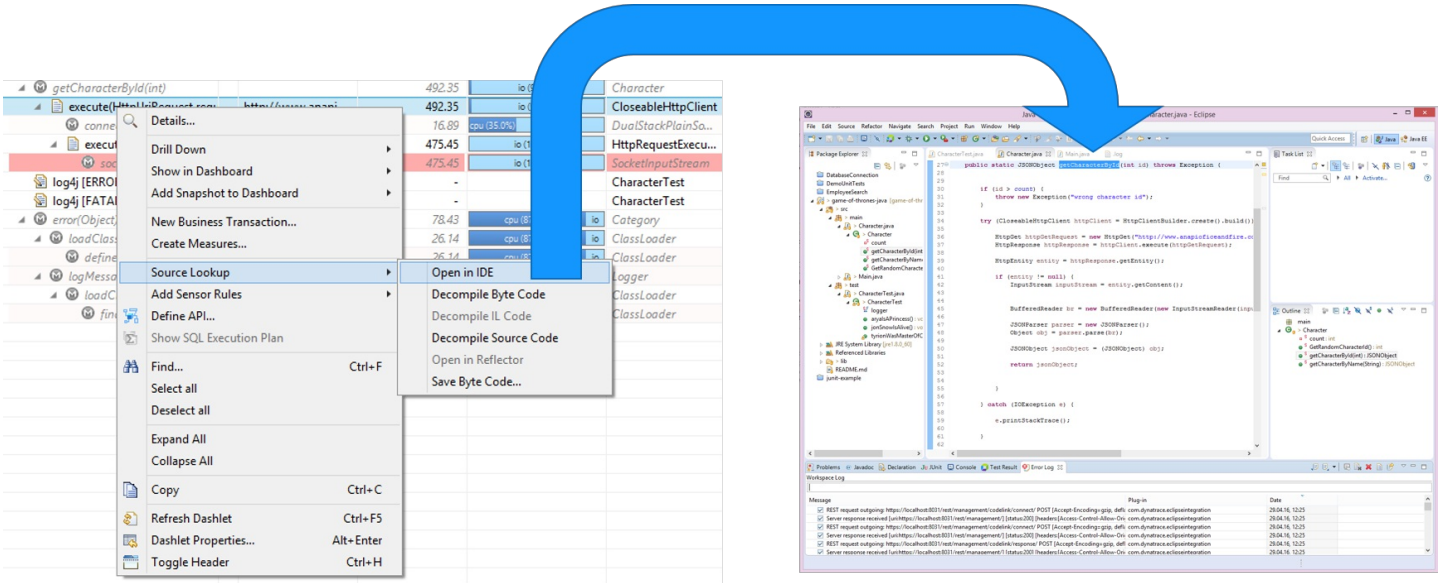
The test results are also visible in the AppMon Client for further drill-down to the PurePath level.



More information about the Test Automation feature: Test Automation Explained

Source code look-up

When AppMon identifies a source code fragment as the root cause of a particular problem, you can go directly from the AppMon Client to the exact location within your Eclipse development environment where you need to make changes.


Details...
Drill Down
Show in Dashboard
Add Snapshot to Dashboard
New Business Transaction...
Create Measures...
Source Lookup
Add Sensor Rules
Define API...
Show SQL Execution Plan
Find... Ctrl+F
Select all
Deselect all
Expand All
Collapse All
Copy Ctrl+C
Refresh Dashlet Ctrl+F5
Dashlet Properties... Alt+Enter
Toggle Header Ctrl+H
Open in IDE
Decompile Byte Code
Decompile IL Code
Decompile Source Code
Open in Reflector
Save Byte Code...

IntelliJ IDEA plugin

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Launcher

Test result

Source code look-up

Downloads

IntelliJ IDEA integration plugin

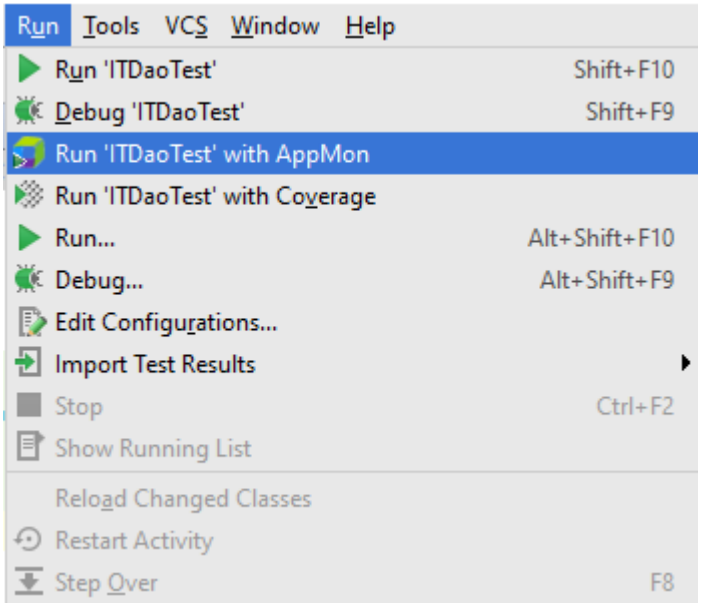
To simplify problem resolution, AppMon integrates with IntelliJ IDEA through the AppMon IntelliJ IDEA Integration Plugin. The AppMon IntelliJ IDEA Integration Plugin enable you to:

- Launch applications with an injected AppMon Agent directly from IntelliJ IDEA.
- Retrieve & display the key architectural metrics (such as number of SQL queries, external API calls, exceptions and log messages) from your tests.
- Perform look-ups of sources files and methods from applications under diagnosis in AppMon.

Manual download, installation and configuration instructions are available on the IntelliJ IDEA Integration Plugin.

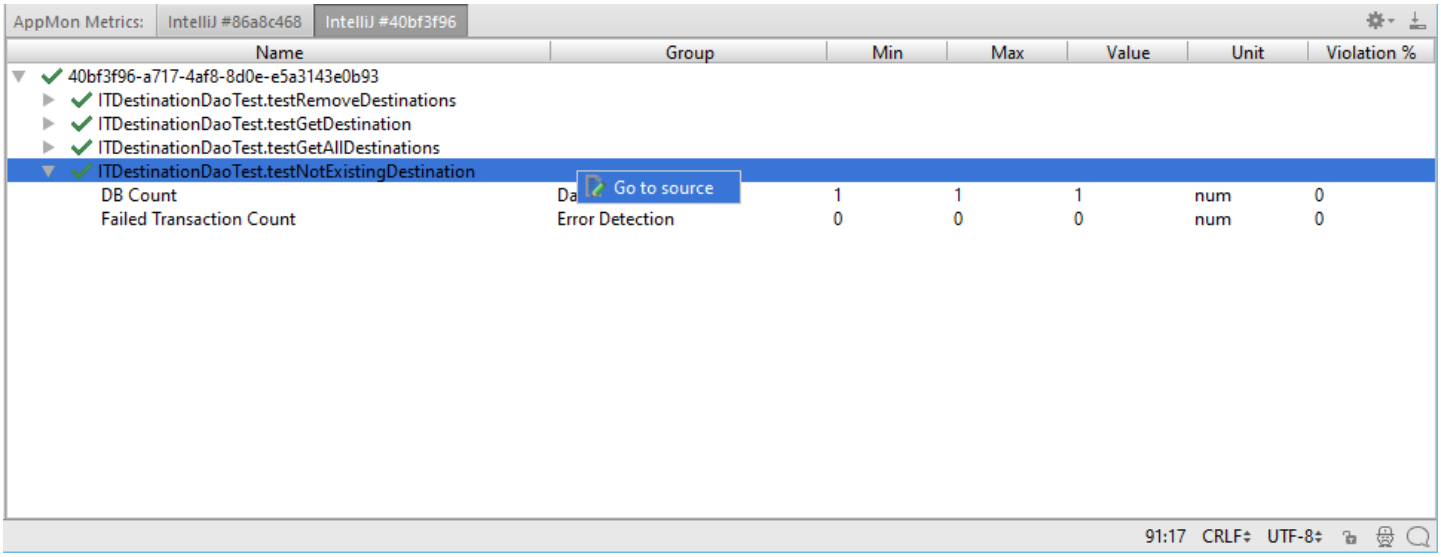
Launcher

With the AppMon Launcher, Auto instrument your Java applications from IntelliJ IDEA:

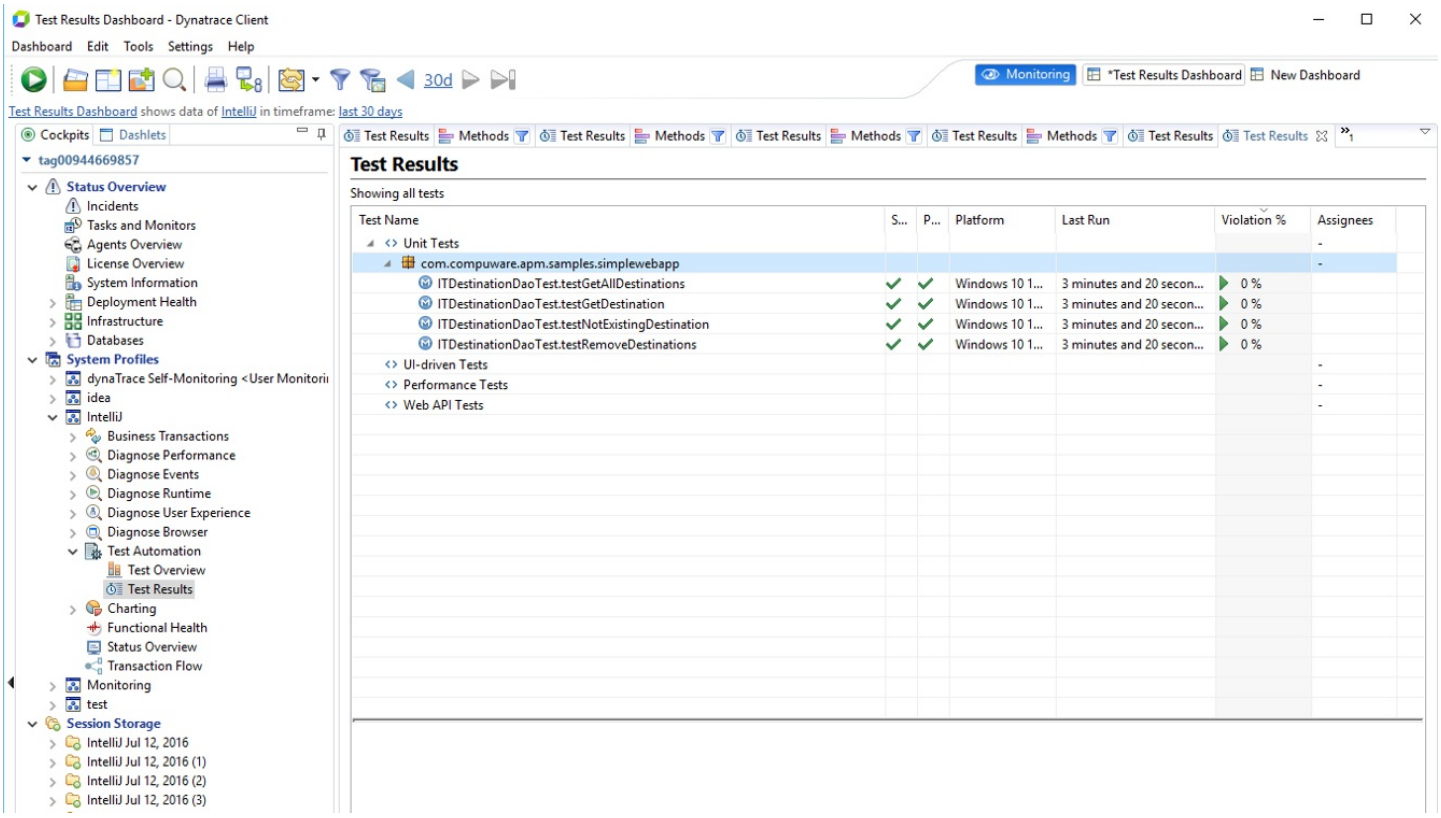


Test result

When lauching JUnit tests, the plug-in will automatically register a test run to the AppMon Server. The results and key architectural metrics are automatically retrieved and displayed in the Test Result tab.



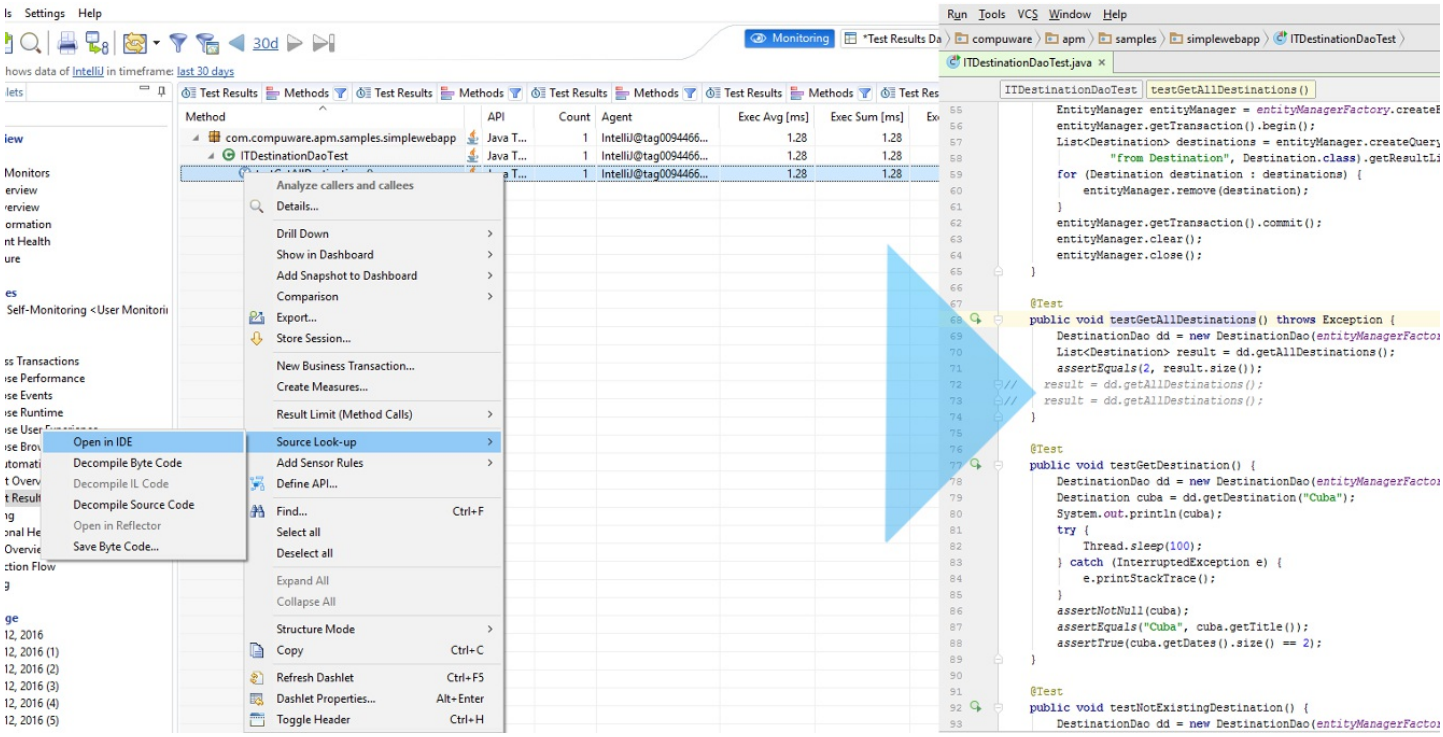
The test results are also visible in the AppMon Client for further drill-down to the PurePath level.



More information about the Test Automation feature: Test Automation Explained

Source code look-up

When AppMon identifies a source code fragment as the root cause of a particular problem, you can go directly from the AppMon Client to the exact location within your IntelliJ IDEA development environment where you need to make changes.


Monitoring
Test Results
Methods
Method
API
Count
Agent
Exec Avg [ms]
Exec Sum [ms]
com.compuware.apm.samples.simplewebapp
ITDestinationDaoTest
Analyze callers and callees
Details...
Drill Down
Show in Dashboard
Add Snapshot to Dashboard
Comparison
Export...
Store Session...
New Business Transaction...
Create Measures...
Result Limit (Method Calls)
Source Look-up
Add Sensor Rules
Define API...
Find... Ctrl+F
Select all
Deselect all
Expand All
Collapse All
Structure Mode
Copy Ctrl+C
Refresh Dashlet Ctrl+F5
Dashlet Properties... Alt+Enter
Toggle Header Ctrl+H
Open in IDE
Decompile Byte Code
Decompile IL Code
Decompile Source Code
Open in Reflector
Save Byte Code...
ITDestinationDaoTest.java
testGetAllDestinations()
@Test
public void testGetAllDestinations() throws Exception {
@Test
public void testGetDestination() {
@Test
public void testNotExistingDestination() {

Visual Studio extension

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Launcher

Source code look-up

Downloads

Visual Studio 2015 extension

Visual Studio 2010, 2012 and 2013 IDE plugins

Visual Studio 2005 and 2008 IDE plugins



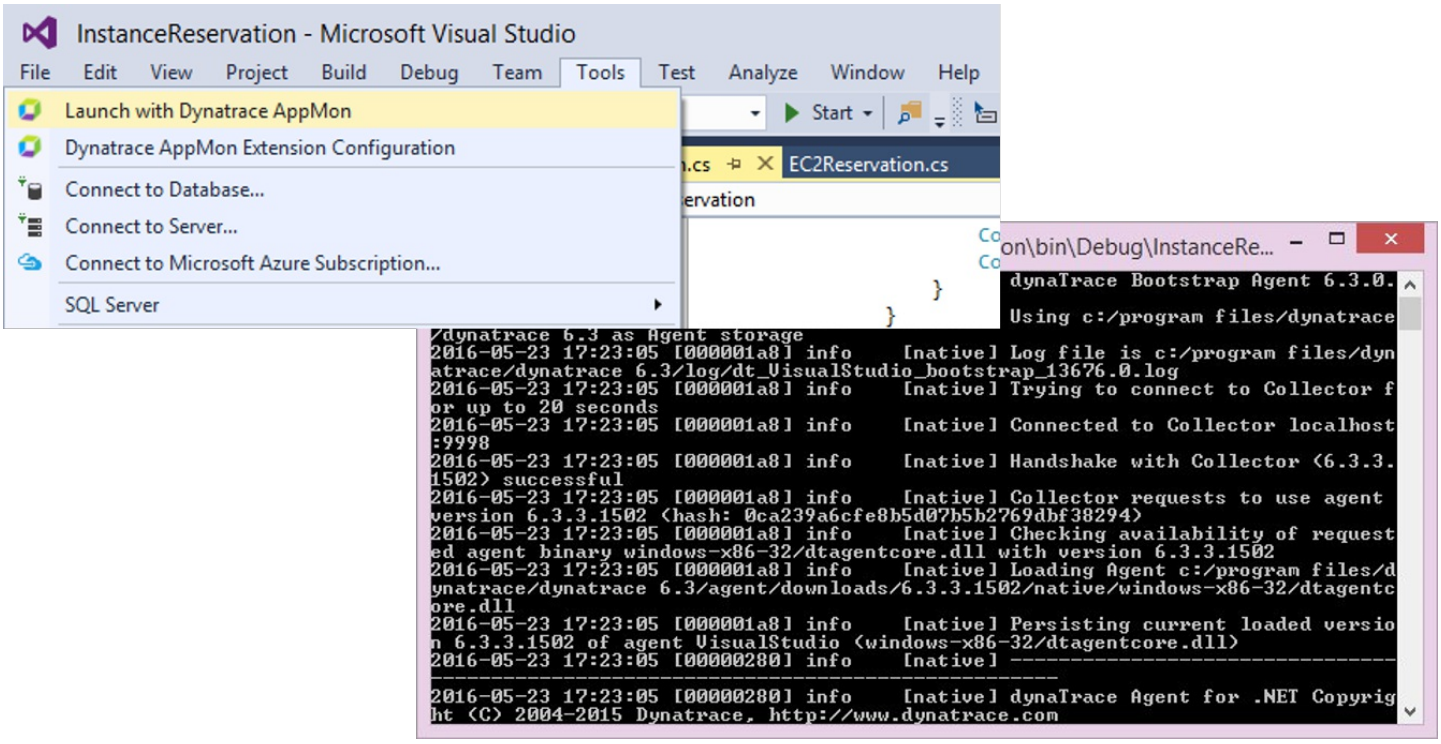
To simplify problem resolution, AppMon integrates with Visual Studio through the AppMon Visual Studio Extension. The AppMon Visual Studio Extension enables you to:

- Launch applications with an injected AppMon Agent directly from Visual Studio.
- Perform look-ups of sources files and methods from applications under diagnosis in AppMon.

Download, installation and configuration instructions are available on the Visual Studio Extension Page.

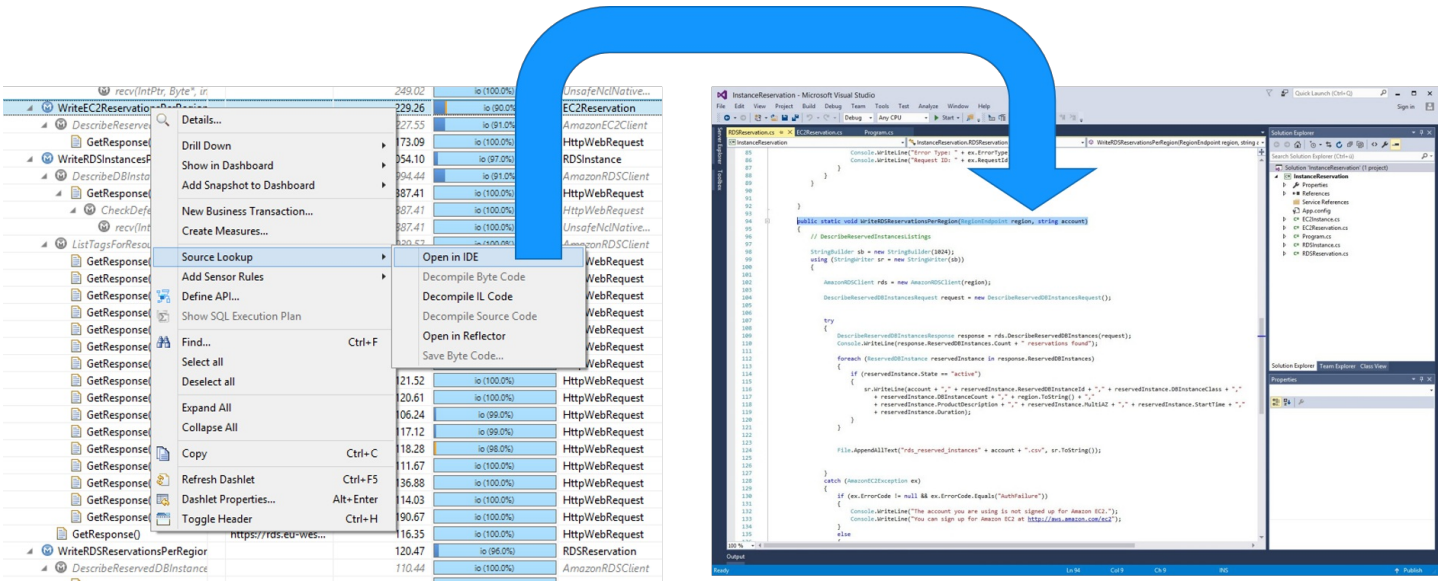
Launcher

With the AppMon Launcher, Auto instrument your Java applications from Visual Studio:



Source code look-up

When AppMon identifies a source code fragment as the root cause of a particular problem, you can go directly from the AppMon Client to the exact location within your Visual Studio environment where you need to make changes.



Source lookup feature

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Lookup types

Views in AppMon Client that offer source code lookup

Source code viewer

Downloads

CodeLink AddIn for Microsoft Visual Studio

Dynatrace Eclipse Integration Plugin

JAD Java Decompiler

The AppMon Client enables you to analyze PurePaths down to the individual methods that have been instrumented in the context of the captured transaction. The PurePath contains information such as method arguments, execution time, and CPU time. When you identify a problematic method either in the **PurePath** view or in the **Methods** view of the AppMon Client, you can use the Source Code Lookup feature provided by the AppMon Client to go deeper in the source code that is associated with the method.

This insight into the source code provides deeper analysis of the captured data.

The AppMon Server keeps all metadata information for all classes that are loaded by the connected AppMon Agent. In combination with the data collected in the PurePath, this metadata enables the AppMon Client to perform different types of Source Code Lookup.

Lookup types

The following types of lookup are available:

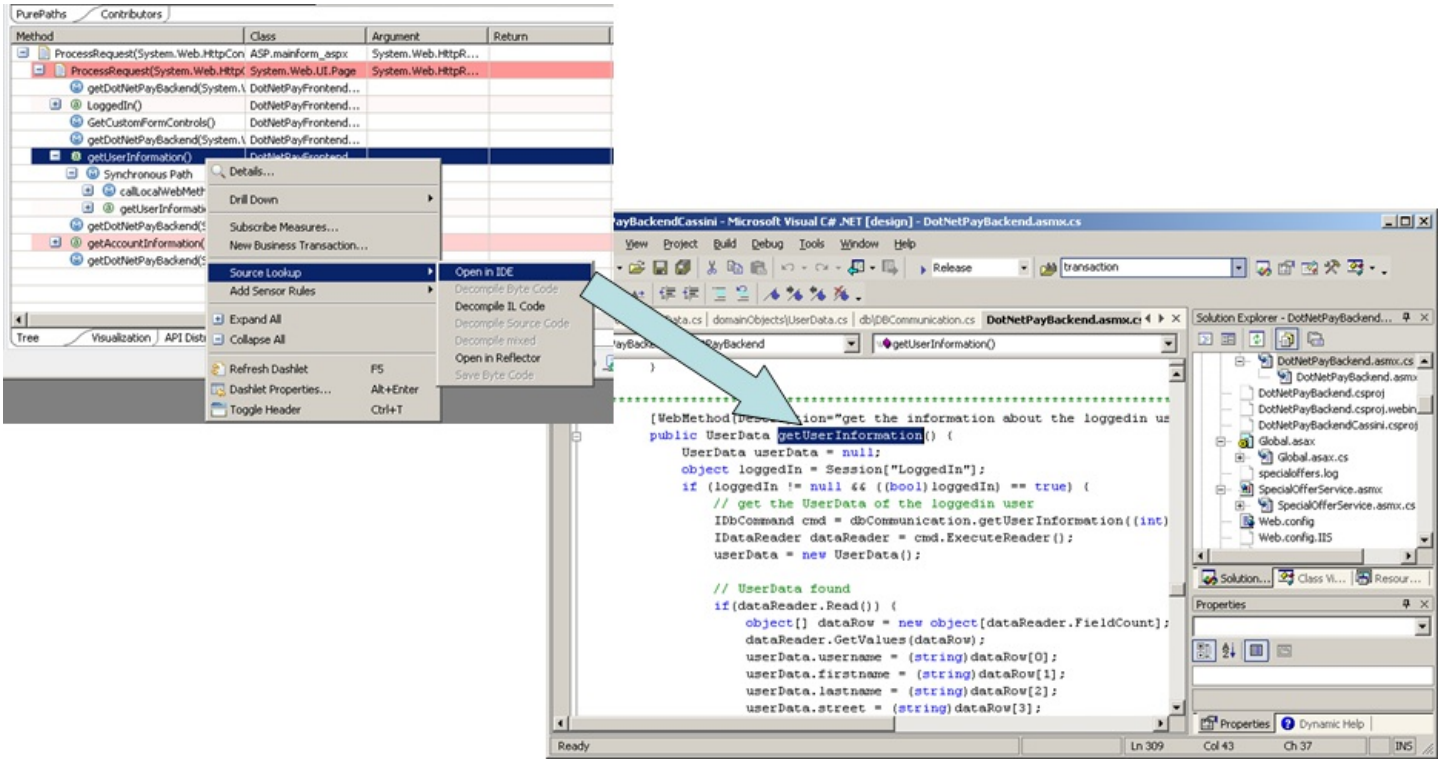
- **Open in IDE** – Jumps to the source code line in your IDE if you have the Visual Studio Solution or Eclipse Project open.
- **Decompile Byte Code** – Decompiles the Java byte code and displays it in a human-readable format.
- **Decompile IL Code** – Decompiles the .NET intermediate code and displays it in a human-readable format.
- **Decompile Source Code** – Decompiles the Java source code and displays it in a human-readable format.
- **Open in Reflector** – Extracts the .NET assembly that contains the selected method and opens .NET Reflector to decompile it.
- **Save Byte Code** – Exports Java byte code to a file that can be analyzed with external tools.

Open in IDE (Eclipse or Visual Studio)

To use this feature, you must:

- Install AppMon CodeLink Add-In For Microsoft Visual Studio or Eclipse Integration Plugin, depending on the IDE you are using.
- Open the Visual Studio Solution or Eclipse Project that contains the source code you want to look up.

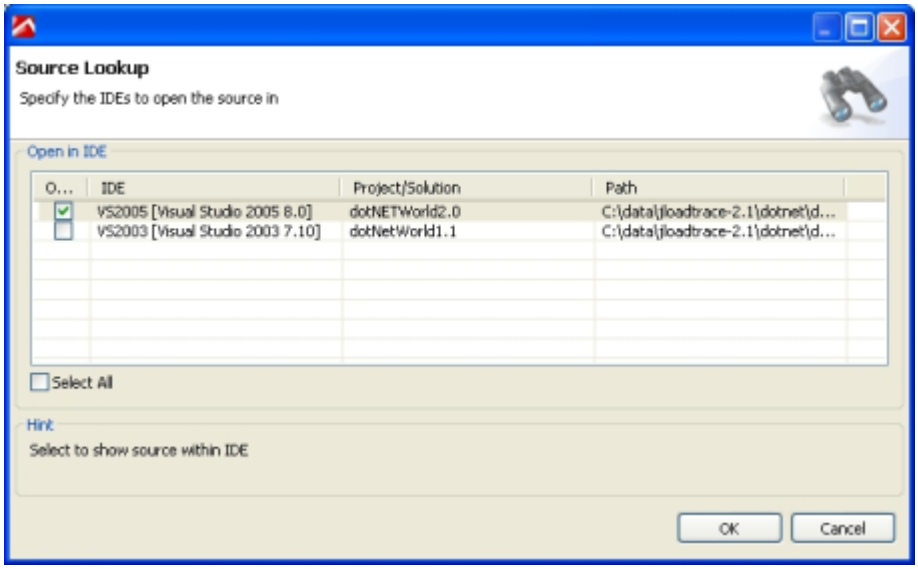
When you have a captured PurePath, right-click a method and select **Source Lookup > Open in IDE**.



Selecting the Open in IDE command

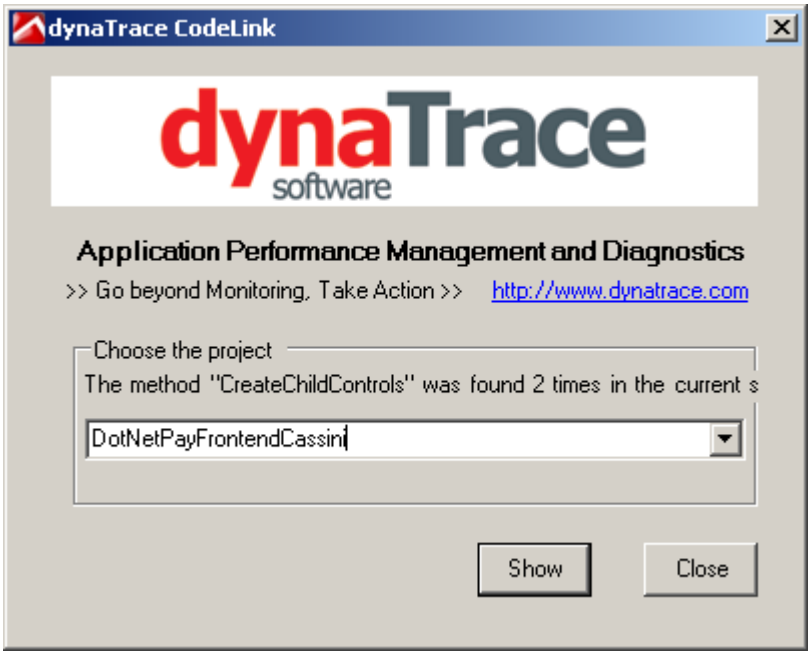
If the source code lookup is not successful, the status bar in the AppMon Client displays the message “Source lookup was not successful.” The likely reason is that the selected method was not part of the open Visual Studio Solution or Eclipse Project.

If you have more than one Visual Studio or Eclipse instance open, you are prompted to select a particular instance to perform the source code lookup.



Selecting the IDE for lookup

If you have multiple projects in your IDE workspace that contain multiple matching method names, you are prompted to select the project for lookup.

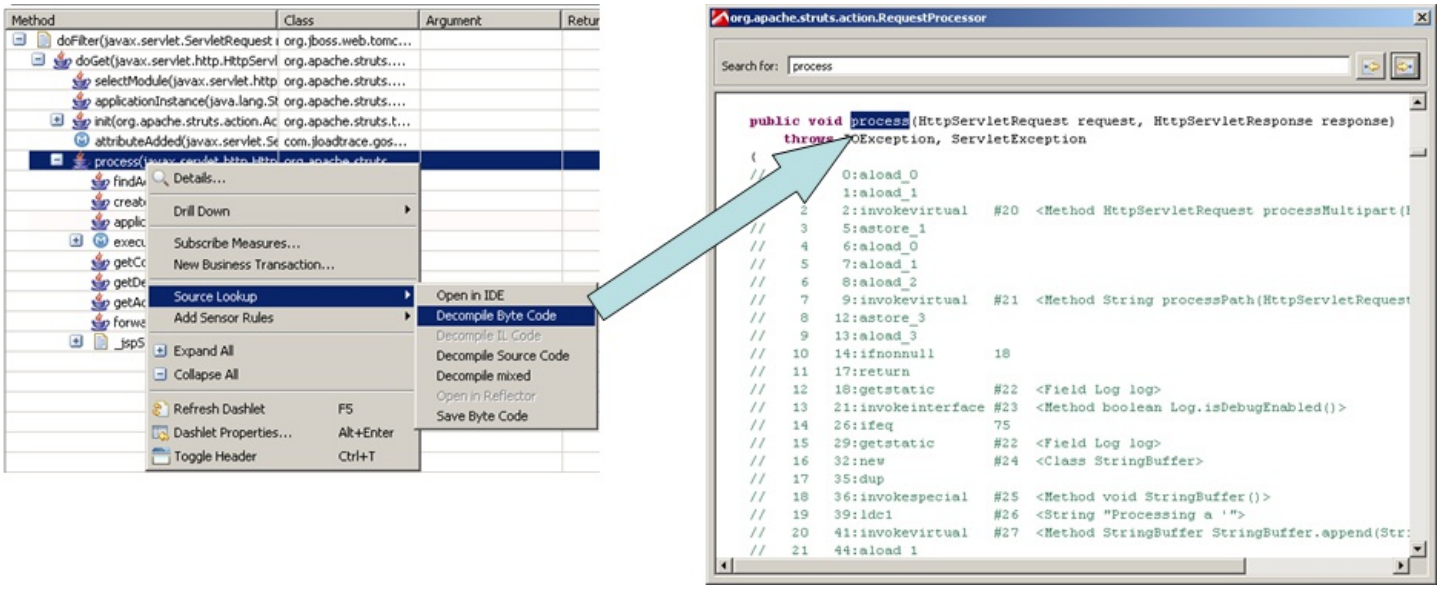


Selecting the project for lookup

Decompile Byte Code

Java code is compiled into byte code that is interpreted by the Java Runtime during execution. Use** Decompile Byte Code** to explore the byte code for any method, either your own code or code that runs in the the Java Application. The byte code view shows a human-readable representation of the Java instructions that make up the method body.

To view the byte code of an instrumented method or the complete instrumented class, right-click the object and select **Source Lookup > Decompile Byte Code**.



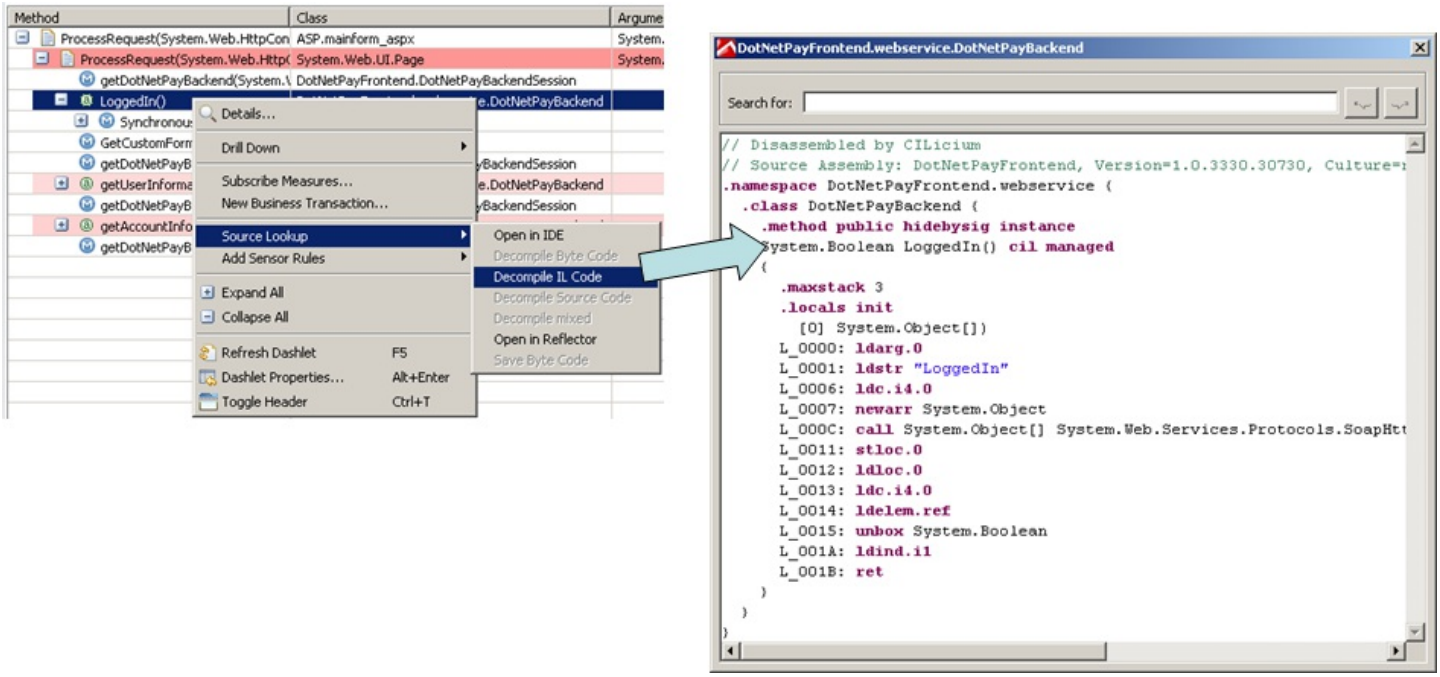
Using the Decompile Byte Code command

Decompile IL Code

Any .NET language (for example, C# and VB.NET) is compiled into intermediate code that is interpreted by the .NET Runtime during execution.

Use** Decompile IL Code** to explore the intermediate code for any .NET method, either your own code or code that runs in the .NET application.

You can look up the intermediate code of an instrumented method or the complete instrumented class. The intermediate code view shows a human-readable representation of the .NET Runtime instructions that make up the method body.

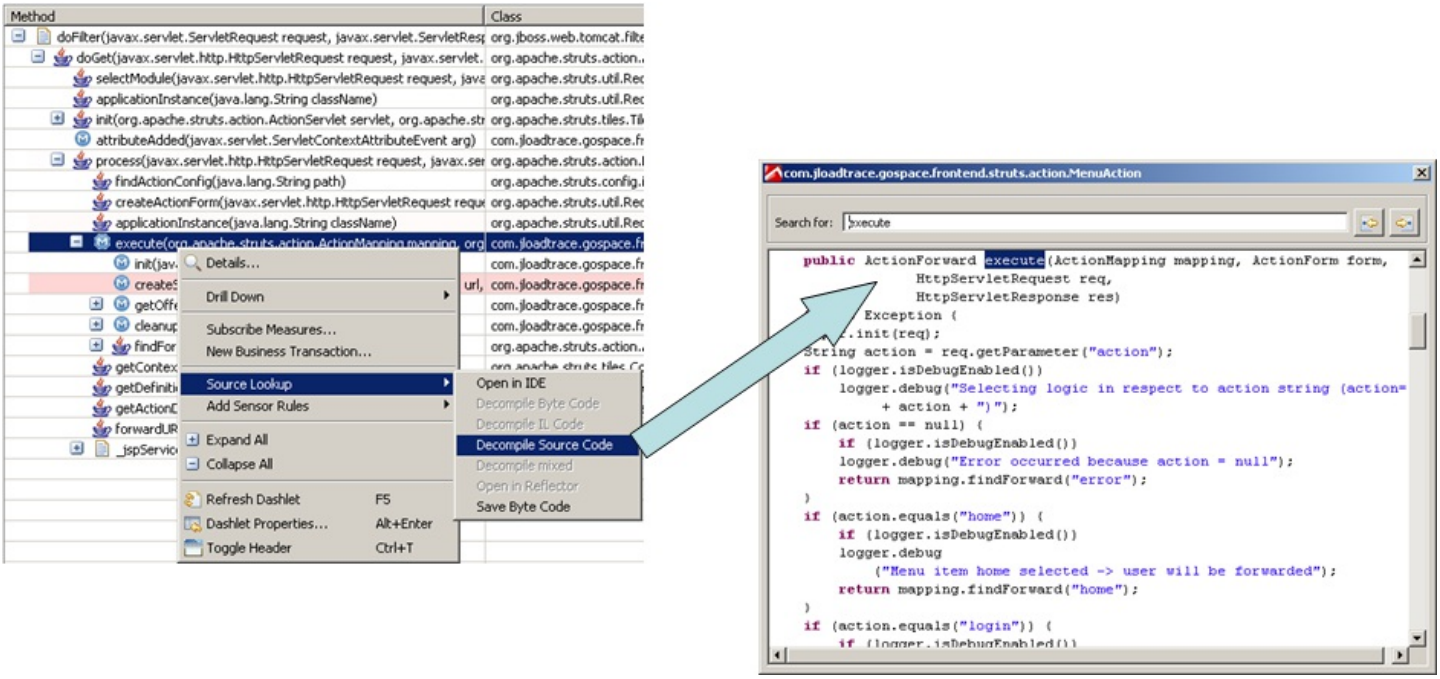


Using the Decompile IL Code command

Decompile Source Code

Java programming code is compiled into byte code that is interpreted by the Java Runtime during execution. Use **Decompile Source Code** to explore the Java programming code that is generated based on the Java byte code, either your own code or code that runs in your Java application.

You can look up the decompiled Java source code of an instrumented method or the complete instrumented class. The *Source Code* view shows a decompiled representation of the Java Runtime instructions that make up the method body.



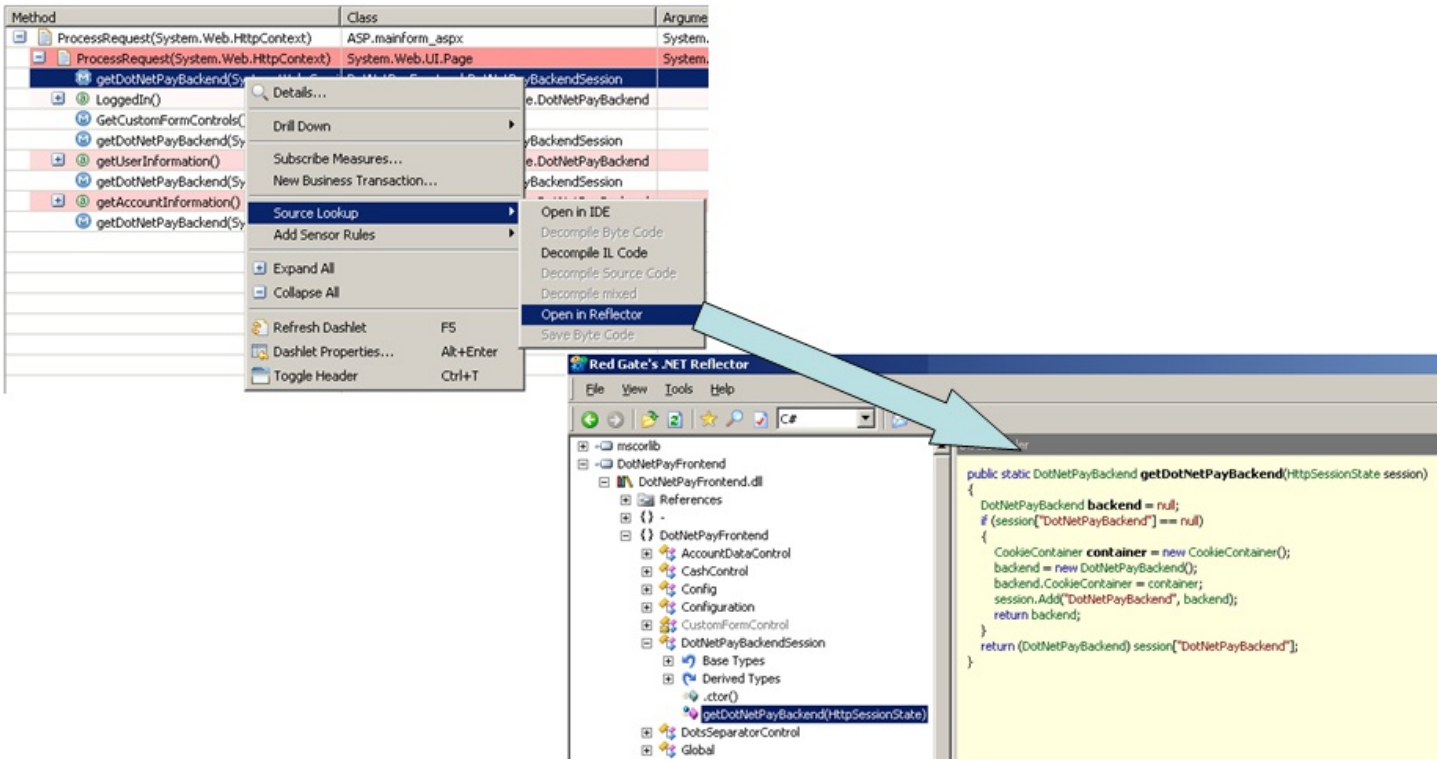
Using the Decompile Source Code command

Open in Reflector

The .NET Reflector is a tool that decompiles .NET Assemblies.

To use the **Open in Reflector** command, you must download and install the .NET Reflector Plugin.

When the plugin is installed and enabled, you can look up a method's source code through .NET Reflector.

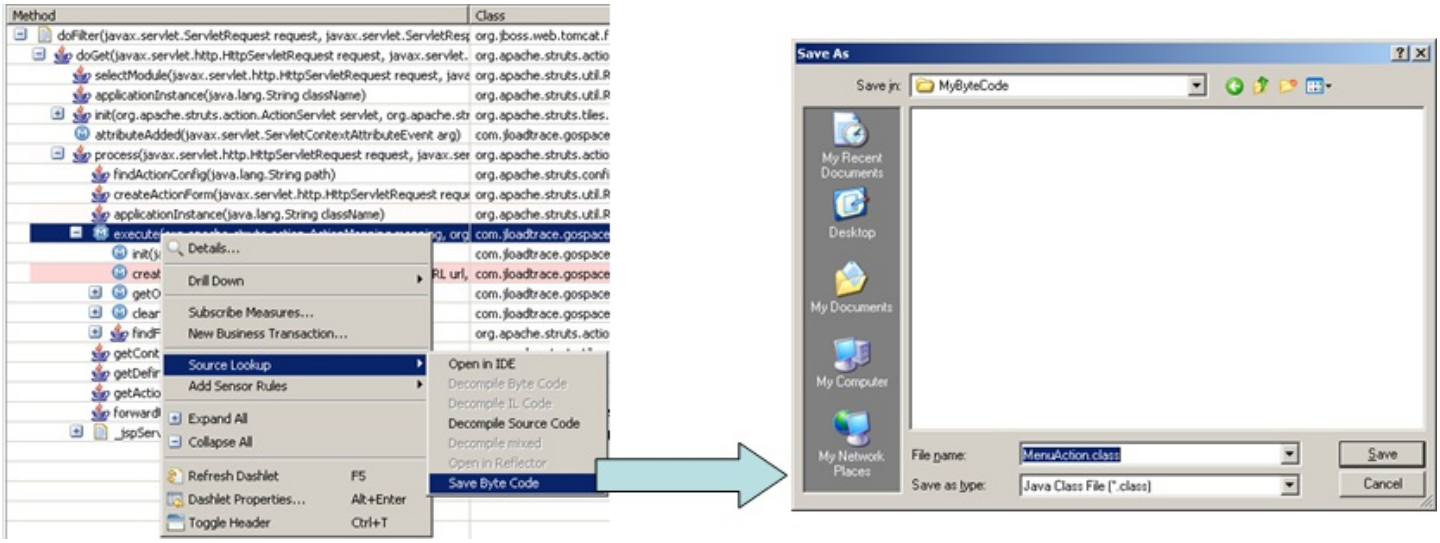


Using the Open in Reflector command

Save Byte Code

Use **Save Byte Code** to export the Java byte code of the Java class that implements the selected method. You can use the exported file with any Java decompiler tool to analyze the internals of this method and all other methods of the class.

Right-click the method and select **Save Byte Code**. You are prompted to specify a target file name. Optionally, navigate to the folder where you want to save the file.



Using the Save Byte Code command

Views in AppMon Client that offer source code lookup

The following views allow Source Code Lookup:

- PurePath Tree
- Methods View
- Deployed Sensor View

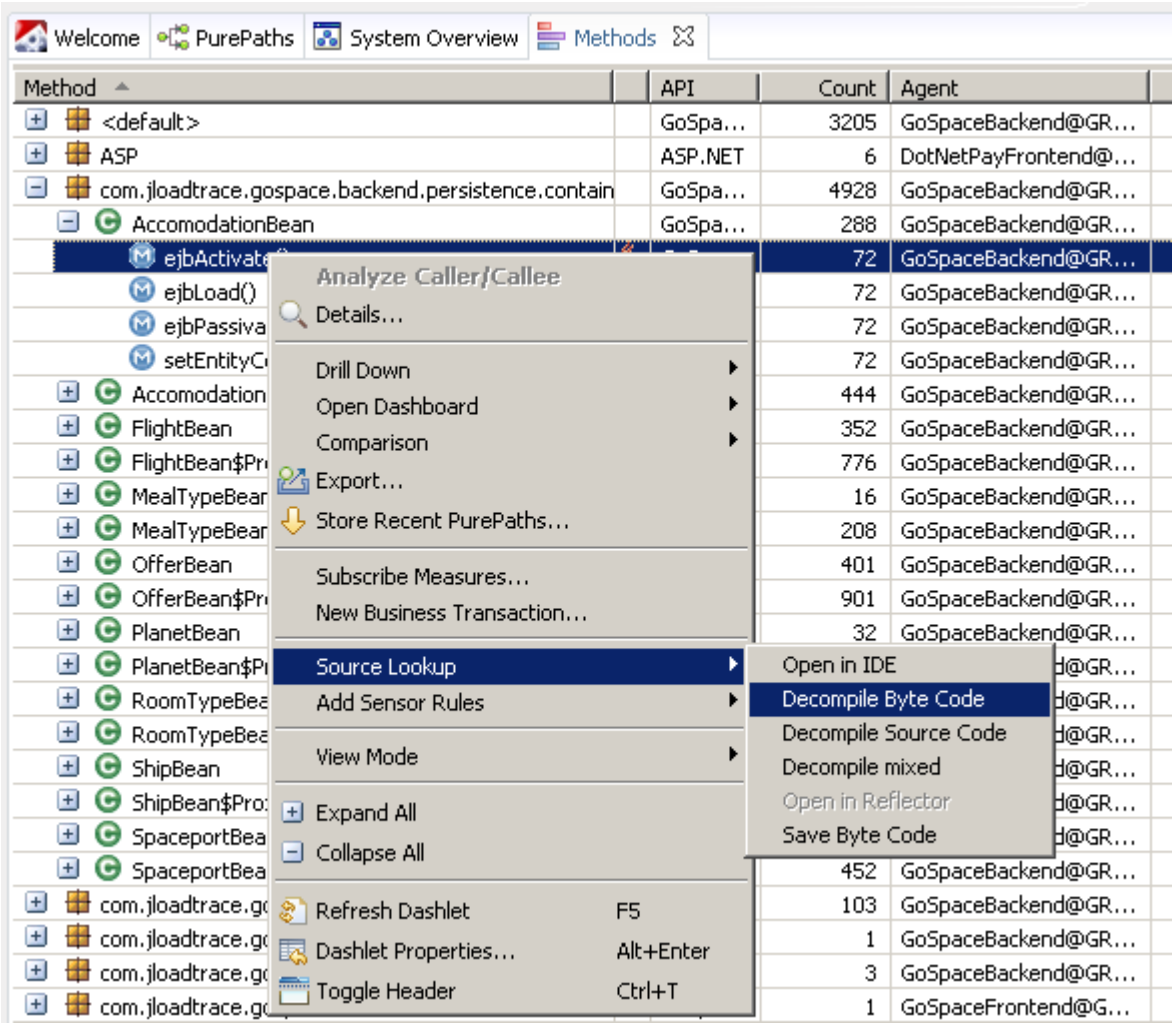
PurePath tree

Click any node in the **PurePath** tree and use the context menu to perform a Source Code Lookup.

Methods view

In the **Methods** view, you can perform Source Code Lookup on packages, classes, and methods:

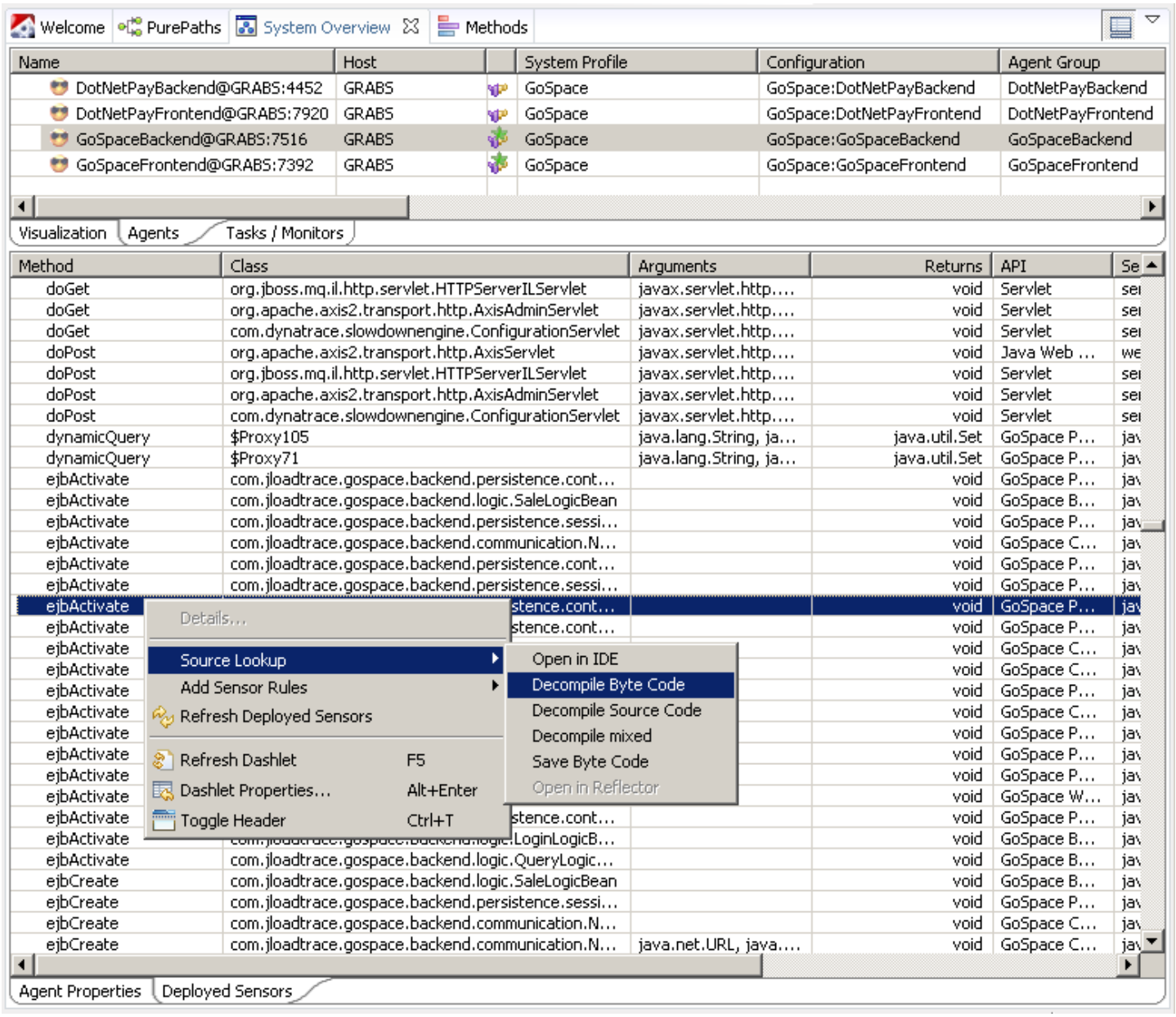
- Packages – You can save the byte code to a file, as described above.
- Classes and methods – You can perform all lookups.



Methods view

Deployed Sensor view

The **Deployed Sensor** view allows you to perform source code lookup on every method that has been deployed in the selected agent. The view is accessible via the **Systems Overview** view.



Deployed Sensor view

Source code viewer

For most of the Source Code Lookup commands, the **Source Code Viewer** shows the human-readable version of the .NET or Java byte code or source code. It first shows the complete decompiled class, then jumps to the first occurrence of the class or method name that you selected for the Source Code Lookup. The first occurrence of this text might not be the method or class that you are looking The viewer includes a search feature to navigate through all occurrences of the text.



Search feature in the source code viewer

Load testing and monitoring tool integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Integration with web load testing and monitoring tools

Watch and learn

Load Testing Made Easy with Dynatrace Webinar

For deep insight and visibility into your application's performance, you can integrate AppMon with the following load testing tools and other monitoring tools:

- Data Center Real User Monitoring (DC RUM)
- Synthetic Monitoring
- Enterprise Synthetic
- LoadRunner
- SilkPerformer

DC RUM integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

System Profile - Integration

Cockpit dashlet

Edit a System Profile

Watch and learn

Integrating with Data Center Real-User Monitoring

Data Center - Real User Monitoring (DC RUM) provides real-time access to information about performance and usage of key business applications. Based on measurement data provided by **network monitoring** devices, it monitors application performance, with the purpose of detecting, identifying and solving problems. Analysis gives insight into business application performance on transaction and operation level.

In combination with the in-depth analysis capabilities of AppMon PurePath technology, you can quickly find the root cause of problems in its code-level context.

To combine DC RUM data with AppMon, configure a connection to your DC RUM appliances in the **Integration** section of the **System Profile Preferences** dialog box. Enabling integration correlates data collected by DC RUM to AppMon PurePaths.

Supported releases

Integrating DC RUM with AppMon 2017 May requires the following:

- DC RUM version 12.4.10 or later.

- Flash installed on the client's browser. Drilldowns from the DC RUM Central Analysis Server (CAS) require Flash.

For detailed instructions on configuring the AppMon - DC RUM integration, see System Profile - Integration.

Synthetic Monitoring integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Synthetic test integration for Application Monitoring

UEM-PurePath integration for Synthetic Monitoring

Related pages

Synthetic test integration for Application Monitoring

UEM-PurePath integration for Synthetic Monitoring

W3C resource timing metrics

Perceived render time

AppMon integrates with Dynatrace Synthetic Monitoring and Keynote Synthetic monitoring to provide a single data access point to view and analyze application monitoring data for your synthetic test assets. Leverage your synthetic test base with APM details and the capability to drill down for performance application-related UEM data. The unified data repository lets your teams collaborate more efficiently than ever and gives analysts the ability to perform deep drilldown and accelerate problem resolution in a simple, easy-to-use interface.

AppMon provides the following types of Synthetic Monitoring integration.

Synthetic test integration for Application Monitoring

AppMon Web lets your teams get custom data views to support role-based information needs while streamlining collaboration. Business owners can get a high-level view of the state of the union, while analysts can see exactly how customers are using the application and drill down to contextual detail for quick problem solving. In the Dynatrace Portal, you can find specific site visits within tests, see all the channels accessed, and choose specific tests to analyze availability and response data.

See Synthetic Test Integration for Application Monitoring for more information on configuring and using this integration.

UEM-PurePath integration for Synthetic Monitoring

You can drill down directly from the Dynatrace Portal into the AppMon Webstart Client, and leverage PurePath and UEM data for your synthetic tests to facilitate issue resolution.

See UEM-PurePath Integration For Synthetic Monitoring for more information on configuring and using this integration.

Synthetic test integration for Application Monitoring

On this page

Overview

Related pages

Synthetic Monitoring configuration

Tile types

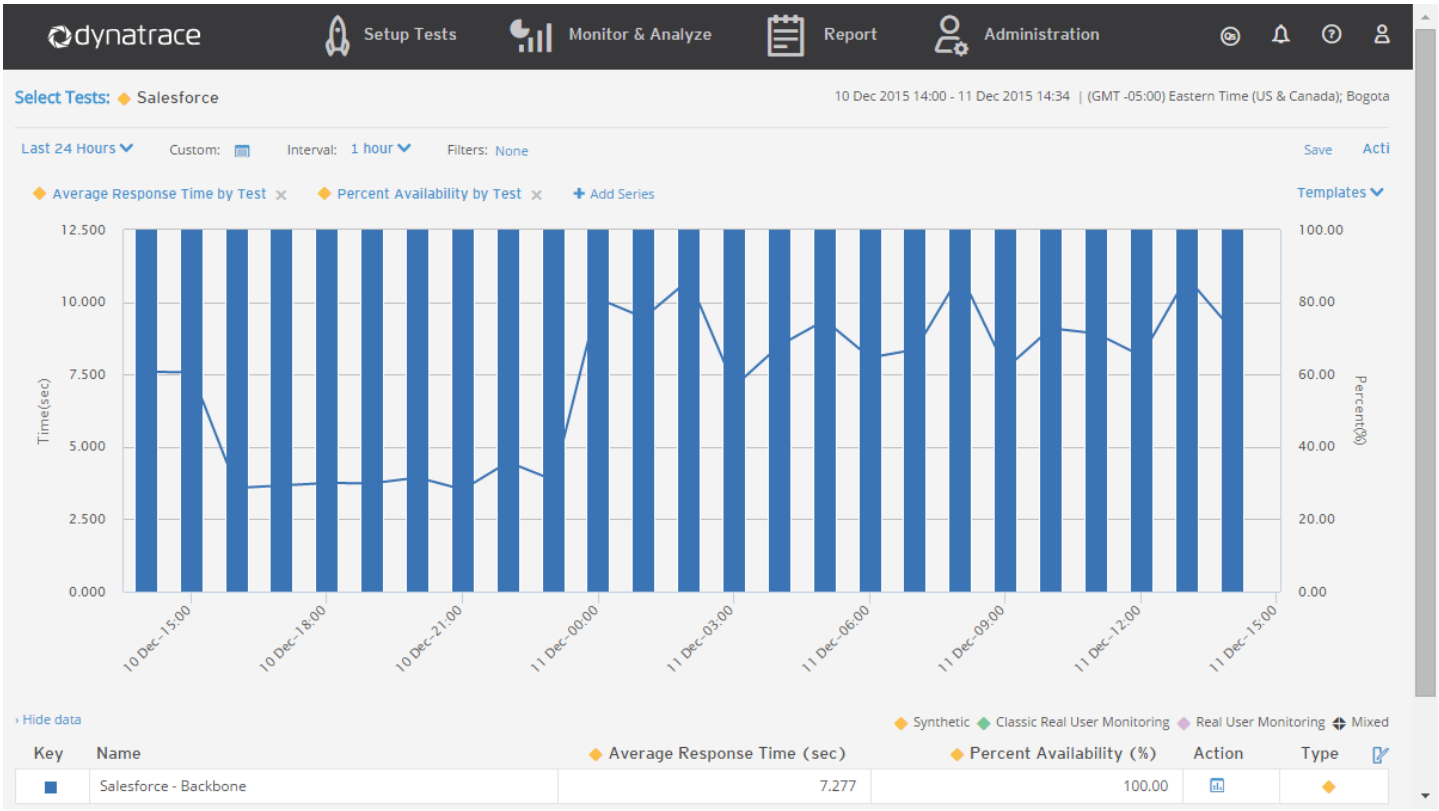
W3C resource timing metrics

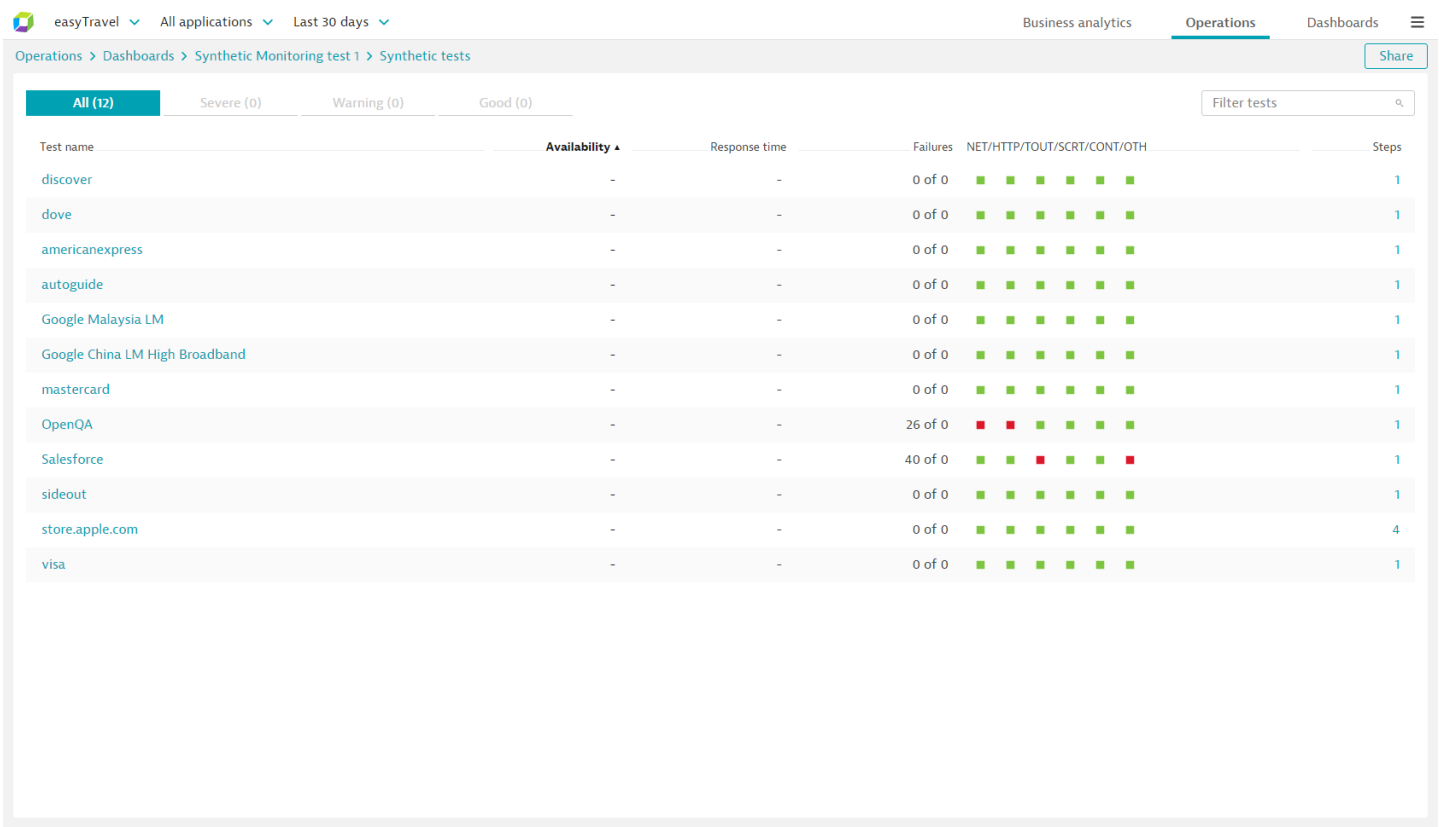
Perceived render time

This page describes how to integrate synthetic tests into AppMon Web. This integration lets you use AppMon Web to analyze the performance details of synthetic tests that are configured and scheduled in the Dynatrace Portal or Keynote Service Center (KSC) (viewed in the MyKeynote portal) for your instrumented web applications.

Overview

AppMon integrates with both Dynatrace Synthetic Monitoring and Keynote Synthetic Monitoring. This integration lets you use AppMon Web to analyze the performance details of synthetic tests that are configured and scheduled in the Dynatrace Portal or Keynote Service Center (KSC) (viewed in the MyKeynote portal) for your instrumented web applications. In addition, you can select a synthetic test as it appears in AppMon Web, and then open an interactive chart in either portal to view details about availability, response times, and failure rates for accurate root cause analysis.


dynatrace
Setup Tests
Monitor & Analyze
Report
Administration
Select Tests: Salesforce
10 Dec 2015 14:00 - 11 Dec 2015 14:34 | (GMT -05:00) Eastern Time (US & Canada); Bogota
Last 24 Hours
Custom:
Interval: 1 hour
Filters: None
Save
Acti
Average Response Time by Test
Percent Availability by Test
Add Series
Templates
12.500
10.000
7.500
5.000
2.500
0.000
Time(sec)
100.00
80.00
60.00
40.00
20.00
0.00
Percent(%)
10 Dec-15:00
10 Dec-18:00
10 Dec-21:00
11 Dec-00:00
11 Dec-03:00
11 Dec-06:00
11 Dec-09:00
11 Dec-12:00
11 Dec-15:00
Hide data
Synthetic
Classic Real User Monitoring
Real User Monitoring
Mixed
Key
Name
Average Response Time (sec)
Percent Availability (%)
Action
Type
Salesforce - Backbone
7.277
100.00


easyTravel
All applications
Last 30 days
Business analytics
Operations
Dashboards
Operations > Dashboards > Synthetic Monitoring test 1 > Synthetic tests
Share
All (12)
Severe (0)
Warning (0)
Good (0)
Filter tests
Test name
Availability
Response time
Failures
NET/HTTP/TOUT/SCRT/CONT/OTH
Steps
discover 0 of 0 1
dove 0 of 0 1
americanexpress 0 of 0 1
autoguide 0 of 0 1
Google Malaysia LM 0 of 0 1
Google China LM High Broadband 0 of 0 1
mastercard 0 of 0 1
OpenQA 26 of 0 1
Salesforce 40 of 0 1
sideout 0 of 0 1
store.apple.com 0 of 0 4
visa 0 of 0 1

Note

Note : An alternative integration allows you to drill down from MyKeynote to view PurePath data in the AppMon Client. See Keynote PurePath Measurements for details.

Configuring synthetic test integration for Application Monitoring

You must configure the credentials used to authenticate to the Dynatrace Portal or MyKeynote portal before you can see a Synthetic Monitoring test list in the AppMon Web Synthetic Monitoring tile.

See Synthetic Monitoring Configuration for details.

Viewing data in AppMon Web

Once the Synthetic Monitoring integration is configured and you have verified a connection to the Dynatrace Portal or MyKeynote, you can view the availability, response times, and failure rates of synthetic tests in AppMon Web. To do this:

1. Create a custom dashboard and add the Synthetic tile.
2. Select synthetic tests for which you want to view availability, response times, and failure rate data.
3. Analyze the performance details of the synthetic tests in AppMon Web.

Creating a custom dashboard and adding a Synthetic tile

1. In AppMon Web, click **Dashboards > Add a Dashboard**. A new dashboard appears with a default tile.
2. Click **Edit Tile**. The Preview and Tile Configurator panes display.
3. Select the Synthetic Tests tile in the Tile Configurator pane. When connected to the appropriate Synthetic Monitoring server, the **Add test** button appears. See the Synthetic Tests tile section in Tile Types for more information on the Synthetic Tests tile type.

Selecting synthetic tests

1. Click the **Add test** button. A list of synthetic tests available for the credentials used to log on to the Synthetic Monitoring server appears in the Tile Configurator pane.
2. Select the tests you want to add to the tile. These are the synthetic tests for which you want to analyze performance details.
3. Click **Done** to add the selected tests to the tile. The selected tests also appear in the Tile Configurator pane, allowing you to remove selected tests as needed.
4. Click **Done**. The tile appears in the dashboard.

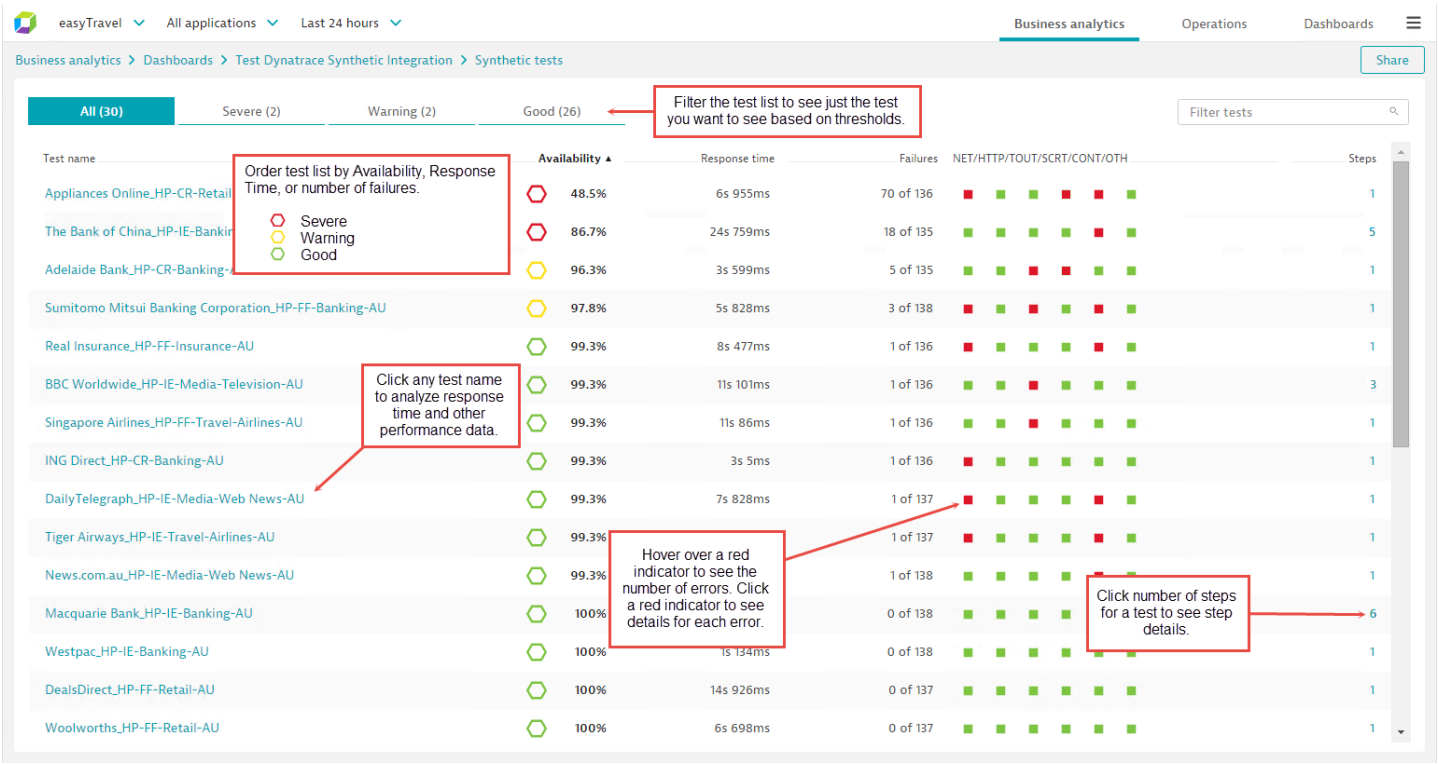
> **Tip**
>
> **Tip:** You can edit the list of selected synthetic tests at any time. Select the tile and click the **Edit** button in the AppMon Web toolbar to add and delete tests. Click the **Done** button in the toolbar after editing the list.

Analyzing Performance Details of Synthetic Tests in AppMon Web

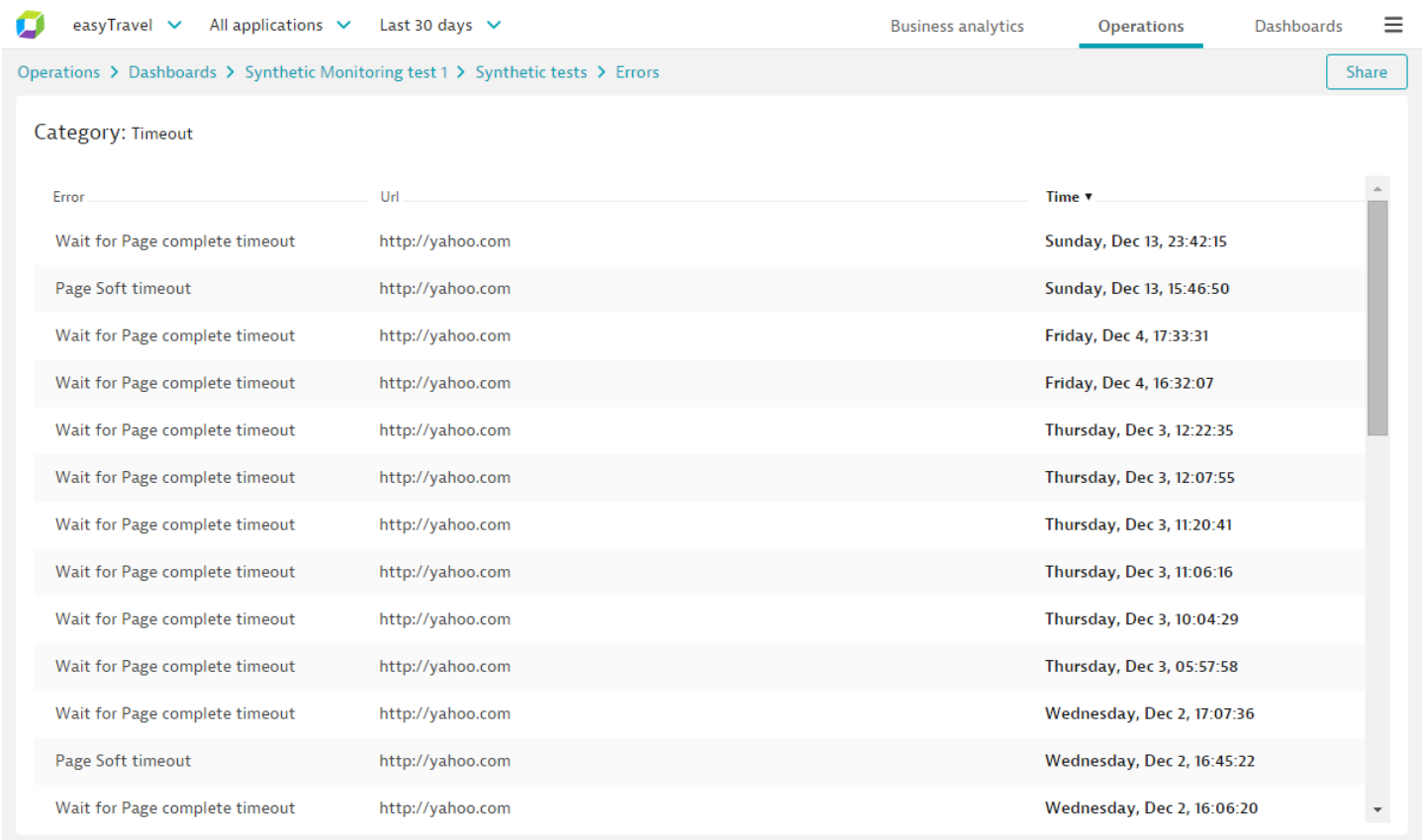
Once you have a Synthetic Tests tile defined with selected tests in AppMon Web, you can view and analyze the performance data of those tests, drill down to view performance details, and perform diagnosis using the Dynatrace Portal or MyKeynote portal.

Click on the tile to display details about the selected tests in the tile. Each selected test shows:

- The name of the test
- Availability
- Response time
- Failures
- Indicators for network, HTTP, timeout, script, contributor group, and other errors. A red indicator signifies errors. You can hover over any red indicator to see the number of errors for the error category. Click a red indicator to see a detail list of all the errors in the test for the associated error category. Each error listed shows information on the type of error, the URL where the error occurred, and the time the error occurred.
- The number of steps in the test. You can click the number in the steps column to view a breakdown of errors by category for each step.



Click on any red indicator light to view details for the error.



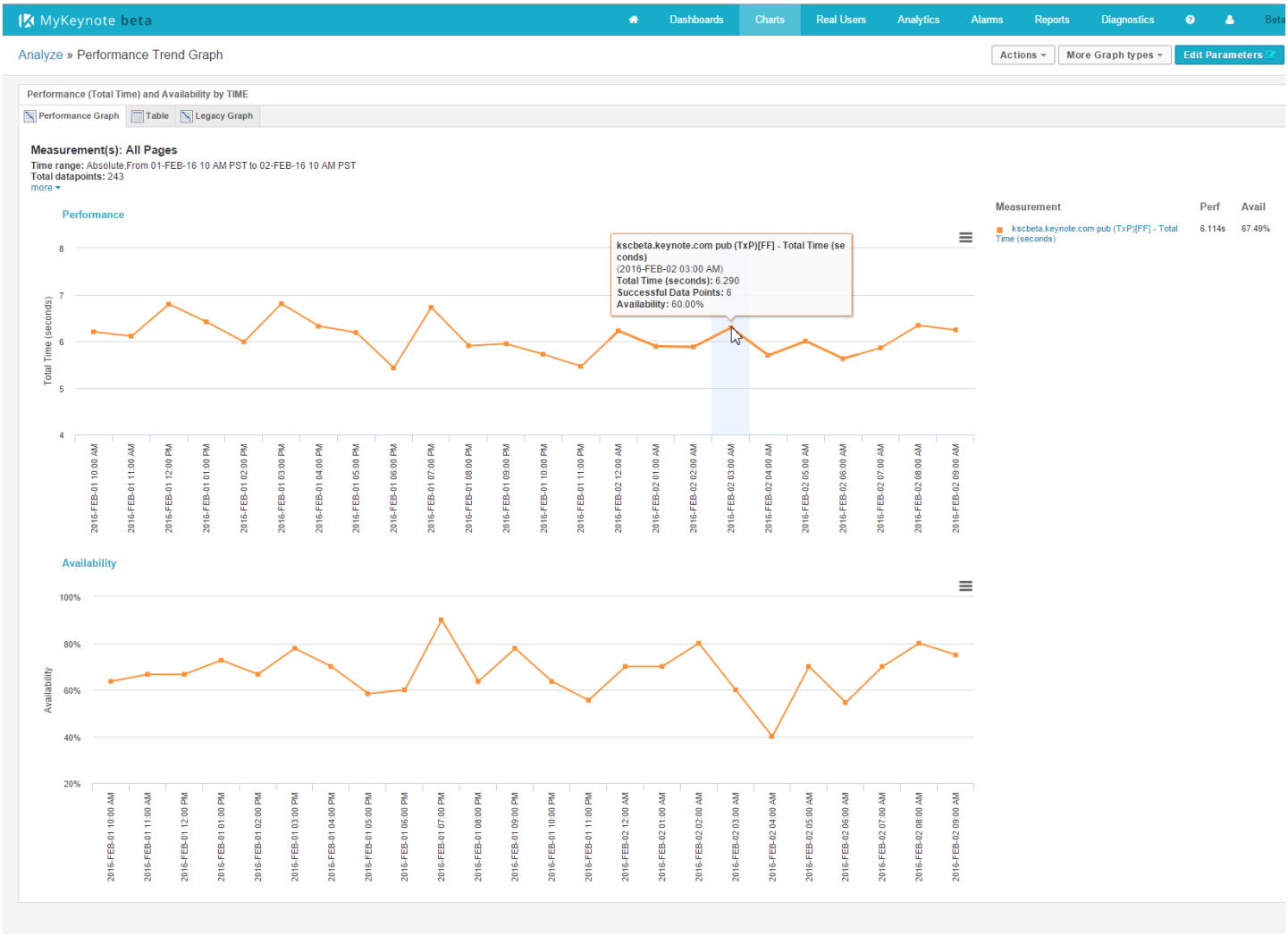
Clicking on any test in the list shows detailed Application Monitoring performance data for the selected test, such as average response time or percent availability. You can chart your data in any meaningful charting format. Hover over data points to see detailed information for the selected data point and drill down from the data point to perform root cause analysis and other Synthetic Monitoring diagnoses.

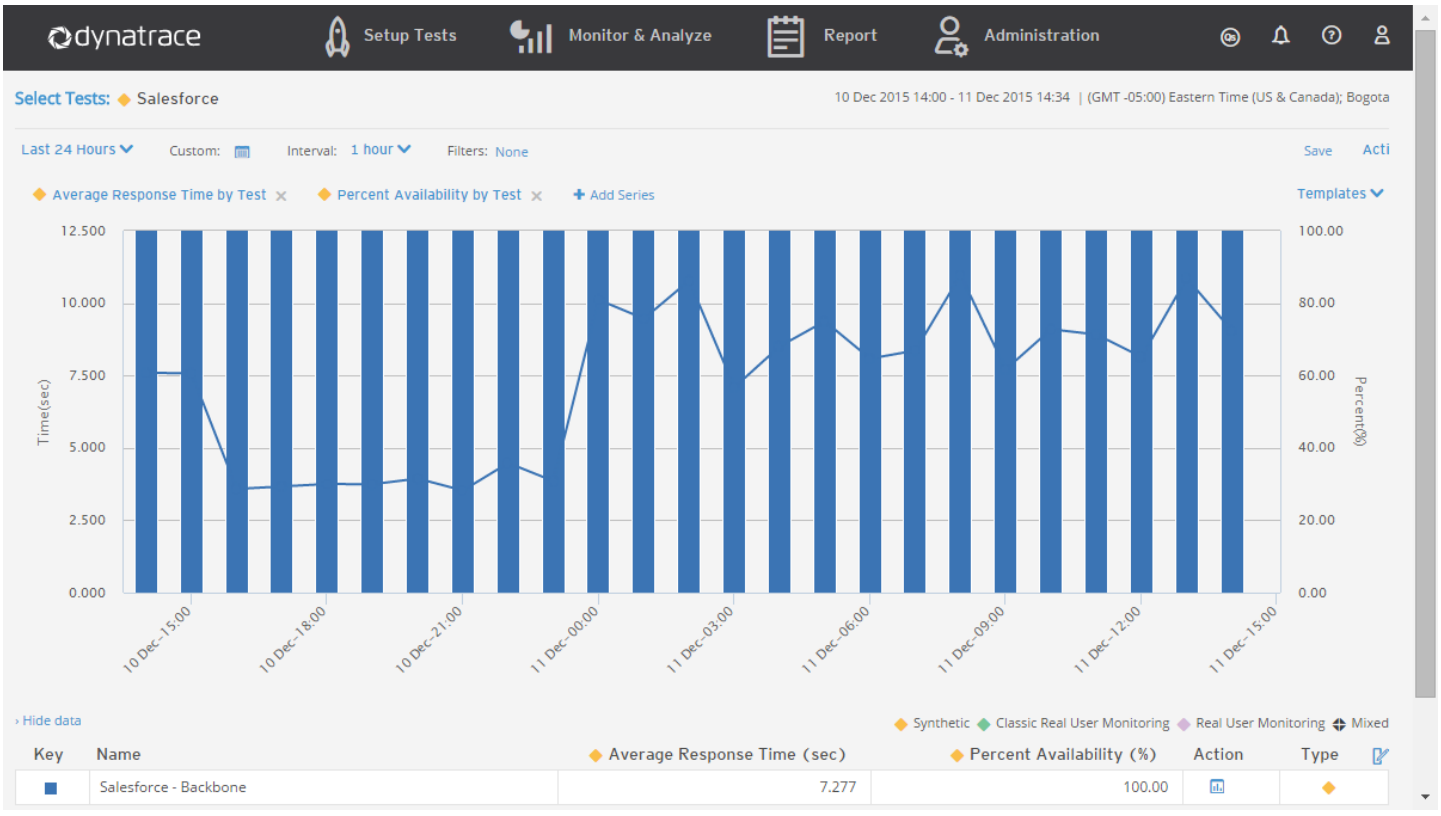
Click a test in the Synthetic Test list to drill down to the Interactive Chart in the Dynatrace Portal or MyKeynote portal, as shown in the following figures. By default, the chart shows response time and availability. You can reconfigure the chart to show data for other metrics, or drill down for more detailed information.

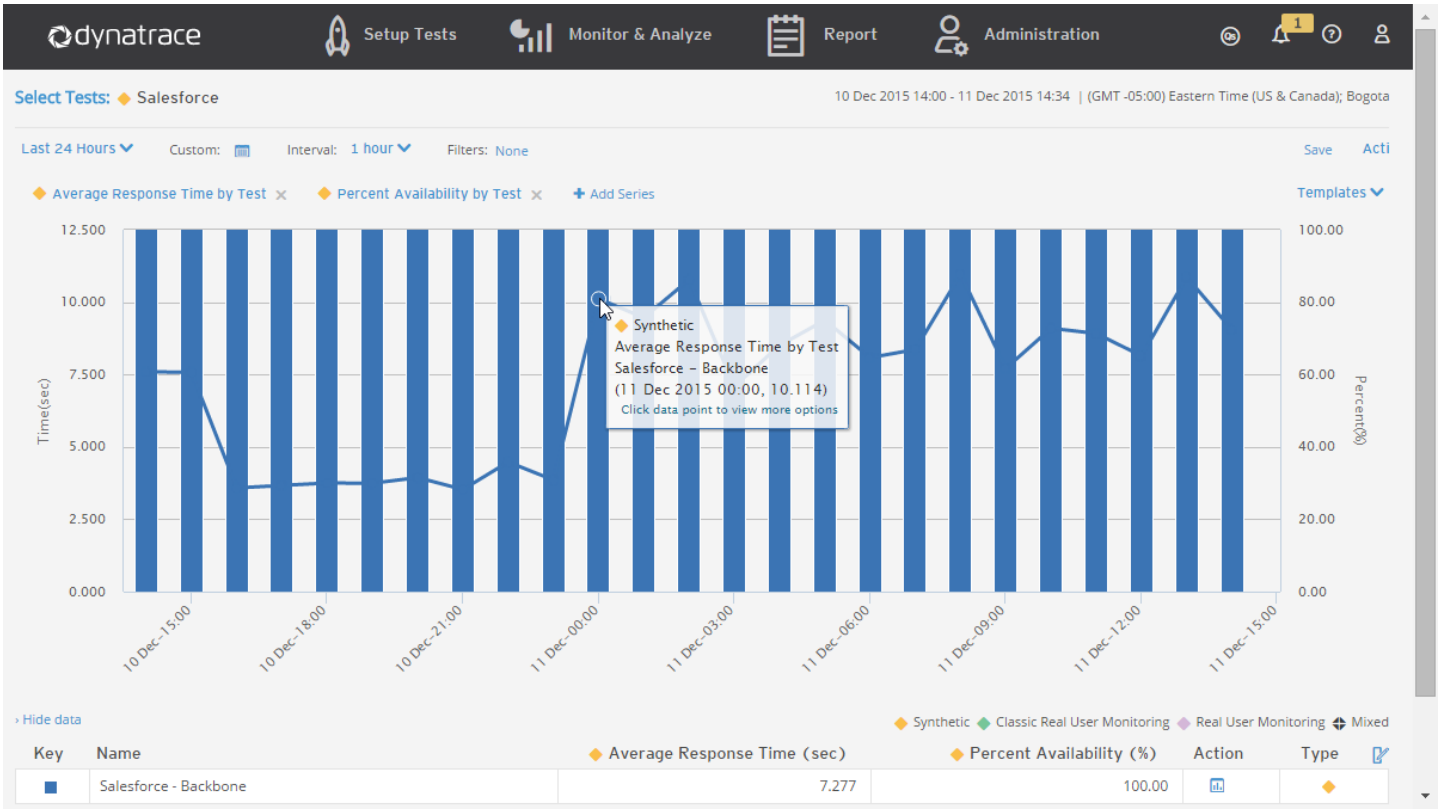
For details of charting the data in the Dynatrace Portal, see Interactive Charts in the Dynatrace Synthetic documentation.

Tip

Tip: When drilling down to an interactive chart in the MyKeynote portal, the MyKeynote Sign In page may appear for the initial drill down. Enter a valid Username or Email address and Password, and click **Sign In**. Refresh the screen after the sign in If the Performance Trend Graph does not appear.


MyKeynote beta
Dashboards
Charts
Real Users
Analytics
Alarms
Reports
Diagnostics
Analyze » Performance Trend Graph
Actions
More Graph types
Edit Parameters
Performance (Total Time) and Availability by TIME
Performance Graph
Table
Legacy Graph
Measurement(s): All Pages
Time range: Absolute,From 01-FEB-16 10 AM PST to 02-FEB-16 10 AM PST
Total datapoints: 243
more
Performance
kscbeta.keynote.com pub (TxP)[FF] - Total Time (seconds)
(2016-FEB-02 03:00 AM)
Total Time (seconds): 6.290
Successful Data Points: 6
Availability: 60.00%
Measurement
Perf
Avail
kscbeta.keynote.com pub (TxP)[FF] - Total Time (seconds)
6.114s
67.49%
Total Time (seconds)
Availability
Availability


dynatrace
Setup Tests
Monitor & Analyze
Report
Administration
Select Tests: Salesforce
10 Dec 2015 14:00 - 11 Dec 2015 14:34 | (GMT -05:00) Eastern Time (US & Canada); Bogota
Last 24 Hours
Custom:
Interval: 1 hour
Filters: None
Save
Average Response Time by Test
Percent Availability by Test
Add Series
Templates
Time(sec)
Percent(%)
Hide data
Synthetic
Classic Real User Monitoring
Real User Monitoring
Mixed
Key
Name
Average Response Time (sec)
Percent Availability (%)
Action
Type
Salesforce - Backbone
7.277
100.00



Troubleshooting

If you see the message `Invalid data` in the tile or in the Tile Configurator pane, the instance of the AppMon Webstart Client used to configure Synthetic Monitoring integration is not the correct instance. Make sure you use the AppMon Webstart Client created by clicking the **Use webstart client** button in the Tile Configurator pane when enabling the integration.

See Tile Types for more information.

UEM-PurePath integration for Synthetic Monitoring

On this page

Viewing Data in AppMon

AppMon drilldowns - landing dashboards

Troubleshooting

Related pages

W3C resource timing metrics

Perceived render time

Configuring account settings

Synthetic test integration for Application Monitoring

Synthetic Monitoring configuration

Watch and learn

Integrating with Dynatrace Synthetic Monitoring

The UEM-PurePath Integration for Synthetic Monitoring lets you drill down directly from the Dynatrace Portal into Application Monitoring, showing the PurePath and UEM data related to your synthetic tests in the AppMon Client.

To use the UEM-Purepath integration for Synthetic Monitoring, you need to change a few settings on both sides of the integration:

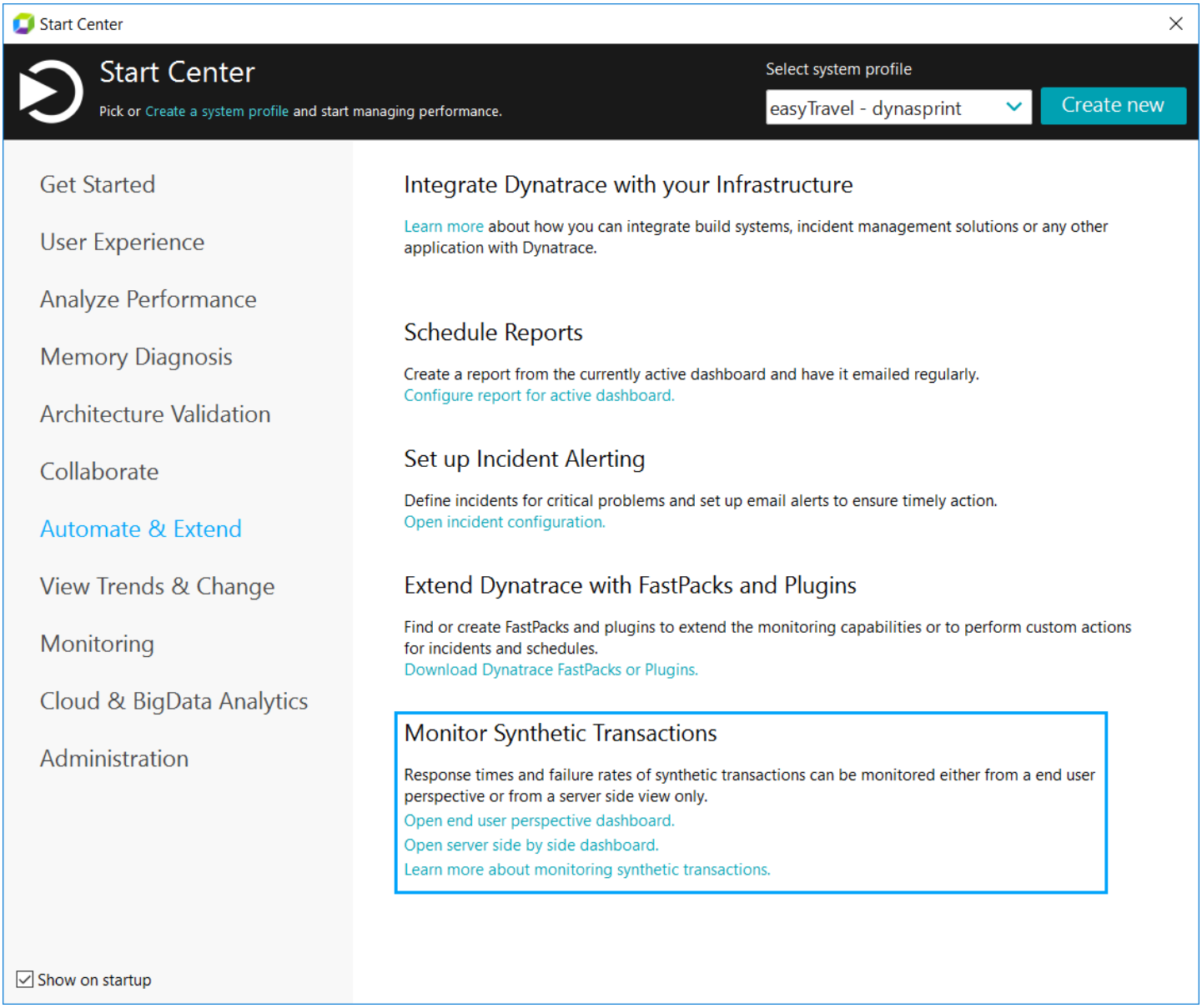
- Configure the AppMon Server integration with the Dynatrace Portal. See Enabling UEM-PurePath Integration for Synthetic Monitoring to learn how. In addition, you may need to verify the security rules related to synthetic monitoring traffic.

- In the Dynatrace Portal, configure the **Dynatyrace AppMon** settings in the **Account Details** page to define the AppMon Server for integration. See Account Settings in the Synthetic Monitoring help for details.

> **Note**
>
> **Note**: You can also set up a similar PurePath integration with Keynote Synthetic Monitoring. See Keynote PurePath Measurements for more details.

Viewing Data in AppMon

To enable the integration with Synthetic Monitoring, go to the **Start Center > Monitor Applications > Monitor Synthetic Transactions** and select the **end user perspective** or **server-side** links that open monitoring dashboards. When using real-browser replay in Dynatrace Synthetic Monitoring, open the end user perspective view. Otherwise, open the server-side view.



AppMon Start Center - Monitor Synthetic Transactions

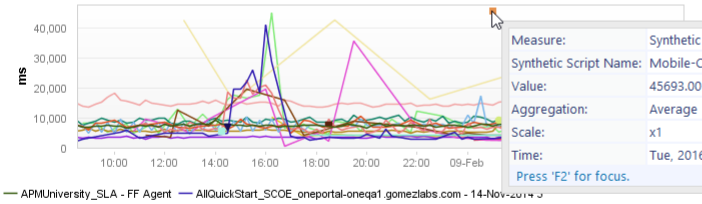
The first time you select this feature, additional measures and business transactions are created that are needed for that integration. AppMon captures Synthetic Monitoring requests and shows details on response times for every monitored transaction in the dashboard.

Synthetic Monitoring - End User Perspective

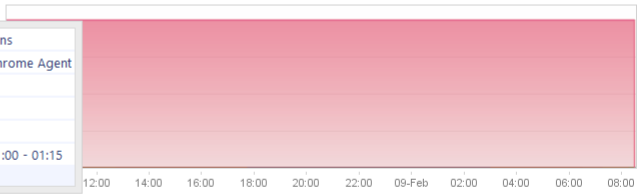
If you have scheduled Synthetic Monitoring tests (Backbone, Last Mile or Private Peer), then these requests will be automatically detected. Learn how to configure Synthetic Monitoring to allow Dynatrace to split transactions that are generated by different tests.
Use the drill down context menu (e.g. Page Actions or Errors) on one of the displayed tests below to start analysis of a test.

Business Transaction	from yesterday 08:32 to today 08:32		last week	
Name	Count	Load Time [avg]	Count	Load Time [avg]
Synthetic Gomez Page Actions (split by Gomez Script Name)	8,844	7896.20 ms	73,105	6597.61 ms
Synthetic Page Actions (split by Synthetic Script Name)	9,855	8652.14 ms	81,479	7281.41 ms
AllQuickStart_SCOE_oneportal-oneqa1.gomezlabs.com - 14-Nov-2014 3	109	5782.18 ms	338	3550.75 ms
AllQuickStart_SCOE_oneportal-oneqa1.gomezlabs.com - 17-Oct-2014 5:28:46 - 2 - FF Agent	1	7262.00 ms	361	3745.60 ms
AllQuickStart_SCOE_oneportal-oneqa1.gomezlabs.com - 17-Oct-2014 5	10	4187.60 ms	20	4413.00 ms
AllQuickStart_SCOE_oneportal-oneqa1.gomezlabs.com - 18-Sep-2014 6:37:17 - 3 - FF Agent	76	8040.01 ms	39	6115.15 ms
AllQuickStart_SCOE_oneportal-oneqa1.gomezlabs.com - 18-Sep-2014 8:49:28 - 1 - FF Agent			89	3515.36 ms
AllQuickStart_SCOE_oneportal-oneqa1.gomezlabs.com - 19-Sep-2014 7:32:02 - 1 - FF Agent			198	3579.13 ms
AllQuickStart_SCOE_oneportal-oneqa1.gomezlabs.com - 24-Sep-2014 2:59:41 - 1 - FF Agent			166	3563.80 ms

Page Load Times by Test Name



Failure Rates by Test Name



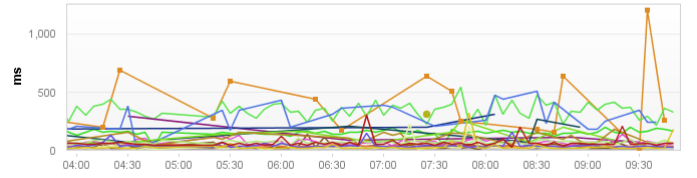
Synthetic Monitoring - Server Side View

If you have scheduled Synthetic Monitoring tests (Backbone, Last Mile or Private Peer), then these requests will be automatically detected. Learn how to configure Synthetic Monitoring to allow Dynatrace to split transactions that are generated by different tests.
Use the drill down context menu (e.g. Response Time Hotspots or Errors) on one of the displayed tests below to start analysis of a test.

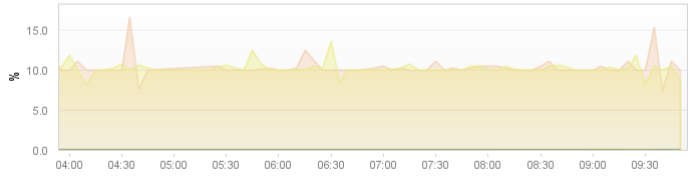
When comparing the response times displayed in this dashboard with Synthetic Monitoring results, you should consider that a single page load might consist of several server side requests, so response times and request counters are not comparable on a one to one base.

Business Transaction	last 6 hours		last week	
Name	Count	PurePath Response Time [avg]	Count	PurePath Response Time [avg]
APMUniversity_SLA - FF Agent	2,377	68.87 ms	76,155	41.80 ms
APMU - North America Chrome	1,459	43.90 ms	281	32.16 ms
APMU - North America Firefox	1,632	27.95 ms	273	15.34 ms
Automation Test 223			17	155.26 ms
Automation Test 252			17	107.41 ms
Automation Test 319			13	67.52 ms
Community_GSA - FF Agent	1,584	297.95 ms	51,263	165.61 ms
copy of monid 25021541-http://community.dynatrace.com	164	468.83 ms	5,052	161.68 ms
DataMigration_smoke_wh_PLM - Chrome Agent	4	306.74 ms	91	170.78 ms

Response Times by Test Name



Failure Rates by Test Name

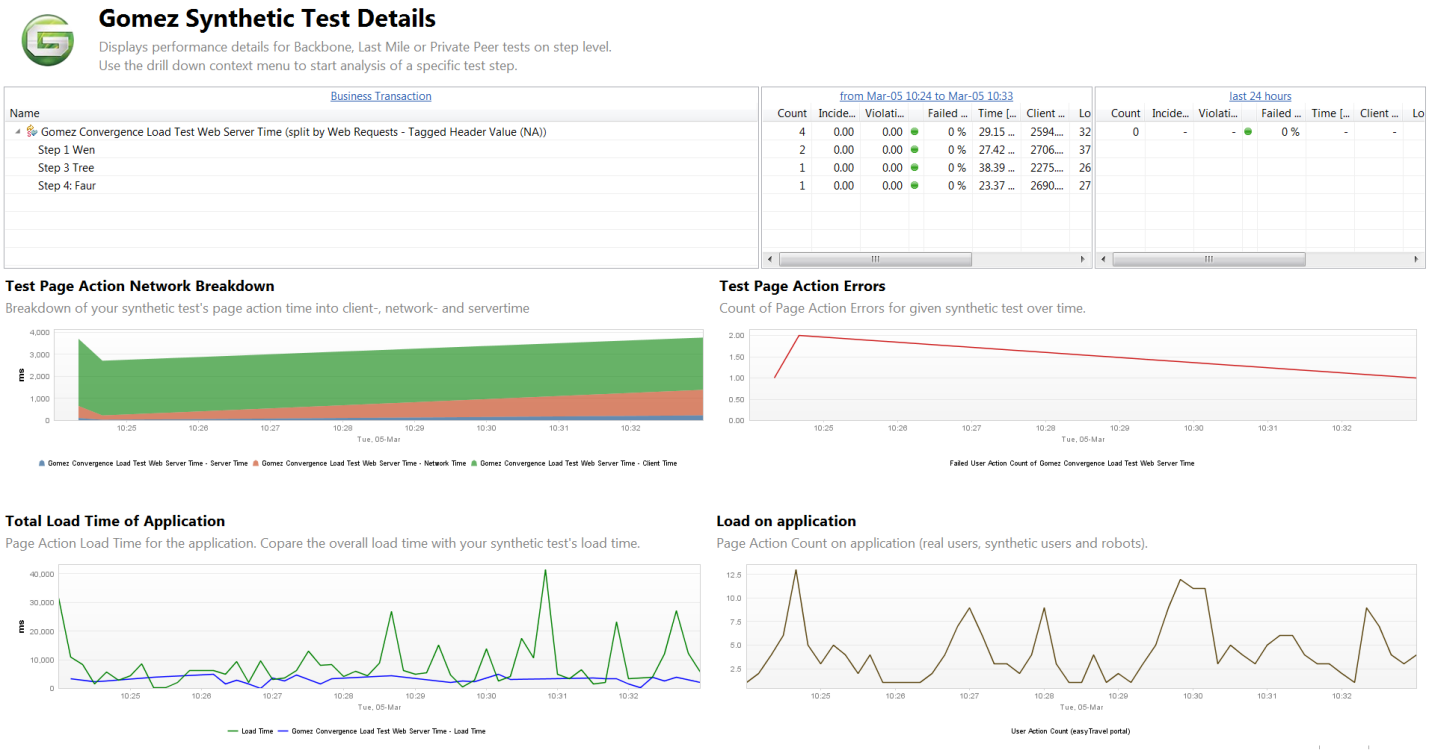


AppMon drilldowns - landing dashboards

The UEM-PurePath Integration for Synthetic Monitoring provides a variety of built-in dashboards to visualize the data of both your synthetic tests and your application's UEM key metrics. The following dashboards are accessible via direct drill-downs from the Dynatrace Portal.

Synthetic test details

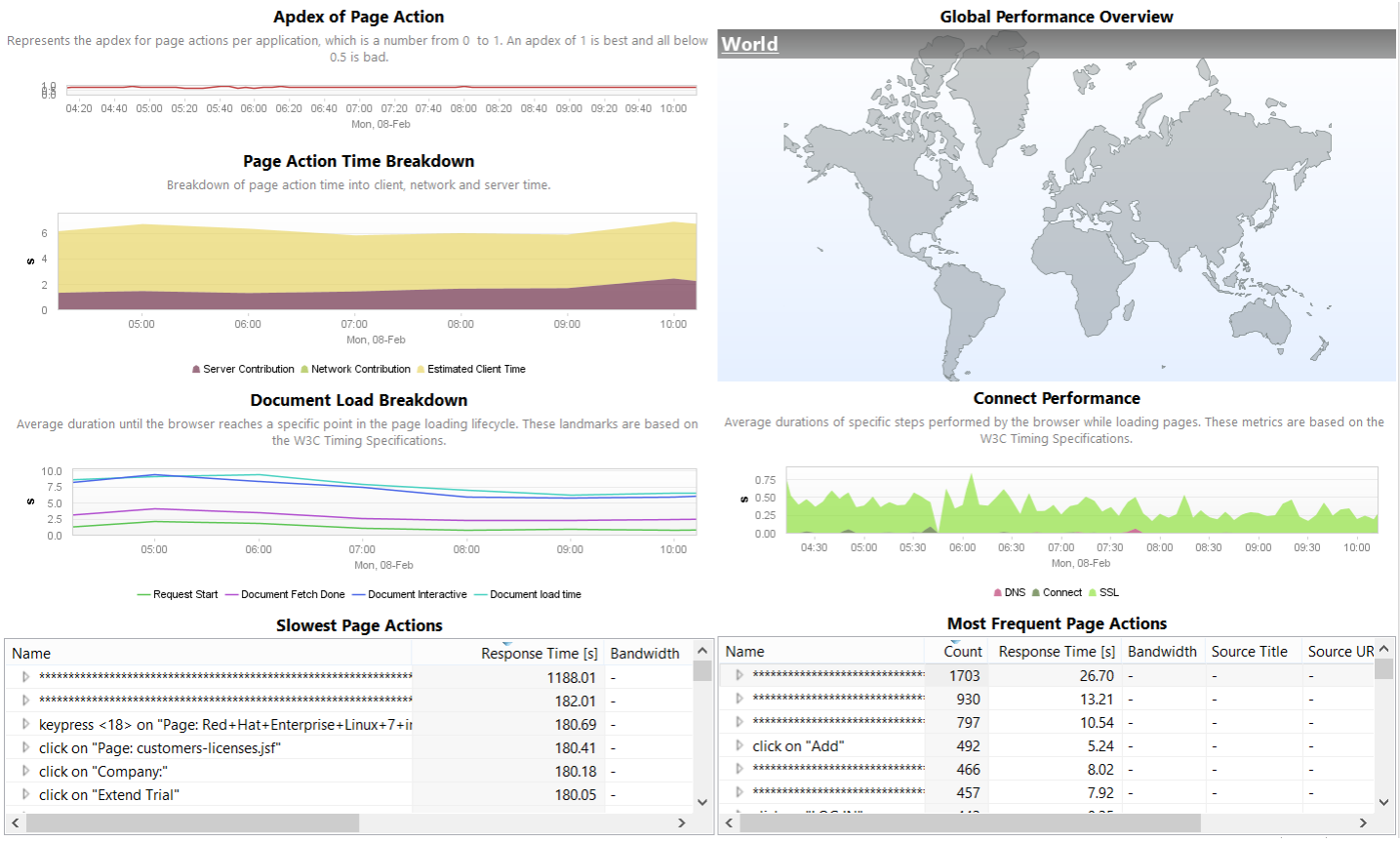
This dashboard shows the main steps of your synthetic test, the network breakdown of the page actions, any errors caused by the synthetic test, and a direct comparison between the total load time caused by the synthetic test and the overall load time of the application.



Synthetic Test Details Dashboard

Page performance

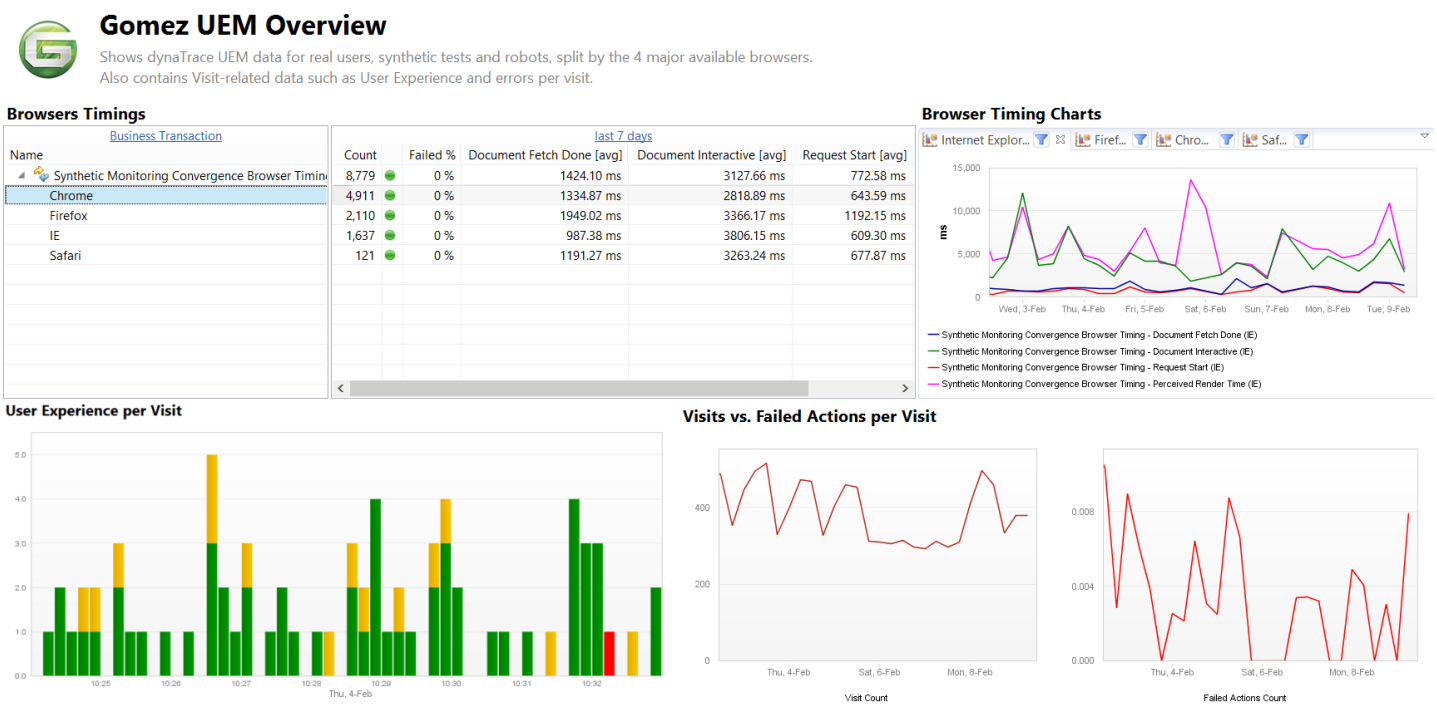
This dashboard provides a common overview of your application's page action and its performance layer breakdown.



Page Performance Dashboard

Synthetic UEM overview

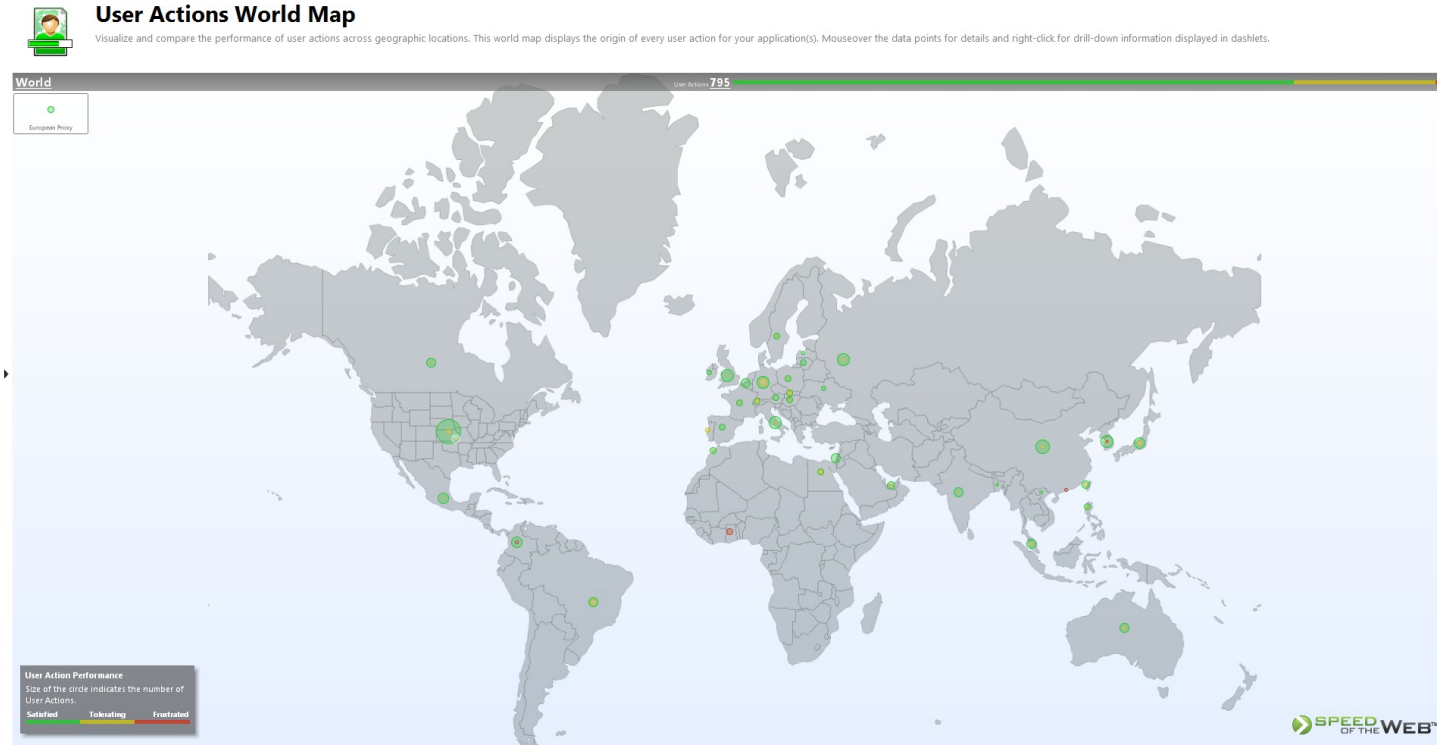
This dashboard provides an overview of the UEM data collected by the Dynatrace Portal integration, as well as the browser timing metrics of the four major browser families: Internet Explorer, Firefox, Chrome, and Safari.



Synthetic UEM Overview Dashboard

User Actions world map

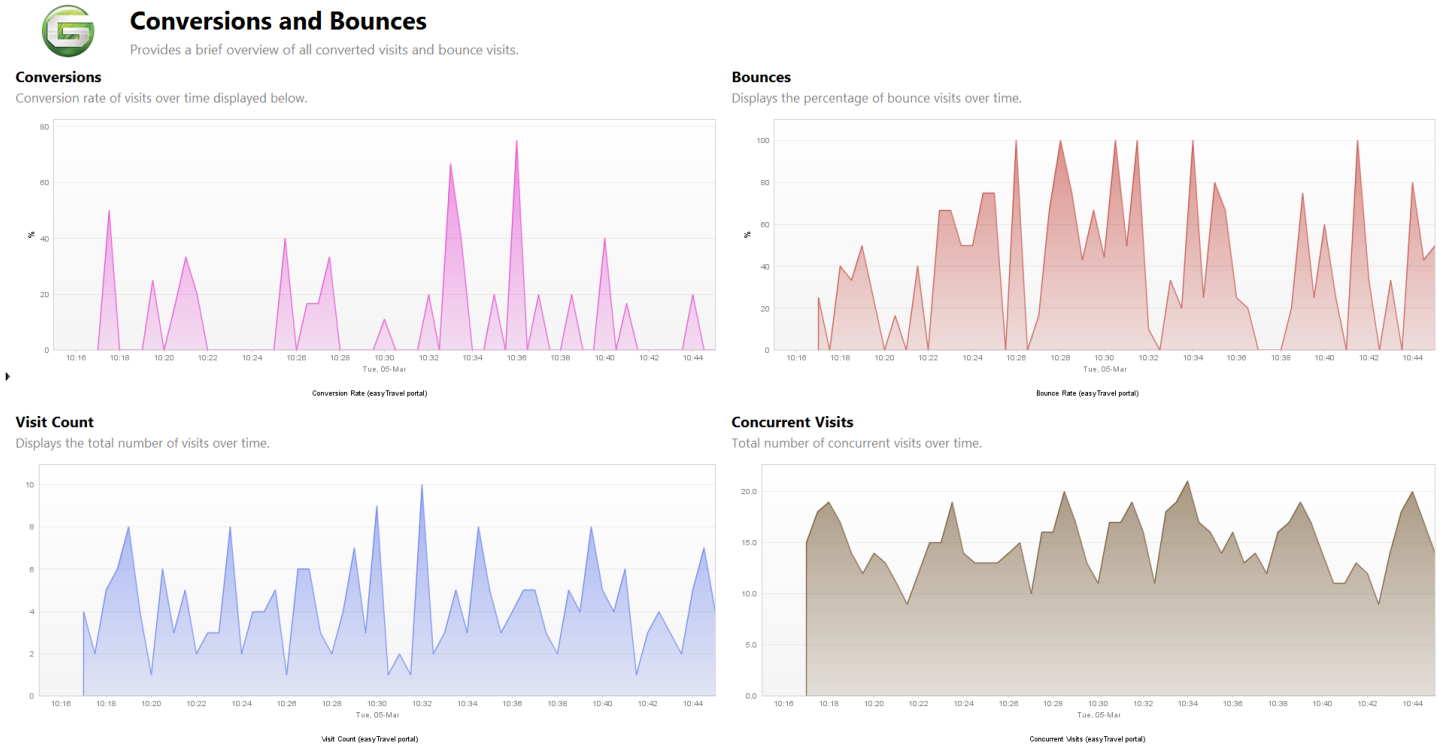
This map visualizes and compares the performance of the user actions of your application across geographic regions.



User Actions world map

Conversions and bounces

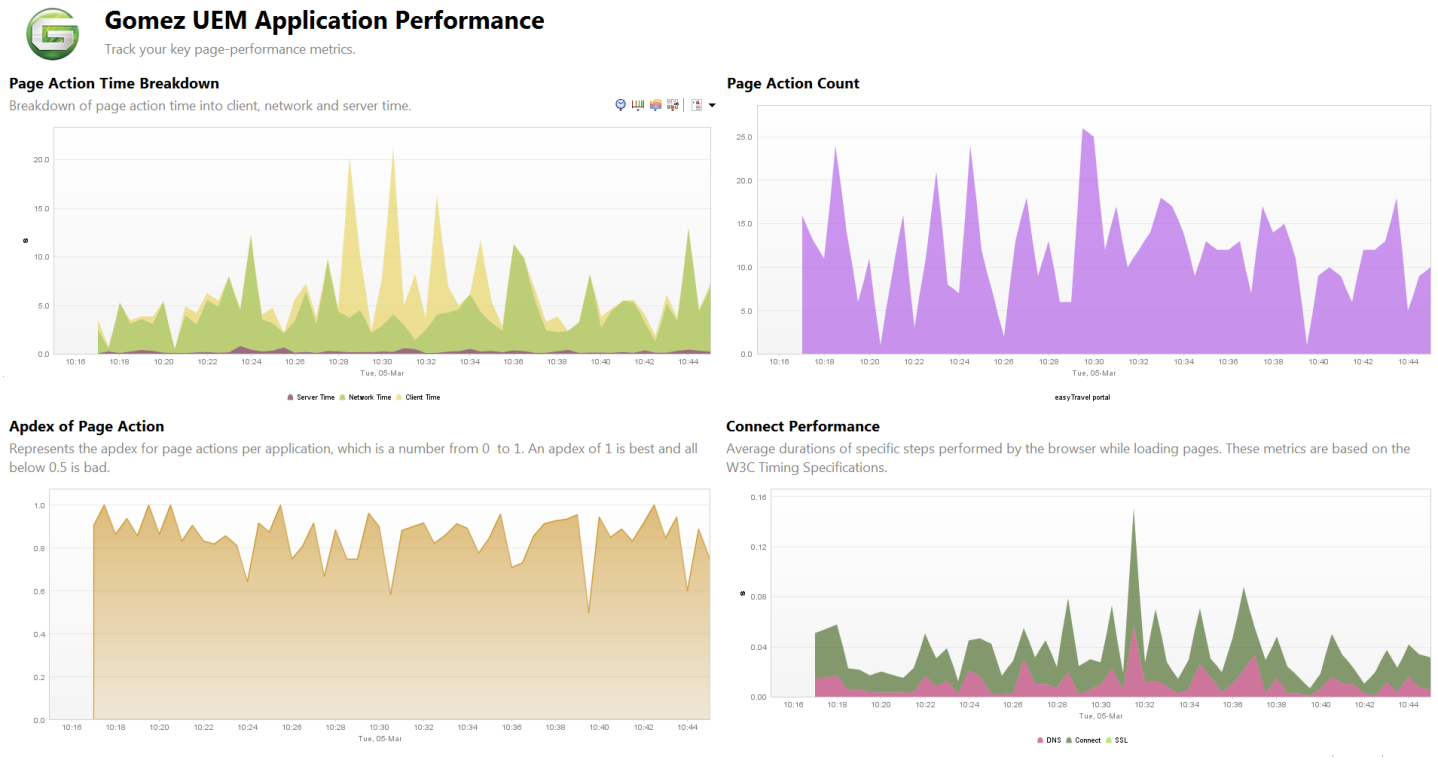
This dashboard provides a brief overview of all converted and bounced visits.



Conversions and Bounces dashboard

Synthetic UEM application performance

This dashboard visualizes the key performance metrics of your application.



Synthetic UEM Application Performance Dashboard

Troubleshooting

Apdex distribution unavailable / incomplete in JSON file

The built-in Business Transaction **Visits by User Experience and Country** is used to calculate the Apdex distribution of the available visits. If your visit-related data seem to be inconsistent or incomplete, verify the following structure of the business transaction:

- Name: Visits by User Experience and Country
- Calculate Results
  - Visit: count, aggregation: sum
- Split Results
  - Country of Visits - metric: Visits - Geo Information
  - User Experience of Visits - metric: Visits - User Experience
  - Application

Enterprise Synthetic Integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Configuring the Dynatrace Application Monitoring and Enterprise Synthetic integration

The integration with Enterprise Synthetic allows you to use AppMon to precisely analyze response times and failure rates of synthetic transactions produced by Enterprise Synthetic. Enabling the AppMon integration in Enterprise Synthetic adds contextual information about each synthetic transaction in the HTTP header string of each transaction request. For example, the script name of the transaction in Enterprise Synthetic can be viewed in the **Timer Name** column on the AppMon **System Profile <name> > Diagnose Performance > Tagged Web Requests** dashlet.

SilkPerformer integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Segue/Borland SilkPerformer and SilkCentral Performance Manager integration

Downloads

SilkPerformer plugin

Segue/Borland SilkPerformer and SilkCentral Performance Manager integration

AppMon can be used with Segue/Borland SilkPerformer and SilkCentral Performance Manager version later than 2017 May. If you perform load tests using the SilkPerformer workbench, you can diagnose web services — for example, SilkPerformer SOAP, Java/axis, and .Net SOAP stacks — with AppMon.

Borland SilkPerformer 2006, 2006 R2, or 2007

SilkPerformer is provided with the AppMon PlugIn. To activate it, select the **Enable in** option in the **System Settings** dialog box of your SilkPerformer Installation.

To use a newer version of the SilkPerformer AppMon PlugIn than the one that is shipped with your AppMon version, copy `AppMon-silkperformer-plugin-<version>.zip` to `<SilkPerformerHome>\Working\Plugins` and unzip it.

For further information, refer to the SilkPerformer documentation.

> **Note**
>
> If you are running the AppMon Client on Windows Vista or Windows 2008, be sure that the AppMon Client is running as administrator.

Segue/Borland SilkPerformer v. 7.3.1

To integrate AppMon with SilkPerformer version 7.3.1, add the `WebSetDiagnostics` function to the `TInit` transaction of your SilkPerformer script. An example integration looks like the following:

```
benchmark SilkPerformerRecorder

use "WebAPI.bdh"

dcluser
user
VUser
transactions
TInit : begin;
TMain : 1;

var

dclrand

dcltrans
transaction TInit
begin
WebSetDiagnostics(WEB_DIAG_dynaTraceDiagnostics);
end TInit;
...
```

Segue SilkPerformer v. 7.2 Integration

If you are using SilkPerformer 7.2, load test transactions can be tagged and traced with AppMon Diagnostics. Use the WebSetHttpTag function in the SilkPerformer Workbench (Model Script) and add the call for the WebSetHttpTag function in the SilkPerformer TInit transaction. The following tags for `dynaTraceDiagnostics` are supported:

- `WEB_TAG_FLAG_Timer`
- `WEB_TAG_FLAG_RequestId`
- `WEB_TAG_FLAG_UserId`
- `WEB_TAG_FLAG_Time`
- `WEB_TAG_FLAG_PageContext`

For additional help with using `WebSetHttpTag`, refer to the SilkPerformer Online Help.

To use the SilkPerformer Workbench with AppMon, modify the Model Script file (`*.bdf`) according to the highlighted parts below.

```
//--------------------------------------------------------------------\-
// Recorded 12/23/2005 by SilkPerformer Recorder v7.2.0.2553
//--------------------------------------------------------------------\-

benchmark SilkPerformerRecorder

use "WebAPI.bdh"

dcluser
user
VUser
transactions
TInit : begin;
TMain : 1;

var

dclrand

dcltrans
transaction TInit
begin
WebSetHttpTag("Dynatrace", WEB_TAG_FLAG_Timer | WEB_TAG_FLAG_RequestId | WEB_TAG_FLAG_Tim
end TInit;
...
```

LoadRunner integration

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

LoadRunner script converter

Modifying LoadRunner scripts for AppMon

Conversion result

LoadRunner script converter

AppMon can be used with HP LoadRunner. HTTP Requests generated by LoadRunner´s Virtual User Generator can be evaluated by the AppMon Server. The **LoadRunner Script Converter** in the AppMon Client modifies LoadRunner scripts to AppMon header-aware scripts.

Modifying LoadRunner scripts for AppMon

Step 1: Record a LoadRunner Script

For details, refer to the LoadRunner documentation.

Start Page | gospace - Web (HTTP/HTML)

vuser_init
Action
vuser_end
globals.h

```
Action()
{
    lr_start_transaction("book_flight");
    web_url("menu.do",
        "URL=http://192.168.10.19:9090/frontend/menu.do?action=special",
        "Resource=0",
        "RecContentType=text/html",
        "Referer=http://192.168.10.19:9090/frontend/userlogin.do;jsessionid={JSESSIONID2}",
        "Snapshot=t10.inf",
        "Mode=HTML",
        LAST);

    web_link("Athenas Hotel",
        "Text=Athenas Hotel",
        "Ordinal=1",
        "Snapshot=t11.inf",
        EXTRARES,
        "Url=images/hotel2.jpg", "Referer=http://192.168.10.19:9090/frontend/list.do?action=more&oid=2375", ENDITEM,
        LAST);

    web_link("buy directly",
        "Text=buy directly",
        "Snapshot=t12.inf",
        LAST);

    web_submit_form("netpay.do",
        "Snapshot=t13.inf",
        ITEMDATA,
        "Name=action", "Value=confirm", ENDITEM,
        LAST);

    web_submit_form("netpay.do_2",
        "Snapshot=t14.inf",
        ITEMDATA,
        "Name=action", "Value=ok", ENDITEM,
        LAST);

    lr_end_transaction("book_flight", LR_AUTO);

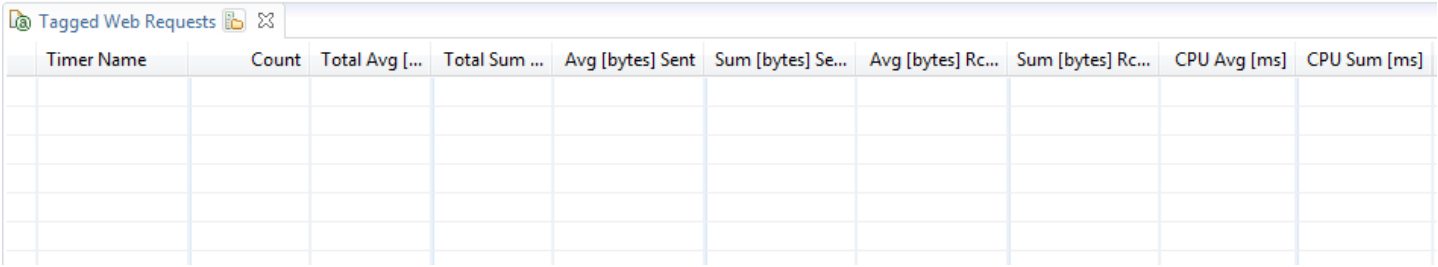
    return 0;
}
```

LoadRunner script

Step 2: Modify the LoadRunner Script

Modify the LoadRunner script as needed. Run the script to make sure it does not contain errors.

The web requests are not listed in the **Tagged Web Requests** dashlet.



Tagged Web Requests dashlet

Step 3: Start the Script Converter

Open the AppMon Client and select **Tools > LoadRunner Script Converter**. The **LoadRunner Script Converter** appears.

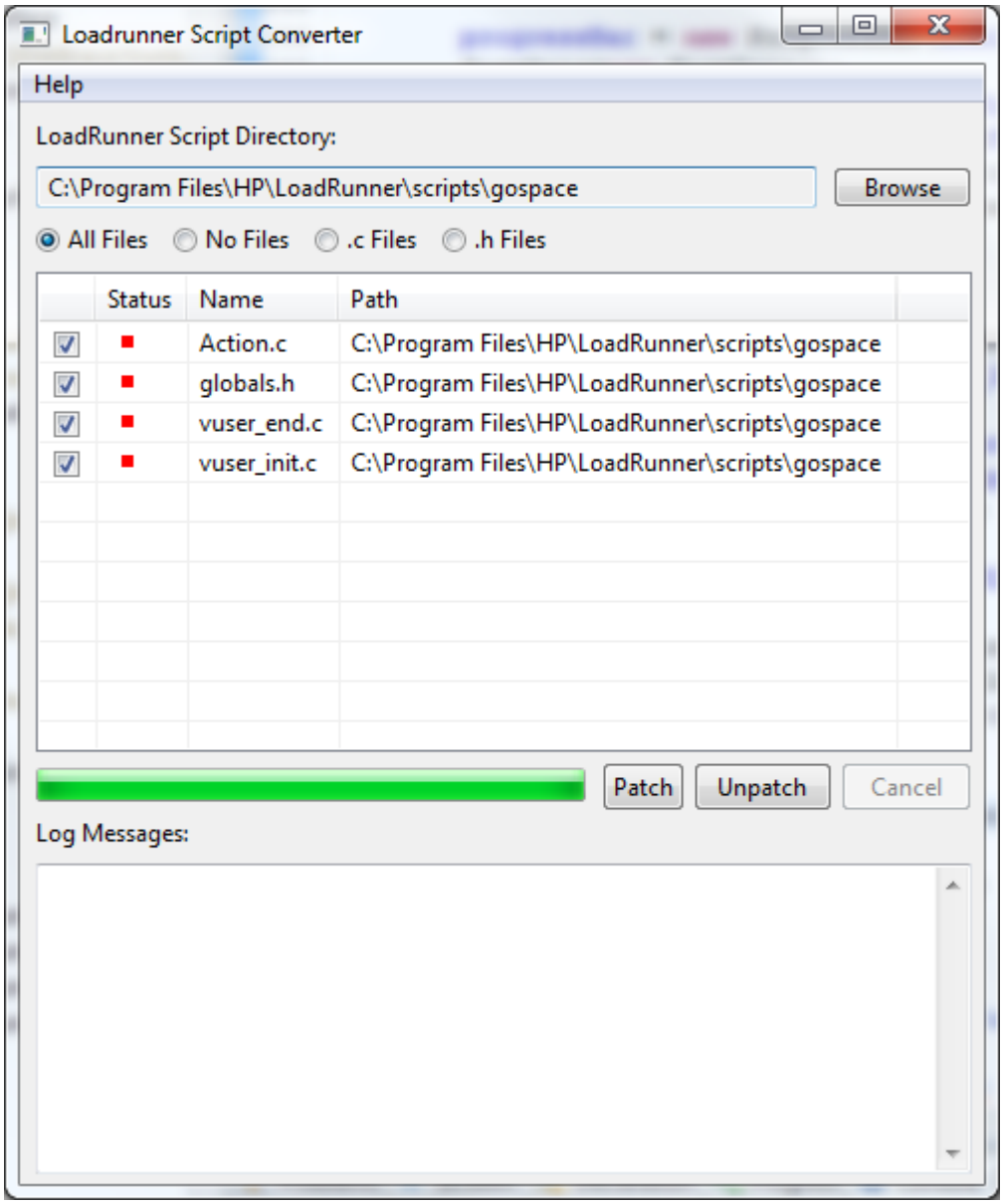
Step 4: Select the Directory That Contains the LoadRunner Scripts

Click **Browse** and navigate to the directory that contains your LoadRunner scripts. When you click **OK**, the **LoadRunner Script Converter** automatically shows all `*.c` files and `globals.h` files that are present in the selected directory.

Step 5: Patch or Unpatch the LoadRunner Scripts

1. Select the files to modify.
   To select specific files, use the checkboxes beside the file names in the **LoadRunner Script Converter** table. To select files with a specific extension or all files, use the radio buttons above the table.

2. Modify the LoadRunner scripts.
   To add the AppMon HTTP header to the selected LoadRunner scripts, click **Patch**.
   To remove all modifications the **LoadRunner Script Converter** made previously, click **Unpatch**.



LoadRunner script converter

</figcaption></figure>

If you need to cancel the conversion, click the **Cancel** button. When you cancel, the **LoadRunner Script Converter** finishes working on the currently active file before ending the operation.

Information about the script conversion is displayed as the conversion proceeds:

- **Progress** – The status icon of each file changes to indicate the status of the operation: currently working on this file, file successfully modified, or error occurred. The progress bar beneath the table shows the overall progress of the operation.
- **Errors** – If the **LoadRunner Script Converter** was unable to modify a file or encountered any other error, the **Log Messages** field displays the error.
  Some common error messages are: **Could not find the file**
  The file may have been renamed or deleted while you were configuring the **LoadRunner Script Converter**. Try reloading the directory by clicking **Browse** and selecting the same directory again.
  - **Was not able to restore the backup file**
    The original file may be opened by another user, or you may not have the permissions to delete the file.
  - **The file is already modified**
    This message only applies to the globals.h file, and only if you previously modified it with the Script Converter. In most cases, this is not an error and you can safely ignore it. If you want to be sure, click **Unpatch** and then click** Patch** again. This time the message should not be displayed.
  - **Access denied**
    Check your permissions for this file. If the filename ends with `.tmp`, make sure you have write permissions for this directory.

Your LoadRunner script is now ready to use with AppMon.

Start Page | gospace - Web (HTTP/HTML)

vuser_init
Action
vuser_end
globals.h

```
Action()
{
    lr_start_transaction("book_flight");
    addDynaTraceHeader("NA=book_flight;PC=menu.do");
    web_url("menu.do",
        "URL=http://192.168.10.19:9090/frontend/menu.do?action=special",
        "Resource=0",
        "RecContentType=text/html",
        "Referer=http://192.168.10.19:9090/frontend/userlogin.do;jsessionid={JSESSIONID2}",
        "Snapshot=t10.inf",
        "Mode=HTML",
        LAST);

    addDynaTraceHeader("NA=book_flight;PC=Athenas Hotel");
    web_link("Athenas Hotel",
        "Text=Athenas Hotel",
        "Ordinal=1",
        "Snapshot=t11.inf",
        EXTRARES,
        "Url=images/hotel2.jpg", "Referer=http://192.168.10.19:9090/frontend/list.do?action=more&oid=2375", ENDITEM,
        LAST);

    addDynaTraceHeader("NA=book_flight;PC=buy directly");
    web_link("buy directly",
        "Text=buy directly",
        "Snapshot=t12.inf",
        LAST);

    addDynaTraceHeader("NA=book_flight;PC=netpay.do");
    web_submit_form("netpay.do",
        "Snapshot=t13.inf",
        ITEMDATA,
        "Name=action", "Value=confirm", ENDITEM,
        LAST);

    addDynaTraceHeader("NA=book_flight;PC=netpay.do_2");
    web_submit_form("netpay.do_2",
        "Snapshot=t14.inf",
        ITEMDATA,
        "Name=action", "Value=ok", ENDITEM,
        LAST);

    lr_end_transaction("book_flight", LR_AUTO);

    return 0;
}
```

Script modified by script converter

Step 6: Run the Load Test Again

This time, the web requests are listed in the **Tagged Web Request** dashlet.

Tagged Web Requests

Timer Name	Virtual User	Page Context	Count	Total Avg [...	Total Sum ...	Avg [bytes] Sent	Sum [bytes] Se...	Avg [bytes] Rc...	Sum [bytes] Rc...
book_flight	-1	menu.do	1	335.89	335.89	247	247	542	542
book_flight	-1	netpay.do_2	1	20.26	20.26	247	247	608	608
book_flight	-1	Athenas Hotel	1	5.04	5.04	247	247	539	539
book_flight	-1	buy directly	1	84.60	84.60	247	247	536	536
login	-1	userlogin.do	1	55.87	55.87	247	247	623	623
logout	-1	menu.do	1	10.91	10.91	163	163	536	536

Tagged Web Request dashlet

Conversion result

The **LoadRunner Script Converter** inserts a short method in your `globals.h` file that populates the AppMon header. It also parses your `.c` files and automatically adds a call to this method before one of the following methods is called:

- `web_url`
- `web_link`
- `web_image`
- `web_submit_form`
- `web_submit_data`
- `web_custom_request`

The `TimerName` is set to the name of the currently running transaction; if the transactions are nested, the transaction names are concatenated. The `PageContext` contains the value of the first parameter passed to the functions listed above.

The `addDynaTraceHeader` function determines the ID of the `vuser` and the current timestamp, and appends this information to the header. It also adds a call in front of the following methods, but it does not contain a `PageContext`:

- `web_browser`
- `web_button`
- `web_check_box`
- `web_edit_field`
- `web_elementweb_file`
- `web_image_link`
- `web_image_submit`
- `web_list`
- `web_map_area`
- `web_radio_group`
- `web_reg_dialog`
- `web_static_image`
- `web_table`
- `web_text_area`
- `web_text_link`

Development kits

The AppMon Agent Development Kit, JavaScript ADK, and Mobile Agent enable you to extend the default AppMon Agent monitoring capabilities by integrating AppMon with technologies such as native C or C++ applications, custom communication protocols, custom JavaScript frameworks, and custom frameworks for mobile applications.

Agent Development Kit (ADK)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Purpose
Components and platforms
Concept
ADK package contents
Licensing
Troubleshooting

Related pages

CICS and IMS Tagging ADK
Native ADK
Tagging ADK for Java
Tagging ADK for .NET

Downloads

Latest update

The AppMon Agent Development Kit (ADK) extends the capability of the AppMon APM solution to areas and scenarios that are not supported by default sensors. For example, you can use the ADK for a proprietary protocol or for instrumenting native applications. The ADK enables you to trace transactions across different platforms and technology types and to correlate the parts of the transaction from all tiers into one PurePath.

Purpose

The AppMon ADK provides:

- Following transactions across all tiers
  End-to-end tracing through Java, .NET, and native applications (C/C++, JNI, and others) in production
- Support for proprietary protocols
  Sensors are available that allow AppMon to follow transactions across system boundaries more or less automatically, by tagging. If proprietary protocols are used, the *Tagging ADK* (available for Java, .NET, and C/C++) traces transactions even across those boundaries.
- Visibility into native applications
  The *Native ADK* can be used to instrument the source code for native applications.

Components and platforms

From the technology perspective, the AppMon ADK has the following components:

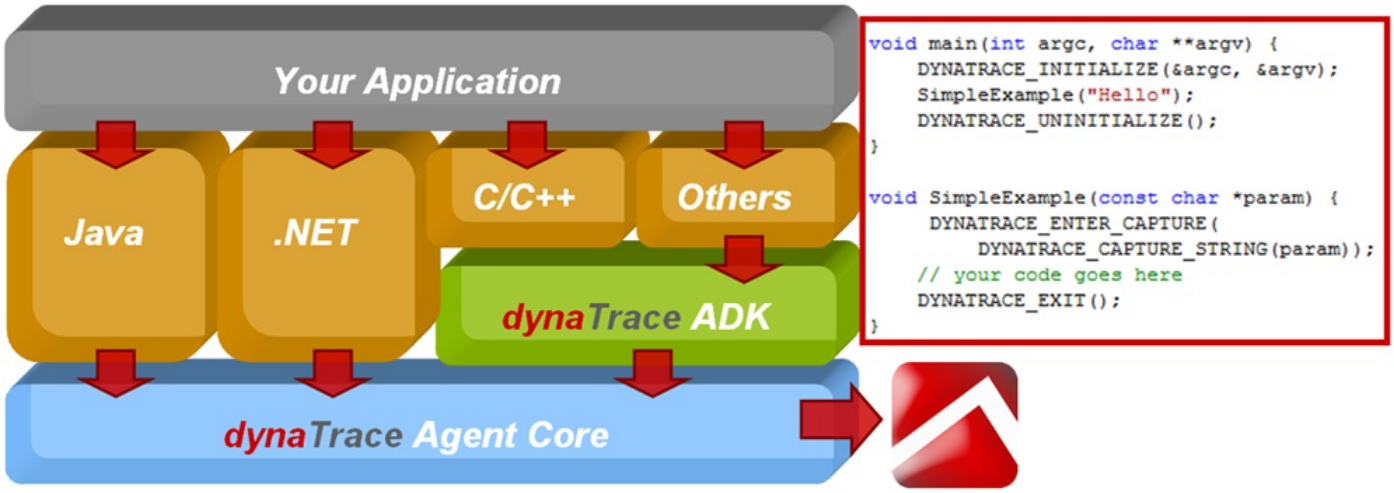
- Tagging ADK for Java and .NET

The Tagging ADK provides support for tagging across custom protocols.

- Native ADK
  The Native ADK provides both tagging and source-code instrumentation for all supported platforms and all applications that can load and call C/C++ libraries.
- CICS and IMS Tagging ADK
  The CICS and IMS Tagging ADK provides an interface to the zOS Agent for tagging in COBOL, PL1, Assembler, and C/C++ in the z/OS CICS and IMS message processing region environments.

Concept

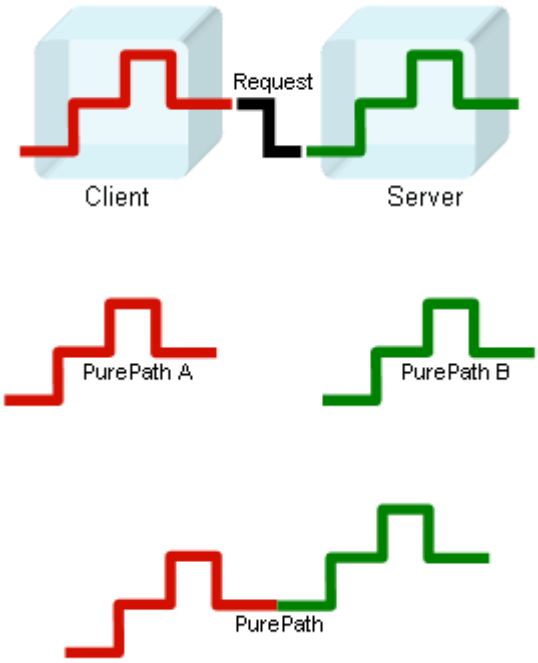
The AppMon ADK is a small layer between your application and the AppMon Agent that handles the communication to the AppMon Server. It provides an interface that enables applications to send necessary transactional information to the AppMon Server. The Server uses these pieces of information, or *tags*, to combine sub-paths into a real end-to-end PurePath.



ADK architecture

Tagging concepts

The Tagging ADK enables you to retrieve the AppMon Trace Tag for the current transaction. This Trace Tag then needs to be sent through the proprietary protocol from the client side to the server side. On the server side, the Trace Tag is placed again. It enables the AppMon Server to link sub-paths of the same transaction together.



Types of tags

The ADK distinguishes between these different types of tags:

- The *Trace Tag* consists of structured data that defines the unique identifier for a specific remote call.
- The *Custom Tag* can be any user-defined, unstructured piece of data. Its content does not matter as long as it is unique for each remote call.

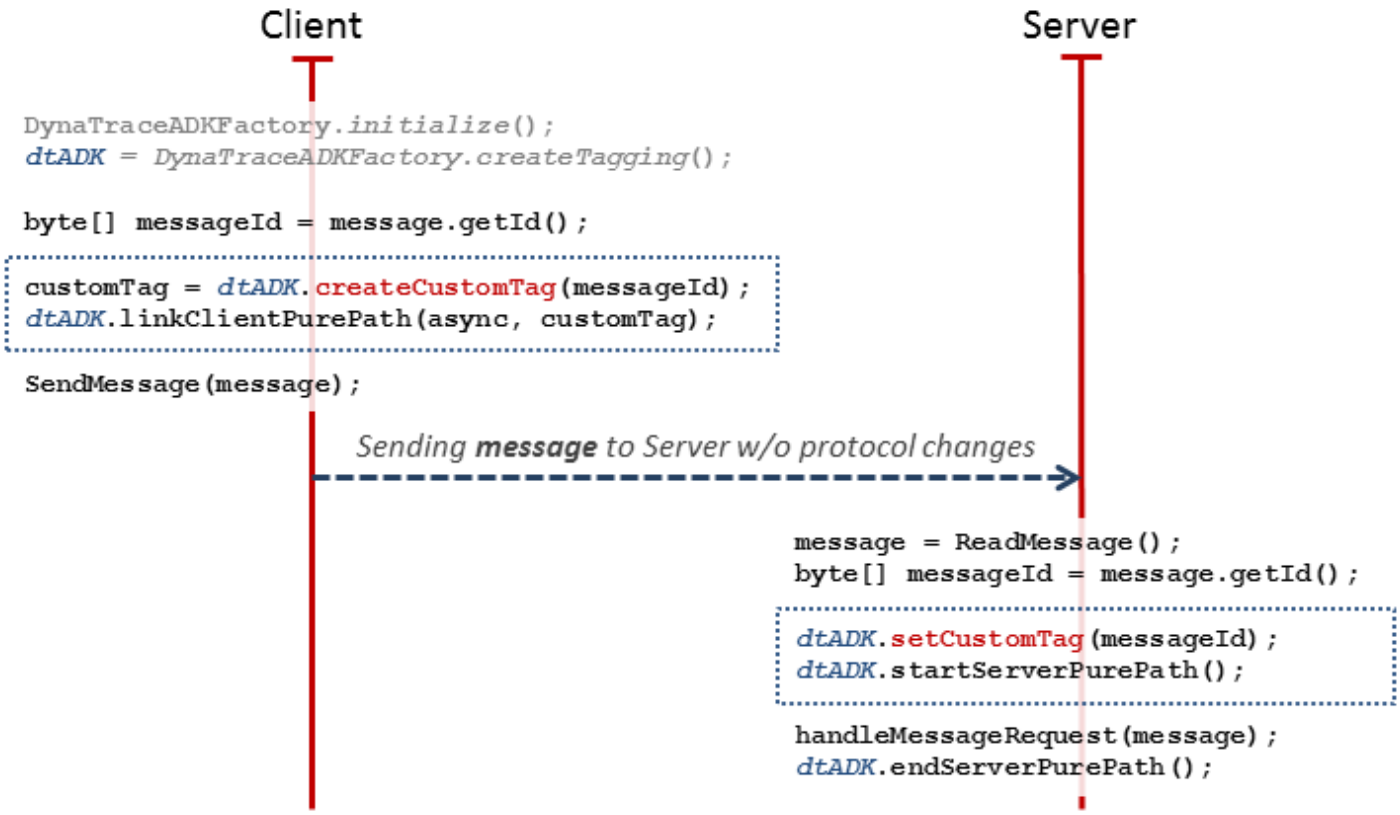
AppMon tagging (active tagging)

Using the Trace Tag is the recommended and most efficient way of tracing transactions across proprietary protocols. It's also called *Active Tagging*, because it actively retrieves a tag from the ADK at the client side and sends it over the protocol to the server side. The AppMon tag can be represented either as a byte array or as a character string, depending on the needs of the client-server protocol to which it has to be attached.

Custom tagging (passive tagging)

When AppMon tagging is technically not possible (the protocol cannot be expanded by adding the AppMon tag information), Custom Tagging might be an option. Custom Tagging uses an arbitrary identifier for linking paths together. This identifier must be unique for each invocation.

If a message sent from the client to the server has a unique message ID, this ID can be used as *a custom tag* instead of using the AppMon tag. Using the ID avoides protocol changes. Therefore, Custom Tagging is also called *Passive Tagging*.



Using existing message ID to link PurePaths

ADK package contents

Package structure

After you install the ADK, the `dynatrace-<version>` folder has the following contents:

```
dynatrace-<version>
+--adk
|  +--doc
|  |  +--c-c++      Interface documentation of the open-source, reference language
                    bindings for C and C++
|  +--include       All necessary header files
|  +--lib           32-bit dynaTrace ADK libraries for specific platform
|  +--lib64         64-bit dynaTrace ADK libraries for specific platform
+--agent
|  +--lib           32-bit dynaTrace Agent for specific platform
|  +--lib64         64-bit dynaTrace ADK libraries for specific platform
+--samples
+--HelloWorld C++   C++ application showing how to use the ADK's basic functionality
+--Java and JNI     Sample application demonstrating tagging java call to JNI functions
+--MultiApp         Sample application implemented in C++, Java and .NET demonstrating
                    tagging over sockets.
+--ADK.profile.xml  System Profile used in all samples
```

Sample applications

After installation, the `samples` subdirectory contains these sample applications:

- **HelloWorld C++**
  C++ application that starts and stops transactions, captures arguments, and traces across thread boundaries.
- **Java and JNI**
  Sample application that demonstrates tagging Java call to JNI functions.
- **MultiApp**
  Sample application implemented in C++, Java, and .NET, which demonstrates tagging over sockets using different technologies. The application can be run as client or server, where the client sends a request to the server to multiply two numbers.

To build the samples, add the Agent ADK library to the `lib` directory (Java) or to the referenced build path (.NET). For detailed information, refer to the `readme` files.

Licensing

- All ADKs are royalty-free with the exception of use on z/OS. AppMon customers can integrate the ADKs into products and deploy as many instances as needed, without charge except on the IBM Mainframe (z/OS).
- To collect data from deployed ADK applications, you need to deploy and license one AppMon Agent for each application instance that will be monitored. Java and .NET ADKs require the standard Java / .NET Agent with no licensing specifics. The Native ADK and CICS ADK require a special build of the AppMon Agent — the ADK Agent — that is licensed separately from the Java / .NET Agent on a per-Agent basis. The Native ADK requires its own license if running on z/OS. The CICS ADK (Agent) licensing is MSU-based.

Troubleshooting

My application does not load the AppMon Agent:

Make sure that the environment variable `DT_AGENTLIBRARY` is set correctly and points to the AppMon Agent (`.dll` or `.so`, depending on your platform). To get a list of defined environment variables, type **set** in the Windows command line.

For more details: set the environment variable `DT_DEBUGADK`, type **set DT_DEBUGADK=true**, and run your application again. You will see additional output indicating the source of the error.

Error handling may not cover all possible error scenarios. If the debug output doesn't help identify the source of the problem, comment to this page to get assistance from the AppMon team.

My application loads the agent, but it doesn't connect to the AppMon Server:

Make sure the AppMon Server is up and running, and has an active license.

I can't link or compile my application:

Make sure the header files and the AppMon ADK libraries can be found.

What compiler was used to build the Windows binaries?

The binaries were built using Microsoft Visual Studio 2010.

Native ADK

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Initialization

Sensor placement and argument capturing

Tagging

Logging

Configuration

Sample applications

Downloads

The Native ADK provides both tagging and instrumentation on the source-code level. It can be used for all supported platforms and for all applications that can load and call a C or C++ library.

The Native ADK provides a set of C and C++ macros to be used for instrumenting the source code. These macros provide the following:

- **Initialization**: Set configuration values. Connect to an AppMon Server (that is, to a stand-alone or Server-integrated Collector).
- **Sensor Placement**: Define entry and exit points. Capture arguments and return values.
- **Tagging**: Trace PurePaths.

Initialization

- `DYNATRACE_SET("<key>=<value>");`: Use this macro to define configuration parameters directly in your code. Use either environment variables or command-line arguments as described below.

The `<key>` can be any known configuration parameter (see Configuration below), but the `dt_` prefix needs to be omitted, for example `DYNATRACE_SET("agentname=MySpecialAgent")` for setting the `DT_AGENTNAME`.

- `int result = DYNATRACE_INITIALIZE(&argc, &argv);`: This macro initializes the AppMon Agent. ADK-specific arguments (arguments starting with `--dt`) will be removed from `argc` and `argv`. Any calls except `DYNATRACE_SET` before this initialization step are ignored. It's good practice to put this macro directly in the `main()` method, because it should be called only once per application. The return value (result) is 0 if the ADK is initialized successfully or was already initialized in a previous call; -1 if any error occurred (for example if the Agent library failed to load).
- `DYNATRACE_REINITIALIZE();`: This call is needed by the child process after a `fork()`, because the AppMon Agent was left in a defunct state.
- `DYNATRACE_UNINITIALIZE();`: This macro terminates and shuts down the AppMon Agent connection, and releases any resources. Any calls after this step are ignored. It's good practice to put this macro at the end of the `main()` method, because it should be called only once when the application terminates.

Sensor placement and argument capturing

Two alternative macro sets are available, one for pure C code and one for C++. The C macros can also be used in C++ environments. They are fully compatible in case the code base to be instrumented consists of both C and C++ applications.

Both macro sets are described in the table.

C and C++ Macros

C Macros	C++ Macros
`DYNATRACE_API("<API>");` Defines the API for a Sensor. This macro must be placed before entering a Sensor.	`DYNATRACE_API("<API>");` Same macro as in C++.

`DYNATRACE_START_PUREPATH();`
`DYNATRACE_EXIT();`
Places a Sensor and marks it as an entry point (i.e., starts a PurePath).
Requires the `DYNATRACE_EXIT` macro for marking the end point of the Sensor.

`DYNATRACE_AUTO_PUREPATH();`
Places a Sensor and marks it as an entry point.
Does not require an exit macro.

`DYNATRACE_ENTER();`
`DYNATRACE_EXIT();`
Places a Sensor without starting a PurePath.
Requires `DYNATRACE_EXIT` for marking the end point of the Sensor.

`DYNATRACE_AUTO_NODE();`
Places a Sensor in C++.
Does not require an exit macro.

```
DYNATRACE_ENTER_CAPTURE(
DYNATRACE_CAPTURE_<type> (<arg1>)
DYNATRACE_CAPTURE_<type> (<arg2>)
...
);
...
```
`DYNATRACE_EXIT();` Argument capturing:
`DYNATRACE_ENTER_CAPTURE` places a Sensor to enable argument capturing.
Individual macros are available the various argument types:
`DYNATRACE_CAPTURE_BOOL(arg)`
`DYNATRACE_CAPTURE_CHAR(arg)`
`DYNATRACE_CAPTURE_INT16(arg)`
`DYNATRACE_CAPTURE_INT32(arg)`
`DYNATRACE_CAPTURE_INT64(arg)`
`DYNATRACE_CAPTURE_FLOAT(arg)`
`DYNATRACE_CAPTURE_DOUBLE(arg)`
`DYNATRACE_CAPTURE_STRING(arg)`

```
DYNATRACE_AUTO_NODE_CAPTURE(
DYNATRACE_CAPTURE << arg1 << a
<< arg *n* ...
);
```
Argument capturing in C++:
`DYNATRACE_AUTO_NODE_CAPTURE` plac
Sensor to enable argument capturing.
`DYNATRACE_CAPTURE` creates a capture instance that you can stream arguments onto.

For compilers that do not support variadic macros, macros are available that have a fixed number of arguments.
For example:
`DYNATRACE_ENTER_CAPTURE_1(arg1)`
`DYNATRACE_ENTER_CAPTURE_2(arg1, arg2)`
For further details, refer to the documentation for the ADK.

Return value capturing:
`DYNATRACE_EXIT_RETURN_<type>(<retval>)` The `DYNATRACE_EXIT_RETURN_<type>` macros mark the end of a Sensor and capture the return value of the type as defined by the macro. The following return value types are supported:

```
DYNATRACE_EXIT_RETURN_BOOL(<retval>)
  DYNATRACE_EXIT_RETURN_CHAR(<retval>)

DYNATRACE_EXIT_RETURN_INT16(<retval>)

DYNATRACE_EXIT_RETURN_INT32(<retval>)

DYNATRACE_EXIT_RETURN_INT64(<retval>)

DYNATRACE_EXIT_RETURN_FLOAT(<retval>)

DYNATRACE_EXIT_RETURN_DOUBLE(<retval>)

DYNATRACE_EXIT_RETURN_STRING(<retval>)
```

Return value capturing (C++ macro):
`DYNATRACE_RETURN_<type>(<retval>)` The `DYNATRACE_RETURN_<type>` macros capture the return value of the type as defined b the macro and return the captured value. The following return value types are supported:

```
DYNATRACE_RETURN_BOOL(<retval>
DYNATRACE_RETURN_CHAR(<retval>

DYNATRACE_RETURN_INT16(<retval

DYNATRACE_RETURN_INT32(<retval

DYNATRACE_RETURN_INT64(<retval

DYNATRACE_RETURN_FLOAT(<retval

DYNATRACE_RETURN_DOUBLE(<retva

DYNATRACE_RETURN_STRING(<retva
```

All standard C/C++ data types and strings such as `char*` can be captured. Pointers as well as some Microsoft type definitions (such as `LDPWORD`) require casting. For example:

```
LPTSTR pchRequest;
LPDWORD pchBytes;
...
// C++ style argument capturing:
DYNATRACE_AUTO_NODE_CAPTURE(DYNATRACE_CAPTURE << (const char*)pchRequest << (int)\*pchByt

// C-style argument capturing:
DYNATRACE_ENTER_CAPTURE(
    DYNATRACE_CAPTURE_STRING(pchRequest),
    DYNATRACE_CAPTURE_INT32((int)\*pchBytes)
);
```

Tagging

Tagging in general closes the PurePath gap between distributed components that use proprietary remoting protocols. It enables continuing a client-side execution path on the server side.

To link a client path to a server path:

1. On the client side, get the tag of the current PurePath and insert a link.
2. Transfer the tag to the server.
3. Start the server-side path, using the tag received from the client. This links the server's PurePath to the PurePath started on the client side.

The AppMon ADK provides support for steps 1 and 3. Transferring the tag from client to server depends on the protocol in use, which is probably proprietary. The transfer procedure is specific to the implementation.

Client-side and Server-side Tags

Client Side	Server Side

`DYNATRACE_GET_TAG(bytebuf, size);`
`DYNATRACE_GET_TAG_AS_STRING(strbuf, size);`
Get the current AppMon tag either as a string (zero-terminated ASCII string) or as a byte array. The size of the string buffer should be at least 100 characters. The size of the byte array is defined as 30 bytes. (This size is subject to change in future versions, but the length will stay constant within an AppMon major version). To determine the exact binary tag length, use the ADK call `DYNATRACE_GET_TAG_SIZE()`.
	`DYNATRACE_SET_T`
	`DYNATRACE_SET_T`
	These macros set the Ap from a string or from a b

`DYNATRACE_LINK_CLIENT_PUREPATH(synchronous, bytebuf);`
`DYNATRACE_LINK_CLIENT_PUREPATH_BY_STRING(synchronous, strbuf);`
Insert a *synchronous* (`true`) or *asynchronous* (`false`) link on the client side using the tag that was received by one of the `DYNATRACE_GET_TAG` macros described above. Synchronous/asynchronous pertains to the type of the PurePath link. Both link types do not result in a blocking call.
	`DYNATRACE_START_`
	Starts a server-side Pur PurePath.

	`DYNATRACE_END_S` Terminates a server-side

`DYNATRACE_GET_TAG_SIZE();`
Returns the amount of space required for the (binary) AppMon tag, in bytes.

`GET_TAG` must be called each time before issuing a remote call to the server side, even if it is still in the same transaction to the same server. The tag that is retrieved by `GET_TAG` identifies just one call, and it cannot be reused. This is also the case when using custom tags. Any request or message ID that is used to identify the remote call must be unique. Another call requires a new, unique identifier.

Important

Make sure `DYNATRACE_LINK_CLIENT_PUREPATH()` is invoked before `DYNATRACE_START_SERVER_PUREPATH()` . It is good practice to place the macro before sending a message to the server or before calling the server-side.

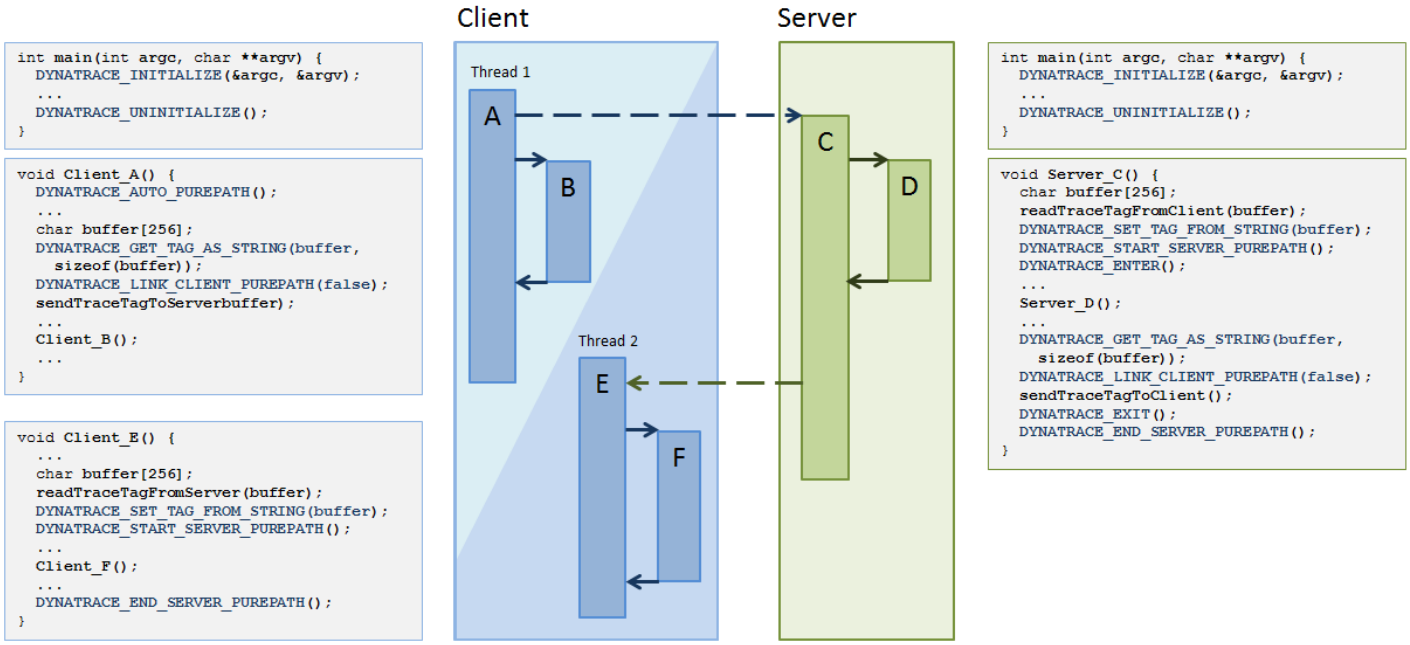
Helper macros

```
DYNATRACE_TAG_TO_STRING(tag, strbuf, strbuf_size);
DYNATRACE_STRING_TO_TAG(strbuf, tag, tag_size);
```

These macros can be used to convert the tag byte array into its string representation, and vice versa.

Tagging in context

The following diagram shows examples of the tagging macros.



Example Tagging Macros

Tag format

Tags are represented either as a 30-byte buffer or as a string. The string representation has the following format:

```
FW<tag-version>;<server id><agent id>;<tag id>;<hop count>;<entry agent id>;<entry tag id>;<path mode indicator>
```

Example: `FW0;1;1481367847;0;4;1481367847;0;0`

Logging

```
DYNATRACE_LOG(loglevel, message, logger_name);
```

This function inserts a custom log message, which shows as *log* node on the related PurePath and in the *Logging* dashlet.

Parameters

`loglevel`	Numeric value for the log level.
`message`	Log message text.
`logger_name`	Name of the logger.

Configuration

You can configure instrumented C/C++ applications by using either environment variables or AppMon-specific command-line parameters.

Configuration Environment Variables and Parameters

ENVIRONMENT VARIABLE Parameter	Default Value	Comment
`DT_AGENTLIBRARY --dt_agentlibrary`	-	Path to the library's file
`DT_AGENTNAME --dt_agentname`	`<appname>`	Agent nam

DT_SERVER --dt_server	localhost:9998	Server (em
DT_AGENTACTIVE --dt_agentactive	true	Set Agent a
DT_DEBUGADK or DT_DEBUG --dt_debugadk or DT_DEBUG	false	Enable (tr
DT_HIRESTIMER --dt_hirestimer	<not set>	Specify the type can b auto – Se runtime cpu – RD (preferred) os – Wind resolution
DT_LOGFILE --dt_logfile	../../log/dt_<appname>_<pid>.log	Log file. Th
DT_LOGLEVELFILE --dt_loglevelfile	INFO	Define the [finest warning
DT_LOGLEVELCON --dt_loglevelcon	INFO	Define the Valid value
DT_LOGFILESIZE --dt_logfilesize	10 * 1024 * 1024	Define the size canno
DT_TRANSFORMATIONSAMPLES --dt_transformationsamples	30	Number of (classes).
DT_TRANSFORMATIONMAXAVGWAIT --dt_transformationmaxavgwait	10	The maxim slow transf for the ADK
DT_WAIT --dt_wait	20	The initial a connecti established uninstrume The AppMo example in releases th can still us when you t (via the lice
DT_SOCKTIMEOUT -	30	The Native receiving d

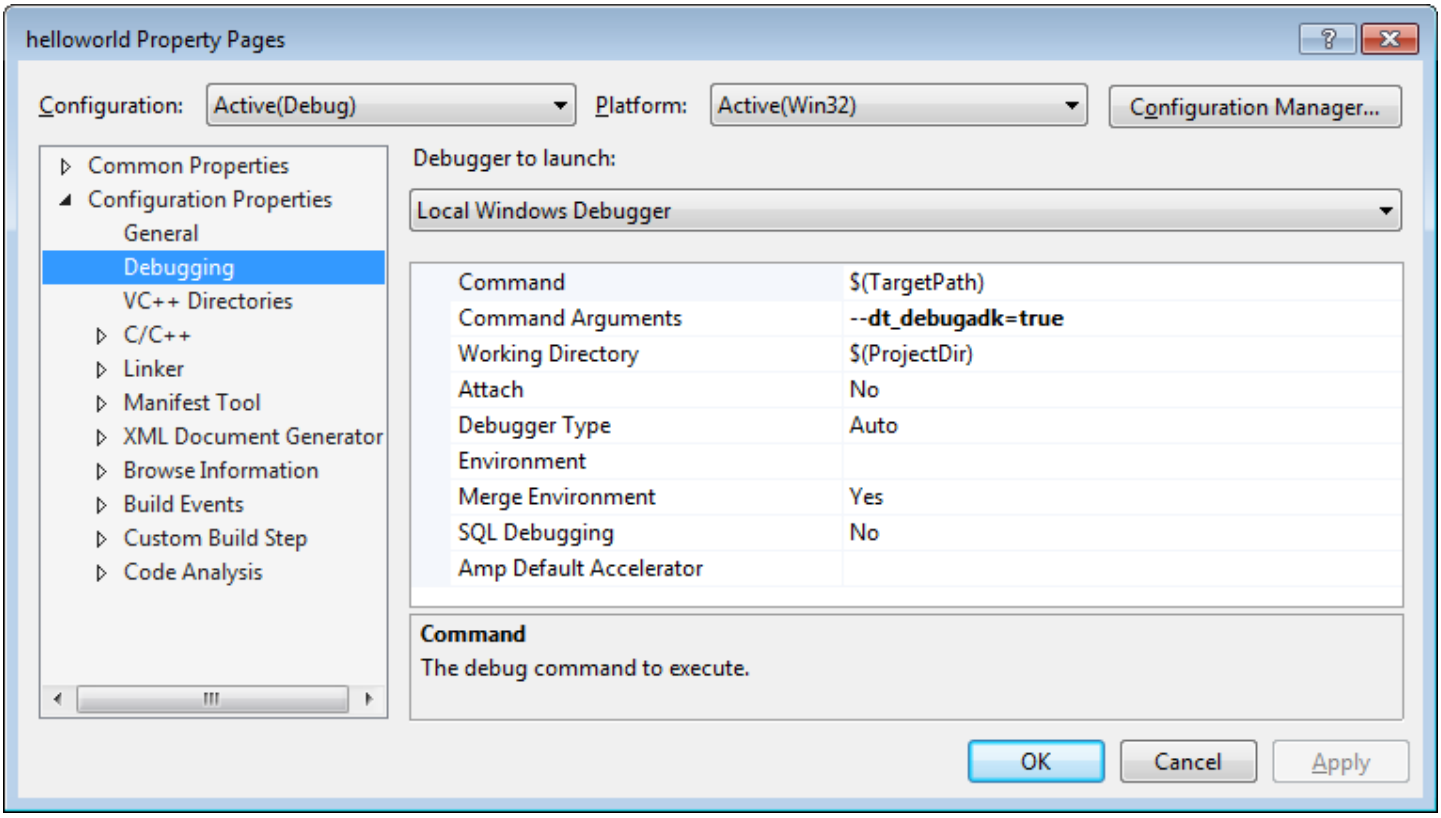
`DT_DEBUGFLAGS --dt_debugflags`	<none set>	Comma-se
		For examp
		`--dt_debu`
		Although a
		explicitly se
		`debugLo`
		`debugCl`
		`debugEv`
		`debugCl`
		`debugCo`
		`debugMe`
		`debugPe`
		`debugMe`
		`debugSy`
		`debugWa`
		`debugMo`
		`debugHa`
		`debugJa`
		`debugSa`
		`debugEx`
		`debugHo`
		`debugEv`
		`debugNo`
		`debugTa`
		`debugPu`

For a 64-bit AppMon installation on a 64-bit Windows machine, the command lines for the default 32-bit build target `helloworld.exe` are:

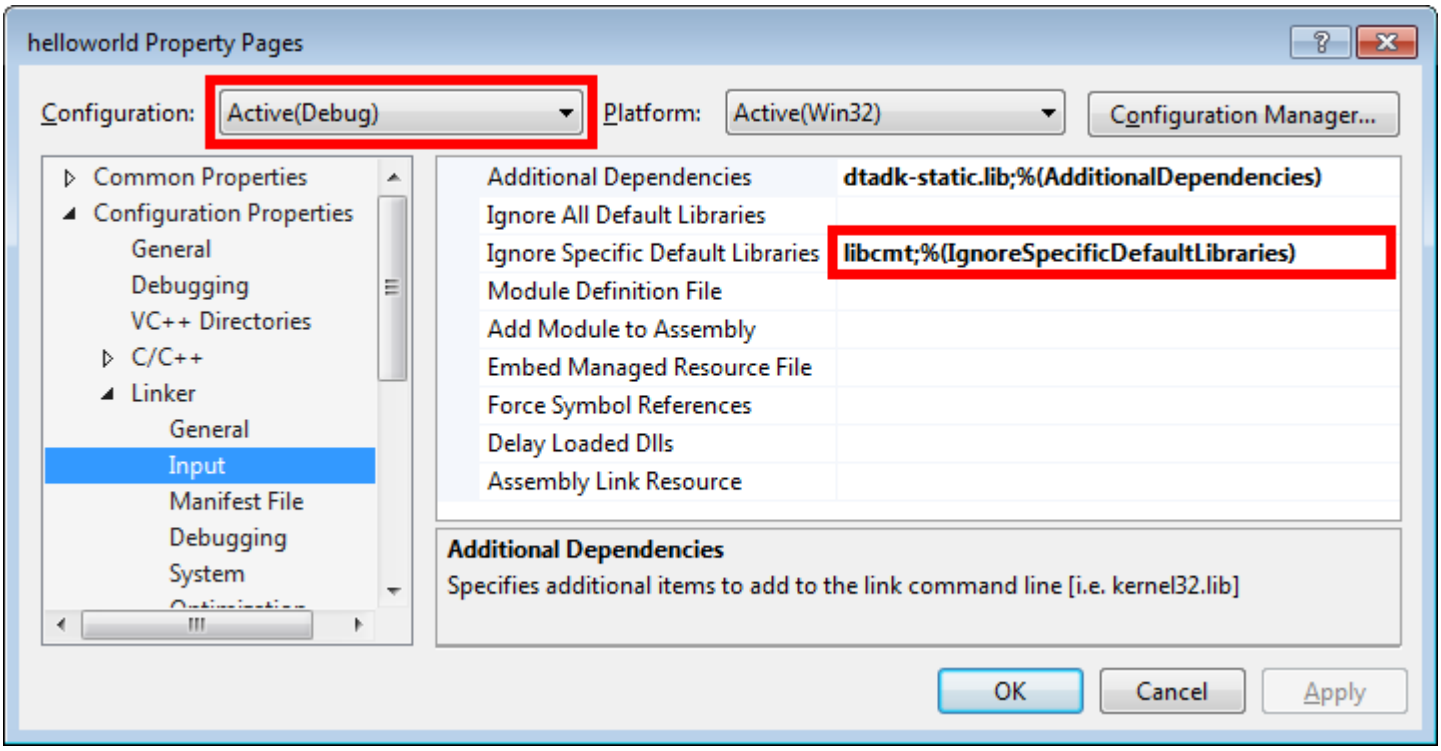
```
set DT_AGENTHOME=C:\Program Files\dynaTrace\dynaTrace <major.minorVersion>\agent\lib hell
```

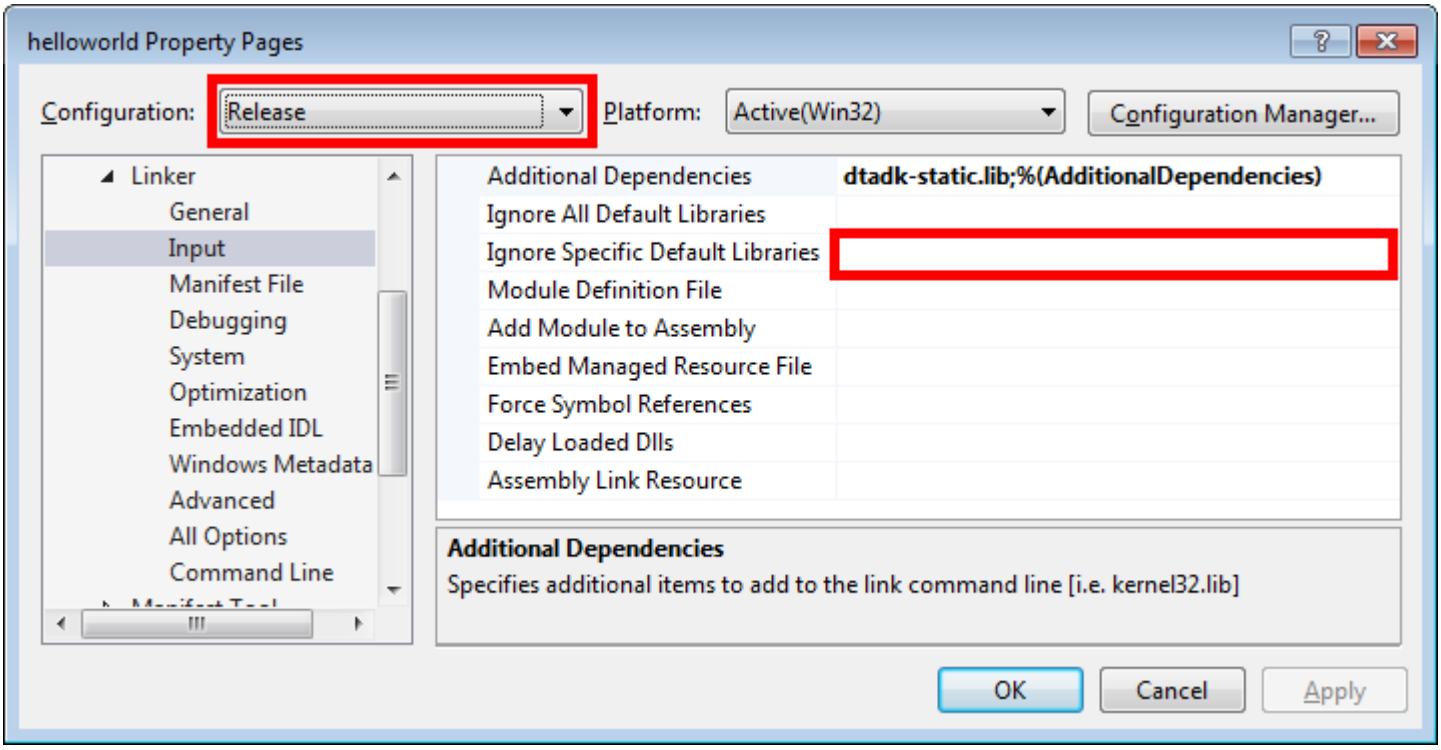
Supply the command-line parameter values as any combination of string literals or environment variables.

If you prefer to test from the VS IDE, supply arguments in the project's *Property Pages* window.



helloworld.exe project properties





Sample applications

Sample applications are available with the AppMon ADK to demonstrate its usage. They are installed the `samples` subdirectory:

- **HelloWorld C++** - A C++ application that starts and stops transactions, captures arguments, and traces across thread boundaries.
- **Java and JNI** - A sample application that demonstrations tagging Java calls to JNI functions.
- **MultiApp** - A sample application implemented in C++, Java, and .NET that demonstrates tagging over sockets using different technologies. The application can be run as Client or Server. The Client sends a request to the Server to multiply two numbers.

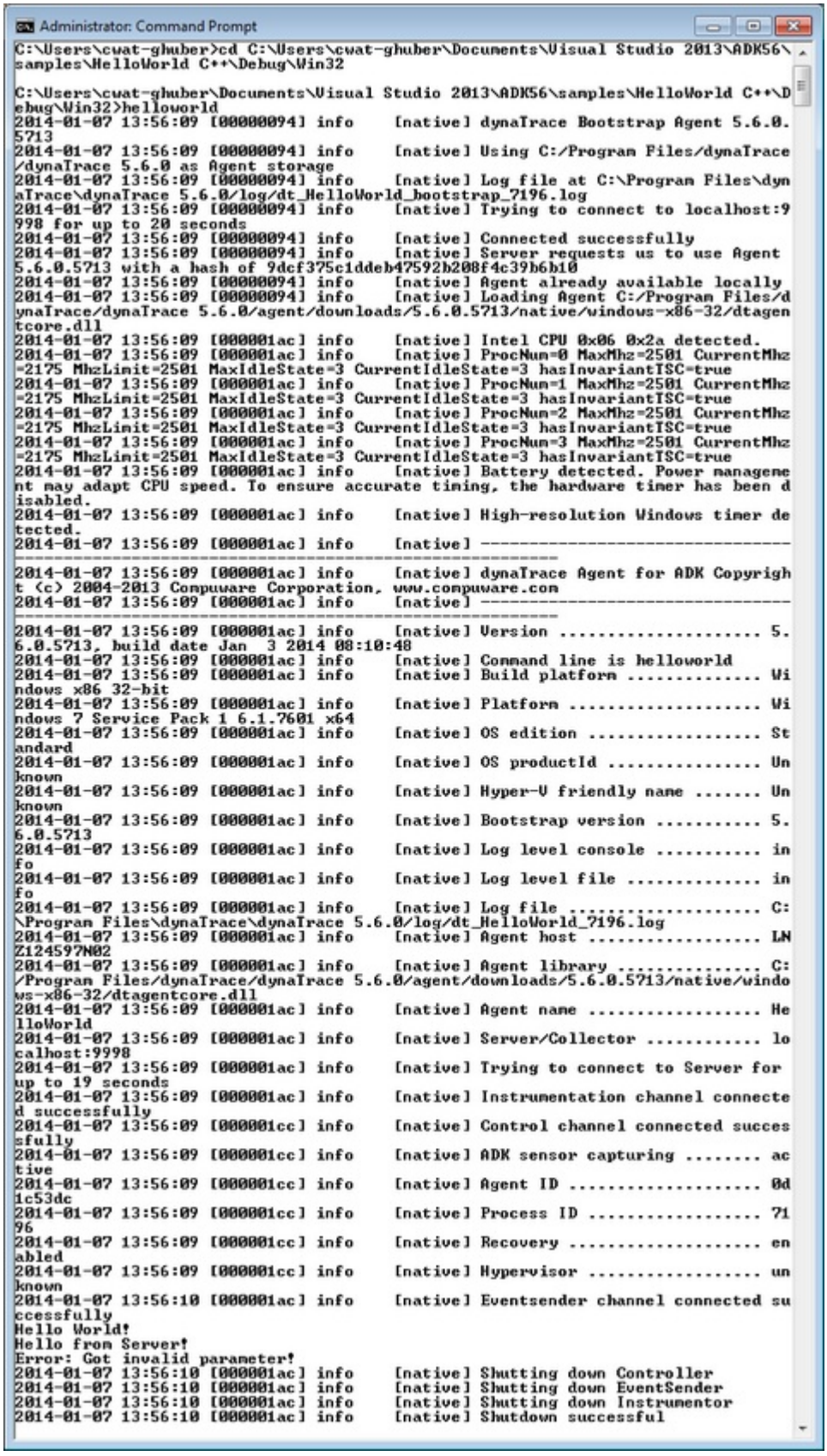
To build the samples, add the Agent ADK library to the `lib` directory (Java) or to the reference build path (.NET). For detailed information. refer to the `readme` files for the applications.

Running helloworld

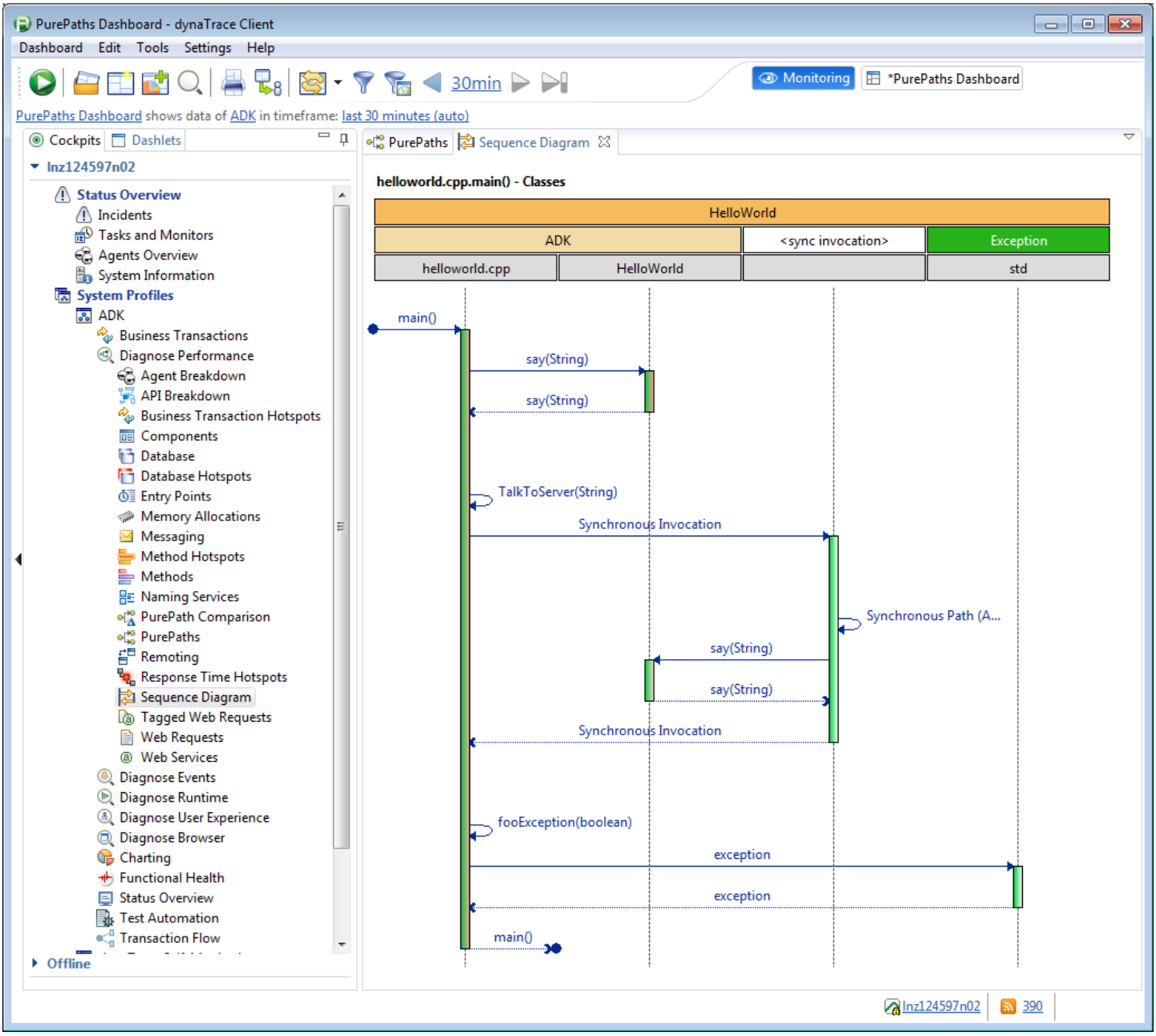
1. Compile `helloworld`:

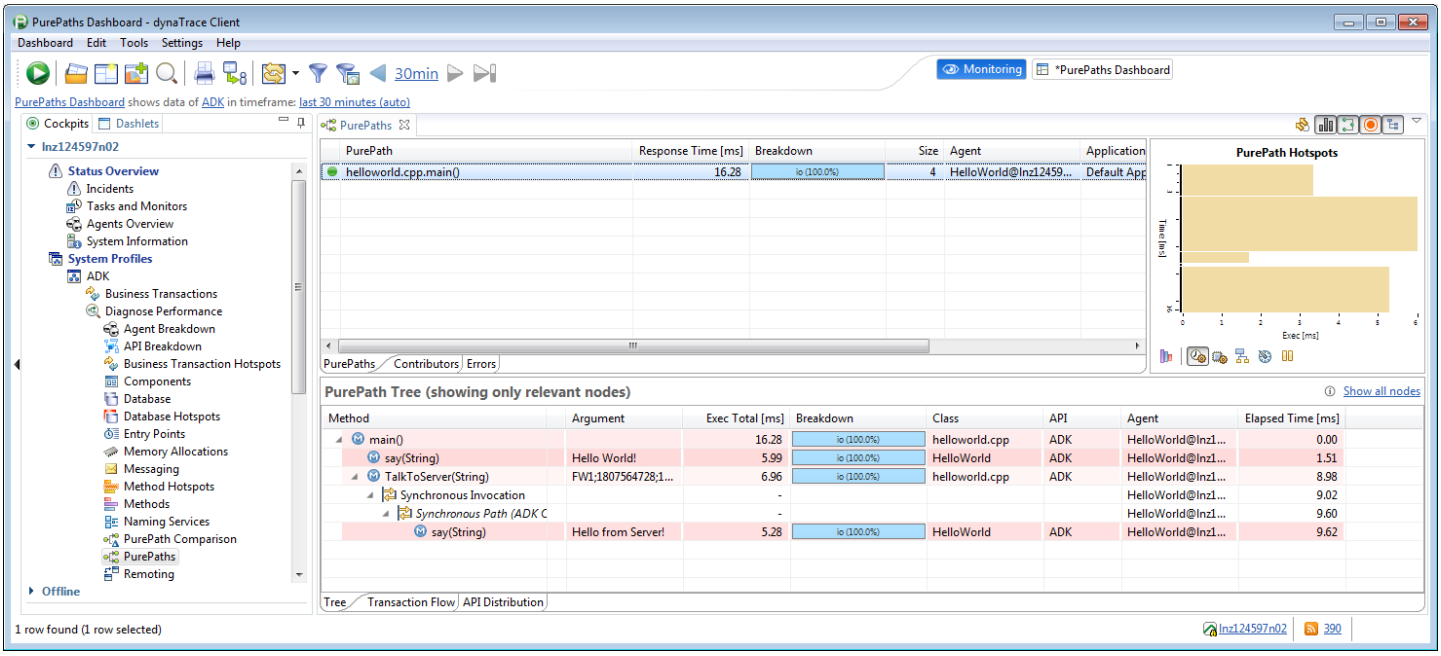
   Note: The revision digit is not part of the AppMon component installation path.
   - On Windows (with Microsoft Visual Studio):
     a. Open `<ADK_base_dir>/samples/HelloWorld C++/helloworld.sln` (where `<ADK_base_dir>` is the ADK directory, e.g. `C:\Users\user_name\Documents\Visual Studio 2013\ADK60`).
     b. Compile and link the solution. The default configuration is debug and the 32 bit platform.
   - On Linux/UNIX (with GNU Make):
     a. Open a shell and switch to the *HelloWorld* sample directory: `cd <ADK_base_dir>/samples/HelloWorld C++` (where `<ADK_base_dir>` is the ADK installation directory, e.g. `/home/user_name/dynatrace-7.0`)
     b. Compile and link by issuing the *make* command in that directory. By default *helloworld* is built as 32-bit application.
2. Configure the AppMon Server.
   a. Launch the AppMon Server and Collector, and connect the AppMon Client.
   b. Import `<ADK_base_dir>/Samples/ADK.profile.xml`: Right-click **System Profiles** in the AppMon Client navigation tree and select **Import System Profile**. Review the profile's **HelloWorld Agent Group > Agent Mappings** for details.
3. Run `helloworld` with tracing.
   a. Open a command-line window.
   b. Navigate to the directory where the `helloworld` executable is compiled, on Windows it's by default: `C:\Users\user_name\Documents\Visual Studio 2013\ADK56\samples\HelloWorld C++\Debug\Win32` on Linux/Unix it's the same directory as above: `<ADK_base_dir>/samples/HelloWorld C++`.

c. Run `helloworld` with command-line parameters (same for all platforms) and environment variables set as described above.



4. Diagnose `helloworld`.
   a. Open the *PurePath* dashlet: **System Profiles > ADK > Diagnose Performance > PurePaths**. The dashlet shows the full trace including captured arguments, execution times, and CPU times.
   b. From the PurePath, you can open or drill down into all AppMon dashlets. For example, right-click on the PurePath and drill down to the Sequence Diagram. All AppMon features are available. You can define measures, incident rules, and Business Transactions to monitor response times of your libraries or SLAs for your services.




PurePaths Dashboard - dynaTrace Client
Dashboard Edit Tools Settings Help
30min
Monitoring
*PurePaths Dashboard
PurePaths Dashboard shows data of ADK in timeframe: last 30 minutes (auto)
Cockpits
Dashlets
lnz124597n02
Status Overview
Incidents
Tasks and Monitors
Agents Overview
System Information
System Profiles
ADK
Business Transactions
Diagnose Performance
Agent Breakdown
API Breakdown
Business Transaction Hotspots
Components
Database
Database Hotspots
Entry Points
Memory Allocations
Messaging
Method Hotspots
Methods
Naming Services
PurePath Comparison
PurePaths
Remoting
Offline
PurePaths
PurePath
Response Time [ms]
Breakdown
Size
Agent
Application
helloworld.cpp.main()
16.28
4
HelloWorld@lnz12459...
Default App
PurePath Hotspots
Time [ms]
Exec [ms]
PurePaths
Contributors
Errors
PurePath Tree (showing only relevant nodes)
Show all nodes
Method
Argument
Exec Total [ms]
Breakdown
Class
API
Agent
Elapsed Time [ms]
main()
16.28
helloworld.cpp
ADK
HelloWorld@lnz1...
0.00
say(String)
Hello World!
5.99
HelloWorld
ADK
HelloWorld@lnz1...
1.51
TalkToServer(String)
FW1;1807564728;1...
6.96
helloworld.cpp
ADK
HelloWorld@lnz1...
8.98
Synchronous Invocation
HelloWorld@lnz1...
9.02
Synchronous Path (ADK C
HelloWorld@lnz1...
9.60
say(String)
Hello from Server!
5.28
HelloWorld
ADK
HelloWorld@lnz1...
9.62
Tree
Transaction Flow
API Distribution
1 row found (1 row selected)
lnz124597n02
390

Tagging ADK for .NET

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Using the Tagging ADK

Related pages

Agent Development Kit (ADK)

Downloads

The *Tagging ADK for .NET* is used for protocol tagging only. In contrast with the Native ADK, it provides no means for instrumenting code, because instrumentation is provided by various .NET sensors. It is part of the Agent Development Kit (ADK). For basic tagging concepts, see Agent Development Kit (ADK).

Using the Tagging ADK

Setup and prerequisite

After installation of the ADK package, the .NET ADK library is located in `<installation-dir>\adk\lib64\DynaTraceADK20.dll`.

Add the respective reference to the ADK library in your .NET project.

Initialize the ADK

The initialization step is required once per process, on the client side as well as on the server side.

After successful initialization, the `DynaTraceADKFactory` lets you create a tagging object, which serves as the interface to the Agent.

```
using DynaTrace.DynaTraceADK;
// [...]
//
// initialize the ADK factory (once on process startup)
DynaTraceADKFactory.initialize();
```

Create a tagging object, which serves as a thin layer between the application and the AppMon Agent.

```
Tagging tagging = DynaTraceADKFactory.createTagging();
```

To free up resources allocated by the `DynaTraceADKFactory`, uninitialize the factory before your application quits.

```
// Uninitializing on process shutdown is necessary in order to disconnect properly from A
// and to free up allocated resources
DynaTraceADKFactory.uninitialize();
```

The `DynaTraceADKFactory` is defined as follows:

```
namespace DynaTrace.DynaTraceADK {
        /// <summary>
        /// <see cref="DynaTraceADKFactory" /> creates an instance of an AppMon ADK.
        /// </summary>
        public class DynaTraceADKFactory {
                /// <summary>
                /// Initializes the <see cref="DynaTraceADKFactory" />
                /// </summary>
                public static void initialize() {
                        DynaTraceADKFactoryImpl.initialize();
                }

                /// <summary>
                /// Creates an instance of <see cref="Tagging" />
                /// </summary>
                /// <returns>a <see cref="Tagging" /> instance</returns>
                public static Tagging createTagging() {
                        return DynaTraceADKFactoryImpl.createTagging();
                }

                /// <summary>
                /// Uninitializes the <see cref="DynaTraceADKFactory" />
                /// </summary>
                public static void uninitialize() {
                        DynaTraceADKFactoryImpl.uninitialize();
                }
        }
}
```

Tag a request

Complete the following steps to tag a remote call.

Client-Side (Caller)

1. Retrieve the current AppMon TraceTag.
2. Specify the location at which to include the server-side path, by inserting a *link*.
3. Send the *AppMon* TraceTag along with the request to the server-side

Server-Side (Callee)

1. Read the remote *AppMon* TraceTag.
2. Place this tag on the server side.
3. Start the server-side PurePath and continue processing.
4. Mark the end of the server-side PurePath, after the request was processed.

Retrieve the TraceTag on the client side

You can retrieve the TraceTag either as a byte array or as a string. You can also use a predefined custom tag.

Option 1: byte array

```
byte[] dtTag = tagging.getTag();
```

Option 2: string

```
string tagAsString = tagging.getTagAsString();
```

Option 3: custom tagging

```
// assume that some kind of unique id exists
byte[] requestId = protocolData.getRequestId();
CustomTag customTag = tagging.createCustomTag(requestId);
```

The `getTag` must be called each time before issuing a remote call to the server side, even if it's still in the same transaction to the same server. The tag that is retrieved by `getTag` identifies just one call and cannot be reused.

When using custom tags, any request or message ID that is used to identify the remote call must be unique. Another call requires a new, unique identifier.

Link on client-side

The next step on the client side is to specify where in the PurePath to include the server-side path, and whether the invocation is synchonous or asynchronous. You may or may not specify the AppMon tag. If you omit the tag parameter, the ADK internally uses the current one, which is identical to the recently retrieved TraceTag.

> **Important**
>
> Calling an ADK function between `getTag*` and `linkClientPurePath` is not permitted. If it is necessary to use any other calls between them, or if there are other instrumented methods in between, it is **required** to specify the AppMon TraceTag explicitly.

Use cases

Synchronous use case, using current trace tag:

```
tagging.linkClientPath(false);
```

Asynchronous use case, specifying the trace tag explicitly:

```
tagging.linkClientPath(true, dtTag);
```

Asynchronous use case, using a custom tag. In this case the tag must be specified:

```
tagging.linkClientPath(true, customTag);
```

> **Warning**
>
> Make sure that `linkClientPath(...)` is invoked before `startServerPath()`: call `linkClientPurePath` before actually transmitting the tag data to the server.
> Calling any ADK functions (logging, other `getTag` calls, etc…) between `getTag/getTagAsString` and `linkClientPath` is not permitted, unless you specify the AppMon TraceTag explicitly in the `tagging.linkClientPath(...)` call.

Place the TraceTag on the server side

You can set the TraceTag either as a byte array or as a string, depending on how you retrieved the tag on the client side.

> **Warning**
>
> Make sure that you set the tag before you start or end the server-side subpath.

Option 1: byte array

```
byte[] tag = [read trace tag];
tagging.setTag(tag);
```

A custom tag is always handled as a byte array:

```
byte[] customTag = [get the id from the request];
tagging.setCustomTag(customTag);
```

Option 2: string

```
string tagAsString = [read trace tag];
tagging.setTagFromString(tagAsString);
```

Mark the server-side path's beginning and end

On the server side, you must specify where your path starts and ends. Enclose the actual processing code with `startServerPurePath()` and `endServerPurePath()`.

```
public void waitForRequest() { ... }

public void execute(Request req) { ... }

public void handleRequest(Request req) {
  tagging.setTag(req.getTag());
  tagging.startServerPurePath();
  execute(req);
  tagging.endServerPurePath();
}

public void run() {
  while(true) {
    Request req = waitForRequest();
    handleRequest(req);
  }
}
```

The tagging interface

The Tagging object provides all methods that an application needs to attach to or continue PurePaths.

```
namespace DynaTrace.DynaTraceADK {
        /// <summary>
        /// <see cref="Tagging" /> provides the user with everything needed for
        /// implementing tagging for proprietary protocols.
        /// Use <see cref="DynaTraceADKFactory" /> to acquire an instance
        /// </summary>
        public interface Tagging {
                /// <summary>
                /// Gets the current tag on the client side represented
                /// as byte array for serialization by the user.
                /// </summary>
                /// <returns>a byte array representing the tag</returns>
                byte [] getTag();

                /// <summary>
                /// Gets the current tag on the client side represented
                /// as string for serialization by the user.
```

```
            /// </summary>
            /// <returns>a string representing the tag</returns>
            string getTagAsString();

            /// <summary>
            /// Sets the tag on the server side from a byte array as
            /// deserialized by the user.
            /// </summary>
            /// <param name="tag">a byte array representing the tag</param>
            void setTag(byte [] tag);

            /// <summary>
            /// Sets the tag on the server side from a string as
            /// deserialized by the user.
            /// </summary>
            /// <param name="tag">a strin representing the tag</param>
            void setTagFromString(string tag);

            /// <summary>
            /// Inserts a synchronous or asynchronous link on the client side.
            /// </summary>
            /// <param name="asynchronous">true if asynchronous, false if synchronous
            /// <remarks>
            /// There cannot be any instrumented calls between  <see cref="M:getTag()
            /// resp. <see cref="M:getTagAsString()" /> and this call.
            /// </remarks>
            void linkClientPurePath(bool asynchronous);

            /// <summary>
        /// Inserts a synchronous or asynchronous link on the client side.
        /// </summary>summary>
        /// <param name="asynchronous">true if asynchronous, false if synchronous</param>
        /// <param name="tag">specify a certain tag to be linked with (this allows
        ///   having instrumented calls between {@link #getTag()} and this call).
        ///   The tag can be specified either as byte array or as string.</param>
        void linkClientPurePath(bool asynchronous, object tag);

            /// <summary>
            /// Starts a server-side PurePath
            /// </summary>
```

```
            void startServerPurePath();

            /// <summary>
            /// Ends a server-side PurePath
            /// </summary>
            void endServerPurePath();

            /// <summary>
            /// Convert a tag to its string format
            /// </summary>
        /// <param name="tag">dynaTrace tag as byte array</param>
        /// <returns>the dynaTrace tag's string representation</returns>
        string convertTagToString(byte[] tag);

            /// <summary>
            /// Convert a dynaTrace string tag to it's byte representation
            /// </summary>
        /// <param name="tag">tag in string format</param>
        /// <returns>the tag's byte array representation</returns>
        byte[] convertStringToTag(string tag);

            /// <summary>
            /// Check if specified tag is valid.
            /// </summary>
        /// <param name="tag">dynaTrace tag</param>
        /// <returns>true, if the specified tag is a valid tag</returns>
        bool isTagValid(object tag);

        /// <summary>Create a custom tag object</summary>
        /// <param name="tagData">byte array containing the custom tag</param>
        /// <returns>CustomTag object</returns>
        CustomTag createCustomTag(byte[] tagData);

            /// <summary>
        /// Set specified custom tag at server side. The function has the same
        /// functionality as setTag(), but allows to continue a PurePath by using
        /// the specified custom tag.
            /// </summary>
        /// <param name="customTag">Custom tag to be continued</param>
        void setCustomTag(byte[] customTag);
```

```
    }

    /// <summary>
    /// Custom tagging: wrapper class for custom tags
    /// </summary>
    public interface CustomTag {
        /// <summary>
        /// Get current custom tag as byte array
        /// </summary>
        /// <returns>custom tag if present, null otherwise</returns>
        byte[] getTag();

        /// <summary>
        /// Get string representation of this custom tag in the format <currentTag>:<prev
        /// </summary>
        /// <returns>hex string representation of this tag</returns>
        string asString();
    }
}
```

Tagging ADK for Java

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Linking client & server paths

Using the ADK

The tagging interface

Related pages

Native ADK

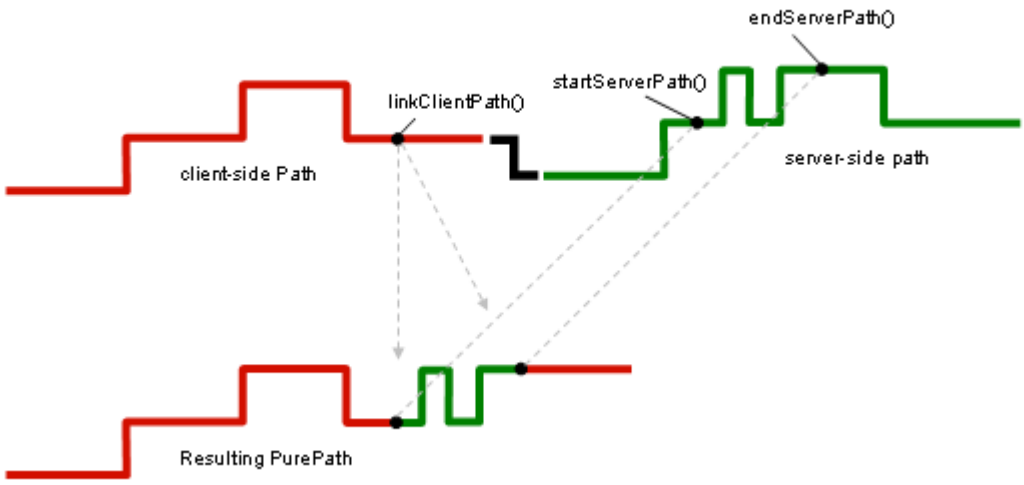
Agent Development Kit (ADK)

Downloads

Use the *Tagging ADK for Java* for protocol tagging only. In contrast to the Native ADK, the tagging ADK provides no means for instrumenting code, because instrumentation is provided by the various Java Sensors.
The Tagging ADK for Java is part of and is delivered with the Agent Development Kit (ADK).

Linking client & server paths

The following diagram illustrates how tagging links the Client-side path of a transaction to the Server-side path to create one PurePath for the entire transaction.



Linking Client-side to Server-side Path

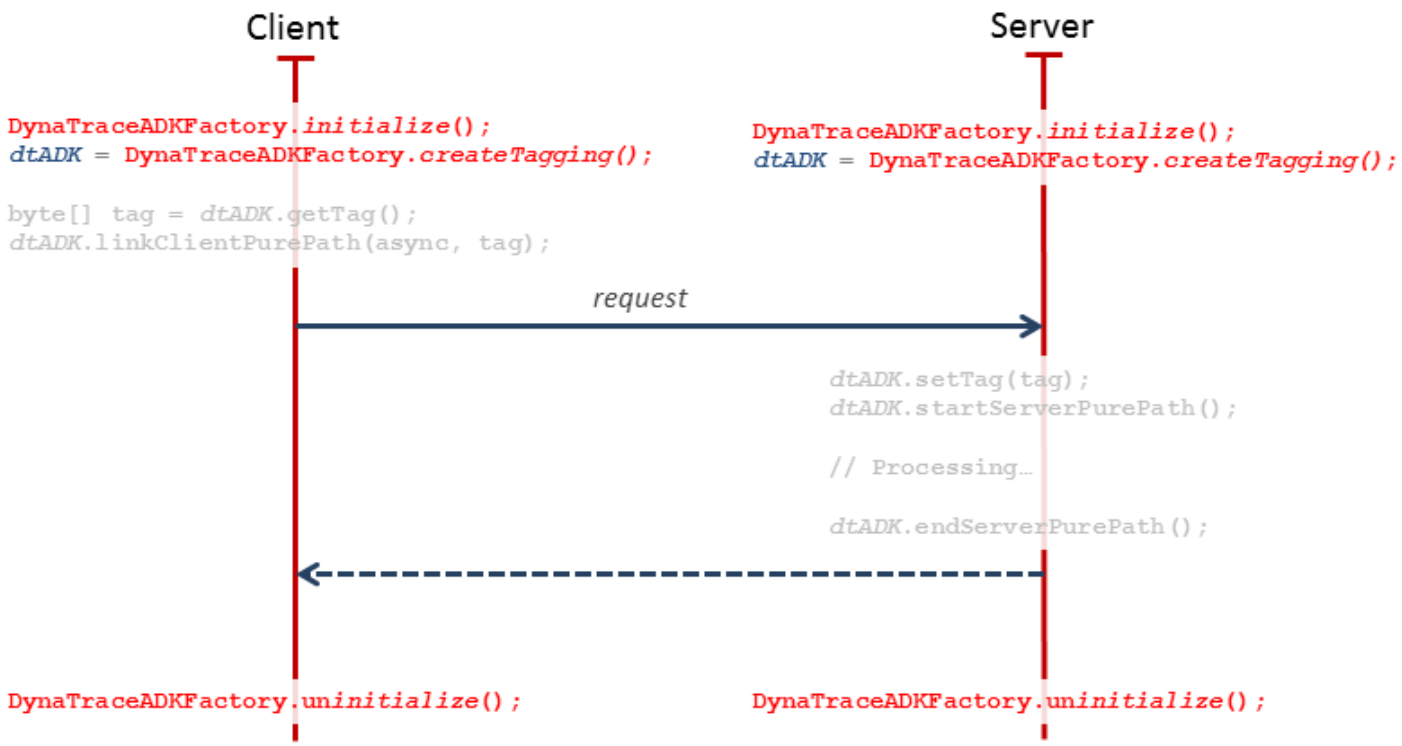
Using the ADK

Prerequisite

Include the library `com.dynatrace.adk.jar` from the ADK package into your project's classpath.

Initialize the Tagging ADK

Coding the initialization is illustrated by the figure below. The initialization step is required on both the client side and the server-side, once per process.



Initializing DynaTraceADKFactory

After successful initialization, `DynaTraceADKFactory` lets you create a `Tagging` object, which serves as the interface to the Agent.

Initialize `DynaTraceADKFactory` before creating a `TaggingADK` object:

```
DynaTraceADKFactory.initialize();
```

Use the following code to retrieve a `TaggingADK` object:

```
Tagging tagging = DynaTraceADKFactory.createTagging();
```

To free up resources allocated by the `DynaTraceADKFactory`, uninitialize the factory before your application quits:

```
DynaTraceADKFactory.uninitialize();
```

Uninitializing the `DynaTraceADKFactory` is necessary to disconnect properly from the AppMon Server and Collector, and to free up allocated resources.

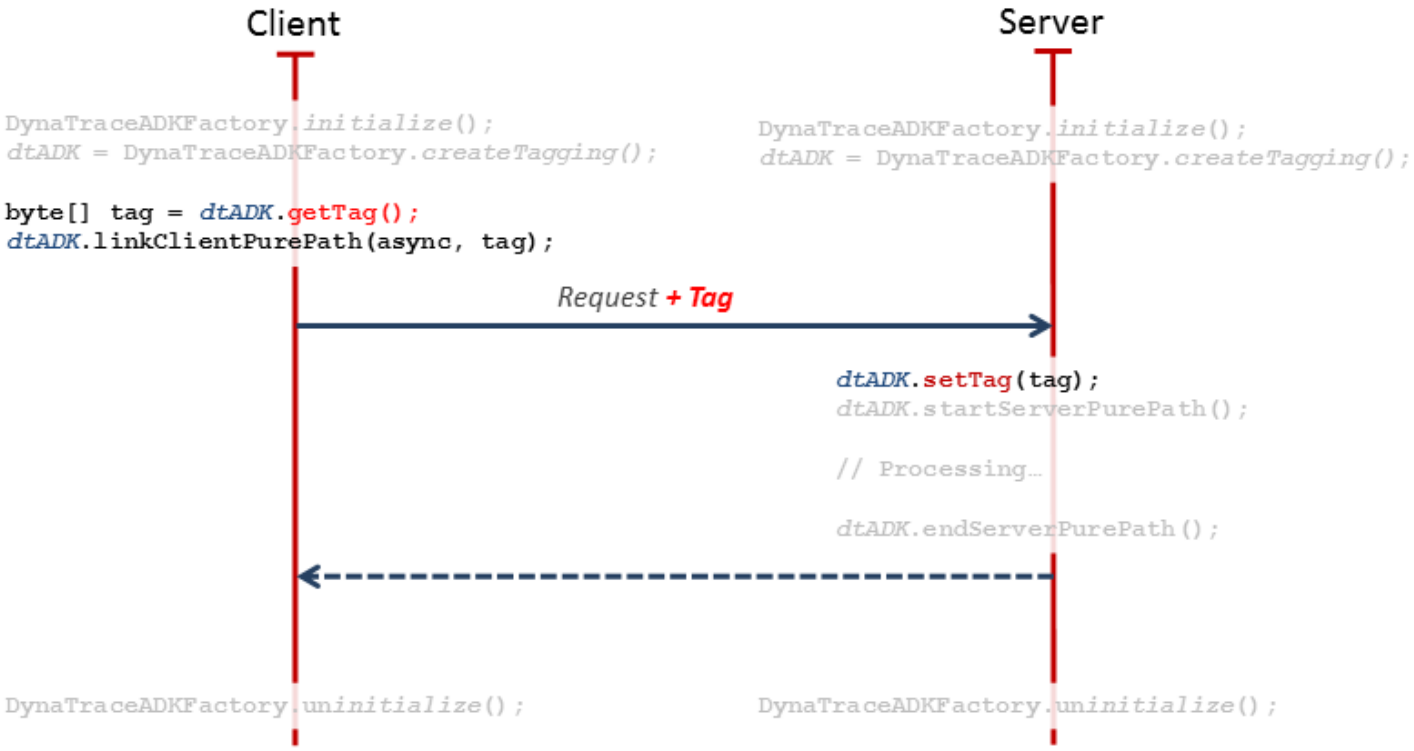
Tag a request

Client-Side (Caller)

1. Retrieve the current *AppMon* TraceTag.
2. Specify where to include the server-side path (insert a link).
3. Send the *AppMon* TraceTag along with the request to the server side.

Server-Side (Callee)

1. Read the remote *AppMon* TraceTag.
2. Place this tag on the server side.
3. Start the server-side PurePath and continue processing.
4. After the request is processed, mark the end of the server-side PurePath.



Tagging a Request

Retrieve the AppMon tag

You can retrieve the AppMon TraceTag either as a byte array or as a string. You can also use a predefined *custom tag*.

Retrieve the TraceTag as a byte array

```
byte[] dtTag = tagging.getTag();
```

Retrieve the TraceTag as a string

```
String tagAsString = tagging.getTagAsString();
```

Use a custom tag to retrieve the TraceTag

```
// assume that some kind of unique id exists
byte[] requestId = protocolData.getRequestId();
CustomTag customTag = tagging.createCustomTag(requestId);
```

You must call `getTag` each time before issuing a remote call to the server, even if it is in the same transaction to the same server. The tag that is retrieved by `getTag` identifies just one call. It cannot be reused.

When you use custom tags, any request or message ID that is used to identify the remote call must be unique. Another call requires a new, unique identifier.

Link on the client side

In the client side, specify the location in the PurePath to include the server-side path, and specify whether the invocation is synchonous or asynchronous.

You may or may not specify the AppMon tag. If you omit the tag parameter, the ADK uses the current parameter, which is identical to the recently retrieved trace tag. This is why calling any ADK functions between `getTag*` and `linkClientPurePath` is not allowed. If you must have any other calls or other instrumented methods between `getTag*` and `linkClientPurePath`, it is **required** to specify the AppMon TraceTag explicitly.

This synchronous use case uses the current TraceTag.

```
tagging.linkClientPurePath(false);
```

This asynchronous use case specifies the TraceTag explicitly:

```
tagging.linkClientPurePath(true, dtTag);
```

This asynchronous use case uses a custom tag. In this case, the tag must be specified explicitly.

```
tagging.linkClientPurePath(true, customTag);
```

> **Note**
>
> Make sure `linkClientPath(...)` is invoked **before** `startServerPurePath()`.
> You must call `linkClientPurePath` before actually transmitting the tag data to the server.
> Calling any ADK functions between `getTag` (or `getTagAsString`) and `linkClientPurePath` (e.g. logging or other `getTag` calls) is not permitted **unless** you specify the AppMon TraceTag explicitly in the `tagging.linkClientPurePath(...)` call.

Place a TraceTag on the server side

You can set the TraceTag either as a byte array or as a string. Use the same type you used to retrieve the tag on the client side.

The TraceTag was transferred as a byte array:

```
byte[] tag = [read trace tag];
tagging.setTag(tag);
```

The TraceTag was transferred as a string:

```
String tagAsString = [read trace tag];
tagging.setTagFromString(tagAsString);
```

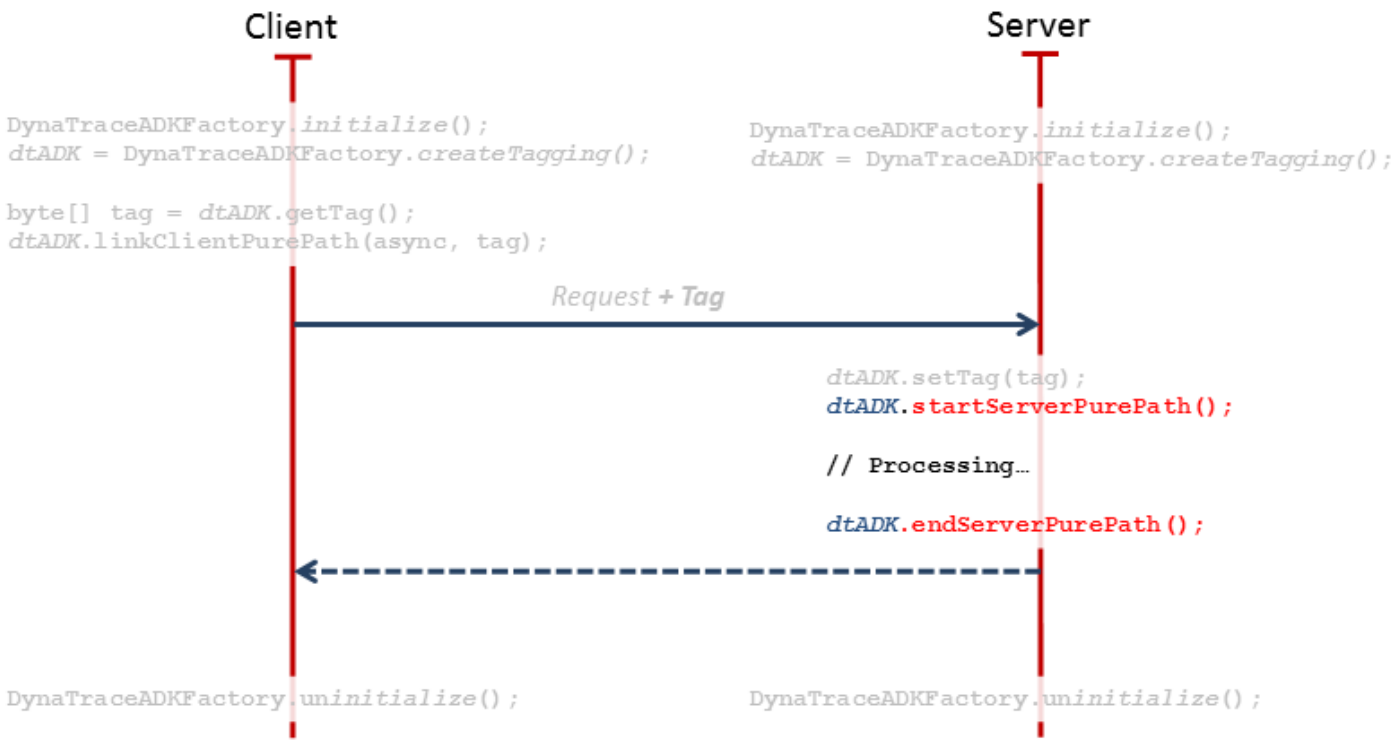
A custom tag is always handled as a byte array:

```
byte[] customTag = [get the id from the request];
tagging.setCustomTag(customTag);
```

> **Note**
>
> Make sure you set the tag before you start or end the server-side subpath.

Mark the server-side path's start and end

On the server side, you must specify where your path starts and ends. Enclose the actual processing code by `startServerPurePath()` and `endServerPurePath()`.



Specifying a PurePath Start and End

```
public void waitForRequest() { ... }

public void execute(Request req) { ... }

public void handleRequest(Request req) {
  tagging.setTag(req.getTag());
  tagging.startServerPurePath();
  execute(req);
  tagging.endServerPurePath();
}

public void run() {
  while(true) {
    Request req = waitForRequest();
    handleRequest(req);
  }
}
```

The tagging interface

The `Tagging` object provides all the methods an application needs to attach to or continue PurePaths.

```
/**
 * Tagging provides the user with everything needed for
 * implementing tagging for proprietary protocols.
 * Use DynaTraceADKFactory to acquire an instance.
 */
public interface Tagging {
    /**
     * Gets the current dynaTrace tag on the client side represented
     * as byte array for serialization by the user.
     * @return a byte array representing the tag
     * Please instrument the calling method\!
     */
    byte[] getTag();

    /**
     * Gets the current dynaTrace tag on the client side represented
     * as String for serialization by the user
```

```
     * as String for serialization by the user.
     * @return a String representing the tag
     * Please instrument the calling method\!
     */
    String getTagAsString();

    /**
     * Sets the dynaTrace tag on the server side from a byte array as
     * deserialized by the user.
     * @param tag a byte array representing the tag
     */
    void setTag(byte[] tag);

    /**
     * Sets the dynaTrace tag on the server side from a String as
     * deserialized by the user.
     * @param tag a String representing the tag
     */
    void setTagFromString(String tag);

    /**
     * Verifies whether a given tag is valid or not
     * @param tag    dynaTrace tag either as byte array or string
     * @return    true, if tag is valid, false otherwise
     */
    boolean isTagValid(Object tag);

    /**
     * Inserts a synchronous or asynchronous link on the client side.
     * Note: There cannot be any instrumented calls between getTag()
     *       resp. getTagAsString() and this call.
     * @param asynchronous true if asynchronous, false if synchronous
     * Please instrument the calling method\!
    */
    void linkClientPurePath(boolean asynchronous);

    /**
     * Inserts a synchronous or asynchronous link on the client side.
     * @param asynchronous true if asynchronous, false if synchronous
     * @param tag specify a certain tag to be linked with (this allows
     *    having instrumented calls between getTag() and this call).
```

```
     *    The tag can be specified either as byte array or as string.
     * Please instrument the calling method\!
     */
    void linkClientPurePath(boolean asynchronous, Object tag);

    /**
     * Starts a server-side PurePath
     */
    void startServerPurePath();

    /**
     * Ends a server-side PurePath
     */
    void endServerPurePath();

    /**
     * Returns a wrapper for a given Runnable which tags the Runnable as a server-side su
     * path. The sub path will start when the run method of the given Runnable is called.
     *
     * Important: To link the client side path with the new sub path,
     * linkClientPurePath(boolean) has to be called immediately before the returned
     * Runnable is scheduled for execution.
     *
     * @param runnable the runnable to tag
     * @return the runnable to use instead of the given runnable
     */
    Runnable createServerPathRunnable(Runnable runnable);

    /**
     * Convert the specified dynaTrace tag to its string format.
     * @param  tag    dynaTrace tag (byte array)
     * @return dynaTrace tag as string
     */
    public String convertTagToString(byte[] tag);

    /**
     * Convert the specified dynaTrace string tag to the tag's byte array representation
     * @param  tag    dynaTrace tag (string format)
     * @return dynaTrace tag as byte array
     */
    public byte[] convertStringToTag(String tag);
```

```
    public byte[] convertStringToTag(String tag);

    /**
     * Custom tagging: wrapper class for custom tags
     */
    public interface CustomTag {
        /**
         * Get current custom tag as byte array
         * @return    custom tag if present, null otherwise
         */
        public byte[] getTag();

        /**
         * Get string representation of this custom tag in the format <currentTag>:<prevT
         * @return hex string representation of this tag
         */
        public String asString();
    }

    /**
     * Create a custom tag object
     * @param tagData     byte array containing the custom tag
     * @return CustomTag object
     */
    CustomTag createCustomTag(byte[] tagData);

    /**
     * Set specified custom tag at server side. The function has the same
     * functionality as setTag(), but allows to continue a PurePath by using
     * the specified custom tag.
     */
    void setCustomTag(byte[] customTag);
}
```

CICS and IMS Tagging ADK

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Link the stub

Include API function declarations for C and PL/I

API naming conventions

API function examples for COBOL, PL/I and C

API return codes

Negative return codes

API notes

API include and sample files

Related pages

Agent Development Kit (ADK)

Downloads

Latest update

The CICS and IMS ADK supports customer-specific tagging that uses either standard AppMon tags or customer-supplied custom tags. AppMon 7.0 and later includes additional APIs that let you add nodes to an existing PurePath and associate arguments and return values with them. This ADK is part of the CICS and IMS Agents. It requires no additional license.

See Agent Development Kit (ADK) for general tasgging information.

Link the stub

`DTSTUB` provides access to ADK functions. Add a `DD` statement and an `INCLUDE` to the binder step to include it in your program. For example:

```
//SZDTLOAD  DD  DISP=SHR,DSN=_hlq_.SZDTLOAD
...
  INCLUDE SZDTLOAD(DTSTUB)
```

Include API function declarations for C and PL/I

PL/I:

```
%INCLUDE DTADKPLI;
```

C or C++:

```
#include <DTADK>
```

See API Include and Sample Files (attachments) at the bottom of this page.

API naming conventions

Each function name starts with `DT`, followed by characters that define the purpose of the function and an optional suffix that indicates the argument type.

```
DTaaaabb
^ ^   ^
| |   `- Parameter type:
| |       - 'TN'  text - null terminated string
| |       - 'TP'  text - PL/I string
| |       - 'TF'  text - fixed length string
| |       - 'A'   byte array
| |       - 'S'   16 bit signed integer
| |       - 'I'   32 bit signed integer
| |       - 'L'   64 bit signed integer
| |
| `-- Method type:
|      - 'SP'   Start Path
|      - 'SLP'  Start Linked Path
|      - 'SCLP' Start Custom Linked Path
|      - 'IL'   Insert Link
|      - 'ICL'  Insert Custom Link
|      - 'IAL'  Insert Async Link
|      - 'IACL' Insert Async Custom Link
|      - 'EP'   End Path
|      - 'EN'   Enter
|      - 'EX'   Exit
|      - 'EXEX' Exit with Exception
|      - 'DC'   Data Capture
|
  `---- 'DT' prefix
```

API function examples for COBOL, PL/I and C

The API supports all common z/OS languages.

Each API call returns an integer value to indicate failure, or zero to indicate success. See the list of return codes below.

Start path

Start a new CICS or IMS PurePath that is unrelated to any existing PurePath.

Note that CICS LINKs, CICS STARTs, and IMS program to program message switches initiated by the transaction being traced automatically create their own linked subpaths.

COBOL examples

```
CALL "DTSP" RETURNING RC.
CALL "DTSPTF" USING FIXED_LEN_STRING, LENGTH, CCSID RETURNING RC.
```

PL/I examples

```
rc = DTSP();
rc = DTSPTN(c_string, ccsid);
rc = DTSPTP(pli_string, ccsid);
rc = DTSPTF(fixed_len_string, length, ccsid);
```

C function declarations

```
int DTSP (void);
int DTSPTN (const char* c_string, const int* ccsid);
int DTSPTP (const void* pli_string, const int* ccsid);
int DTSPTF (const char* fixed_len_string, const int* length, const int* ccsid);
```

Parameters

`c_string`	Path name specified as a null-terminated string.
`pli_string`	Path name specified as a PLI varying string.
`fixed_len_string`	Path name specified as a fixed-length string with an explicit length.
`length`	Length of the specified fixed-length string.
`ccsid`	Coded character set identifier for the string. Specify a value of zero (0) to use the CICS or IMS region's default code page.

Start linked path

Start a CICS or IMS PurePath that is a continuation of another PurePath, typically in response to an inbound service request from another platform.

Note that CICS LINKs, CICS STARTs, and IMS program to program message switches initiated by the transaction being traced automatically create their own linked subpaths.

COBOL examples

```
CALL "DTSLPA" USING BINARYTAG, LENGTH RETURNING RC.
CALL "DTSCLPA" USING CUSTOMTAG, LENGTH RETURNING RC.
CALL "DTSLPTF" USING FIXED_LEN_STRING, LENGTH RETURNING RC.
```

PL/I examples

```
rc = DTSLPA(binarytag, length);
rc = DTSCLPA(customtag, length);
rc = DTSLPTN(c_string);
rc = DTSLPTP(pli_string);
rc = DTSLPTF(fixed_len_string, length);
```

C function declarations

```
int DTSLPA (const void* binarytag, const int* length);
int DTSCLPA (const void* customtag, const int* length);
int DTSLPTN (const char* c_string);
int DTSLPTP (const void* pli_string);
int DTSLPTF (const char* fixed_len_string, const int* length, const int* ccsid);
```

Parameters

`binarytag`	Standard AppMon binary tag specified as a byte array.
`customtag`	Custom tag specified as a byte array.

`c_string`	Standard AppMon string tag specified as a null-terminated string.
`pli_string`	Standard AppMon string tag specified as a PLI varying string.
`fixed_len_string`	Standard AppMon string tag specified as a fixed-length string with an explicit length.
`length`	Length of the specified fixed-length string or byte array. Note that this is the length of the tag, not the buffer.

End path

Terminate the current PurePath created by a **Start Path** or **Start Linked Path** API call.

The **End Path** API may not be required because the PurePath ends automatically at the end of the transaction, or when another **Start Path** or **Start Linked Path** API is used in the same transaction to trace another input message. **End Path** should be used before a long running transaction waits for another input message to arrive. When used, **End Path** must be issued by the same transaction that issued the **Start Path** or **Start Linked Path**, regardless of CICS LINKs, CICS STARTs, or IMS program to program message switches which create their own linked subpaths automatically, which are independent of the ADK.

COBOL example

```
CALL "DTEP" RETURNING RC.
```

PL/I example

```
rc = DTEP();
```

C function declaration

```
int DTEP (void);
```

Parameters

None

Insert link

Link the current PurePath to another PurePath, typically in response to an outbound request from the CICS or IMS application to a server on another platform.

The string or array used with these APIs returns a new AppMon tag that is sent with the outbound request, but `DTICLA` requires the caller to supply a custom tag.

COBOL examples

```
CALL "DTILA" USING BINARYTAG, TAG_BUFFER_LEN RETURNING RC.
CALL "DTICLA" USING CUSTOMTAG, LENGTH RETURNING RC.
CALL "DTILTF" USING FIXED_LEN_STRING, TAG_BUFFER_LEN RETURNING RC.
```

PL/I examples

```
rc = DTILA(binarytag, tag_buffer_len);
rc = DTICLA(customtag, length);
rc = DTILTN(c_string, tag_buffer_len);
rc = DTILTP(pli_string, tag_buffer_len);
rc = DTILTF(fixed_len_string, tag_buffer_len);
```

C function declarations

```
int DTILA (void* binarytag, int* tag_buffer_len);
int DTICLA (const void* customtag, const int* length);
int DTILTN (char* c_string, const int* tag_buffer_len);
int DTILTP (void* pli_string, const int* tag_buffer_len);
int DTILTF (char* fixed_len_string, int* tag_buffer_len);
```

Parameters

`binarytag`	Standard AppMon binary tag returned as a byte array (output).

`customtag`	Custom tag specified as a byte array (input).
`c_string`	Standard AppMon string tag returned as a null-terminated string (output).
`pli_string`	Standard AppMon string tag returned as a PLI varying string (output).
`fixed_len_string`	Standard AppMon string tag returned as a fixed-length string with an explicit length (output).
`length`	Length of the supplied byte array (input).
`tag_buffer_len`	Length of the supplied buffer that is used to return the new tag (input). Also used as an output parameter by `DTILA` and `DTILTF` to return the actual length of the returned array or string.

Insert Async link

Link the current PurePath to another PurePath, typically in response to an outbound request from the CICS or IMS application to a server on another platform. These APIs duplicate the Insert Link APIs described above, but also insert links for outbound requests that do not necessarily operate in a synchronous manner.

The string or array used with these APIs returns a new AppMon tag that is sent with the outbound request, but `DTIACLA` requires the caller to supply a custom tag.

COBOL examples

```
CALL "DTIALA" USING BINARYTAG, TAG_BUFFER_LEN RETURNING RC.
CALL "DTIACLA" USING CUSTOMTAG, LENGTH RETURNING RC.
CALL "DTIALTF" USING FIXED_LEN_STRING, TAG_BUFFER_LEN RETURNING RC.
```

PL/I examples

```
rc = DTIALA(binarytag, tag_buffer_len);
rc = DTIACLA(customtag, length);
rc = DTIALTN(c_string, tag_buffer_len);
rc = DTIALTP(pli_string, tag_buffer_len);
rc = DTIALTF(fixed_len_string, tag_buffer_len);
```

C function declarations

```
int DTIALA (void* binarytag, int* tag_buffer_len);
int DTIACLA (const void* customtag, const int* length);
int DTIALTN (char* c_string, const int* tag_buffer_len);
int DTIALTP (void* pli_string, const int* tag_buffer_len);
int DTIALTF (char* fixed_len_string, int* tag_buffer_len);
```

Parameters

`binarytag`	Standard AppMon binary tag returned as a byte array (output).
`customtag`	Custom tag specified as a byte array (input).
`c_string`	Standard AppMon string tag returned as a null-terminated string (output).
`pli_string`	Standard AppMon string tag returned as a PLI varying string (output).
`fixed_len_string`	Standard AppMon string tag returned as a fixed-length string with an explicit length (output).
`length`	Length of the supplied byte array (input).
`tag_buffer_len`	Length of the supplied buffer that returns the new tag (input). Also used as an output parameter by `DTIALA` and `DTIALTF` to return the actual length of the returned array or string.

Enter Node

Add a node to the currently active CICS or IMS subpath to indicate the execution start of a new program or activity.

The number of Exit Node and Exit Node with Exception API calls must equal the number of Enter Node API calls or the Purepath is marked as corrupted. Typically, this API is used by the program calling the named program but it could also be issued at the beginning of the called program. The Enter API returns a 32-but integer token that must be saved so it can be specified on the associated Exit or Exit Exception API call.

COBOL examples

```
CALL "DTENTF" USING FIXED_LEN_STRING, LENGTH, TOKEN RETURNING RC.
```

PL/I examples

```
rc = DTENTN(c_string, token);
rc = DTENTP(pli_string, token);
rc = DTENTF(fixed_len_string, length, token);
```

C function declarations

```
int DTENTN (const char* c_string, int* token);
int DTENTP (const void* pli_string, int* token);
int DTENTF (const char* fixed_len_string, const int* length, int* token);
```

Parameters

`c_string`	Program name specified as a null-terminated string.
`pli_string`	Program name specified as a PLI varying string.
`fixed_len_string`	Program name specified as a fixed-length string with an explicit length.
`length`	Length of the specified fixed-length string.

`token`	A 32-bit binary integer token used to associate this node with the corresponding Exit or Exit Exception (output).

Exit Node

Indicate that the program or activity associated with the most recent Enter Node API call has ended.

Each Exit Node or Exit Node with Exception request pairs with the most recent Enter node request. Subpaths with unbalanced numbers of Enter and Exit APIs are marked as corrupted.

Related Enter Node and Exit Node APIs should be issued by the same program in most cases. This is so an Exit does not attempt to end a node associated with a program with its own node due to a LE call, CICS LINK, or IMS program to program message switch invoked since the most recent Enter Node API call.

COBOL example

```
CALL "DTEX" USING TOKEN RETURNING RC.
```

PL/I example

```
rc = DTEX(token);
```

C function declarations

```
int DTEX (const int* token);
```

Parameters

`token`	The 32-bit integer token that was returned by the most recent DTEN API call.

Exit Node with Exception

Indicate that the program or activity associated with the most recent Enter Node API call ended with an error.

Each Exit Node or Exit Node with Exception request pairs with the most recent Enter node request. Subpaths with unbalanced numbers of Enter and Exit APIs are marked as corrupted. Related Enter Node and Exit Node APIs should be issued by the same program in most cases. This is so an Exit does not attempt to end a node associated with a program with its own node due to a LE call, CICS LINK, or IMS program to program message switch invoked since the most recent Enter Node API call.

COBOL example

```
CALL "DTEXEX" USING TOKEN RETURNING RC.
```

PL/I example

```
rc = DTEXEX(token);
```

C function declarations

```
int DTEXEX (const int* token);
```

Parameters

`token` The 32-bit integer token that was returned by the most recent DTEN API call.

Data Capture

Provide an argument capture value for a subsequent Enter Node API, or the return value for a subsequent Exit Node or Exit Node with Exception API.

Multiple Data Capture APIs may be issued before an Enter Node API, but only one return value may be associated with an Exit Node or Exit Node with Exception API.

An Enter, Exit, or Exit with Exception API should always immediately follow one or more Data Capture APIs to prevent the captured value being unintentionally associated with another node.

COBOL examples

```
CALL "DTDCS" USING INT16_VALUE RETURNING RC.
CALL "DTDCI" USING INT32_VALUE RETURNING RC.
CALL "DTDCL" USING INT64_VALUE RETURNING RC.
CALL "DTDCTF" USING FIXED_LEN_STRING, LENGTH, CCSID RETURNING RC.
```

PL/I examples

```
rc = DTDCS(int16_value);
rc = DTDCI(int32_value);
rc = DTDCL(int64_value);
rc = DTDCTN(c_string, ccsid);
rc = DTDCTP(pli_string, ccsid);
rc = DTDCTF(fixed_len_string, length, ccsid);
```

C function declarations

```
int DTDCS (const short* int16_value);
int DTDCI (const int* int32_value);
int DTDCL (const long long* int64_value);
int DTDCTN (const char* c_string, const int* ccsid);
int DTDCTP (const void* pli_string, const int* ccsid);
int DTDCTF (const char* fixed_len_string, const int* length, const int* ccsid);
```

Parameters

`c_string`	Capture value specified as a null-terminated string.
`pli_string`	Capture value specified as a PLI varying string.
`fixed_len_string`	Capture value specified as a fixed-length string with an explicit length.
`length`	Length of the specified fixed-length string.

`ccsid`	Coded character set identifier for the string. Specify a value of zero (0) to use the CICS or IMS region's default code page.
`int16_value`	A 16-bit signed binary value such as a COBOL PIC S9(5) COMP, PLI FIXED BIN 15. C short int, or assembler fullword.
`int32_value`	A 32-bit signed binary value such as a COBOL PIC S9(9) COMP, PLI FIXED BIN 31. C int, or assembler fullword (F).
`int64_value`	A 64-bit signed binary value such as a COBOL PIC S9(18) COMP, PLI FIXED BIN 63. C long long int, or assembler doubleword (FD).

API return codes

All APIs return an integer value.

Return Code	API Types	Action	Description
0	All	As documented	Success
4	SP SLP SCLP	None	A path is already in progress for this transaction.
4	DC EN EX EXEX EP IL ICL IAL IACL	None	No path is in progress for this transaction at this time.
8	EP	None	A different sensor started the path that is in progress for this transaction.
8	SLP	None	The specified AppMon tag's format is invalid or the specified string length doesn't match the actual string tag length.
8	IL IAL	None	The supplied buffer is too small to hold the tag.
8	EX EXEX	None	The supplied token is not valid.
12	SP	None	The specified path name is too long.

12	SLP SCLP ICL IACL	None	The specified tag length is negative or too long.
12	DC	None	The specified capture string is too long (currently limited to 200 bytes).
12	EN	None	The specified node name is too long (currently limited to 8 bytes).
16	SLP	None	A dummy AppMon tag was specified.
2000	Any	Correct calling program	A program check occurred in the CICS or IMS agent while handling an ADK call. This is presumably caused by an incorrect argument.

Negative return codes

A negative return code generally indicates that the AppMon agent is not available, so the application program should continue on the assumption that tracing isn't being performed.

In a CICS environment, the CICS DFHRMCAL interface passes ADK requests to the CICS agent's task related user exit (TRUE). DFHRMCAL returns undocumented negative values such as -2 if the agent is not active. If negative return codes are encountered, you can use the DTAX transaction to see whether the AppMon agent is enabled.

In an IMS environment, a negative return code also indicates that the AppMon IMS agent is not available. Return code -1 means the IMS agent is not installed in this IMS subsystem. Return code -4 means the IMS agent is installed but has been disabled. Return code -5 indicates that the installation process is not completed in this dependent region. Return code -5 is expected if the ADK starts a Purepath in the first transaction to execute after the message region starts or the IMS agent is enabled. Initialization may take a second or two, so this condition may occur for more than one transaction at startup.

Contact AppMon support if the ADK returns negative return codes when AppMon is actively tracing other transactions in the affected dependent region.

API notes

The CICS and IMS ADK supports COBOL, PL/I, Assembler, and C/C++. PL/I and C provide `include` members that declare the API functions.

COBOL programs must specify a type such as `PIC S9(9)` that corresponds to a binary fullword for integer variables such as `RC`, `LENGTH` (not a valid COBOL variable name), `CCSID`, and `TAG_BUFER_LEN`. Array or string variables can be declared as `PIC X(##)`.

All character string values assume the default EBCDIC code page of the CICS or IMS region unless the API accepts an explicit CCSID override.

The binary tag length is 30 bytes and a string tag may be up to 77 bytes. The `tag_buffer_len` specifiing the area length to hold an output tag argument must be set as follows:

- `DTILA` and `DTIALA` API – 30 or more.
- `DTILTF` and `DTIALTF` API – 77 or more.
- `DTILTN` and `DTIALTN` API – 78 or more to allow for a null terminator byte.
- `DTILTP` and `DTIALTP` API – 79 or more to allow for the halfword string length.

Use a buffer size somewhat larger than these minimum sizes to minimize compatibility issues with future AppMon releases.

Input string arguments are currently limited to a maximum length of 200 bytes. This limit does not include the two byte length of a PLI varying string or the null terminator byte of a null terminated string, just the text contents of the string.

API include and sample files

- ADKASMPG - IMS assembler example
- ADKPLIPG - IMS PL/I example
- ADKCOBLP - IMS COBOL example
- DTADKDPL - COBOL program to start a path and CICS LINK to a specified program
- ADKASM - CICS assembler example
- DTADKPLI - PL/I include file
- DTADK - C header file
- ADKCOBOL - CICS COBOL example
- ADKC - CICS C example
- ADKPLI - CICS PL/I example

JavaScript ADK and JavaScript Ajax ADK

On this page

Introduction

Requirements

Using the JavaScript ADK for Flash and Silverlight applications

Related pages

Collecting streaming media measures from Flash players

Introduction

The Javascript ADK provides functions to extend the default behavior of the JavaScript Agent. You can use this ADK to start custom user actions, report events, or extend third-party content detection on a web page.

Both the web server and the Java Agent can handle user experience monitoring performed by the JavaScript Agent, extensible with the JavaScript ADK. To get the data, the User Experience sensor must be placed and active for the web server or the application server.

Requirements

Since 6.5, the JavaScript ADK is bundled with the JavaScript Agent itself. So as long as the agent is present on the page, the ADK functionality is available as well. Thus, the only requirement is to ensure your webpage is injected either automatically or manually.

Asynchronous injection

If asynchronous injection is enabled for the JavaScript Agent, this also means that the ADK will be available as soon as the rest of the code is. Use caution when attempting to use ADK functionality early in the webpage's lifecycle.

You can verify that the JavaScript Agent is injected by checking the `window.dT_` property:

```
if(typeof window.dT_!='undefined'){
  //JavaScript agent is present
}
```

The namespace `dynaTrace` is available, which provides all ADK functions. See the Javadoc included in the samples project (see below) for a list of functions at `http://<serveraddress>/samples/doc/index.html`.

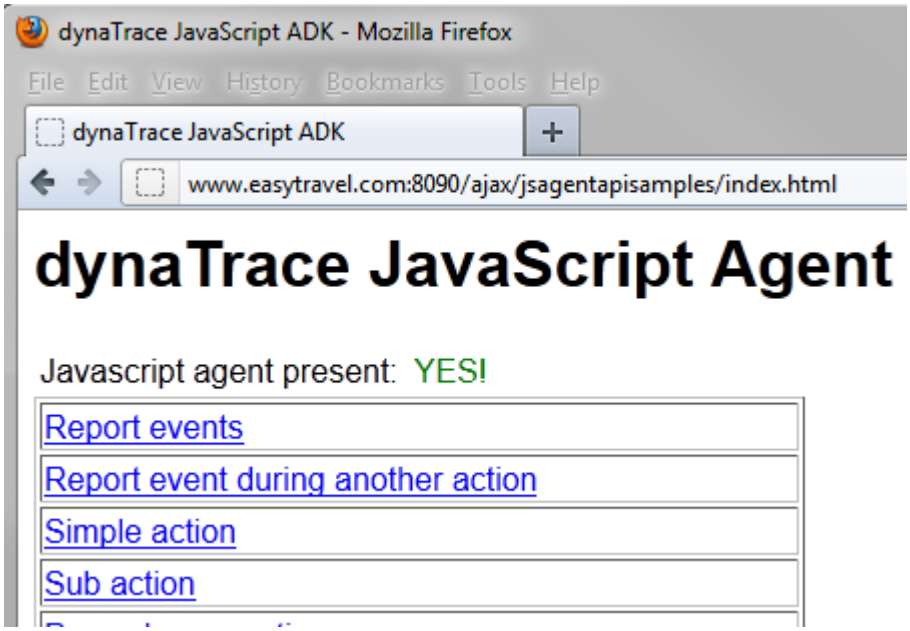
JavaScript ADK samples project

A samples project is available for download to help understand use cases listed below for the JavaScript ADK.

Download and unzip the package, and place the files from the `samples` directory on a web server where the JavaScript Agent is injected.

If, for example, you place the files in the `htdocs/samples` directory of an Apache server, the samples are accessible via `http://<serveraddress>/samples/index.html`.

If the JavaScript Agent is injected on the server, the index page appears as shown in the figure.



JavaScript ADK Index Page

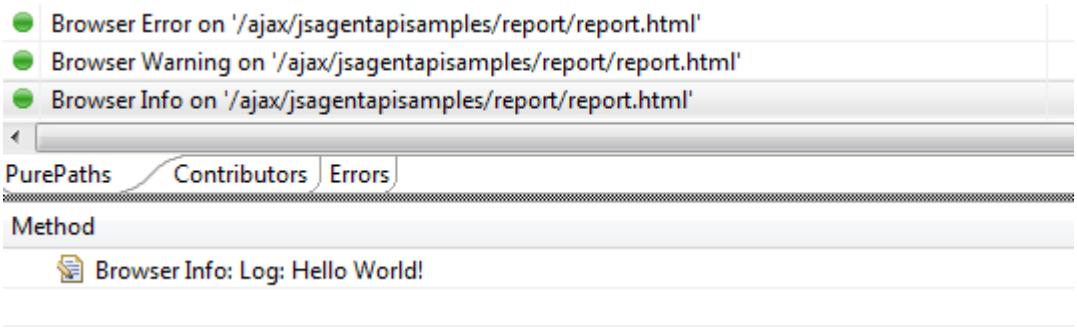
Reporting events, warnings, errors, strings, and values

The JavaScript ADK provides functions to report information to the Server. This information is displayed in the *User Action PurePaths* dashlet:

- `reportEvent(msg)`
- `reportWarning(warning)`
- `reportError(error)`
- `reportValue(key, value)`
- `reportString(key, string)`

Example: Execute Report Functions (/report/report.html)

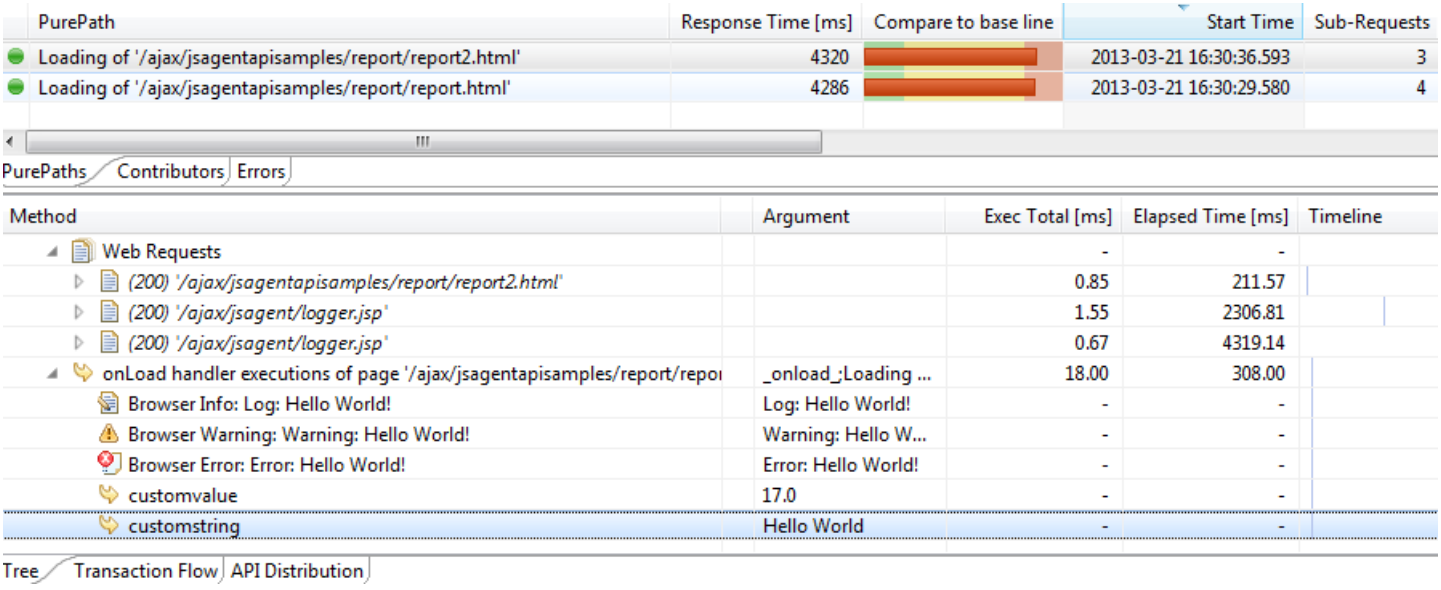
This example provides buttons that execute report functions. The resulting user actions are listed in the dashlet as shown in the figure.



User Actions for report function

Example: Call Report Functions (/report/report2.html)

This example calls the report functions in an `onload` event handler. The messages are automatically appended to the currently running `onload` action. The resulting user actions are listed in the dashlet as shown in the figure.



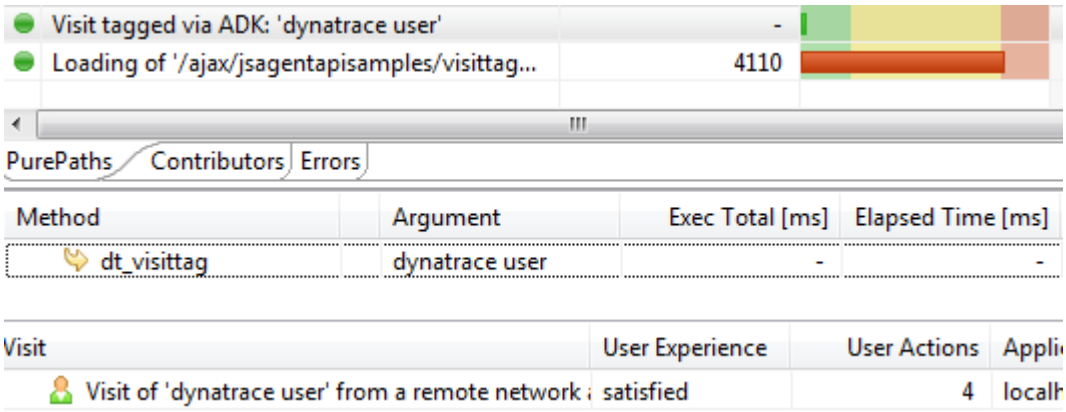
User Actions for onload event handler

Visit tagging

The JavaScript ADK enables you to tag your visits with any value you want. Use `tagVisit(anystring)` to tag the current visit with the specified value.

Example: Tag Visits (/visittagging/report.html)

This example tags a visit with the value of an input field. Enter any string and click **Tag Visit**. The function uses `reportString` with the special key `dt_visittag`. The server handles this special request by tagging a non-tagged visit with the provided string.



Tagged visit

End visit

Example: End Visit (/endVisit/endvisit.html)

By default a visit ends after a configurable inactivity time (default is 30 min.). Via the ADK a visit can be ended forcibly, e.g. when a user logs out from an application.

Simply call `dynaTrace.endVisit()` and the visit will be ended without waiting for the inactivity time. For correlation purposes the visit will still show up as active in the client for about 3 minutes but actions sent after `dynaTrace.endVisit()` will added to a new visit.

Set the application version

The application's version can be manually set on the visit via the function `setAppVersion`.

Call `dynaTrace.setAppVersion(version)` as soon as the ADK has finished loading, where the version String is what you want to set for each visit. Note that this will override the configuration setting specified in the System Profile's **UEM** section.

Starting custom actions

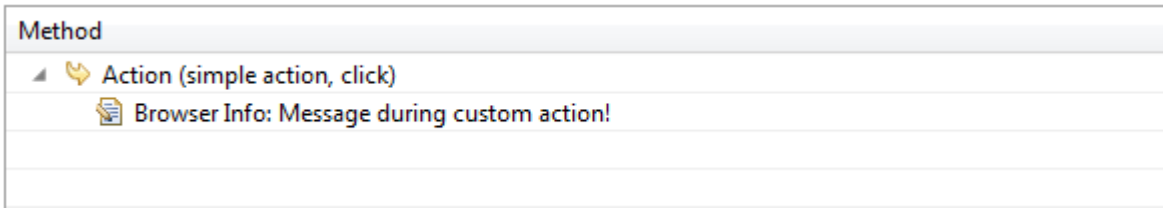
The JavaScript ADK includes functions to start and stop custom user actions.

- `dynaTrace.enterAction(name,type,startTime,parentAction)` starts a new action. If no `parentAction` is provided, a new root action is started. The function returns the name of the new action. This name can differ from the name given as a parameter, when multiple actions with the same name happen.
- `dynaTrace.leaveAction(actionName)` stops the action with the specified name.

Example: Start and Stop an Action Within a Page (/actions/simpleaction.html)

This example provides a button that starts and stops a simple action in an `onclick` handler.

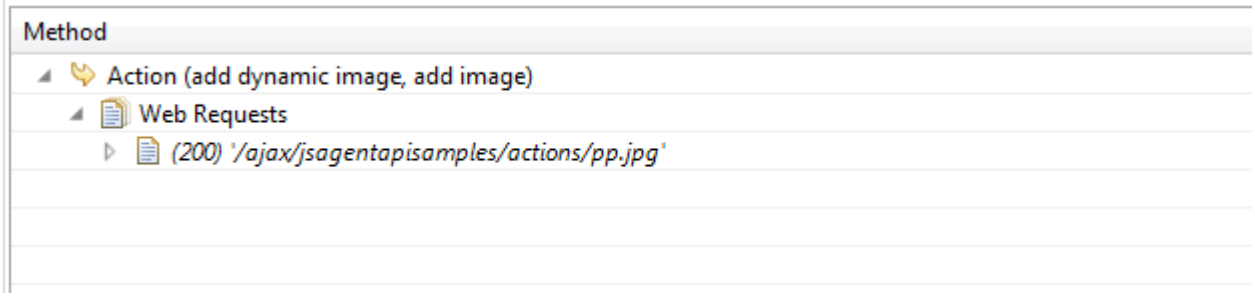
The User Action is shown in the following figure.



onclick Handler User Action

Example: Start and Stop an Action, with Web Requests (/actions/advancedaction.html)

This example provides a button that loads an image dynamically. It calls `enterAction` before loading and `leaveAction` when loading is finished. The web request for the image is automatically added to the User Action in the PurePath, as shown in the following figure.

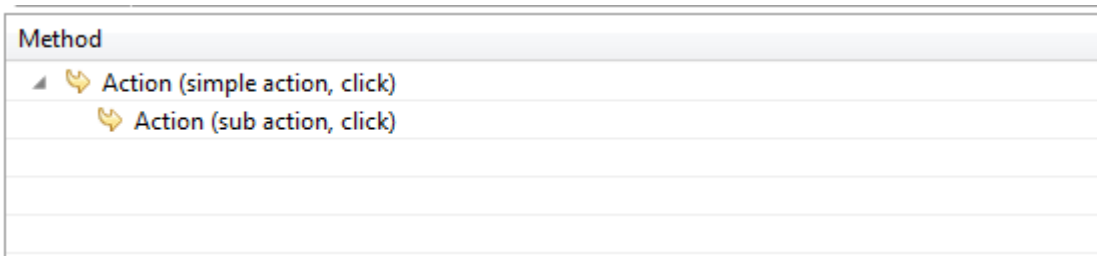


User Action with web request

Example: Create an Action with a Subaction (/actions/subaction.html)

This example provides a button that creates an action with a subaction.

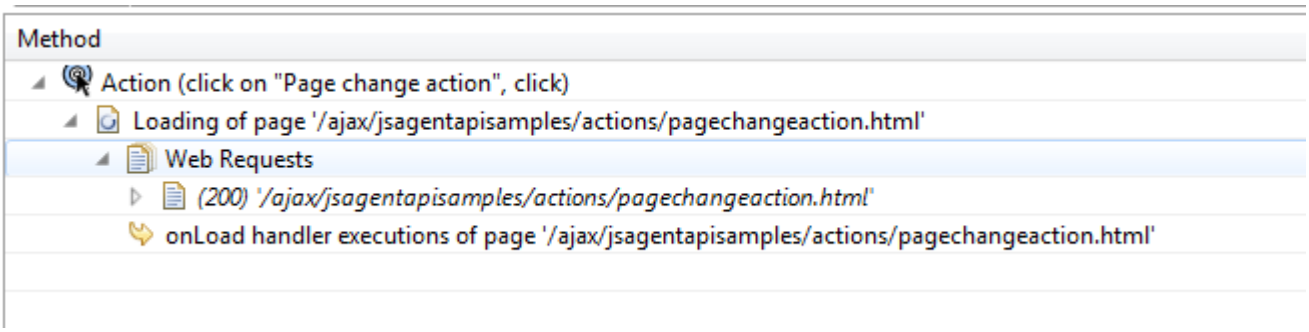
The User Action is shown in the following figure.



Example: Start an Action That Triggers a Page Change (/actions/pagechangeaction.html)

This example starts an action that triggers a page change, with a `setTimeout` call 5 seconds after `onload`. This construction is useful when no user input can be associated with an action, so that the automatic user action detection does not work. This situation can happen when an action is triggered, for example, by `setTimeout` or `setInterval`.

The User Action is shown in the following figure.



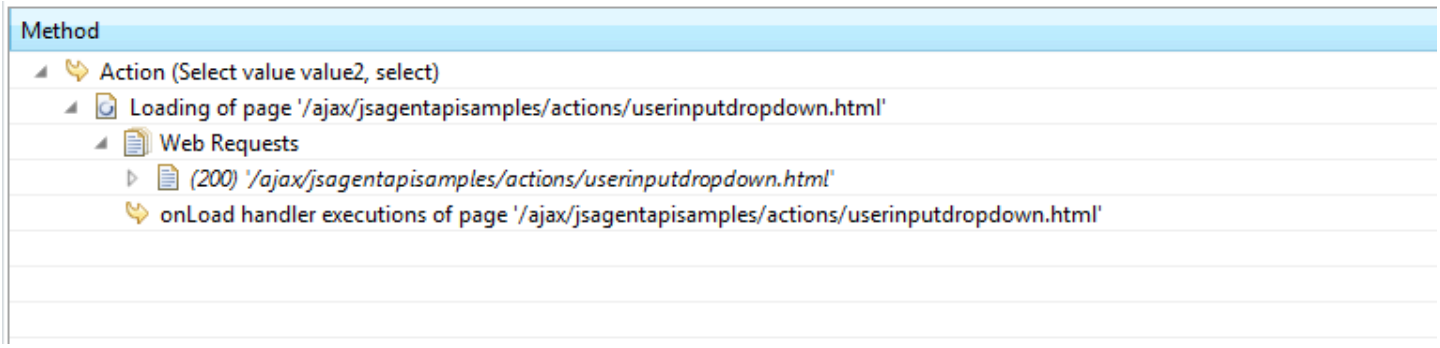
Page change action

Example: Override the Default Action Detection (/action/userinputdropdown.html)

Sometimes it is necessary to override the default action detection. In this example, the `onchange` event of a dropdown list is modified so that the newly selected value is added to the action name.

1. The default action detection is disabled via `dynaTrace.setAutomaticActionDetection(false);`.
2. A custom action is started. The action name contains the selected dropdown value: `dynaTrace.enterAction('Select value '+value, 'select',null,null);`.

The User Action is shown in the following figure.



User Action with override

Extending support for third-party content

The JavaScript Agent supports detection of third-party resources. The automatic detection works for images and scripts.

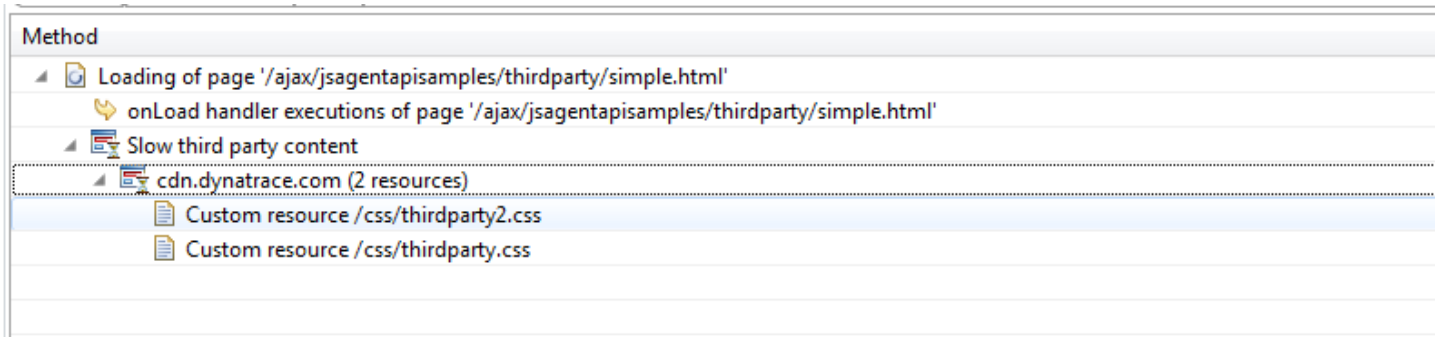
Use the following functions to enable automatic detection of additional resources:

- `dynaTrace.startThirdParty(type,url)` indicates the start of a third-party resource. The type can be `i` (image), `s` (script) or `c` (custom resource).
- `dynaTrace.stopThirdParty(url,success,start,stop)` indicates when the third-party resource is loaded.

Example: Add Custom CSS Files (/thirdparty/simple.html)

This example demonstrates how to add custom resources. It adds a CSS file where start and stop times are set automatically, and another CSS file where start and stop time are set explicitly.

The User Action is shown in the following figure.

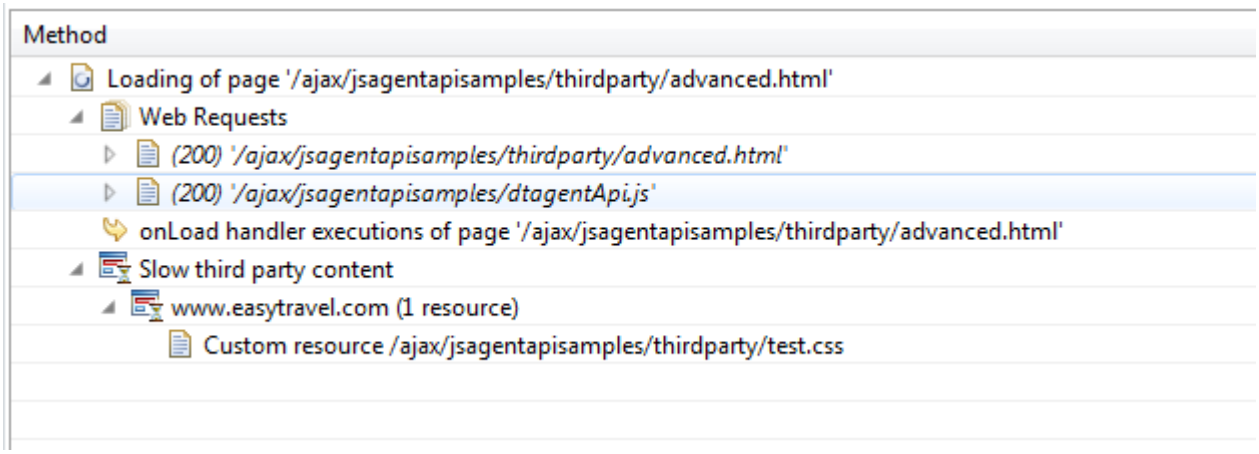


Custom resources

Example: Detect When a CSS File Is Loaded (/thirdparty/advanced.html)

This example demonstrates how to detect the successful loading of a CSS file. It dynamically adds a `link` tag for the CSS, and then checks whether the number of styles is greater than zero, in 20-millisecond intervals.

The PurePath is shown in the following figure.



CSS file detected

Reporting playback of media streams

The JavaScript ADK provides a function to report information to the AppMon Server about the playback of media streams.

```
addStreamingNode(source, duration, userTriggered, watchedCompletely, maxPlayTime, playTime, bufferingCount, bufferingTime, type)
```

This function provides data for media streams that cannot be collected by the Streaming JavaScript plugin because of that plugin's reliance on HTML5 media tags. If your streams are embedded using, for example, Adobe Flash technology, the Streaming plugin cannot detect them. If you can collect the same data by using JavaScript or a plugin API that the media player offers, then you can leverage this function to send similar data to the AppMon Server.

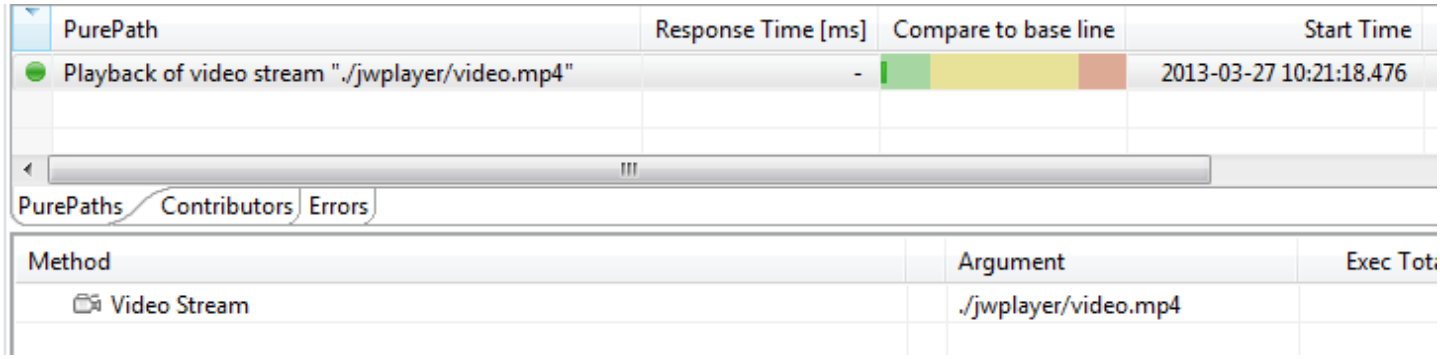
Parameters

`source`	URI of the media stream.
`duration`	Duration in seconds of the media stream.
`logger_name`	Name of the logger.
`userTriggered`	`true` if the user had to press a play button to start playback of the media stream; `false` if the playback of the video started automatically (e.g. as soon as the page load was finished),
`watchedCompletely`	`true` if the user watched the video completely (until playback ended automatically); `false` if the user stopped the media stream or left the page before playback was completed.
`maxPlayTime`	The farthest second of the media stream that was played back.
`playTime`	Total time (in seconds) the user spent consuming the media stream.
`bufferingCount`	Number of times the playback was interrupted by rebuffer events.
`bufferingTime`	Total duration (in seconds) of the rebuffer events.
`type`	`_video_` for video streams; `_audio_` for audio streams

Example: Report Playback Information (/streaming/streaming.html)

This example uses the JavaScript API of the JWPlayer (http://www.longtailvideo.com/jw-player), a Flash based media player, to collect information about the playback of a media streams and report this information to the AppMon Server using the JavaScript ADK.

The resulting PurePaths are displayed in the *User Action PurePaths* dashlet.



Video stream PurePath

Capturing Custom XHR calls

Example: Capture Custom XHR Calls (/xhrapi/simple.html)

This example demonstrates how to instrument a simple `XMLHttpRequest` in a page. Clicking the **Load Data** button performs an Ajax call to the URL `ajax.txt` is performed. The result is displayed in the `div` with ID `resultDiv`.

The `loadData` function starts an XHR call via `ActiveXObject` in Internet Explorer and via `XMLHttpRequest` in all other browsers.

The following code tells the JavaScript Agent that an XHR action starts:

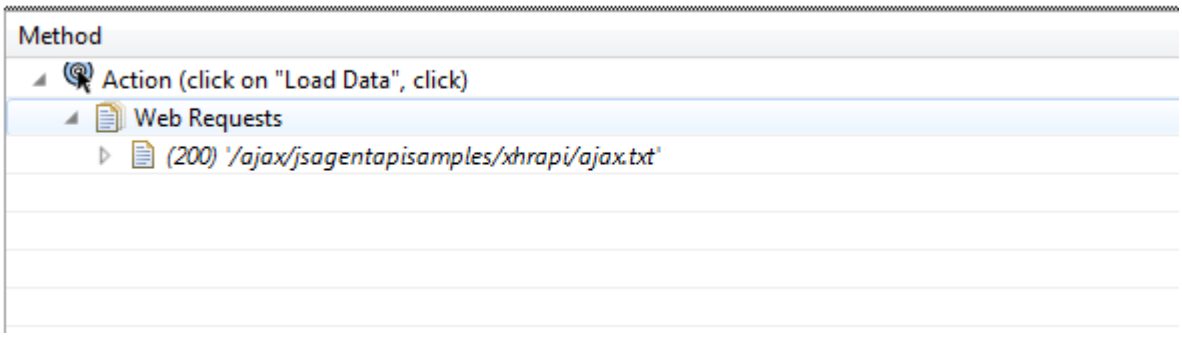
```
var action=dynaTrace.enterXhrAction();
```

This code tells the JavaScript Agent that the XHR action ends in the `onreadystatechange` callback:

```
dynaTrace.leaveXhrAction(action);
```

The JavaScript Agent automatically detects the user input that triggers the action.

The generated User Action is shown in the following figure.



User Action generated by XHR call

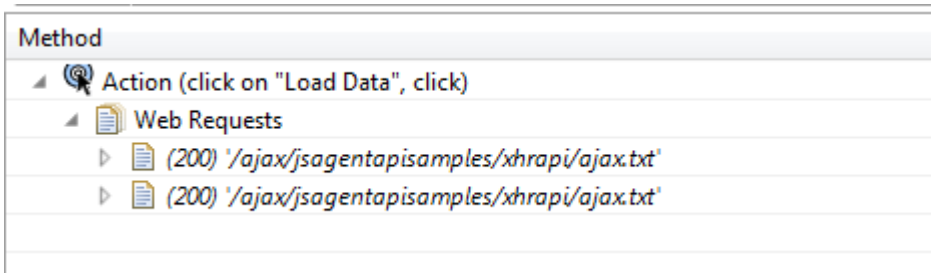
Example: Capture Cascading Custom XHR Calls (/xhrapi/cascading.html)

When an XHR call triggers a new XHR in the `onreadystatechange` callback, this XHR action should be added as a subaction to the first action as shown in the following code.

```
dynaTrace.enterXhrCallback(action); //callback code dynaTrace.leaveXhrCallback(action);
```

This code tells the JavaScript Agent that no extra XHR action is created when an XHR request is started within this callback.

Clicking the **Load Data** button in the example results in the User Action shown in the following figure. There is only one Ajax action, but two requests for `ajax.txt`.



Cascading custom XHR calls

Example: Capture Dynamic Call to a JavaScript File (/xhrapi/scripttag.html)

Sometimes Ajax calls are performed not via `XMLHttpRequest`, but by dynamically appending `script` tags to the page. When `script` tags are used, the resulting script file usually tries to call a callback function to handle the result when the call is finished.

In the example, the `loadData` function adds a `script` tag to the `head` tag that loads the script file `ajax.js`. This is usually dynamically generated on the server side (via a mechanism such as PHP, JSP, or servlet). The `ajax.js` file calls the globally defined function `showResult` when it is loaded.

Instrumenting this call requires adding two calls to the JavaScript Agent:

In `loadData`:

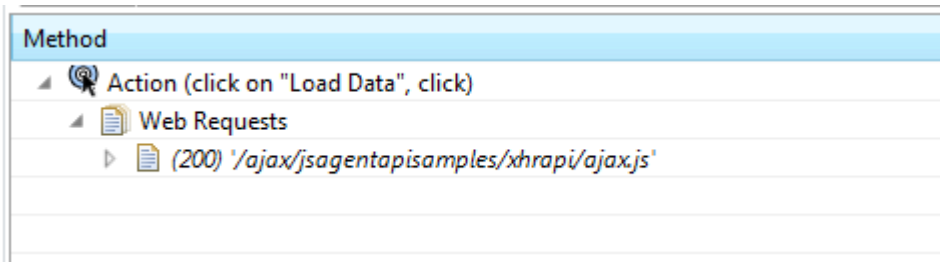
```
action=dynaTrace.enterXhrAction();
```

In the `showResult` callback function:

```
dynaTrace.leaveXhrAction(action);
```

If the callback function can also start new Ajax calls, you should add the `enterXhrCallback` and `leaveXhrCallback` calls to the callback function.

Clicking the **Load Data** button results in the User Action shown in the figure.



Dynamic call to a JavaScript file

Capturing frame loads during an XHR callback

Example: Capture frame loading during an xhr callback (/xhrapi/frameloadingduringxhr.html)

This example demonstrates how to extend an XHR action to include the loading of a frame action that occurs during an XHR callback function.

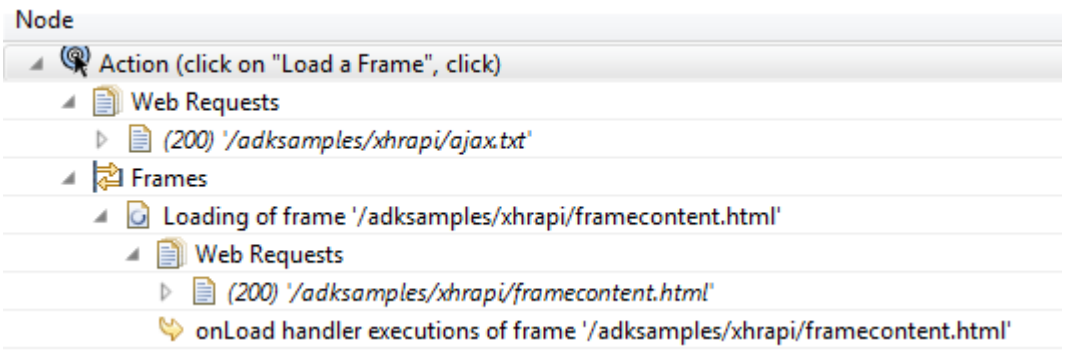
Without the ADK, the XHR action finishes and causes the monitor signal to be sent before the frame gets a chance to start loading. As a result, two separate actions are created, when just one would be expected. The ADK provides the ability to keep these actions unified.

Make a call to `enterXhrAction` before making the AJAX request, add listeners to the `onload` event of the frame's element, and then call `leaveXhrAction` here, passing the return value of the corresponding `enterXhrAction` call.

```
var xhrAction = dynaTrace.enterXhrAction("frame load xhr");
var xhr = window.XMLHttpRequest? new XMLHttpRequest(): new ActiveXObject("microsoft.xmlht
xhr.open('GET', "ajax.txt", false);
xhr.onreadystatechange = function(){
        if(xhr.readyState == 4){
                var e = document.createElement("IFRAME");
                e.src = "framecontent.html";
                document.body.appendChild(e);
                if (e.addEventListener) {
                        e.addEventListener("load", frameLoadedHandler, false);
                } else if (e.attachEvent) {
                        e.attachEvent("onload", frameLoadedHandler);
                }
        }
}
xhr.send(null);

function frameLoadedHandler() {
        dynaTrace.leaveXhrAction(xhrAction);
}
```

After clicking a button with the text "Load a Frame" that then performs the code above, the structure should appear like this in the Client:



Adding custom user input

Example: Add Custom User Input for Automatic Action Detection (/xhrapi/userinput.html)

This example demonstrates how to add custom user input that is used by automatic action detection. When an Ajax call is started or a page change is triggered, the automatic action detection checks whether user input is currently active. If it finds active user input, it creates a User Action.

The example page contains an HTML input text box that triggers an Ajax call via jQuery. JQuery is instrumented by the JavaScript Agent by default, so we do not need to consider XHR wrapping for this example.

The Ajax call is only performed when 800 milliseconds elapses with no key pressed by the user. This delay is common for autosuggest implementations, to avoid sending XHRs when the user is still typing.

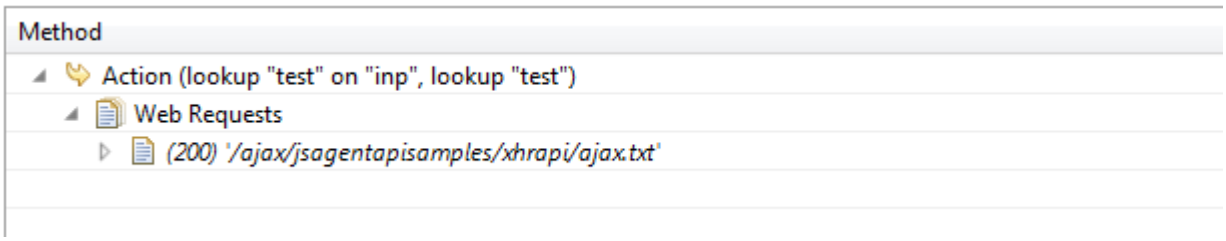
When the Ajax call is performed, no user input is active, so the call `beginUserInput` tells the JavaScript Agent that there is user input:

```
var inp=document.getElementById('inp'); var value=inp.value; var uinp=dynaTrace.beginUser
```

The call `endUserInput` ends the user input :

```
dynaTrace.endUserInput(uinp);
```

Entering the text **test** results in the User Action shown in the figure.



User input for automatic action detection

Handling framework onload behavior

Many Ajax frameworks start actions, including XHR calls and asynchronous operations, in `onload` handlers. Some of these actions finish after the `onload` event of the browser is triggered. Because these actions all belong to the `onload` event of the browser, it is possible to extend the `onload` action by using the API functions.

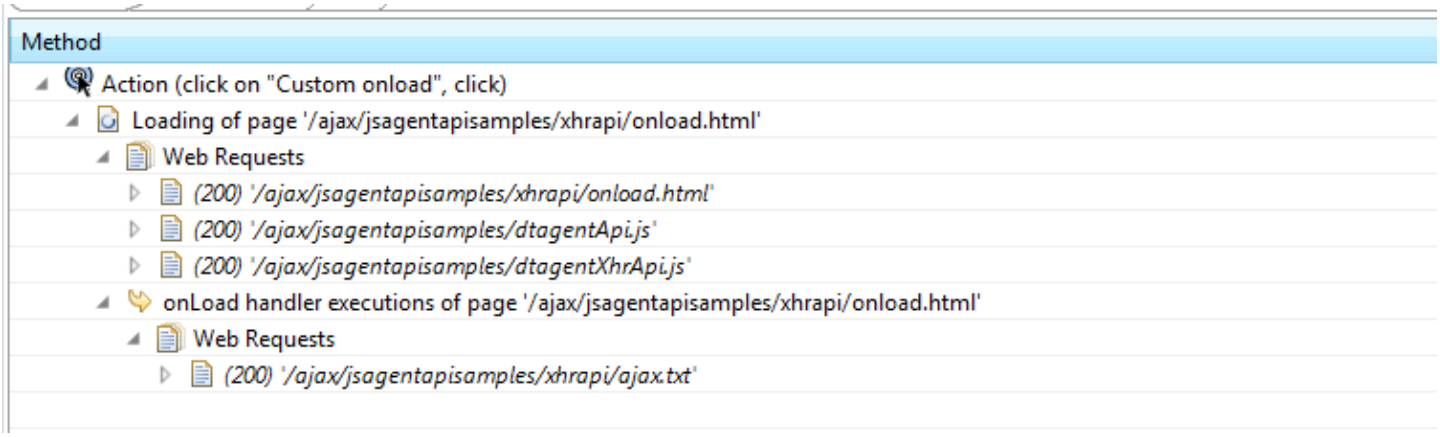
- `signalOnLoadStart` indicates the start of `onload`. It can be called to indicate when `onload` of a framework starts.
- `incrementOnLoadEndMarkers` is necessary to tell the JavaScript Agent that the end of the `onload` action will be indicated via `signalOnLoadEnd`. For every call to `incrementOnLoadEndMarkers`, `signalOnLoadEnd` has to be callled.
- `signalOnLoadEnd` indicates that an `onload` end was reached. A framework can have more than one `signalOnLoadEnd`. The last one ends the `onload` action.

No new actions are created during `onload`. All web requests will be correlated to the `onload` action.

Example: Extend onload Handling

The example shows how to extend the `onload` action. It starts an XHR call in the browser's `onload` callback and ends the `onload` action in the callback of the XHR.

The resulting PurePath is shown in the figure.



Extended onload handling

Handling redirects after the onload event

Sometimes a page may trigger a redirect after it has finished loading. If the JSAgent has time to finish processing and sends the monitor signal before the page actually begins redirecting, this will appear as two separate actions when you would expect the monitor signal to be a subaction of the first. This almost always happens when using the meta tag `<meta HTTP-EQUIV="REFRESH" content="0; url=redirectedpage.html"></meta>` for URL redirection, especially if the timer from `"content=num"` is set to a large value.

To force the signal to wait for another onload, you can call the ADK's `incrementOnLoadEndMarkers()` function immediately after loading the ADK script. It will properly set the monitor signal as a source action on the next page.

Setting custom metadata

Example: Custom metadata (/metadata/metadata.html)

Metadata can be set also via the ADK's setMetaData(name,value) function. Any data set using this function will be added to any subsequent actions on the same page, along with any normally detected metadata configured in the system profile's **UEM** settings.

```
$("#clickme").on("click", function() {
        var val = $("#md").val();
        dynaTrace.setMetaData("customValue at " + Date.now(),val);
});
```

Using the JavaScript ADK for Flash and Silverlight applications

Technologies like Adobe Flash and Microsoft Silverlight are often used for implementing complex user interfaces, animation, or video playback. Although AppMon UEM cannot instrument these technologies “out of the box”, it provides mechanisms for these technologies to interact with the web page in which they are embedded. These mechanisms make it possible to call the JavaScript functions of the AppMon JavaScript ADK and make the technologies accessible for AppMon UEM.

To instrument a Flash or Silverlight component, include the sample project’s utility classes (see below) in your Flash or Silverlight project.

Similar to the Introduction for JavaScript code, the ADK classes for Flash and Silverlight are also based on actions. The utility classes provide similar methods.

> **Important**
>
> To correlate the defined actions to any web requests belonging to that action, it is necessary to use the utility class to prepare these web requests by adding the header `x-dtReferer`. This head is required by the AppMon correlation engine.

Using the ActionScript class dynaTraceFlashAdk.as

This ActionScript class contains all functions of the JavaScript ADK that delegate to the JavaScript ADK, plus a few utility functions that simplify the tagging of web requests for the correlation engine.

The following code example shows how to use the utility class:

```
private var action:String;

//create instrumentation object for ActionScript class
private var dynaTraceAdk:DynaTraceFlashAdk = new DynaTraceFlashAdk();
[...] protected function buttonClickHandler(event:MouseEvent):void

{
  var url:String = "ajax.jsp";
  // create a new action string for the performed click event
  action = dynaTraceAdk.enterAction("Click on Button", "click");
  urlLoader = new URLLoader();
  var request:URLRequest = new URLRequest();
  request.url = url;

  //prepares the URLRequest by adding a custom header "x-dtReferer" in order to correlate
  request = dynatraceAdk.prepareURLRequest(request);
  urlLoader.addEventListener(Event.COMPLETE, completeLoadContent);
  try
  {
    urlLoader.load(request);
  }
    catch (error:Error)
  {
    //if an error occurred, report to dynaTrace server
    dynaTraceAdk.reportError("Error during load of image");
  }
}
protected function completeLoadContent(event:Event):void
{
  [...]
  // mark action as completed dynatraceAdk.leaveAction(action); [...]
}
[...]
```

Using class DynaTrace.cs for Silverlight applications

This C# class is similar to the ActionScript utility. It also contains delegates for the JavaScript Agent ADK functions and some utilities for tagging Web requests.

The following code example shows how to use the class:

```
private string action;
// create the instrumentation object for SilverLight private JavaScriptAdk
javaScriptAdk = new JavaScriptAdk();
[...]
private void button1_Click(object sender, RoutedEventArgs e)
{
  WebClient client = new WebClient();
  // starts a new action
  javaScriptAdk.enterAction("Click on button","click");
  // prepares the web request by adding the custom header "x-dtReferer" in order to corre
  javaScriptAdk.PrepareWebClient(client);
  client.DownloadStringCompleted += new DownloadStringCompletedEventHandler(complete);
  client.DownloadStringAsync(new Uri("ajax.jsp"));
}
private void complete(object sender, DownloadStringCompletedEventArgs args)
{
  if (args.Error != null)
  {
    //if an error occurred, report to dynaTrace server
    javaScriptAdk.reportError(args.Error);
  }
  // mark action as completed javaScriptAdk.leaveAction(action);
}
[...]
```

In Silverlight versions earlier than version 4, cookies were not automatically set on `WebClient` and `HttpWebRequest` objects. For these earlier versions, you have to copy cookies onto the request objects. This is also done in the `PrepareWebClient` and `PrepareWebRequest` methods.

Instrumentation result

After instrumenting your Flash or Silverlight application, you should be able to retrieve PageAction PurePaths, as shown in the following figure.

Method	Exec Total [ms]	Elapsed Time [ms]	Timeline
Action (Click on Button, click)	132.00	0.00	
Web Requests	-	-	
(200) '/ajax/testframework/tests/ajax.jsp'	0.38	105.50	

PageAction PurePath

Collecting streaming media measures from Flash players

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Requirements

Implementation

Related pages

JavaScript ADK and JavaScript Ajax ADK

JWPlayer API documentation

HTML5 media tags have a standardized API that enables the AppMon JavaScript Agent to collect metrics for streaming media out of the box. However, no such standardized API is available for the many different Flash media players that populate the web, which makes it unfeasible to support collection of streaming media metrics for Flash media players out of the box.

As long as the Flash media player offers a public API, either for ActionScript or for JavaScript, it is usually straightforward to implement the missing link between the Flash media player and the AppMon Server, because the different APIs offer the same functionality and only syntax details vary among the different implementations. This topic describes how the implementation would work in principle. It does not provide implementation details for a specific Flash player.

Requirements

AppMon

The requirements for the AppMon installation are as follows:

- The AppMon release must be 5.5 or later.
- A UEM license must be available.
- The AppMon JavaScript ADK must be embedded in the HTML pages that contain the Flash player. (As of 6.5, it is bundled with the JavaScript Agent, so this is no longer necessary).

Flash media player

The requirements for the Flash media player are as follows:

- The player must have a public API that can be used to access the state of the media player. Manipulation of the state is not necessary.
- An Event-API is available to register event listeners (e.g. start and stop) and media stream events (e.g. ready for playback and buffering start) for the player. Ways to implement the desired functionality without listening to events (e.g. polling the state of the media player) are not covered in this guide, because events are widely supported.

Some media players that fulfill these requirements are:

- JWPlayer http://www.longtailvideo.com/jw-player
- Limelight Player http://www.limelightvideoplatform.com
- FlowPlayer http://flash.flowplayer.org
- YouTube Player https://developers.google.com/youtube/

Required events

It is assumed that the Flash media player allows registering listeners for the following events.

Event	Description
beforeStart	This event happens before the start of the playback, when the media stream is loaded. It enables accessing the metadata (URL, type, duration, autostart) of the media stream.
stateChange	This event is called whenever the state of the player changes. Available values for this event are the new state and the old state. Possible states are: • IDLE – The initial state of the player. • BUFFERING – The player has to stop playback and wait until additional data is loaded. • PLAYING – The player is playing the media stream.
playing	This event happens at least once every second while the playback of the media stream is in progress. It updates timing information (`maxPlayTime` , `playtime`). If a `playing` event is not present, it can be simulated using the `setInterval` JavaScript function.

complete	This event happens when playback of the media stream is complete.
error	This event happens when a error occurs during playback.

The exact naming and syntax for registering event listeners differs among media players. Most players will support the events above listed in some way. A larger API is generally offered to deal with advanced use cases (e.g. playlist handling or starting playback of a different media stream in the same player). Refer to the medial player or API documentation for more information.

Required values

The values that must be collected from the Flash player to submit streaming media information to AppMon are described in JavaScript ADK and JavaScript Ajax ADK .

Implementation

Implementation involves three steps:

1. The local variables that track the state are initialized and the event listeners are registered.
2. The player plays the streaming media and, when an event occurs, calls the callback functions that were registered in step 1. Depending on the type of events that occur and the player state, information is sent to the AppMon Server.
3. Collected values are reported to AppMon whenever the page is unloaded. It is assumed that only one video is displayed per page.

Each step is described in detail below.The example code is from the plugin developed for the JWPlayer. Comments and some special case handling are removed from the example code for clarity. It is recommended that you review the fully detailed, commented code of the media player implementation before starting with your own implementation.

Step 1. Initialize Variables and Register Event Listeners

When the media player (or the page containing the media player) is loaded, variables for each of the values that need to be collected are initialized.

JWPlayer code example - initialize of variables

```
var mediaType = null;
var source = null;
var duration = -1;
var playTime = 0;
var playTimeForSending = 0;
var maxPlayTime = 0;
var autoplay = config.autoplay; // this information can be retrieved from the player conf
var watchedTillEnd = false;
var bufferingCount = 0;
var bufferingTime = 0;
```

Event listeners are registered for the player. Because the JWPlayer has a separate function for each type of statechange, the same function is registered for all statechanges.

JWPlayer code example - register events

```
// assign event listeners to player
player.onReady(setup);

function setup(evt) {
        player.onComplete(onCompleteEvent);
        player.onTime(onTimeEvent);
        player.onBeforePlay(updateVideoInformation);
        player.onError(onErrorEvent);
        player.onMeta(updateVideoInformation);

        // state change events
        player.onBuffer(onStateChange);
        player.onIdle(onStateChange);
        player.onPlay(onStateChange);
        player.onPause(onStateChange);
};
```

Step 2. Call Event Listeners

During the playback of the media the player will then call the event listeners whenever an event occurs. The following table describes what actions need to be taken whenever an event occurs. Some actions are only to be performed if a certain condition is fulfilled. In that case, the condition is described in the **Condition** column.

beforeStart event

Event	Condition	Actions
beforeStart		Source, type, duration, and `userTriggered` are retrieved from the Flash player and stored in the respective variables. The other variables are initialized to false or 0, respectively.

JWPlayer code example - beforeStart event listener

```
function updateVideoInformation() {
        var videoInformation = player.getPlaylist()[0];
        if (!mediaType) {
                if (videoInformation.type == "video") {
                        mediaType = "_video_";
                } else if (videoInformation.type == "audio") {
                        mediaType = "_audio_";
                }
        }

        if (!source && videoInformation.file) {
                source = videoInformation.file;
        }

        if (duration < 0 && videoInformation.duration > 0) {
                duration = Math.round(videoInformation.duration);
        }
}
```

stateChange event

Event	Condition	Actions
stateChange	If new state is BUFFERING	Increase `bufferingCount` by 1. Store current timestamp as `bufferingStartTime`.

	If new state is PLAYING and previous state was BUFFERING	Calculate the duration of the buffer event, based on the stored `bufferingStartTime` and the current timestamp, and add it to `bufferingTime`.

JWPlayer code example - stateChange event listener

```
function onStateChange(stateChange) {
        if (stateChange.newstate == "BUFFERING") {
                bufferingCount++;
                bufferingStart = getCurrentTime();
        } else if (stateChange.oldstate == "BUFFERING") {
                bufferingTime += getCurrentTime() - bufferingStart;
                bufferingStart = -1;
        }
}
```

playing event

Event	Condition	Actions
playing		Add the time elapsed between the current playing event and the previous playing event to `playTime`.
	If current media stream position is greater than `maxPlayTime`	Set `maxPlayTime` to the current media stream position.

JWPlayer code example - playing event listener

```
function onTimeEvent(mediaTime) {
        if (Math.round(mediaTime.position) > maxPlayTime) {
                maxPlayTime = Math.round(mediaTime.position);
        }
        var currentTime = getCurrentTime();
        if (lastTimeEvent > 0) {
                playTime += currentTime - lastTimeEvent;
        }
        lastTimeEvent = currentTime;
}
```

complete event

Event	Condition	Actions
complete		Set `watchedCompletely` to `true`, and set `maxPlayTime` to the duration of the video to prevent inconsistencies.

JWPlayer code example - complete event listener

```
function onCompleteEvent() {
        maxPlayTime = duration;
        watchedTillEnd = true;
}
```

error event

Event	Condition	Actions
error		Report the error, using the AppMon JavaScript ADK.

JWPlayer code example - error event listener

```
function onErrorEvent(message) {
        dynaTrace.reportError(message);
}
```

Step 3. Report Collected Values to the AppMon Server

Assuming the Flash media player only replays a single video, collected values should be reported to the AppMon Server before the `onbeforeUnload` event. The easiest way to achieve this is to use the function `addPageLeavingListener()` provided by the AppMon ADK, as shown in the following example.

JWPlayer code example - sending streaming media information

```
dynaTrace.addPageLeavingListener(reportMediaStateToDynaTraceAgent);

function reportMediaStateToDynaTraceAgent() {
        dynaTrace.addStreamingNode(source, duration, autoplay, watchedTillEnd,
                maxPlayTime, playTime, bufferingCount, bufferingTime, mediaType);
}
```

Integration with web load testing and monitoring tools

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Request header

Response header

Viewing Tagged Web Requests in AppMon

Synchronizing session recording with load test runs

WebAPI tests

Usage

Related pages

Tagged Web Requests dashlet

Watch and learn

Load Testing Made Easy with Dynatrace Webinar

Overview

AppMon integrates well with load testing and monitoring products, especially on the transactional / request level.

AppMon automatically starts a PurePath on a servlet, ASP.NET or web service when the AppMon HTTP trace tag is present.

The Dynatrace Community Webinars show how to use AppMon in load testing and also show how it uses tagged web requests to integrate with any HTTP-based load testing solution.

Request header

To use other load testing and monitoring tools for diagnosing web services in AppMon, add the following information to the HTTP headers generated by the load testing tool:

```
x-dynaTrace: VU=1;PC=.1;ID=4;NA=SearchPage
```

Below is a sample HTTP header. The final line of the example shows the added code.

```
POST /onca/soap?Service=AWSECommerceService HTTP/1.1
Content-Type: text/xml; charset=utf-8
Host: soap.amazon.com
Content-Length: 2566
Connection: Keep-Alive
Accept: */*
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT)
x-dynaTrace: VU=1;PC=.1;ID=4;NA=ListResults
```

The following table describes the HTTP header string for AppMon diagnostics.

Key	Description	APMaaS Name	Usage
ID	The unique request **ID** (serial number). This string should be unique for one web request or a set of web requests that together make up a step/transaction execution. `ID=<Id>`	Transaction GUID	optional
PC	The *Page Context* contains information about what document in the currently processed page is loaded. The following syntax is recommended, though not required: • If it is a named frame, then the value starts with the frame name. • The document number, unique for the page, is appended after a period. If embedded documents are cached, this number need not be progressive. `PC=<FrameName>.<DocId>`		optional

VU	The unique number of the *Virtual User* that sends the request. `VU=<Id>`	Virtual User ID	optional
NA	The timer **NA** me of the request. This can be the timer or transaction name used in the load test script to identify the response time measure, or the document/page title, or any other human-readable URL encoded identifier for that document. `NA=<timername>`	Transaction Name (SYM), Step Name (SaaS)	**required**
SI	The *Source ID* can be used to identify the product that triggered the request: For example **WLT** (Web Load Testing), SYM (Dynatrace Synthetic Monitoring), BB (Backbone), or LM (Last Mile).	Source ID	optional
GR	*Geographic Region*, useful only for the Synthetic Monitoring solution. This contains arbitrary text.	Location	optional
AN	*Agent Name*: the logical name of the Agent from which the request originated. This information is used by Synthetic Monitoring.	Agent Name	optional
SN	*Script Name*. This groups a set of requests that make up a multi-step transaction, for example making an online purchase.	Script Name	optional
TE	*Test Name* is the name of the entire load test. It uniquely identifies a test run.	Test Name	optional

Response header

```
HTTP/1.1 200 OK
x-dynaTrace: RS=session20060620091846;PT=31;PA=1;PS=76562898
Content-Type: text/html
Transfer-Encoding: chunked
Date: Tue, 20 Jun 2006 07:19:16 GMT
Server: Apache-Coyote/1.1
```

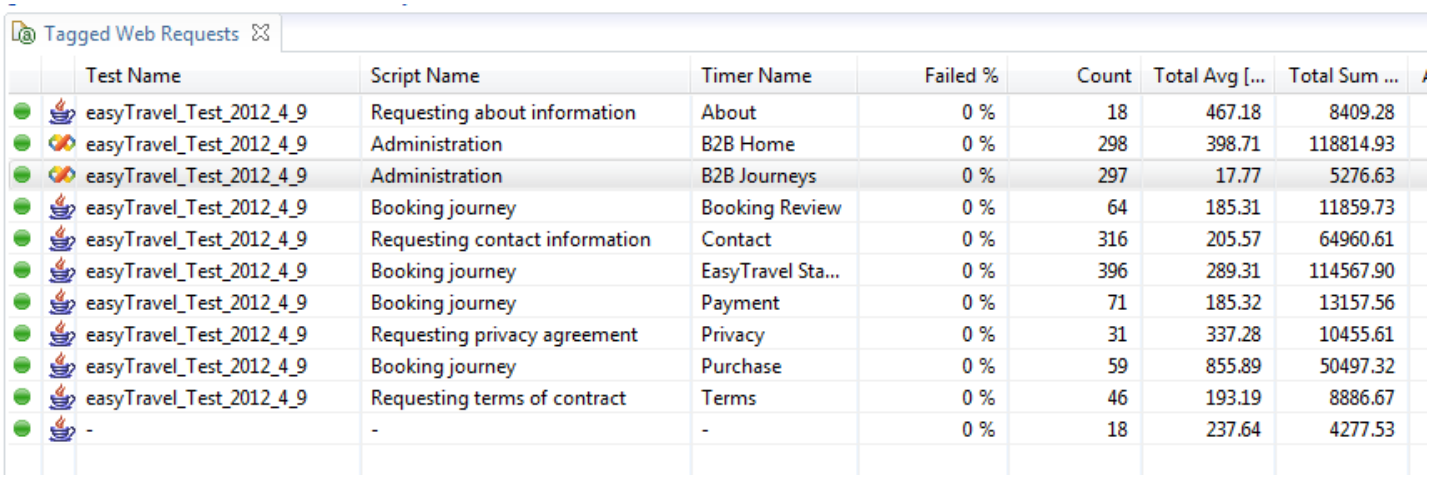
All the attributes together uniquely define a single object request, i.e. a single PurePath.

Key	Description	Usage
RS	Name of the AppMon *Recorded Session* that is currently being recorded.	only set when session recording is active
PT	*PurePath* identifier – The *Trace* number.	required
PA	*PurePath* identifier – The entry point *Agent* number.	required

PS	*PurePath Server* identifier – The AppMon Server that captured this PurePath.	required
SP	*System Profile* – Set to the active System Profile if manual session recording is not active; i.e., if either continuous session recording or no session recording is active.	available

Viewing Tagged Web Requests in AppMon

AppMon groups the resulting web page requests specifically for load tests. See Tagged Web Requests Dashlet for details.



Test Name	Script Name	Timer Name	Failed %	Count	Total Avg [...	Total Sum ...
easyTravel_Test_2012_4_9	Requesting about information	About	0 %	18	467.18	8409.28
easyTravel_Test_2012_4_9	Administration	B2B Home	0 %	298	398.71	118814.93
easyTravel_Test_2012_4_9	Administration	B2B Journeys	0 %	297	17.77	5276.63
easyTravel_Test_2012_4_9	Booking journey	Booking Review	0 %	64	185.31	11859.73
easyTravel_Test_2012_4_9	Requesting contact information	Contact	0 %	316	205.57	64960.61
easyTravel_Test_2012_4_9	Booking journey	EasyTravel Sta...	0 %	396	289.31	114567.90
easyTravel_Test_2012_4_9	Booking journey	Payment	0 %	71	185.32	13157.56
easyTravel_Test_2012_4_9	Requesting privacy agreement	Privacy	0 %	31	337.28	10455.61
easyTravel_Test_2012_4_9	Booking journey	Purchase	0 %	59	855.89	50497.32
easyTravel_Test_2012_4_9	Requesting terms of contract	Terms	0 %	46	193.19	8886.67
-	-	-	0 %	18	237.64	4277.53

Tagged Web Requests

Synchronizing session recording with load test runs

AppMon Server offers both a web service and RESTful service API that can be used to record an AppMon session for the time of the load test run. In earlier versions of AppMon the command line interface could be used via the `dtcmd` utility. You can still use this command line utility, but we recommend using the web-based APIs.

To get a list of all web-based APIs, browse to the AppMon Server web frontend, which by default is http://dtservername:8020.

From there, you have the option to obtain the SOAP/HTTP web service WSDL (http://dtservername:8020/wsdl/ManagementServerService.wsdl) or interact with the RESTful interface via an HTML interface (http://dtservername:8020/rest/html/management/server) or via an XML-based interface (http://dtservername:8020/rest/management/server).

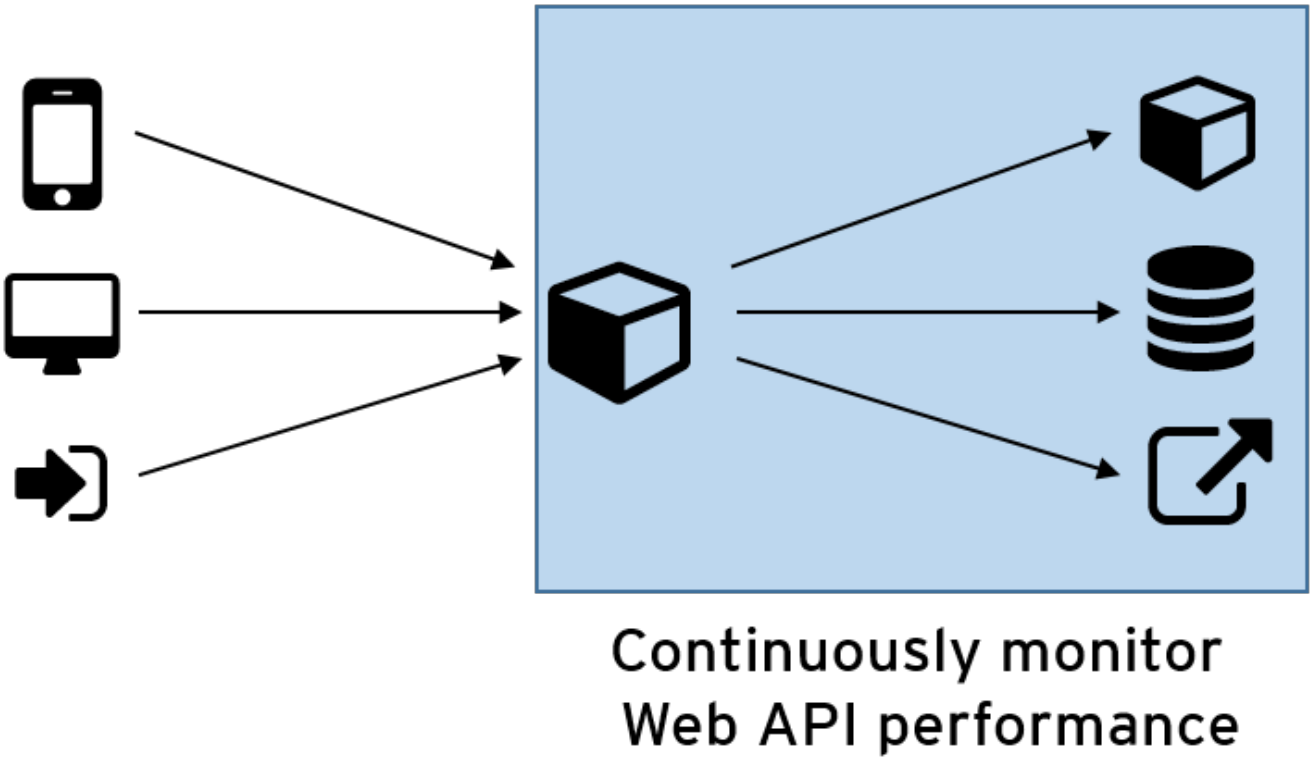
The HTML interface enables you to explore all options the RESTful interface offers. Clicking the XML link on each HTML page shows you the links to the RESTful service that should be used when accessing the service from an external tool. The response of the XML-based interface is also XML, which is easier to parse and analyze.

The RESTful interface makes it easy to interact with the AppMon Server. For synchronizing session recording with load test runs, you can use the RESTful calls to start and stop session recording for a specific System Profile. If your load testing tool offers a way to execute web requests at the beginning and at the end of the load test run, you can add the necessary web request calls to the AppMon Server.

See Start Session Recording and Stop Session Recording for details on how to start and stop session recording using the REST Interfaces.

WebAPI tests

Web APIs are interfaces provided by a web server that can be used by a variety of clients like smartphone apps, rich-client applications, or other web applications. It is crucial to continuously ensure API performance, and with the Web API test category it is now possible to continuously monitor performance metrics for HTTP-based web service APIs. See Capture Performance Data from Tests for a list of metrics available for the Web API category.



Usage

Perform the following steps to execute a Web API test.

1. Register a new test run through the REST interface using the **webapi** category.
2. Send one or more HTTP requests to the URI page you want to test. These requests have to include the X-dynaTrace header where the test run ID returned by the AppMon Server REST Interface is specified to let AppMon know that these requests belong to a test. See below for a description of fields in the X-dynaTrace header.
3. Analyze the test results in the Test Automation dashlet, or retrieve them using REST API.

In case you don't want to provide specific test metadata, you only need to provide the test name using the **TN** field in X-dynaTrace header (see below). That way test run will be registered automatically.

X-Dynatrace header possible key-value pairs

The table lists possible key-value pairs. They can be provided in any order, separated with a semicolon.

Note

The **TN** field should be used to provide test name. The **TR** and **RC** fields can still be used if you want to provide test run id or the expected response code.

Key	Description	Usage
TN	Test Name (TN) should be provided using this field. Metrics for requests with the same name are aggregated. You can execute as many tests as desired within the same test run, but give them different names if they are not testing the same thing.	Required. You can still use **NA** and **TR** fields instead.
NA	Old way of passing Test **Na** me.	**Deprecated** Requires **TR** field to be provided
TR	TestRunId (TR), from a registered test run. See REST API POST testruns for more information.	Optional when **TN** field us used. If not provided, a test registers automatically. Required with **NA** field.
RC	Expected Response Code (RC). If the test results in any other response code, the test is marked as failed.	Optional. The default is 200.

X-Dynatrace header sample

Using test run id registered using Test Automation REST API, expecting 200 response code:

```
X-dynaTrace: TN=SampleTestName;TR=7a0520fc-eda9-4f78-a8fa-4cb7d04ac8b8;RC=200
```

Relying on automatic test run metadata registration (no test metadata provided), expecting default response code (200):

```
X-dynaTrace: TN=SampleTestName
```

Measures

Category	Metric Name	Notes
Webrequests	Bytes Sent	

Webrequests	Bytes Received
PurePaths	PurePath Duration
PurePaths	PurePath Response Time
Methods	Invocation
Logging	Count
Exceptions	Count
Database	Execution Count

Walkthroughs

See Integrate Web API Performance Monitoring in JMeter or Integrate Web API Performance Monitoring in SoapUI for more information how to use the Web API test category.

Plugin architecture

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Plugin types

Plugin development environment

Related pages

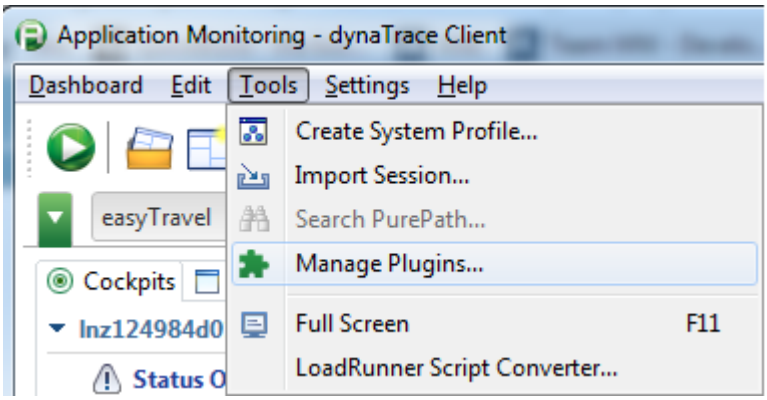
Building FastPacks

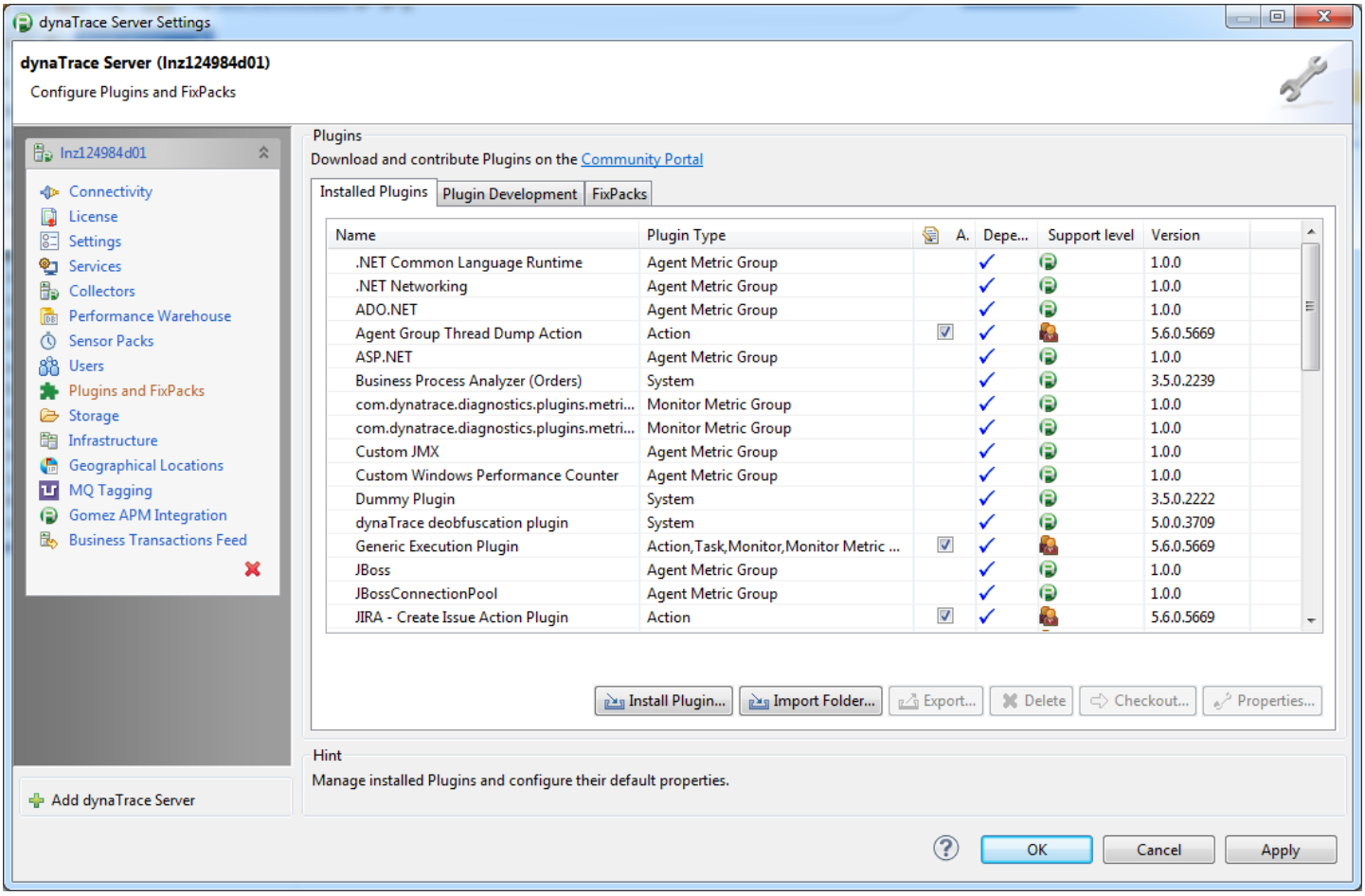
Manage and develop user plugins

Manual uninstallation of a system plugin

Plugins extend AppMon with features and functions beyond the functionality in the installed product.

All Plugins are displayed and managed in the **Plugins** pane of the **Server Settings** dialog box. To display the dialog box, select **Tools > Manage Plugins** in the AppMon Client.





Plugin types

Plugin Type	Sub Type	Description & Properties
System Plugin	System	*System Plugins* are provided by AppMon to extend the AppMon system functionality (e.g. Transaction Flow and Test Automation). System Plugins cannot be deactivated, deleted, or edited by the user. System Plugins are installed on the AppMon Server and are automatically deployed to all connected AppMon Clients or AppMon Collectors as necessary. Installation of a System Plugin requires a restart of the extended Server, Collector, or Client.
User Plugin	Action	Actions are added to Incident Rules and are executed whenever the conditions defined in the Incident Rule are met.
	Library	Any valid OSGi bundle can be added as a library. The only exception is a bundle that is used internally by the AppMon installation. Internally used bundles cannot be altered by the user because the alteration could harm the integrity of the AppMon installation. However, internally used bundles can be used to fulfill bundle requirements of user-defined Plugins, such as Tasks, Actions, or Monitors.
	Metric Group	Metric Groups are divided into Agent Metric Groups and Monitor Metric Groups. An *Agent Metric Group* is used to specify metrics based on Java JMX MBeans, IBM WebSphere PMI counters, and Microsoft Windows performance counters. Measures based on metrics from Agent Metric Groups provide measurements from one or multiple Agents and do not require a Monitor. A *Monitor Metric Group* defines metrics for a specific Monitor. Measures based on metrics from Monitor Metric Groups provide measurements from one specific instance of a Monitor.

	Monitor	*Monitors* are scheduled for repetitive execution on an AppMon Collector. Each Monitor has a set of configuration parameters and provides an execution status. A Monitor also has a set of measure subscriptions that describe the measurements the Monitor has to perform and provide measurement data after execution.
	Task	A *Task* is scheduled for repetitive execution on an AppMon Server or AppMon Collector. Each Task has a set of configuration parameters and provides an execution status.
Core Plugin	Webstart	Client
FastPack		A *FastPack* is a bundle of common resources that is installed via the standard installation but is not displayed in the list of installed Plugins.

Plugin development environment

Some User Plugins are provided by AppMon. User Plugins may also be developed by end users, using the Plugin Development Environment (PDE) within the AppMon Client. They are typically executed on the Server, the Collector, or the Agent. Installation of a User Plugin does not require a restart of the Server or Collector.

User Plugins must follow strict rules and interfaces defined by AppMon. See Manage and Develop User Plugins for details.

Building FastPacks

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

File structure

Resource types

Related pages

OSGi manifest file rules

Plugin.xml documentation

eServices

Overview

An AppMon FastPack, also called a *Resource Pack*, bundles different resources into one installation file that can be imported into AppMon just like a Plugin.

A Fastpack may contain the following resources:

- System Profiles
- Dashboards
- Sensor packs
- License files
- User Plugins
- Any other resource that is not executable (e.g. text files)

FastPacks are not listed in the Plugin overview after installation. Any selection of the FastPack items can be removed manually in the shell.

> **Note**
>
> You can use a FastPack to import a `.key` or license file, but you will need to activate the file manually in eServices and then import the activated license file. A quicker and more flexible way to import a license file is to use the Client. Select **Settings > Dynatrace Server > License** to display the license **Information** pane, then click **Import** and follow the prompts.

File structure

FastPack Installation files are archive files that define a valid OSGi bundle. A FastPack may have the file extension `.jar` or `.dtp`.

A Fastpack needs at least a manifest file `META-INF/MANIFEST.MF` to define the FastPack version, and a `[plugin.xml](/support/doc/appmon/images/integrations-and-extensions/plugin-architecture/building-fastpacks/plugin.xml)` file to define the FastPack's contents. The `[plugin.xml](/support/doc/appmon/images/integrations-and-extensions/plugin-architecture/building-fastpacks/plugin.xml)` file always has to reside in the root directory of the FastPack. An extension for the extension point `com.dynatrace.diagnostics.InstallerContent` has to be defined in the `plugin.xml` of the FastPack:

`Plugin.xml` basic structure:

```
<plugin>
   <extension id="Installer Content" point="com.dynatrace.diagnostics.InstallerContent">
      <metainfo installer_type="resourcepack" name="<YOUR_FASTPACK_NAME>">
         <!-- Your resource entries, see below -->
      </metainfo>
   </extension>
</plugin>
```

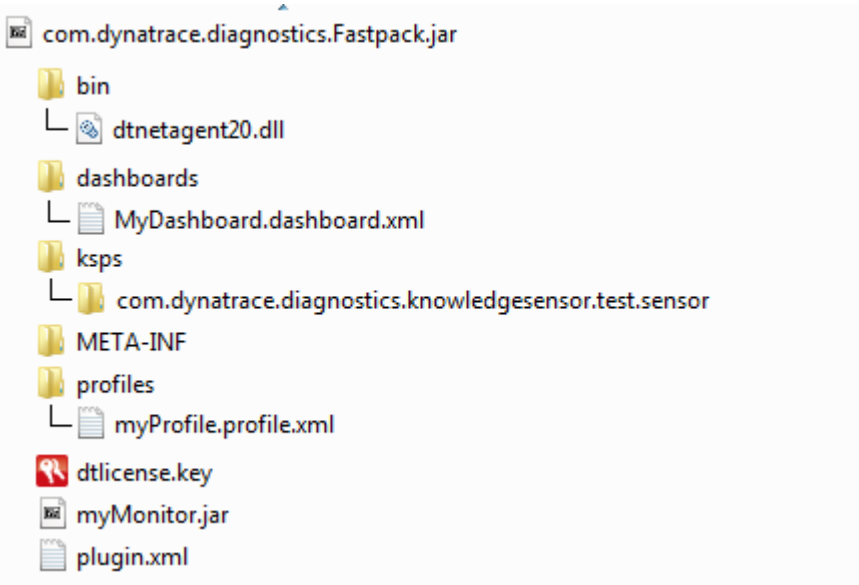
Resources in the FastPack can be organized into subdirectories. You can download a sample FastPack to review the typical contents.

Example Structure

This example contains:

- A User Plugin (specifically, a Monitor) and a license file in the root directory.
- A System Profile, a dashboard, a Sensor pack, and a generic resource (`.dll`) in different subdirectories.

Example FastPack Contents



`plugin.xml` content:

```
<plugin>

   <extension id="Installer Content" point="com.dynatrace.diagnostics.InstallerContent">
      <metainfo installer_type="resourcepack" name="DemoFastPack">
         <!-- Your resource entries -->
         <resource resource="ksps/com.dynatrace.diagnostics.knowledgesensor.test.sensor/"
         <resource resource="bin/dtnetagent20.dll" resource_type="resource" target_dir="l
            <instance instance="server"/>
         </resource>
         <resource resource="dashboards/MyDashboard.dashboard.xml" resource_type="dashboa
         <resource resource="profiles/myProfile.profile.xml" resource_type="systemProfile
         <resource resource="dtlicense.key" resource_type="licenseFile"/>
         <resource resource="myMonitor.jar" resource_type="userPlugin"/>
      </metainfo>
   </extension>
</plugin>
```

A `MANIFEST.MF` file needs to be in a `META-INF` directory.

```
Manifest-Version: 1.0
Bundle-ManifestVersion: 2
Bundle-Name: Demopatch
Bundle-SymbolicName: com.dynatrace.diagnostics.demopatch;singleton:=true
Bundle-Version: 4.1.0.2599
Bundle-Vendor: dynaTrace software GmbH
Implementation-Version: 4.5.0.2599
Require-Bundle: com.dynatrace.diagnostics.sdk
```

Most resources are identified to the AppMon system by the `resource_type` tag in the `plugin.xml` file. This tag tells the installation system where to install a particular resource. For generic resources, it is necessary to define the installation location in the `[plugin.xml] (/support/doc/appmon/images/integrations-and-extensions/plugin-architecture/building-fastpacks/plugin.xml)` file: specifically, the installation instance and `target_dir` relative to the root directory.

In the example, the generic resource is installed on the Server in the directory `<server-root>/lib`.

Resource types

The table describes the resource types that you can include in a FastPack. Note that the value for `resource_type` is case-sensitive.

Resource Type	Value of `"resource_type"`	Remarks
System Profile	`systemProfile`	
Dashboard	`dashboard`	
Sensor Pack	`sensorPack`	Must be the directory containing the **sensor's** `plugin.xml` (compare with an unzipped `.dtcs` file).
License File	`licenseFile`	
User Plugin	`userPlugin`	May be a directory or `.jar` file.
Generic Resource	`resource`	May be a directory or a single file. Needs at least one installation instance (`server` or `collector`) and a subdirectory entry.

Restrictions

FastPacks are not allowed to install core executable code of the AppMon system (resource_type: coreBundle and agentRes).

Manage and develop user plugins

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Introduction

Plugin types

Library

Related pages

Developing a user plugin

Plugin best practices and examples

Plugin frequently asked questions

Plugin management

URL monitor

Introduction

User Plugins allow you to extend the product's functionality for automation and monitoring. You can generate basic Plugin skeletons through the AppMon Client, which you also use to compile, test, and deploy the Plugins.

Plugin types

A User Plugin can be any of four types: Action, Task, Monitor, or Agent Metric Group. The Action, Task, and Monitor User Plugins contain code that is executed by the AppMon Server or Collector. An Agent Metric Group User Plugin only contains configuration data that defines Metric Groups, metrics, and their default configurations.

The following table summarizes how the User Plugin types are executed.

Plugin Type	Execution on Server	Execution on Collector	Execution on Agent
Action	x	x	
Task	x	x	
Monitor	(embedded. Collector)	x	
Agent Metric Group			x

Action

Actions are not scheduled for repetitive execution. An Action can be added to an Incident Rule so it is executed whenever the conditions defined by the Incident Rule are met. Like Tasks, Actions get configuration parameter values for execution and provide an execution status after execution. Actions also get information about the condition that caused the Incident, e.g. a measurement that violated a threshold.

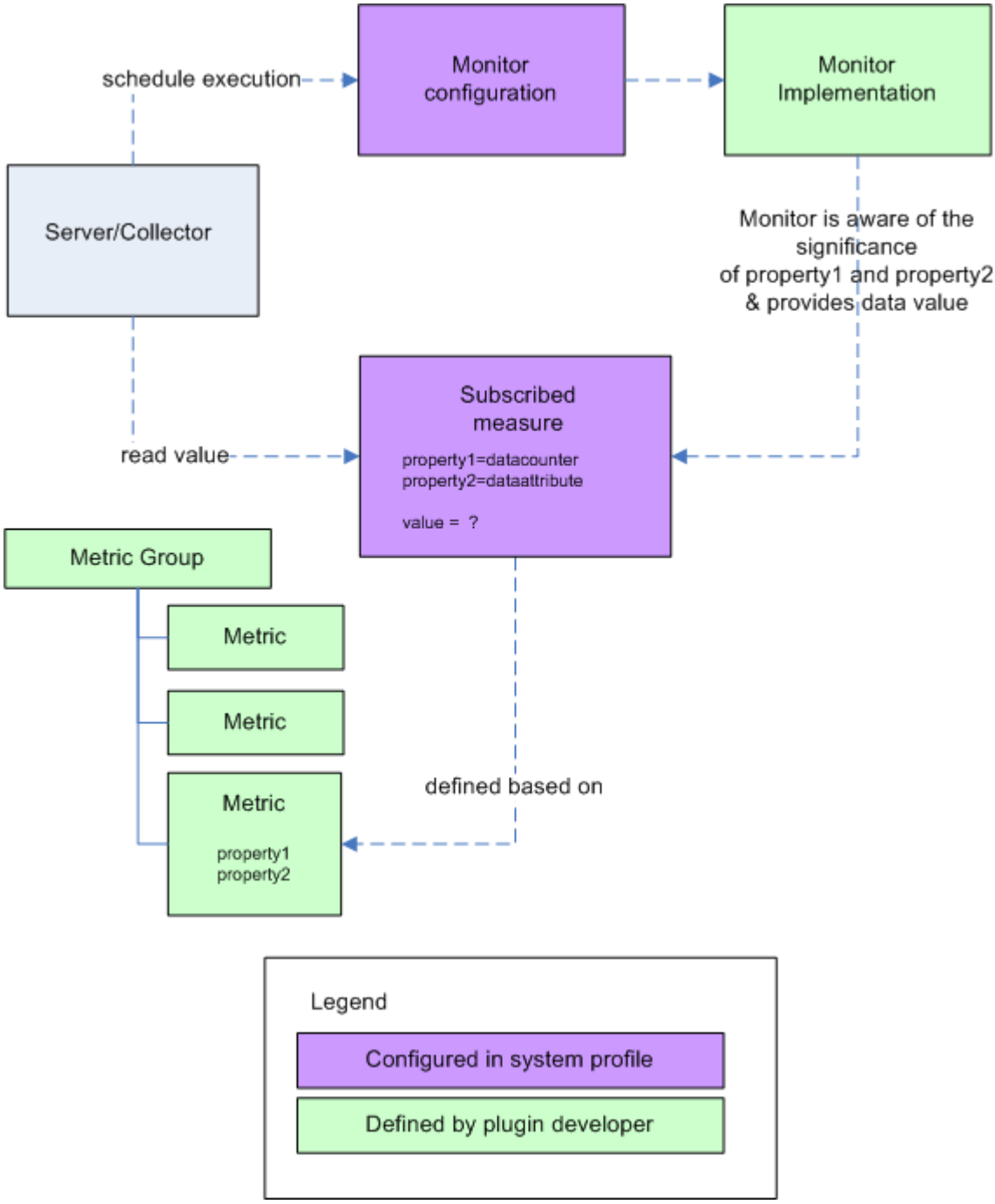
Task

A Task can be scheduled for repetitive execution on the Server or on a connected Collector. You can define execution schedules and share them among different Tasks and Monitors. On execution, a Task gets a set of configuration parameter values from the AppMon Plugin architecture. When execution is completed, the Task provides an execution status containing data about the execution.

Monitor

Monitors are scheduled for repetitive execution on a Collector. Like Tasks and Actions, Monitors get configuration parameters and provide an execution status. A Monitor also gets a set of measure subscriptions that describe the measurements the Monitor has to perform, and it provides measurement data after execution.

The figure illustrations how a Monitor interacts with an AppMon Server or Collector.



Monitor interaction with server/collector

Dynamic monitor measures

Monitor Plugins can generate dynamic measures on their own. While it is still necessary to define and subscribe a base measure for the Monitor, you can create additional submeasures that are analyzed independently. See Plugin Best Practices and Examples for a code sample.

Monitor hosts

You can configure a set of hosts for each Monitor configuration, so that the Monitor is run once per scheduled execution for each of the configured hosts. See Plugin Best Practices and Examples for an example how to use the configured host in the Monitor code, and URL Monitor for an example how to configure a host set.

Metric group

A Metric Group can be either an Agent Metric Group or a Monitor Metric Group.

An Agent Metric Group is used to specify metrics based on Java JMX MBeans, IBM WebSphere PMI counters, and Microsoft Windows performance counters. Measures based on metrics from Agent Metric Groups provide measurements from one or multiple Agents. These measures do not require a Monitor.

A Monitor Metric Group defines metrics for a specific Monitor. Measures based on metrics from Monitor Metric Groups provide measurements from one specific instance of a Monitor.

Library

Any valid OSGi bundle can be added as library. The only exception are bundles that are used internally by the AppMon installation. You cannot alter those bundles because the alteration could harm the integrity of the AppMon installation. However, internally used bundles can be used to fulfill bundle requirements of User Plugins, just like Tasks, Actions, or Monitors.

Plugin management

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Manage plugins

Plugin properties

Overview

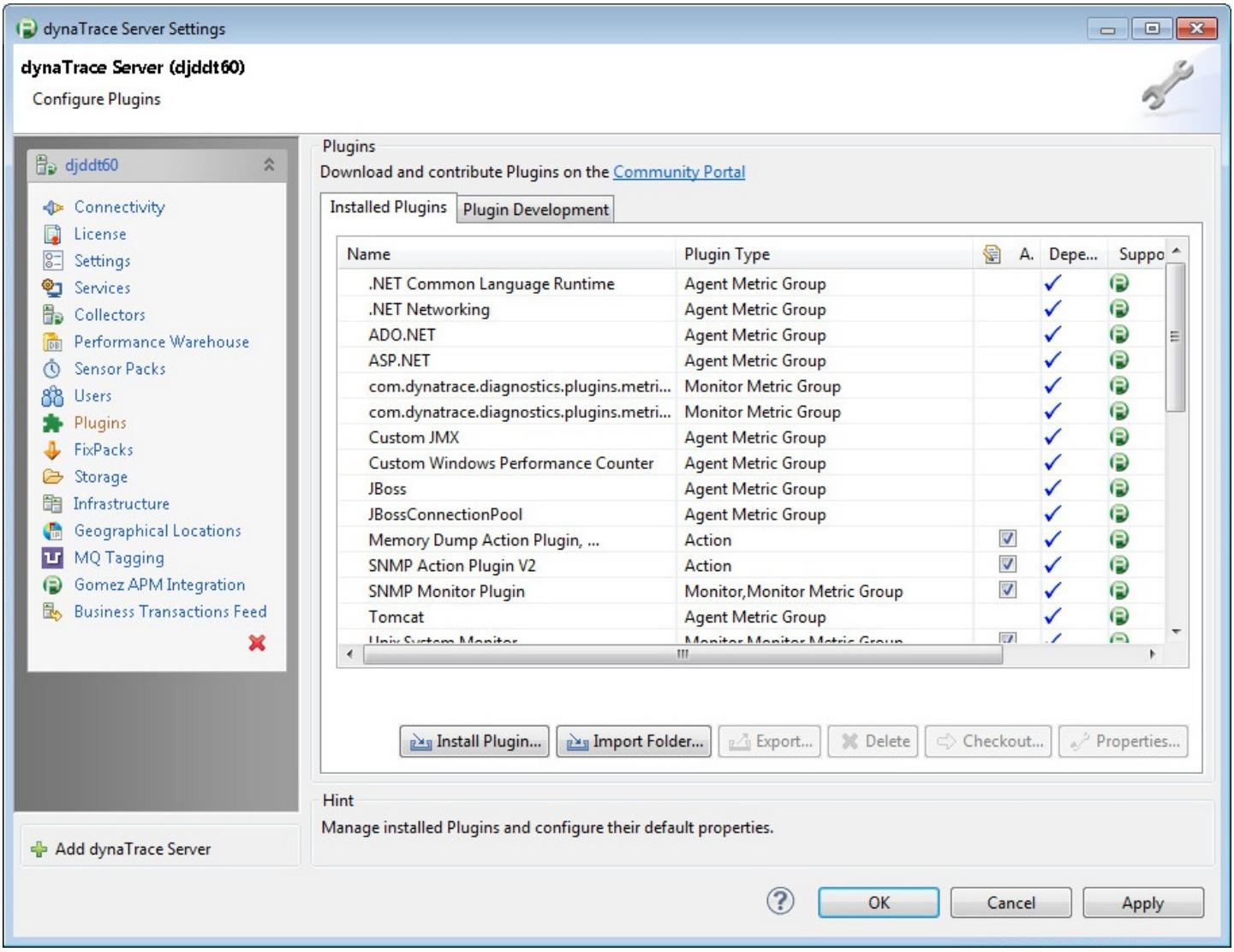
This section covers installation, uninstallation, and configuration of User Plugins.

Manage plugins

Manage AppMon Plugins through the the **Plugins** pane of the **dynTrace Server Settings** dialog box. To open the dialog box, select **Tools > Manage Plugins** in the AppMon Client.

Installed plugins

The **installed Plugins** tab provides an overview of all dynaTrace Plugins that are installed on the server.



Installed Plugins

The tab contains the columns described in the following table.

Column names may be truncated, depending on the column width. To adjust a column width, click-and-drag the column's right border. To view the full name without adjusting the column, hover over the name.

Column	Description
Name	The name of the Plugin, as it is referenced in System Profiles.
Plugin Type	In addition to the interface type (Task, Action, Monitor), other type definitions are listed to fully describe the Plugins. For example, a Plugin type may be listed as **Monitor, Monitor Metric Group**.

Active	Plugins that can be disabled have a checkbox in this column. If the checkbox is selected, the Plugin is active. When you clear the checkbox for a Plugin, all subsequent executions of that Plugin are canceled; this has the same effect as suspending the Plugin schedule. If the Plugin is currently executing when you clear the checkbox, it continues until execution is complete. Select the checkbox again to restore the Plugin to active status.
Dependencies resolved	This column Indicates the installation state of a Plugin. A blue checkmark indicates that the bundle is correctly installed and all bundle requirements could be resolved. A red exclamation mark indicates that the bundle could not be resolved and is not working. Review the detailed information about a Plugin to identify dependencies that could not be resolved.
Support level	A dynaTrace icon in this column means that a plugin is supported by AppMon. The community icon means that this Plugin is supplied and supported by the Dynatrace Community, which includes the AppMon development team, the Community team, partners, and customers. See Support Levels for more details.
Version	The version of the installed Plugin.

You can display these details for a Plugin in a tooltip by hovering over the Plugin row. To copy any or all of the information in the tooltip, press F2 when the tooltip is displayed.

Right-click a Plugin to display a context menu that provides the following functions. Some of these functions are also available through buttons at the bottom of the pane. If a function cannot be performed on a Plugin (for example, the Plugin's properties cannot be changed), the menu item and button are disabled.

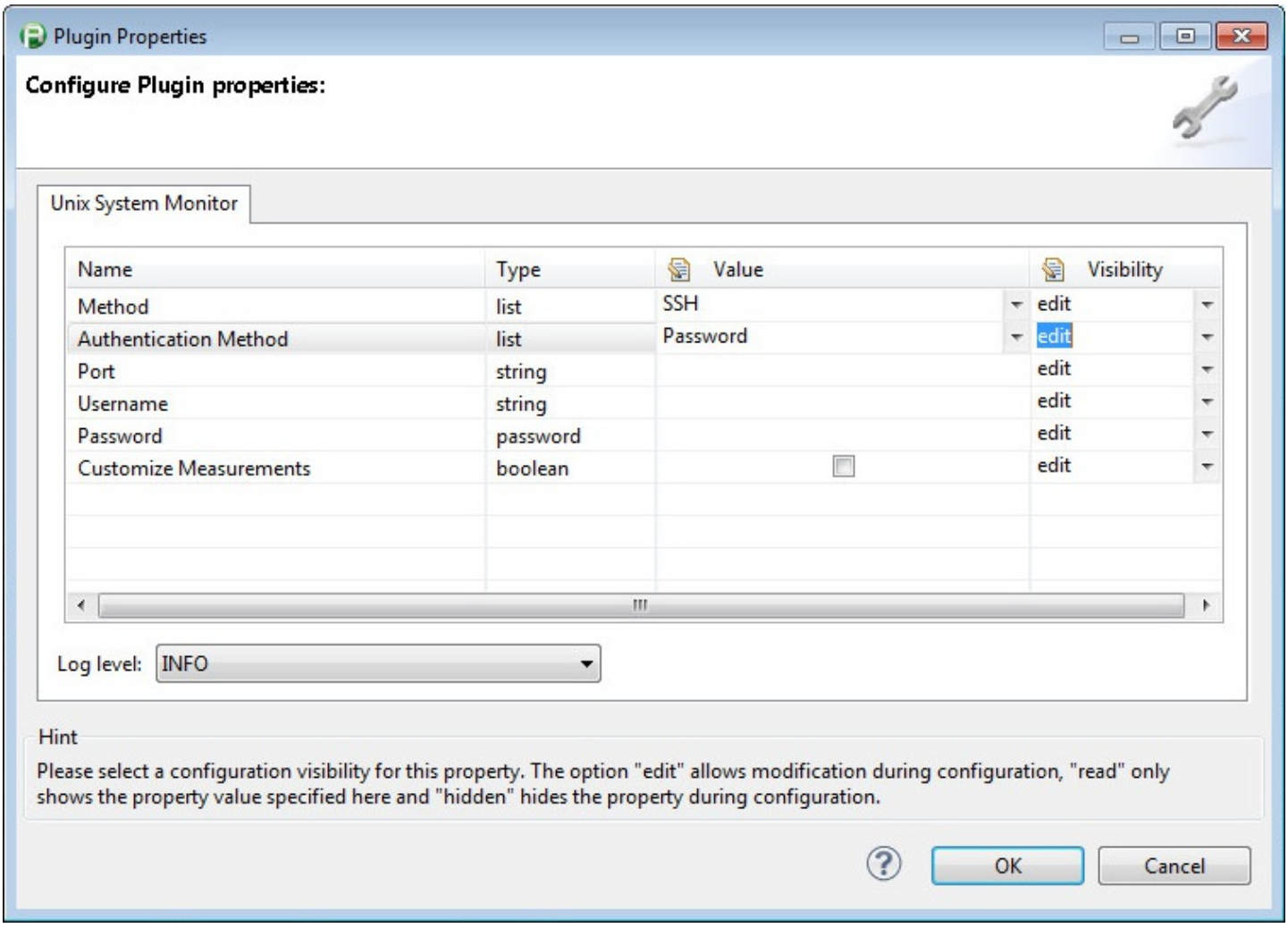
> **Warning**
>
> Actions marked with an asterisk (*) are implemented immediately, and cannot be undone by clicking the **Cancel** button.

Function	Description
Details	Displays the Plugin details in a popup window, similar to hovering over the Plugin to display a tooltip.
Install Plugin*	Select a Plugin `.jar` file to install.
Import Folder*	Select a Folder that contains a bundle to install.
Export*	Download the selected Plugin from the AppMon Server and save it to the specified location. It is recommended that you export a Plugin to a backup directory before making any changes to it.

Delete*	Delete the selected Plugin and its default configuration from the Server. The execution of the Plugin will be stopped, but the Plugin schedule is not removed. If you re-import the Plugin, the scheduled execution will resume.
Checkout*	Download the selected Plugin from the server, extract it to the specified location, and automatically add it to the Development page. Only the Task, Action, and Monitor Plugin types can be checked out. For all other types, the **Checkout** button is disabled. Some Plugins may not contain any source because of their license policies, so the checkout for these Plugins only includes the libraries but no source.
Properties	Open the **Plugin Properties** dialog box to configure the Plugin's properties. See below for details.
Find	Find a specific Plugin in the list, or filter the list to show only Plugins with the specified string in their names. Select the filter icon to display a filtered list; deselect the icon to find one search result at a time.

Plugin properties

Use the **Plugin Properties** dialog box to specify Server-wide default settings for the selected Plugin.



Plugin Properties

The dialog box contains the following columns.

Column	Description
Name	The Plugin's properties, which are specific to the Plugin.
Type	The data type of the property.
Value	The value assigned to the property. For properties that have Boolean values, this column contains a checkbox that is selected for TRUE (YES) or cleared for FALSE (NO).
Visibility	You can use this column to restrict the configuration of the Plugin. Three options are available: • **edit**: This is the default option. The property's value can be modified. • **read**: The value is read-only and cannot be modified. • **hidden**: The property will not be displayed (and thus cannot be modified). When the visibility is set to **read** or **hidden**, the property value is used for all Plugin instances. When the visibility is set to **edit**, the property value is used as the default value, but it can be modified for individual instances. Changing the visibility from **read** or **hidden** to **edit** restores the value on each instance that was entered before visibility was set to **read** or **hidden**.

Developing a user plugin

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Using the plugin development tab

Using the New Plugin wizard

Testing plugins

Status codes

Plugin files and directory structure

API / PDK documentation download

Related pages

Plugin best practices and examples

Plugin management

FastPacks and plugins

Downloads

Overview

The Plugin Development Environment (PDE) enables you to create bundle specifications for AppMon Plugins: a valid manifest file, a valid `plugin.xml` file, source file skeletons, and optional libraries, resource files, and bundle dependencies. You also use the PDE to compile and build a Plugin, test it on a connected AppMon Server or Collector, and to deploy it to a Server.

The PDE does not provide an IDE for developing and debugging Plugin source code. You can use your preferred IDE for development. A `.project` file for Eclipse is generated automatically. See Best Practices and Examples for details.

OSGi and Eclipse extensions mechanism

The AppMon architecture is based on OSGi bundles that use the Eclipse Extensions Mechanism. All Plugins created by the PDE are also bundles of those types. The Eclipse Extensions Mechanism enables you to define Extension Points and Extensions. *Extension Points* define the interfaces that an application or component provides/ An *Extension* defines a specific implementation for a specific Extension Point.

The AppMon Plugin architecture provides an Extension Point for defining the configuration parameters of a Plugin, and an Extension Point for the Plugin types Action, Monitor, Task, Agent Metric Group, and Monitor Metric Group. These Extension Points enable you to share an Extension that defines the Plugin configuration among different Plugin types. For example, a configuration Extension for a web service, containing a URI and a set of parameters, can be used by a Monitor that collects performance data for a web service and by an Action that uses a web service to post incident data.

The example `Plugin.xml` code, below, shows the relations between different Extensions.

The example includes:

- A Plugin configuration Extension named **Signal Generator**.
- A Monitor Extension named **Signal Generator Monitor**, which uses the **Signal Generator** configuration because its `configId` points to the `id` of the **Signal Generator** extension.
- A Monitor Metric Group Extension named **Signal Generator Metric Group**, for the Monitor Extension **Signal Generator Monitor** because its `monitorId` points to the `id` of the extension **Signal Generator Monitor**.

```
<extension point="com.dynatrace.diagnostics.pdk.pluginconfig"
   id="com.dynatrace.diagnostics.plugins.SignalGenerator"
   name="Signal Generator">
        <!-- Definition of configuration parameters -->
        ...
</extension>
<extension point="com.dynatrace.diagnostics.pdk.monitor"
   id="com.dynatrace.diagnostics.plugins.SignalGeneratorMonitor"
   name="Signal Generator Monitor">
  <plugin class="sg.SignalGenerator" configid="com.dynatrace.diagnostics.plugins.SignalGe
 </extension>
 <extension point="com.dynatrace.diagnostics.pdk.monitormetricgroup"
   id="com.dynatrace.diagnostics.plugins.SignalGeneratorMonitorMetricGroup"
   name="Signal Generator Metric Group">
        <metricgroup monitorid="com.dynatrace.diagnostics.plugins.SignalGeneratorMonitor"
               <!-- Definition of monitor metric group -->
               ...
        </metricgroup>
</extension>
```

Reuse, extend, or develop from scratch?

Before starting to develop a new Plugin, it is always useful to check whether a Plugin that does the job is already available, or an existing Plugin can be altered or extended to fulfill the new requirements. Check the Plugins provided with the AppMon installation, and the Community Downloads Overview. If you find a candidate for alteration or extension among the already installed Plugins, check it out to a folder and revise it. If you find a candidate in the Community, download and extract it to a folder, and add the extracted Plugin to the Plugin development projects. Detailed information about these tasks is provided below.

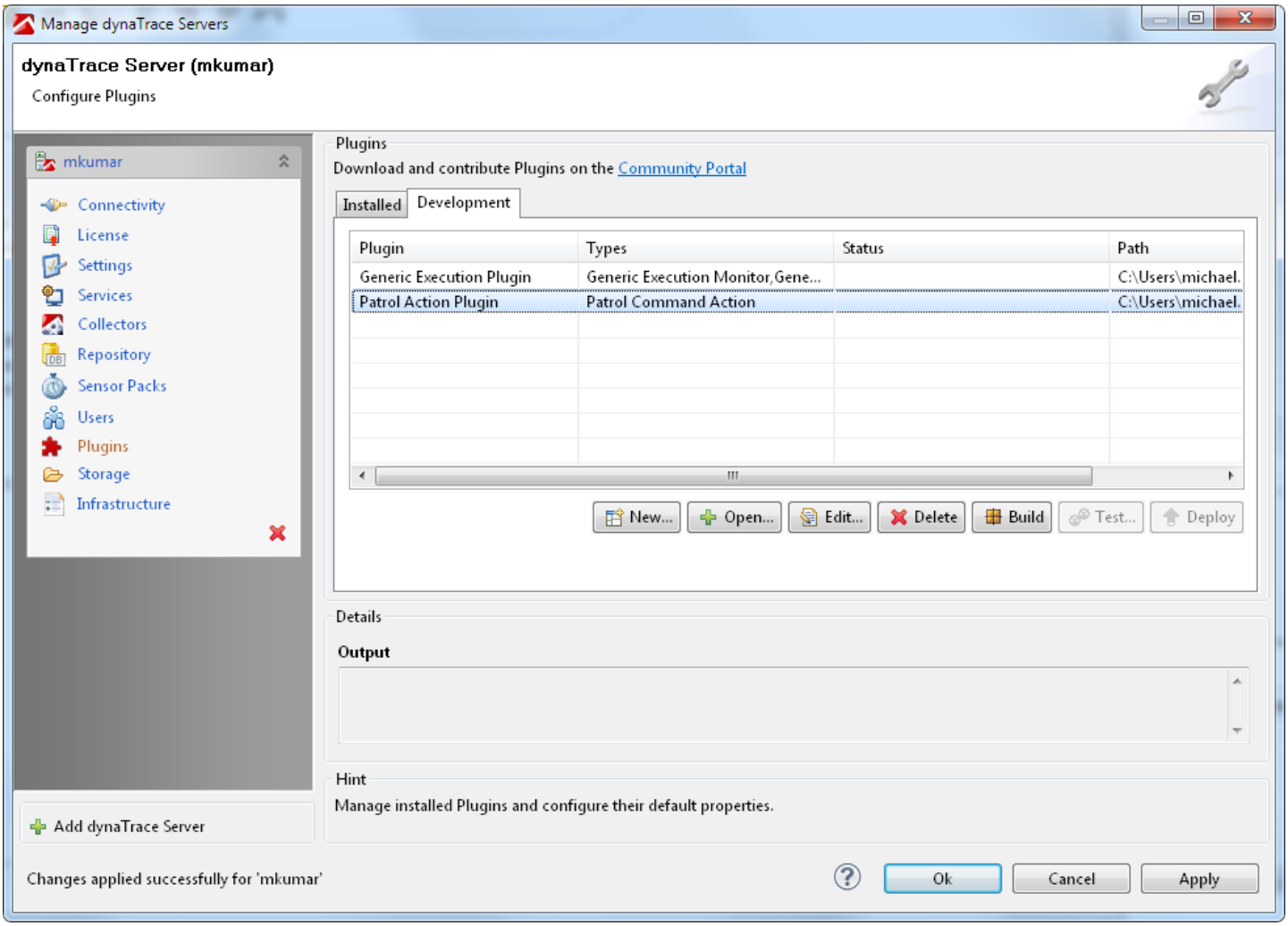
> **Note**
>
> Some Plugins may not contain source code, because of licensing restrictions. In this case, appropriate source skeletons are created on checkout. These skeletons implement all required interfaces, but do not contain functionality.

If no adequate candidate can be found, create a new Plugin project.

After you complete your work on a Plugin is finished, consider sharing it with the Dynatrace Community. See the Community Downloads Overview for detailed information about how to share your Plugins.

Using the plugin development tab

Develop new Plugins and revise Plugins through the **Plugin Development** tab of the **Dynatrace Server Settings** dialog box. To display this tab, select **Tools > Manage Plugins** in the AppMon Client, then select the tab.



Plugin Development Tab

The tab displays the following information about each listed Plugin:

- **Plugin** – The name of the configuration Extension of the bundle, e.g. **Signal Generator**.

- **Types** – The names of all Action, Monitor, and Task Extensions of the bundle, e.g. **Signal Generator Monitor**.
- **Status** – Information about the current state of the Plugin.
  Note that The status **loading failed** indicates that the structure of the Plugin files or directories is invalid. You must correct this problem manually with a text editor or File Manager.
- **Path** – Displays the root path of the Plugin project in the file system.

You can perform the actions listed in the table below through the **Plugin Development** tab.

Actions marked with an asterisk (*) will be performed immediately, and cannot be undone by clicking **Cancel**.

Action	Description
New*	Open the Plugin wizard to create a new Plugin.
Open	Add an existing Plugin to the list by choosing the directory that contains the extracted Plugin.
Edit*	Open the **Edit Plugin** wizard to change the configuration of the selected Plugin.
Delete*	Remove the Plugin from the list, and optionally delete the directory that contains the Plugin files.
Build*	Compile and package the Plugin into a single JAR file. This action cleans the build directory and downloads all dependent JAR files to the Client's `temp` directory. Hidden files are excluded from the build.
Test	Test the Plugin on the AppMon Server.
Deploy*	Install the built and tested Plugin on the AppMon Server.

To use the **Build**, **Test**, and **Deploy** actions, the Client must be connected to an AppMon Server. All other actions can be performed while the Client is not connected.

Using the New Plugin wizard

Use the **New Plugin Wizard** to create new Plugin projects, as well as to edit existing Plugin projects.

New Plugins are created automatically in the PDE base folder. You can specify the base folder either the first time you create a Plugin or in the **Directories** pane of the **Client Preferences** dialog box.

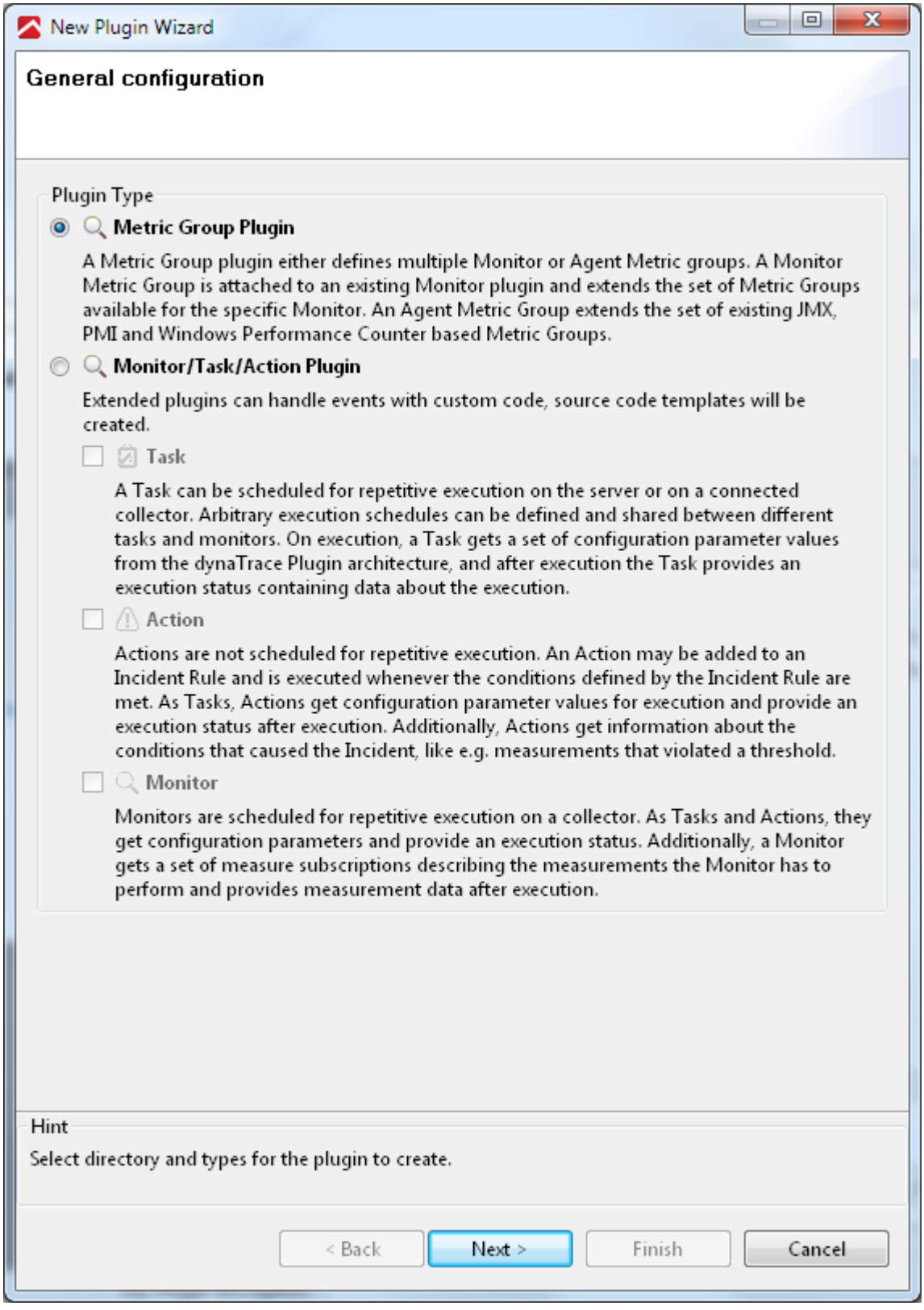
When you create a new Plugin project, the **New Plugin Wizard** generates a new Java class skeleton that implements the required interfaces, depending on the selected Plugin type(s): Action, Task, Monitor, or a combination of these.

General configuration

In the first screen of the **New Plugin Wizard**, select the Plugin type(s):

- **Metric Group Plug In**
- **Monitor/Task/Action Plug In** – If you select this type, you must also select **Task**, **Action**, **Monitor**, or a combination of these before you can continue.

After the Plugin is created, its type cannot be changed.

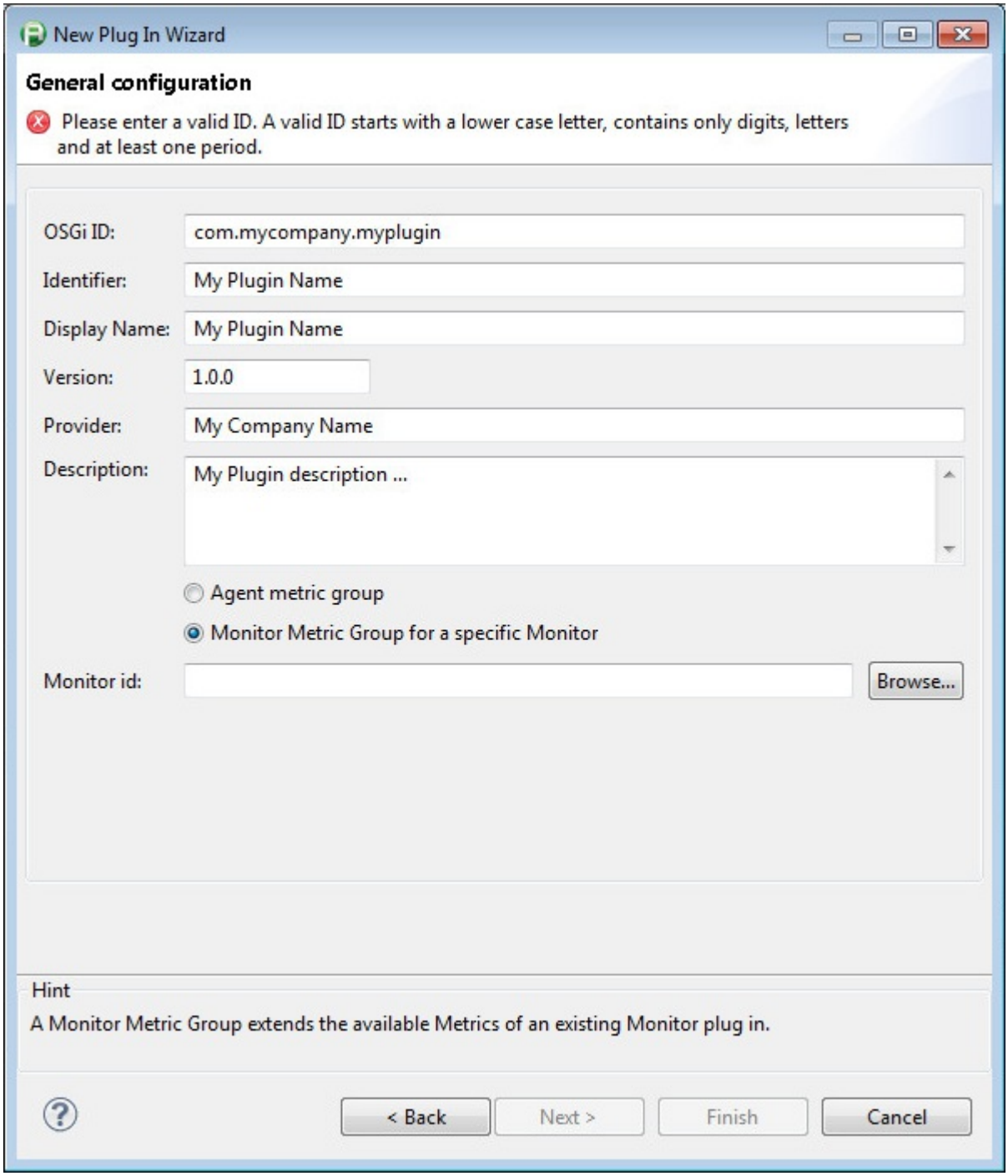


New Plugin wizard - general configuration

General configuration - metric group

When you select **Metric Group Plug In** as the Plugin type and click **Next**, the New Plugin Wizard displays configuration options for this type, with default entries in the fields. Replace these defaults with the information for your Plugin:

- **OSGi ID** – A unique identifier for the Plugin's JAR file. Usually the ID is a lowercase, dot-separated string that starts with `com` or `org` followed by the company's or organization's short name and the project name. It **must not** contain special characters like quotes (") or parentheses.
- **Identifier** – A descriptive name for the Plugin.
- **Display Name** – The name of the Plugin that will be shown in the **Server Preferences**.
- **Version** – The version of the Plugin, which can have four components: major, minor, and micro version and a qualifier string. The qualifier string may be, for example, a build number. At least the major version needs to be specified.
- **Provider** – The full name of the company or organization that provides this Plugin.
- **Description** – Optional text that describes the Plugin.
- **Metric Group type**: Select the subtype of this Metric group:
  - An **Agent Metric Group** defines metrics that are provided directly by the AppMon Agent.
  - A **Monitor Metric Group** defines additional metrics for an existing Monitor with a known ID.
- **Monitor id** – This field becomes available when you select **Monitor Metric Group**. Enter the identifier of an existing monitor. Use the browse button to pick a Monitor that is currently installed on the server.

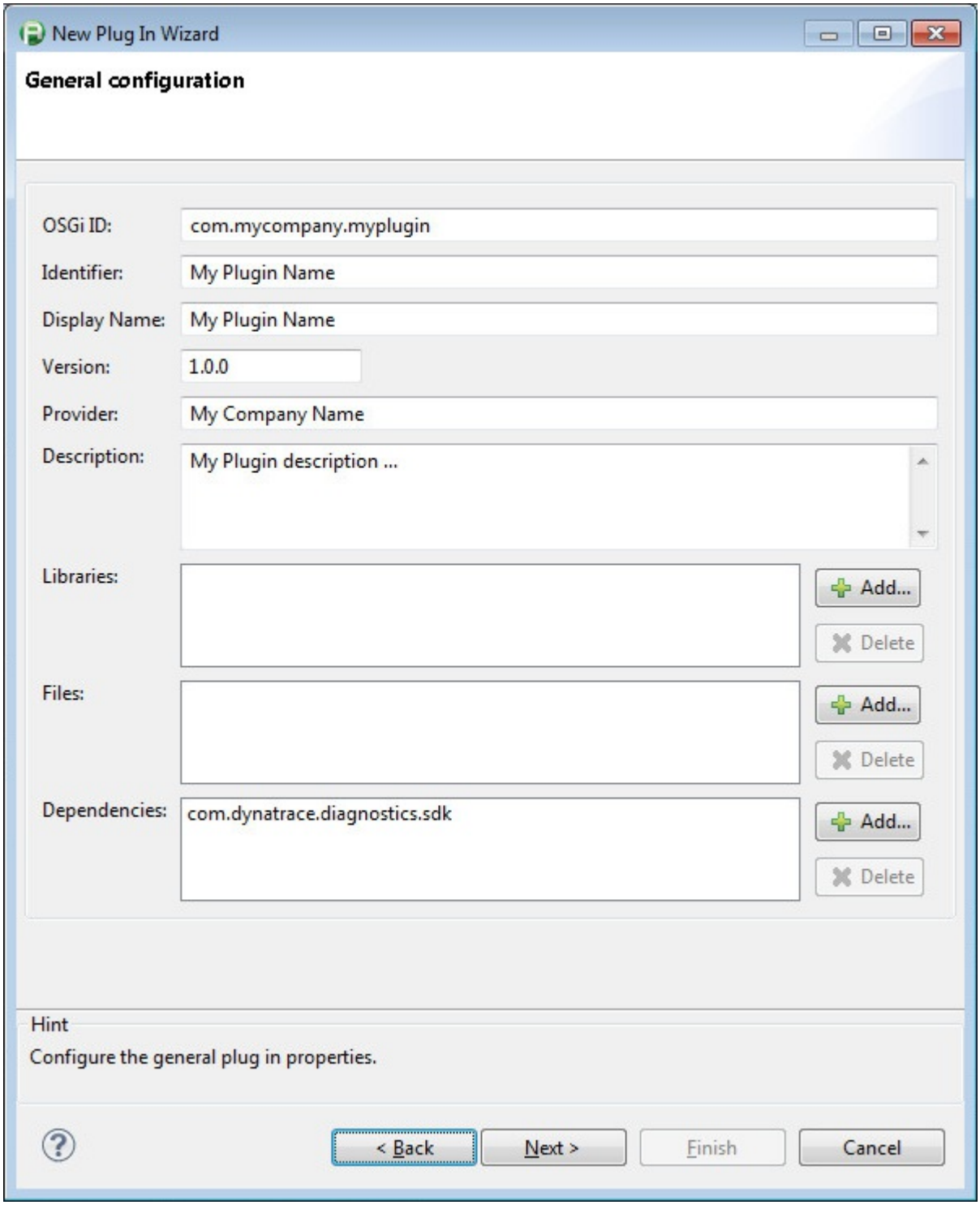


General Configuration - Metric Group

General configuration - monitor/task/action plugin

When you select **Monitor/Task/Action Plug In** as the Plugin type and click **Next**, the New Plugin Wizard displays configuration options for this type, with default entries in the fields. Replace these defaults with the information for your Plugin:

- **OSGi ID** – A unique identifier for the Plugin's JAR file. Usually the ID is a lowercase, dot-separated string that starts with `com` or `org` followed by the company's or organization's short name and the project name. It **must not** contain special characters like quotes (") or parentheses.
- **Identifier** – A descriptive name for the Plugin.
- **Display Name** – The name of the Plugin that will be shown in the **Server Preferences**.
- **Version** – The version of the Plugin, which can have four components: major, minor, and micro version and a qualifier string. The qualifier string may be, for example, a build number. At least the major version needs to be specified.
- **Provider** – The full name of the company or organization that provides this Plugin.
- **Description** – Optional text that describes the Plugin.
- **Libraries** – A list of libraries that will be packaged with the Plugin. Only JAR files can be added as libraries. The files will be copied to the `lib` subdirectory of the Plugin and will be added to the build classpath.
- **Files**– A list of resource files that will be packaged with the Plugin. You can add any file here, as needed. The files will be copied to the `res` subdirectory of the Plugin.
- **Dependencies** – if the Plugin requires other OSGi bundles to work, add them as dependencies. For an AppMon Plugin, you **must** set `com.dynatrace.diagnostics.sdk` as a dependency; it is entered in the field by default.
  The **Please select a dependency** dialog box that appears when you click **Add** has a filter at the top. To display only objects that begin with a specific string, type that string in the filter. Use a wildcard to filter for objects that have the string anywhere in the name. For example, typing `\*ja` in the filter includes `com.dynatrace.diagnostics.javascript_6.0.0.6620` and `javax.servlet_2.5.0.v201103041518` in the filtered list.

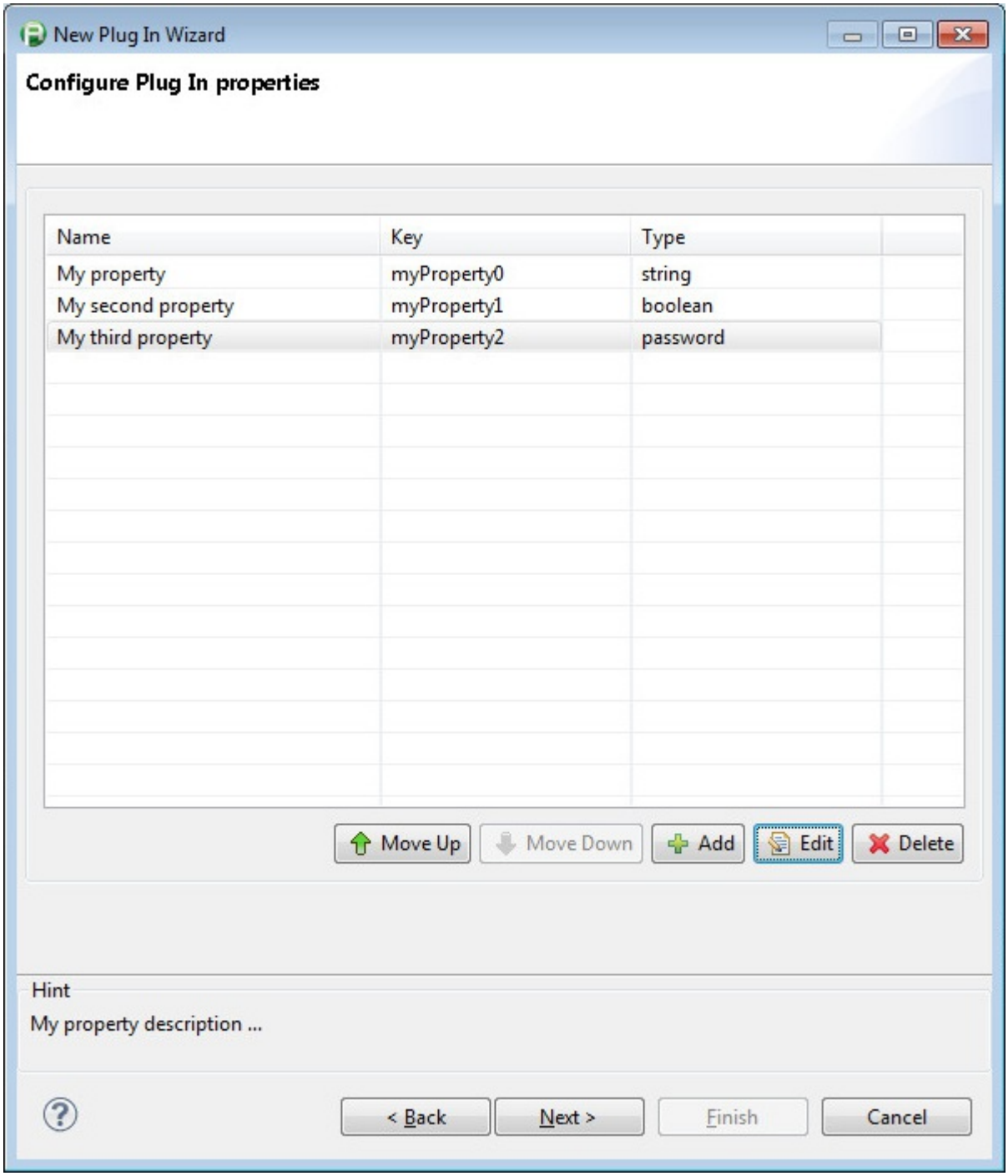


General configuration - monitor/task/action

Configure plugin properties

This screen only appears when you have selected **Monitor/Task/Action** as the Plugin type.

The configuration properties are Plugin configuration items of arbitrary size.

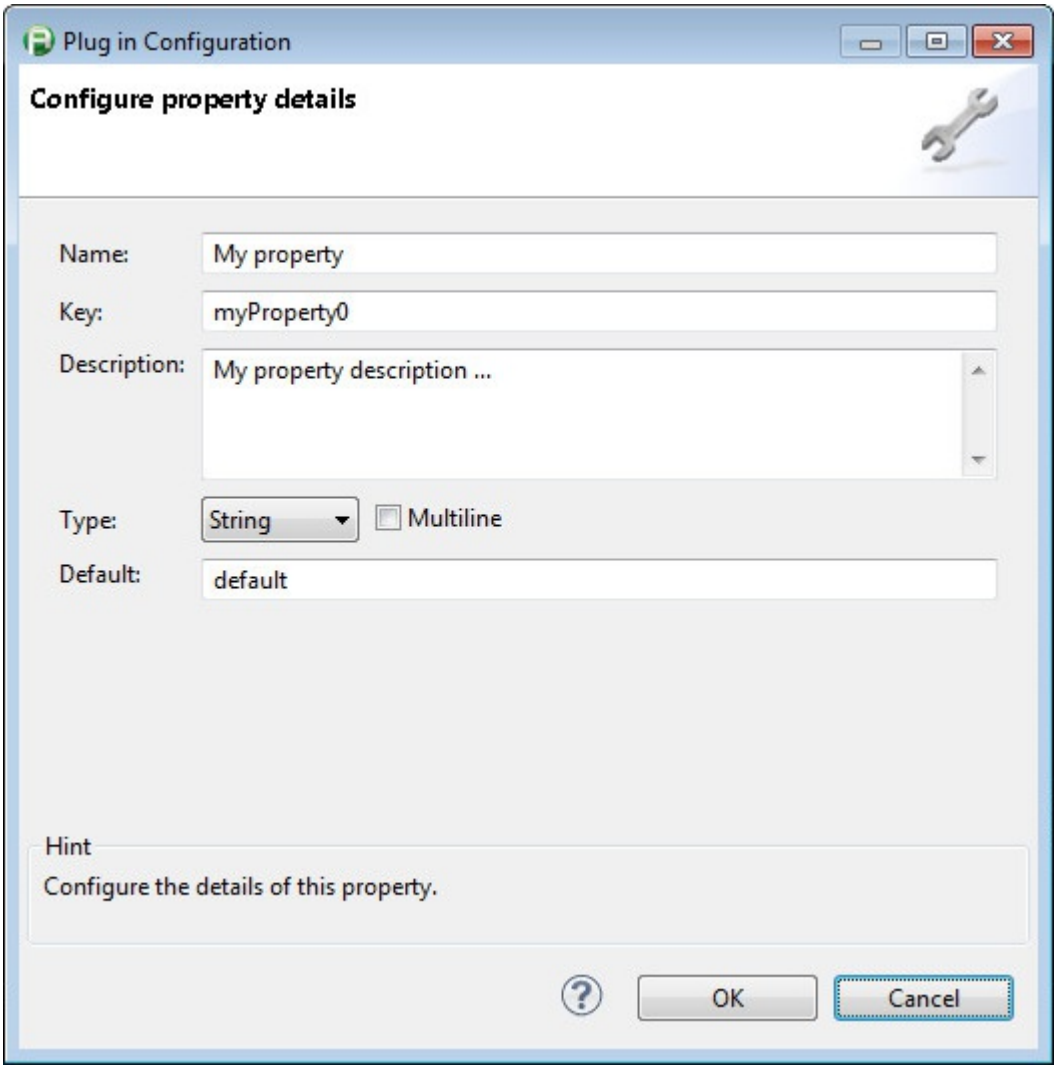


Configure Plugin Properties

Click **Add** to create a new property. Define the properties of the Plugin and set the default values.

- **Name** – The label for the property that will be displayed in the configuration dialog boxes.
- **Key** – A unique identifier for the property. It is used to retrieve the property value from the Plugin source code.
- **Description** – Optional text that describes the property.

- **Type** – The data type of the property. Supported types are:
  - String
  - Password
  - Boolean
  - Long – Integer numeric values
  - Double – Floating-point numeric values
  - Date
  - List – Set of predefined selectable strings
  - URL
- **Default value** – The default value for this Plugin. This default can be overridden in the **Server Settings** dialog box (see Plugin Management). When you select **List** as the property type, the **Default** field becomes a table in which you can type the strings; select a radio button in the **Default** column to specify which string is the default value.



Configure Property Details

You can control the visibility of Plugin properties based on the values of other properties. For example, the Boolean property **Authentication** controls the visibility of a String property **User** and a Password property **Password** so that **User** and **Password** are only visible if **Authentication** is set to `true`. However, you cannot set visibility through the **New Plugin Wizard**. You must modify the `plugin.xml` file to add visibility control.

The following code snippet implements the **Authentication** example.

```
<property key="auth" type="boolean" default="false"
          label="Authentication" description ="Use Athentication."/>

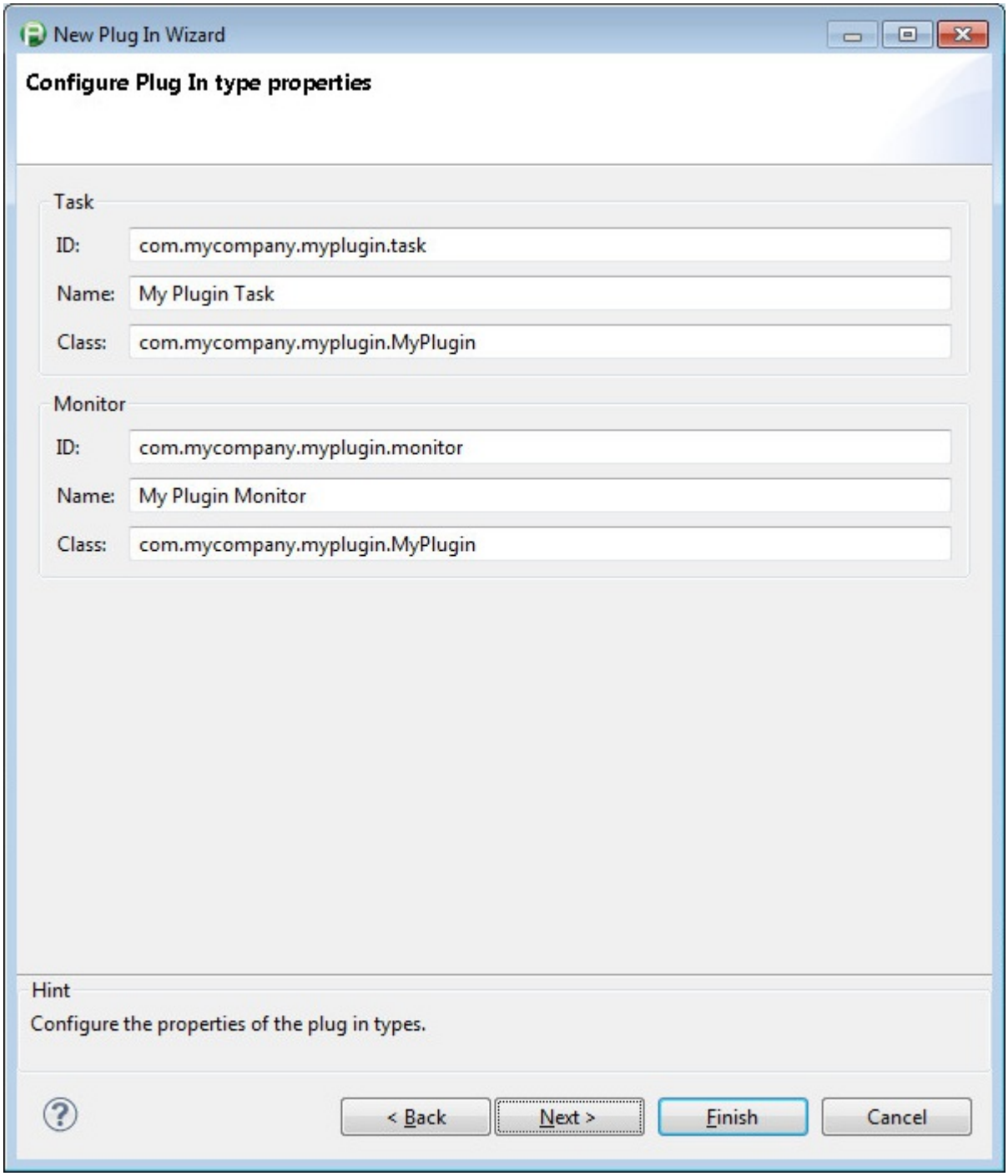
<property key="username" type="string" default="" label="Username" description="Enter the
          <visibility>
                <rule key="auth" value="true" />
          </visibility>
</property>

<property key="password" type="password" default="" label="Password" description="Enter t
          <visibility>
                <rule key="auth" value="true" />
          </visibility>
</property>
```

Configure plugin type

This screen only appears when you have selected **Monitor/Task/Action** as the Plugin type.

For each selected Plugin type (Action, Task, or Monitor), you must select a Java class to implement the the type-specific interface and a name to represent the Plugin type in the AppMon Client. If no class with the specific name exists, a new class will be created. If you have selected more than one Plugin type (for example, both Task and Monitor), you can use one Java class for all the selected types. The **Configure Plugin type properties** screen displays fields for each type, as shown in the figure.



Configure Plugin Type Properties

Enter the information for each Plugin type:

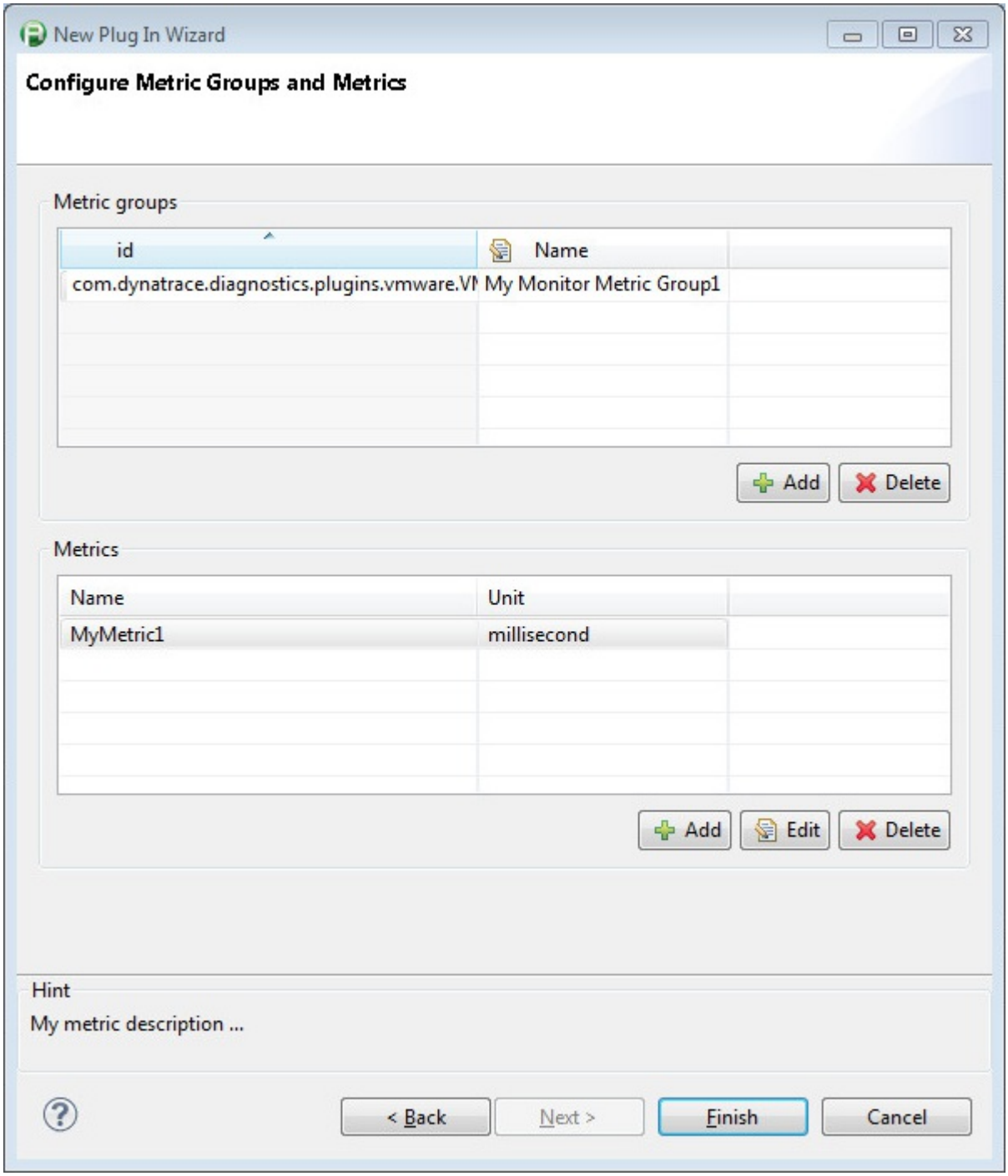
- **ID** – The ID is entered automatically. It is the OSGi ID you entered in the General configuration screen, with the type appended.
- **Name** – The name of the Plugin and type that will be shown when setting up the Plugin.

- **Class** – The fully qualified Java class name (*packagename.classname*) for this Plugin type. A simple template file will be created if it doesn't already exist. For the package name, the same naming rules apply as for the **ID** field. The class name is usually written in CamelCase.

Configure metric groups and metrics

This screen only appears for the Plugin types **Monitor**, **Monitor Metric Group**, and **Agent Metric Group**. Use the screen to add metric groups and add metrics to groups. You can also edit the Metric Group names. The names of Metric Groups and metrics must be unique.

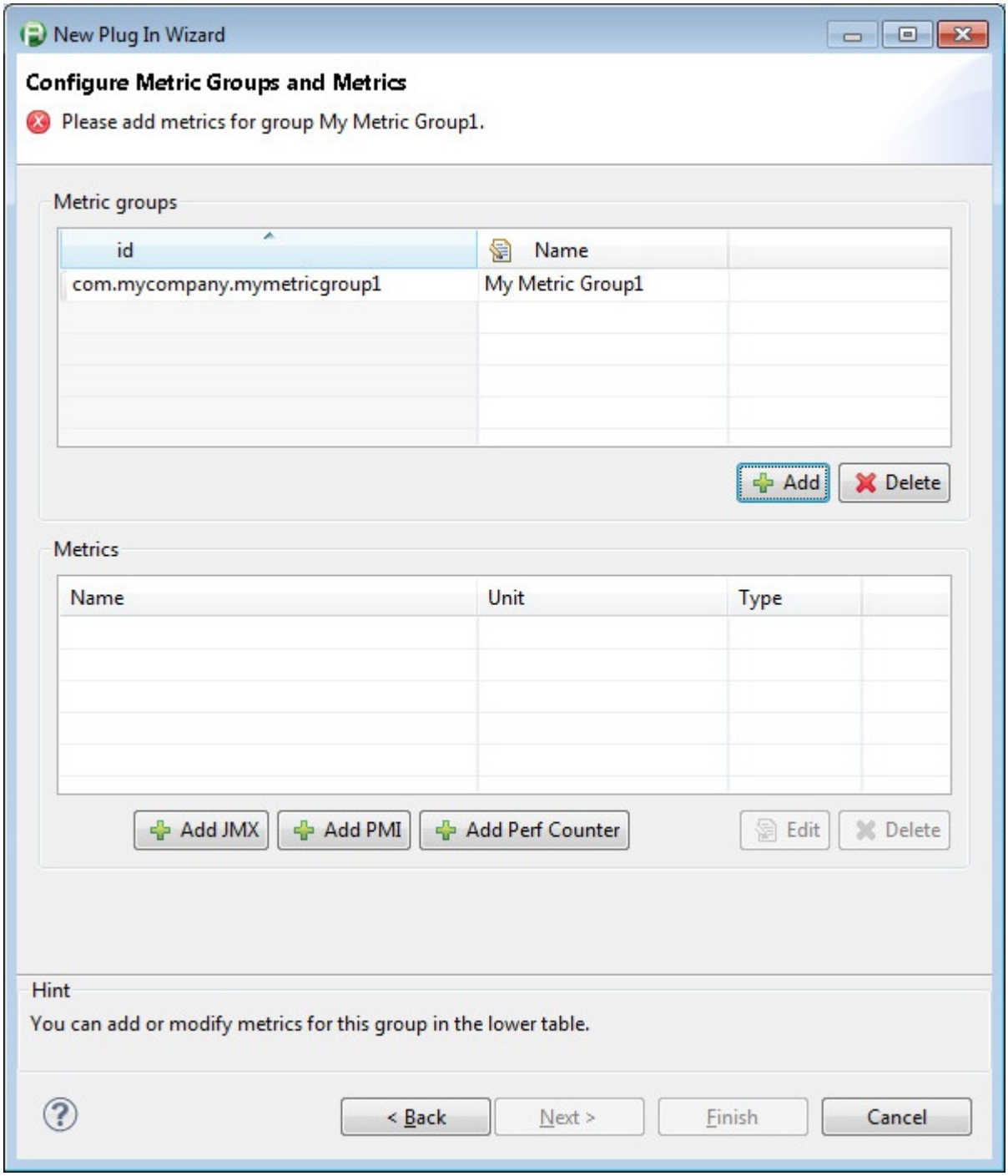
If the Plugin type is Monitor Metric Group, then when you add a **Metric group**, an **Add** button appears below the **Metrics** list. Clicking **Add** enters default values for the metric **Name** and **Unit**. Select the new metric and click **Edit** to define the metric name and properties as described below.



Configure Monitor Metric Groups and Metrics

If the plugin was configured as Agent Metric Group in step 2 this page will look slightly different: Instead of a generic add button buttons to explicitly add PMI, PerfMon and JMX Metrics are available.

If the Plugin type is Agent Metric Group, then when you add a **Metric group**, three buttons appear below the **Metrics** list: **Add JMX**, **Add PMI**, and **Add Perf Counter**. Clicking any of these buttons enters default values for the metric **Name** and **Unit**. Select the new metric and click **Edit** to define the metric name and properties as described below.



Configure Agent Metric Groups and Metrics

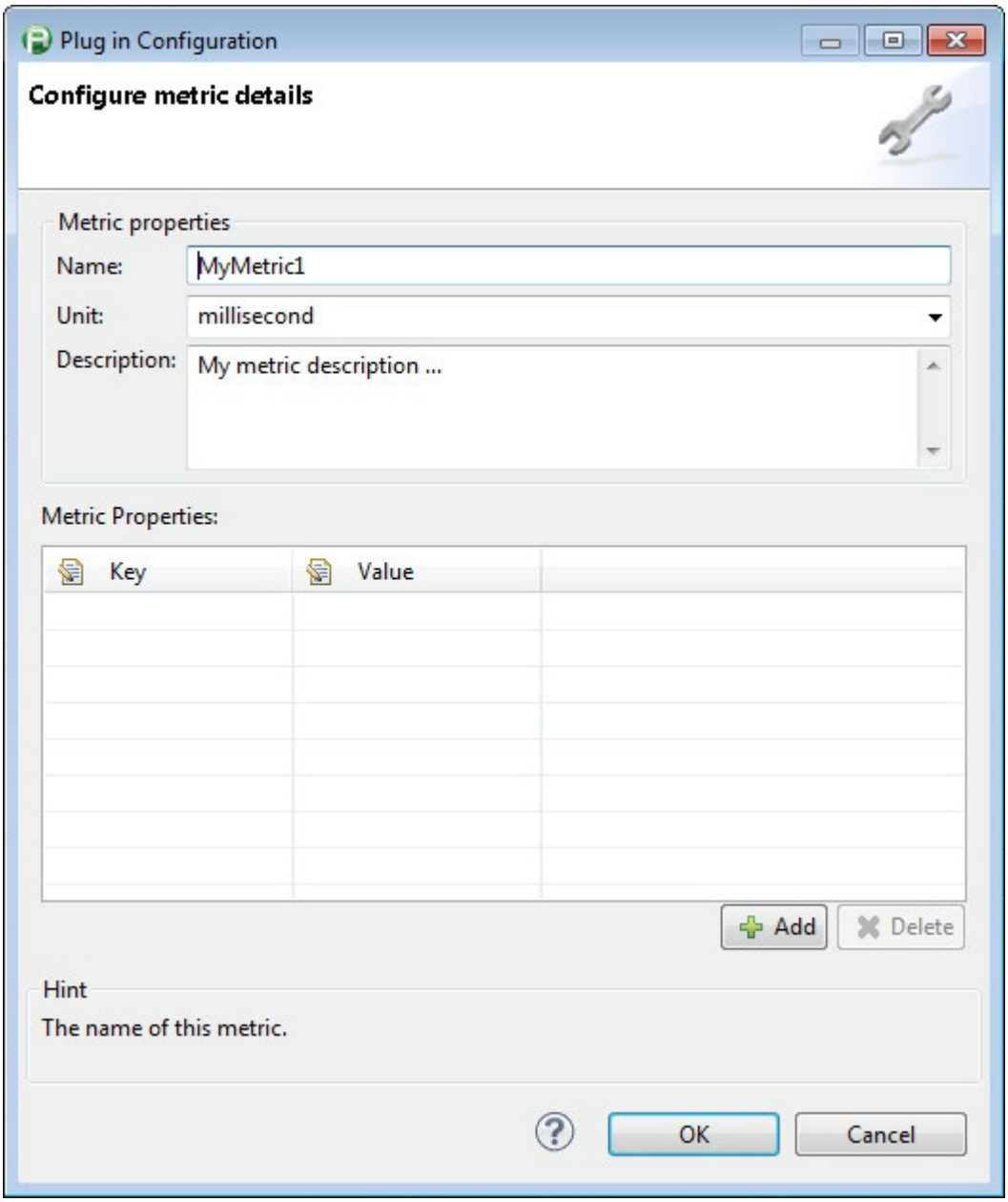
Configure metric details

When you click **Edit** in the **Configure Metric Groups and Metrics** screen, the **Configure metric details** dialog box opens. The contents vary depending on the metric type.

These fields are provided for all types of Metrics:

- **Name** – The metric name has to be unique within the Metric Group.
- **Unit** – The unit in which the measurements are provided. Select the unit from the list.
- **Description** – Optional text that describes this metric.

For Monitor Metric Groups, the screen includes a table for defining the **Metric Properties**, parameters that may be required by the Monitor.



Monitor Metric Details

For Agent Metric Groups, the the fields in the **Configure metric details** screen depend on the metric type. For JMX and PMI metrics, click **Browse Metrics** to select an Agent and then select a metric collected by the Agent, to populate the fields.

For JMX Metrics, provide these mandatory details:

- **Domain** – Mandatory MBean domain of the object to monitor, e.g. `jboss.*`.
- **Key Properties** – Properties to find a matching object. You must specify at least one property **Key**. If **Value** is left empty, all objects with the specified key will match. If you select the **Exclude** checkbox, all objects with the specified key will be ignored.
  Optionally, select **Include similar MBeans** to include MBeans that match the specified properties but also have other properties set.
- **Attribute Name** – Attribute of the object to monitor.

For PMI Metrics, provide these mandatory details:

- **Module Path** – The path to match the object to monitor, e.g. `beanModule`.
- **Performance Counter** – The measure name of this object, e.g. `ActivateCount`.
- **Statistic Value** – The aggregation method, e.g. `count`.
  See the IBM WebSphere PMI documentation for a list of available performance modules, counters, and statistic values.

For PerfCounter Metrics, enter these mandatory details:

- **Performance Object** – The object to measure, e.g. `Processor`.
- **Performance Counter** – The measure name of this object, e.g. `%% Processor Time`.
- **Instance** – The instance to use to query the Performance Counter, e.g. `_Total`.

Testing plugins

After you build the Plugin, you must test it successfully before it deploying it. To test a Plugin, you must specify the Plugin type or role (e.g. Action, Task, or Monitor) and the test location (Server, embedded, or out-of-process Collector). Optionally, specify the test setting configuration for the Plugin configuration parameters.

During the test, the Plugin is transferred to the test location and temporarily installed there. Afterwards, the methods for setup, execution, and tear down are called.

The criteria for passing the test are:

- The Plugin can be successfully installed.
- The setup, execution, and tear down do not hang.

- The setup and execute methods return `Success`.

Test location restrictions

Testing of Plugin types is restricted to the locations on which the Plugins can be deployed. This means that Monitors can be tested on a Collector (embedded or out of process), Actions can only be tested on the Server, and Tasks can be tested on all AppMon locations. If no appropriate testing location is available (e.g. no Collector is connected), the **Test** Button is disabled.

Status codes

The runtime status of a Plugin is determined by the returned status code. Plugins should be designed to use the best fitting status code available. All available status codes are in the `com.dynatrace.diagnostics.pdk.Status` class.

Each of the status categories applies to a different stage of Plugin execution:

- 100 Success – Return these status codes only if you are sure that the Plugin executed as expected.
- 200 Internal AppMon Server errors – These status codes are only returned if the AppMon Server has an internal problem. **Do not use these status codes in your Plugin.**
- 300 Plugin internal errors – The Plugin code has errors or cannot execute because environmental limitations. For example, the Plugin does not have required permissions).
- 400 Infrastructure errors – The Plugin code can execute properly, but the target service is not reachable because the infrastructure between the plugin and the target service has errors. For example, a network error occurred.
- 500 Target service errors – The Plugin code and the infrastructure are working correctly, but the target service did not complete as expected. For example, the target service is not installed on the target machine.

The following table describes all available status codes.

Status	Status Code	Base Status Code	Message	Description
Success	100	100	Success	Plugin execution was successful.
PartialSuccess	101	100	Partial Success	Plugin execution was partially successful.

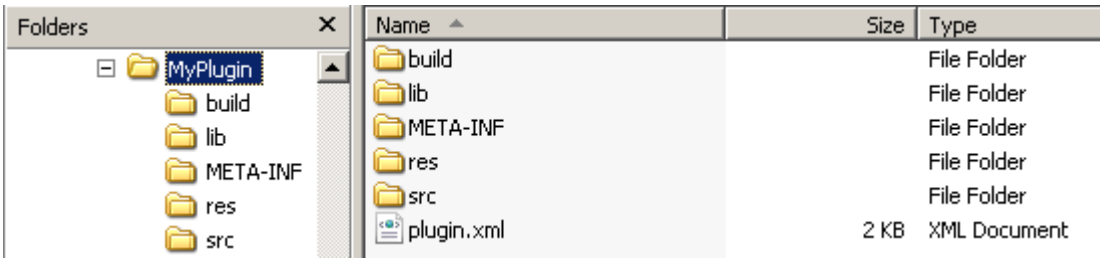
ErrorDynatraceServer	200	200	Server error	And internal AppMon Server error occurred.
ErrorLicense	201	200	License error	The license does not support Plugins.
ErrorInternal	300	300	Plugin error	A Plugin internal error occurred.
ErrorInternalException	301	300	Plugin exception	The Plugin caused an exception.
ErrorInternalResourceUnavailable	302	300	Plugin resource unavailable	A necessary Plugin resource is unavailable.
ErrorInternalUnauthorized	303	300	Plugin unauthorized	The Plugin does not have all required permissions.
ErrorInternalConfigurationProblem	304	300	Plugin configuration problem	The Plugin configuration is invalid.
ErrorInfrastructure	400	400	Infrastructure error	Plugin execution failed because of an infrastructure problem.
ErrorInfrastructureUnreachable	401	400	Target service unreachable	The Plugin is unable to reach its target service.
ErrorInfrastructureUnauthorized	402	400	Permission error	The Plugin does not have all required permissions to communicate with the target service.
ErrorInfrastructureProtocolError	403	400	Protocol error	A Plugin protocol error occurred.

ErrorInfrastructureTimeout	404	400	Timeout	A timeout occurred before the Plugin reached its target service.
ErrorTargetService	500	500	Target service error	A target service error occurred.
ErrorTargetServiceExecutionFailed	501	500	Target service execution failed	The Plugin was unable to execute the target service.
ErrorTargetServiceUnauthorized	502	500	Target service unauthorized	The target service does not have all required permissions.

Plugin files and directory structure

All Plugin directories contain the following subdirectories and file:

- **`build`** – Directory that contains the built Java classes and the Plugin JAR file.
- **`lib`** – Directory that contains optional libraries.
- **`META-INF`** – Directory that contains the `MANIFEST.MF` file.
- **`res`** – Optional directory that contains resource files that are packaged with the Plugin.
- **`src`** – Source directory, which contains Java files.
- **`plugin.xml`** – The main Plugin file, which contains general information, properties, Plugin types, and Metric Groups. These items are grouped into *Extensions*.



Plugin directory structure

If required directories or files are missing, the status on the development page will be **loading failed** (see Plugin development).

Java plugin class file

The Java Plugin class must implement the interface of the respective Plugin type (Task, Action, Monitor).

A Plugin can have multiple types. The Java Plugin class can implement multiple Plugin type interfaces. This feature makes sharing and reusing code easy.

The plugin.xml file

The contents of the `plugin.xml` file are grouped into Extensions:

- **`pluginconfig`** - This Extension contains Plugin information and properties.
- **`task`**, **`action`**, **and `monitor`** – These are the extensions for the Plugin types. They link to the `pluginconfig` Extension and provide the Java class name.
- **`monitormetricgroup`** – This Extension contains a metric group definition, including all metrics and links to a `monitor` Extension.

Be careful when manually editing this file. **Back up the file before opening it for editing.** Make sure each extension ID is unique and the references to classes and other extensions are valid.

Plugin license

If a software license applies to a Plugin, the license text may be packaged with the Plugin.

To display this license text in the Plugin Properties in the AppMon Client, add a `license` element to the `information` section of the `plugin.xml` file, as shown in the following example. Only plain text files are supported.

```
<information>
  <description value="The description of this plugin." />
  ...
  <license value="license.txt" />
</information>
```

API / PDK documentation download

The API / Plugin Development Kit documentation is available as a JavaDoc Bundle.

Plugin best practices and examples

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Working with a revision control system (RCS)

Working with an IDE (Eclipse)

Debugging a plugin

Examples

Dynamic monitor measures

Working with a revision control system (RCS)

Working with a revision control system doesn't involve any special actions.
If you have a Plugin in your RCS, do a checkout with your revision control client. Then add that Plugin to the development page.

Many revision control systems store versioning information in special hidden files or directories. These hidden files (including files starting with a period) will be excluded from the build.

Working with an IDE (Eclipse)

You can use the Eclipse integrated development environment to develop Plugins. It is available at http://www.eclipse.org/.

Eclipse setup

To enable compilation of AppMon Plugins in Eclipse, you need to set up the target platform correctly. Open Eclipse preferences, navigate to Plug-in Development > Target Platform and add a new platform entry which points to the `plugins` directory in your local dynaTrace Client installation, usually (for Windows) `c:\Program Files (x86)\dynatrace\dynaTrace (version)\client\plugins`. If you are using the webstart client and have no local client installation you will need to ask your Dynatrace administrator to provide you with the file `com.dynatrace.diagnostics.sdk.jar` which can be found in the `plugins` directory of the server or collector installation. Place the file in a directory **outside** of the plugin project and configure the target platform as described above. Do **not** place the file in the `plugin/lib` directory.

Editing an existing plugin from a plugin jar file

1. Extract the jar file into a directory of your choice.
2. Import the plugin into the PDE using the **Open** button.
3. Open and save the edit dialog at least once to create the Eclipse project files. (The reason for this workaround is a small known bug which will be fixed in a future version)
4. Import the project into eclipse as described below.

Editing a plugin that is managed in the AppMon Client PDE

When creating a project that contains source files, the PDE automatically creates project files for Eclipse (3.4 and higher) that enable you to import the project into Eclipse without further configuration.

1. Create and configure the Plugin to your needs.
2. Open Eclipse with any workspace, and select **File > Import**.
3. In the import dialog box, select **Existing Projects into Workspace** and navigate to the base folder of your plugin.
   The currently configured base folder for plugin development is shown in the **Client Preferences** of the AppMon Client. The default folder name for a Plugin is the Plugin name.
4. Select your project, then select **Finish**.

> **Warning**
>
> It is recommended that you manage the libraries used by the Plugin only in the PDE. The PDE will re-create the Eclipse project files to update the classpath every time the Plugin configuration is saved. This means any configuration that was done in Eclipse will be lost.

Debugging a plugin

A plugin that is executed on an AppMon Server or Collector has two options to transfer debug messages: as status messages (for Monitor and Task Plugins) or via a log file (for all plugins).

Status messages

Each lifecycle method has to return a Status object, which has a message property that can be set in the Plugin. This message is shown in the plugin details of the plugin schedule.

Log messages

Each plugin sends log messages either to `<dtinstallation>/log/server` or to Collector log directories, depending on the execution target of the Plugin. The log file name starts with the ID of the Task, Monitor, or Action..

Examples

For detailed procedures for creating a Plugin, see Developing a User Plugin.

Creating a performance monitor

1. Create a new Plugin.
2. Set the type to **Monitor**.
3. Configure the Monitor Metric Group: Set the name to **My Performance Monitor**, the first metric key to **MyTimeValue**, and the unit to percent
4. **Finish** the **New Plugin Wizard**.
5. Open the Plugin Java class file in your preferred text editor.
   If, for example, you used `com.mycompany.myplugin.MyPlugin` as the class name in the wizard, you can find the file in `PluginPath/src/com/myplugin/`.

This example is a simple Plugin for demonstration purposes only. Most `try/catch` blocks and other checks are omitted to make the code easier to read.

```
package com.mycompany.myplugin;
```

```
import com.dynatrace.diagnostics.pdk.*;
import java.util.*;


public class MyPlugin implements Monitor {
        // constants
        private static final String HOST = "host";
        private static final String METRIC_GROUP = "MyMetricGroup";
        private static final String METRIC_MY_TIME = "MyTimeValue";

        private static final Logger log = Logger.getLogger(MyPlugin.class.getName());

        private String host;

        @Override
        public Status setup(MonitorEnvironment env) throws Exception {
                // this will be one of the hosts which is configured for this monitor ins
                host = env.getHost().getAddress();

                // your implementation to create connection to host etc.

                return new Status(Status.Code.SUCCESS);
        }

        private Long getMeasurement() {
                // add your implementation here
                return 42;
        }

        @Override
        public Status execute(MonitorEnvironment env) throws Exception {

                // get the measurement
                Long measurement = getMeasurement();
                if (measurement == null) {
                    log.warning("useful information why no measure could be read");
                    return new Status(Status.StatusCode.ErrorInternal, "failed to get mor
                }
```

```
                // get the monitor measures for the processor time metric
                Collection<MonitorMeasure> measures = env.getMonitorMeasures(METRIC_GROUI
                if (measures == null) {
                        return new Status(Status.Code.INTERNAL_ERROR, "failed to get moni
                }

                // set the measurement
                for (MonitorMeasure measure : measures) {
                        measure.setValue(measurement * 0.001); // the value is scaled by
                }

                return new Status(Status.StatusCode.Success);
        }

        @Override
        public void teardown(MonitorEnvironment env) throws Exception {

            // disconnect from your host

        }
}
```

Creating an action plugin

1. Create a new Plugin.
2. Set the type to **Action**.
3. Configure the Plugin properties: Create one property with the key `property1` and type `string`.
4. **Finish** the **New Plugin Wizard**.
5. Open the Plugin Java class file in your preferred text editor. If, for example you used `com.mycompany.myplugin.MyAction` as the class name in the wizard, you can find the file in `PluginPath/src/com/myplugin/`.

```
package com.mycompany.myplugin;

import java.util.Collection;
import java.util.logging.Logger;
import com.dynatrace.diagnostics.pdk.*;

public class MyAction implements Action {
```

```
public class MyAction implements Action {
        private static final Logger log = Logger.getLogger(MyAction.class.getName());

        private static final String ENV_CONFIG_STRING_PROP1 = "property1";

        @Override
        public Status setup(ActionEnvironment env) throws Exception {
                try {
                        //setup code
                } catch (Exception e)  {
                        return new Status(Status.StatusCode.ErrorInfrastructure, "Setup e
                }
                return new Status(Status.StatusCode.Success);
        }

        @Override
        public void teardown(ActionEnvironment env) throws Exception {
                //shutdown code if necessary
        }

        @Override
        public Status execute(ActionEnvironment env) throws Exception {
                String prop1 = env.getConfigString(ENV_CONFIG_STRING_PROP1); //access to

                Collection<Incident> incidents = env.getIncidents();
                for (Incident incident : incidents) {
                        String message = incident.getMessage();
                        log.info("Incident " + message + " triggered.");
                        for (Violation violation : incident.getViolations()) {
                                log.info("Measure " + violation.getViolatedMeasure().getN
                        }
                }
        return new Status(Status.StatusCode.Success);
        }
}
```

Dynamic monitor measures

This snippet extends the preceding example by creating two dynamic submeasures for any measures subscribed for the metric `mymetric`.

```
... (other execute() code omitted)

Collection<MonitorMeasure> monitorMeasures = env.getMonitorMeasures("mymetricgroup", "mym
for (MonitorMeasure subscribedMonitorMeasure : monitorMeasures) {

    //this will book to the "classic" monitor measure
    //this is optional and only to show how it used to be.
    subscribedMonitorMeasure.setValue(42);

    //for this subscribed measure we want to create a dynamic measure
    MonitorMeasure dynamicMeasure = env.createDynamicMeasure(subscribedMonitorMeasure, "
    dynamicMeasure.setValue(24);

    //now we create another one for a different queue name
    dynamicMeasure = env.createDynamicMeasure(subscribedMonitorMeasure, "Queue Name", "Q
    dynamicMeasure.setValue(32);

}
...
```

Plugin frequently asked questions

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Plugin best practices and examples

The build failed because no JRE/JDK can be found. what should I do?

If the build fails because the Java compiler could not be found, copy the file `tools.jar` from a JDK installation to `<DT_HOME>/jre/lib/`. If the Client is running from Webstart, copy `tools.jar` to the `lib` directory of your local JRE installation.

Is thread safety a concern when executing the plugin?

Yes. The Plugin life cycle methods may be executed at any time and from multiple threads. Consider this possibility when writing custom Plugin code.

My plugin depends on another bundle. should I add it as a bundle dependency or as a library?

If the bundle is only used by one Plugin, it is best to add it as a private library, because a library is part of the Plugin bundle and that eases Plugin deployment.

If the bundle is used by multiple Plugins and is of considerable size, add it as a bundle dependency, because this enables sharing of the bundle between different Plugins. However, this may also break the isolation of the Plugins that use the bundle, especially if the bundle has any sort of state.

The secure option, though it consumes more resources, is to add the bundle as a library to each Plugin that uses it.

How can I deploy the plugin without executing the mandatory test run?

After building the Plugin, switch to the **Installed** tab and use **Import Jar** to import the built Plugin JAR file. The JAR file can be found in the `build` folder of the Plugin project.

Do this only as a workaround. Installing an untested plugin is **not** recommended.

Manual uninstallation of a system plugin

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Removal procedure

Overview

The AppMon Client does not have an uninstallation mechanism for plugins. You must manually remove a plugin when the AppMon system is shut down.

> **Important**
>
> Always back up the installation directory before deleting any files.

Removal procedure

See the table below for default locations of the plugin directories.

1. Shut down the affected AppMon Server, Collector and Client instances.
2. Find the plugin in the `deployment` folder of the Server.
3. Remove the whole plugin directory.
4. Find the plugin bundle files in the `dtplugins` folder of all affected instances**.
   ** For Webstart Clients, the `dtplugins` folder is located in the user directory, not in the installation directory.
5. Remove the bundle files.
6. Restart the affected instances.

7. Check that the plugin is no longer listed in the **Plugins** pane of the **Dynatrace Server Settings** dialog box.
   In the AppMon Client, select **Tools > Manage Plugins** to display the pane.

Plugin directories

Server `<DT_HOME>/server/<pluginPath>`	
`deployment`	Deployment directory for System Plugins
`dtplugins/plugins`	Installation directory for System Plugins
`conf/plugins`	Installation directory for User Plugins
`server/plugins`	Installation directory for Core Plugins
`server/backup`	Backup directory for Updates
Client	
`<User_Home>/.dynaTrace/dynaTrace <version>/client/dtplugins/plugins`	Installation directory for System Plugins
`<DT_HOME>/client/plugins`	Installation directory for Core Plugins
Collector `<DT_HOME>/collector/<pluginPath>`	
`dtplugins/plugins`	Installation directory for System Plugins
`plugins`	Installation directory for Core Plugins
`backup`	Backup directory for Updates

Remove plugins from extra instances

If you got extra instances of the default Collector instance (as installed; see Set up Collectors) you may need to remove the plugin there as well.

For extra Collector instances the path is `instances/<instance-name>` relative to the default instance path `<DT_HOME>/collector`.

If you have extra Server instances (deprecated! – see Set up the AppMon Server), look in `<DT_HOME>/server/instances/<instance-name>`.

Realtime streaming

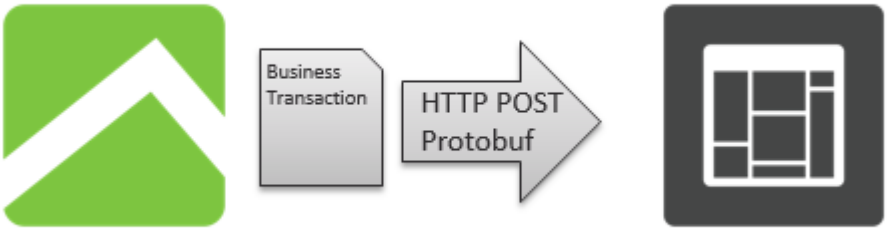
There are several other ways to integrate AppMon with other tools like reporting Dashboards via REST or sending data directly to your synthetic solution.

Are there other ways to export AppMon data?

If you would like to export PurePath data you should use the BT feed.

If you are looking to do business analytics based on UEM data, the PureLytics stream is the perfect solution for you.

Which one to choose?



A protobuf definition makes it easy to create your own consumer for the BT stream.

The business transaction feed allows to export business transactions (BTs) in real time. It can be enabled on a per-BT basis. BTs can be configured to be calculated for visits, user actions and for PurePaths. If multiple BTs get calculated for the same PurePath, the same PurePath will be exported multiple times.

What is the Real Time Business Transactions Feed?



When enabled, the PureLytics stream automatically sends JSON data for all visits and user actions to the configured HTTP endpoint. An export can also be triggered on demand for a specific time frame or session.

Use Elasticsearch to do ad-hoc analysis on large amounts of visit or user action data, and combine the data with analytics mined from other sources. You can then consume the combined data stored in the external data source using a visualization tool like Kibana for detailed charting and other data visualization, providing detailed UEM analysis.

The PureLytics stream sends real time UEM data from the AppMon Server to an external data source such as Elasticsearch so you can you leverage UEM data as part of big data analysis. Data that can be streamed includes user visits, user actions and client errors.

What is the PureLytics Stream?

- PureLytics Stream
- Real Time Business Transactions Feed

AppMon captures high quality data that can be used for many other purposes aside of APM. AppMon offers different ways to integrate with external systems. This page describes the realtime streaming capabilities available with AppMon:

PureLytics Stream

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Using the PureLytics Stream with ElasticSearch and Kibana

Technical details and limitations

Related pages

Realtime streaming

Real time Business Transactions feed

User experience management

PureLytics Stream (REST)

Overview

The PureLytics stream sends real time UEM data from the AppMon Server to an external data source like Elasticsearch so you can you leverage UEM data as part of big data analysis. Streamed data includes user visits, user actions and client errors.

Use Elasticsearch to do ad-hoc analysis on large amounts of visit or user action data, and combine the data with analytics mined from other sources. You can then consume the combined data stored in the external data source using a visualization tool like Kibana for detailed charting and other data visualization, providing detailed UEM analysis.

When enabled, the PureLytics stream automatically sends JSON data for all visits and user actions to the configured HTTP endpoint. An export can also be triggered on demand for a specific time frame or session.



About ElasticSearch

Elasticsearch is a highly scalable open-source search engine with a full-text search-engine library. It includes a distributed real-time document store where all fields are indexed and can be searched. Streaming live UEM data to Elasticsearch provides multi-dimensional Ad-hoc analysis on large data sets over long time frames.

The JSON document format

Visit, User Action, and Client Error data export as JSON documents. Data for any Business Transactions defined for Visits and User Actions also export as part of the documents.

JSON document examples

The following table shows examples of exported JSON documents.

Visit	User Action	Client Error
Browser Visit	Browser User Action	Javascript Error
Mobile Visit	Mobile User Action	Mobile Web Request
		Mobile Error Code
		Mobile Exception
		Mobile Crash

Javascript Warning & Info

Using the PureLytics Stream with ElasticSearch and Kibana

This video shows an example of how to visualize data from the PureLytics stream in Kibana.

The following sections describe the general process to visualize PureLytics Stream data using ElasticSearch and Kibana.

Satisfy requirements and prerequisites

Requirements and prerequisites

- Elasticsearch 1.5.2 or later. Elasticsearch 2.0.0 or higher is recommended because of the major stability improvements and improved resource use introduced in the 2.0.0 release.
- Elasticsearch requires a recent version of Java on the machine where Elasticsearch runs. You can install the latest Java version from www.java.com.
- Most later versions of Kibana work seamlessly with Elasticsearch. However, the latest version of Kibana is recommended because of the increased level of integration with Elasticsearch and more robust analytics and charting features.
- If running Elasticsearch on Windows, install cURL so you can quickly and conveniently submit requests to Elasticsearch. Download cURL from `http://curl.haxx.se/download.html` .
- Familiarity with Elasticsearch is needed for deploying Elasticsearch, creating nodes and clusters, indexing, and executing Elasticsearch queries.
- Familiarity with Kibana is needed for visualizing data.

Install streaming target software and visualization software

If not yet installed, install Elasticsearch and install Kibana.

Start ElasticSearch and set up a cluster

You must have Elasticsearch started and have at least one cluster set up in one node to interact with it and create JSON documents. Run an Elasticsearch cluster with at least 2 nodes for optimal results.

Additional information about nodes and clusters

- It is recommended to create and use multiple nodes rather than one node with more CPU cores.
- Node memory should be between 8 GB and 64 GB.
- Get the fastest SSD you can get to boost performance.
- Once started, communicate with Elasticsearch using the JSON based REST API residing at localhost port 9200. You can query Elasticsearch using cURL in Windows, or use graphical tools such as Fiddler or RESTClient on other platforms. You can also use Elasticsearch's Sense plug-in, which is a simple user interface specifically for using ElasticSearch's REST API that includes auto-completion.
- Use the Elasticsearch queue size to specify the (maximum) size of the request to Elasticsearch cluster. You can update the queue size using the cluster-setting API. A request to the endpoint point is sent either if the queue size reaches its maximum or after the configured dispatch interval.
- See the official Elasticsearch documentation on Hardware and Sizing Elasticsearch for more information.

See the guide on the Elasticsearch Web site for more information on starting and using Elasticsearch.

Apply a dynamic template

Apply a **dynamic template** to the cluster. Adapt the number of shards to the number of your data nodes.

The template contains the JSON data for the Elastic, and the command to apply it.

The following shows an example `cURL -X PUT` command used to send the template to the Elasticsearch cluster.

```
curl -X PUT 'http://localhost:9200/_template/dynatrace' -d '{"order":
0, "template": "dt_*", ...}'
```

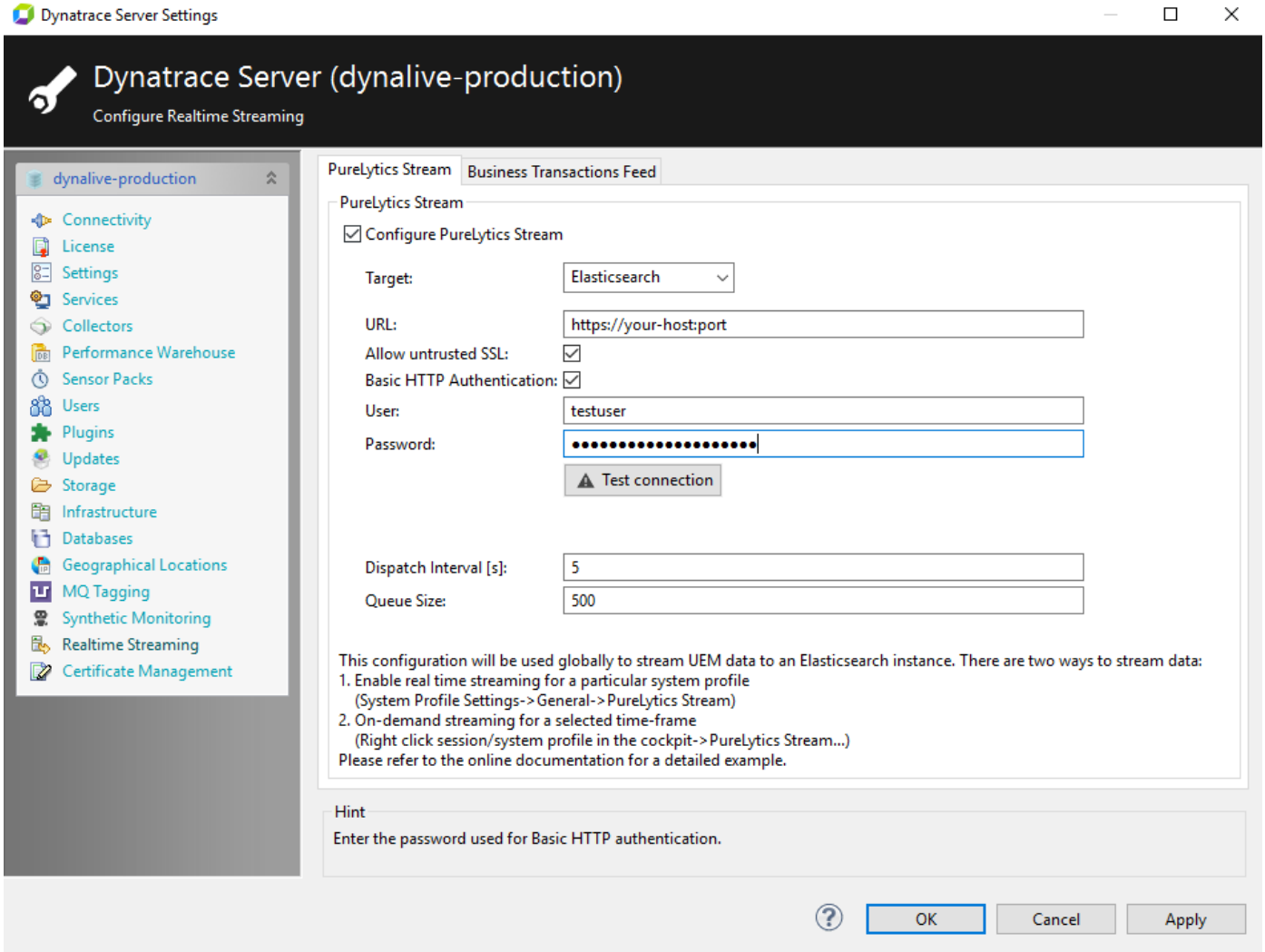
You can use `"dt_*"` as an index pattern get the streamed data. If you're sending the JSON data as a cURL string, escape the double quotes. Otherwise, it is not needed.

Configure the PureLytics Stream

To configure AppMon to stream PureLytics data, you enable the PureLytics stream, specify your elasticsearch endpoint and other AppMon Server Settings for realtime streaming, then enable the PureLytics stream for each System Profile for which you want to export data.

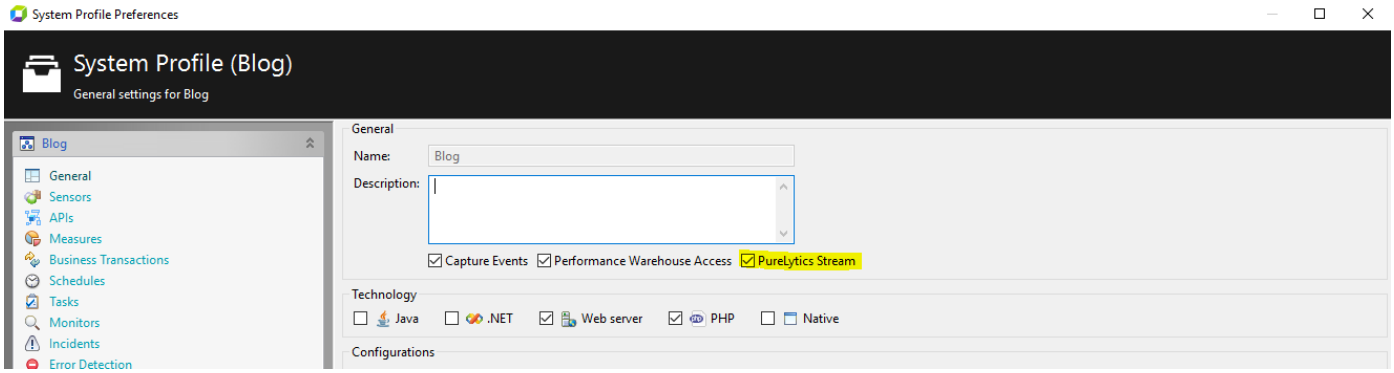
Configure the PureLytics Stream

1. Open Realtime Streaming in the AppMon Client Settings (**Settings > Dynatrace Server > Realtime Streaming > PureLytics Stream**).
2. Check **Configure PureLytics Stream**.
3. Choose the target you want to stream the JSON documents to. You can either stream them to an Elasticsearch Cluster or stream them to an generic HTTP endpoint (it will use the same Elasticsearch bulk format).
4. then enter the URL to your Elasticsearch cluster or generic HTTP endpoint (REST port) in the **URL** field. **ATTENTION:** the PureLytics Stream will append ***"/_bulk"*** to the entered URL).
5. Optionally select **Allow untrusted SLL** if you want to use HTTPS and use unofficial certificates.
6. Use **Dispatch Interval(s)** and **Queue Size** to set the amount of data sent with every bulk request and the interval between the requests. A request is sent when the internal document queue hits the size specified in **Queue Size** and/or after the amount of time specified in **Dispatch Interval(s)** has passed (*Dispatch Interval*). Request size is approximately the number of documents* times 2 kB. * This** depends heavily on factors such as configured Business Transactions and resource timings.



7. Click **OK**.

8. In the **Cockpits** tab, double-click the desired System Profile to open the Preferences dialog box for the profile.
9. In the General tab, **PureLytics Stream** and click **OK**.



Enable the PureLytics Stream for each System Profile you want to export

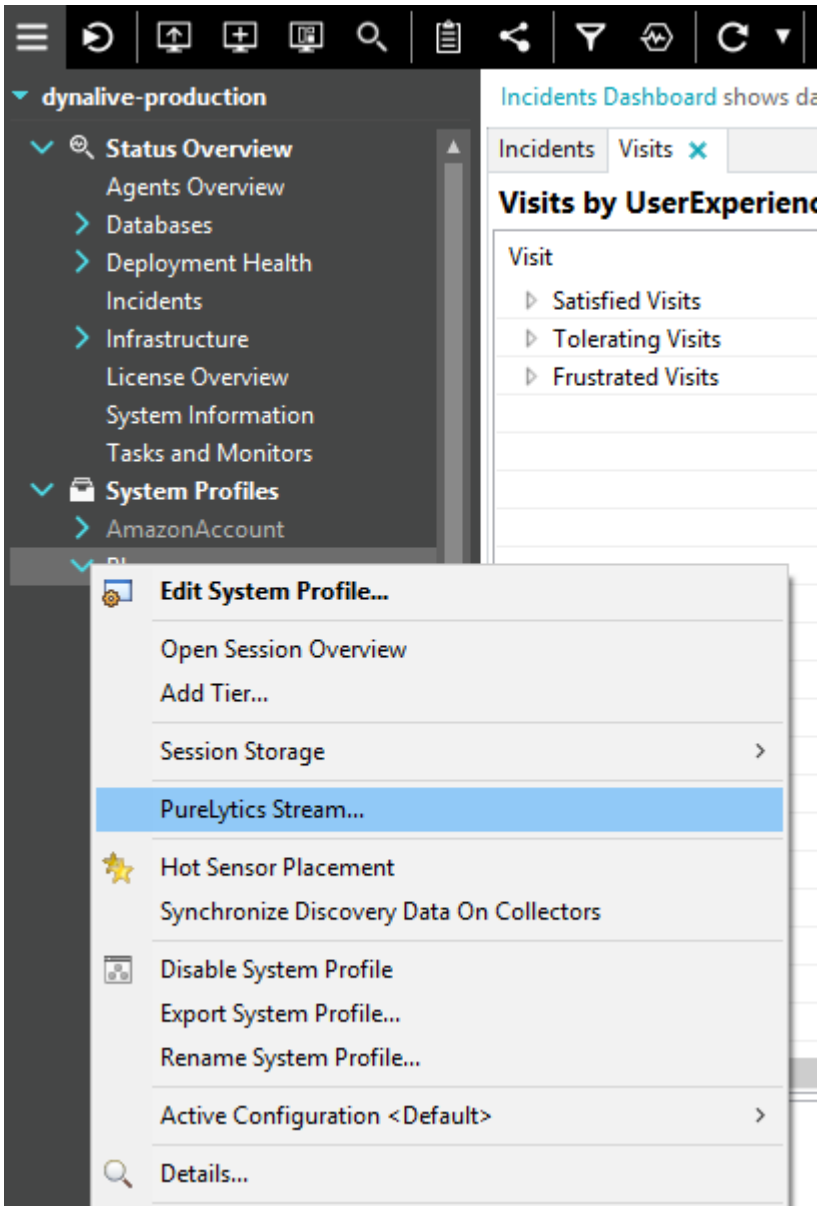
</figcaption></figure>

On-demand export

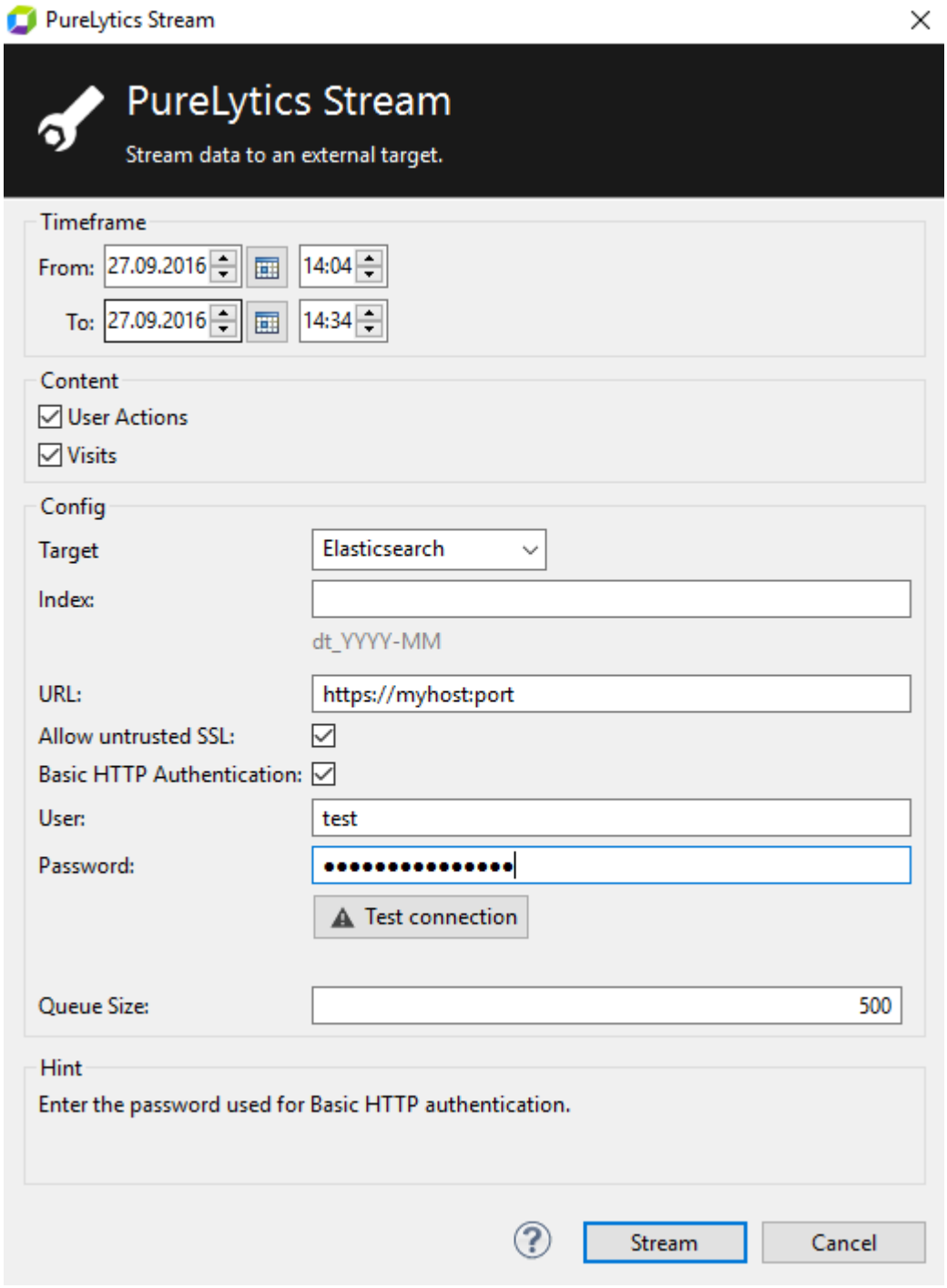
You can export data on-demand for a specific session in the AppMon Client, or use the PureLytics Stream REST API to export on-demand.

using DynaTrace Client

- Right-click on a stored or live session containing the data to export and choose **PureLytics Stream**.



- Specify the desired time range for captured PureLytics stream data and which document content you want to export (User Actions and/or Visits). Change the stream configuration as needed and click **Stream**.



using REST interface

Use the REST API to trigger and monitor an on-demand export. For more information see PureLytics Stream (REST).

- Create a job that exports data with a POST request. For example:
  `POST https://<server>:8021/rest/management/profiles/<profilename>/analytics`
- Monitor the job with a GET request. For example:
  `GET https://<server>:8021/rest/management/profiles/<profilename>/analytics`

Visualize data in Kibana

You must configure at least one index pattern to use Kibana. An index pattern identifies the index for which you want to search and visualize PureLytics data. Each index pattern you specify must match the name of one of your Elasticsearch indices.

Technical details and limitations

- Documents are exported after they are analyzed. For visits data, this means documents export after the visit completes. It can take some time until a visit is visible in Elasticsearch, depending on the visit length and the configured visit timeout.
- Documents of every configured system profile are stored in *indices per time frame*. An index is created for every month. and named accordingly. For example, visits, user actions and client errors which started in January 2016 (using the document field `data.startTime`) are stored in index named `dt_2016-01`.
- PureLytics streaming uses the Elasticsearch bulk API (https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-bulk.html). AppMon UEM appends `/_bulk` to the configured URL that was configured (only if target “Elasticsearch” is configured). A single request contains multiple documents of different types. The bulk API also requires additional meta data alongside the actual data containing various information, including the used index, document type, and the document ID.

```
{"index":{"_index":"dt_2016-03","_type":"visit","_id":"1860046861_14036"}}
{"serverID":1860046861,"serverName":"lnz124742d03","systemProfile":"easyTravel","data":{"
{"index":{"_index":"dt_2016-03","_type":"visit","_id":"1860046861_14037"}}
{"serverID":1860046861,"serverName":"lnz124742d03","systemProfile":"easyTravel","data":{"
{"index":{"_index":"dt_2016-03","_type":"useraction","_id":"1860046861_1385397273_1","par
{"serverID":1860046861,"serverName":"lnz124742d03","systemProfile":"easyTravel","data":{"
(...)
```

AppMon reference

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

AppMon platform overview

Client UI overview

Documentation downloads

Glossary

Keyboard shortcuts

REST interfaces

Reference material for AppMon Application Monitoring.

Client UI overview

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Menu bar and toolbar (1)

Default Monitoring Perspective & dashboard tabs (2)

Breadcrumb navigation buttons (3)

Cockpit sidebar (4)

Configuration menu (5)

Dashboard content (6)

Product and security notifications (7)

Client UI elements

Start Center

Monitoring Overview dashboard

Application Overview dashboard

Infrastructure monitoring dashboards

Dashboards

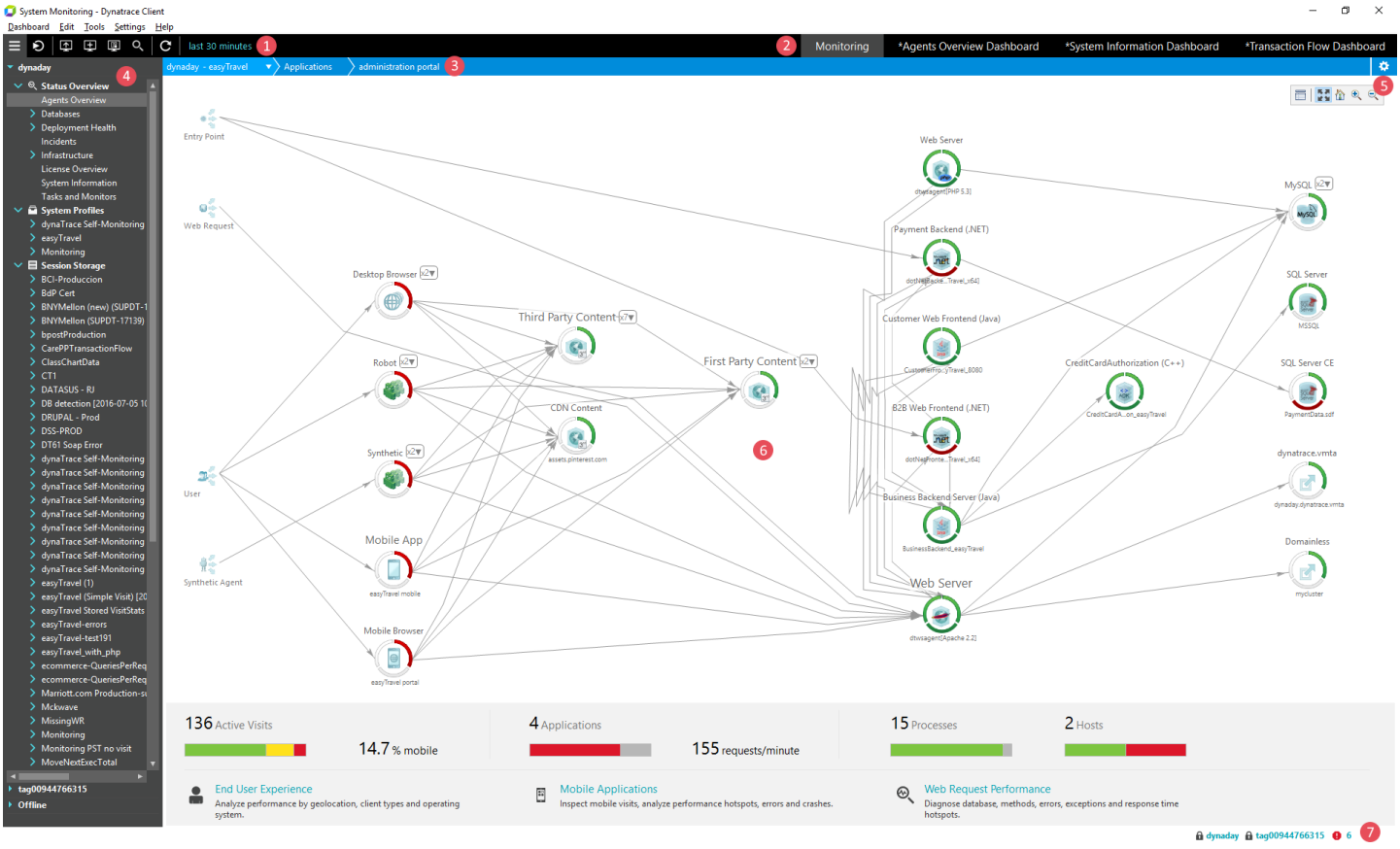
Take the AppMon Client wherever you go

The AppMon Client is a visually rich user interface that provides intuitive access to the many facets of your application's performance and system health data captured by AppMon. There are two types of Clients:

- **Standalone Client**: A full featured and locally installed client from which you can connect to multiple AppMon Servers. See Set up the AppMon Client for installation and configuration details.

- **Webstart Client**: A full featured client that uses Java Web Start technology to access the client from a supported web browser. The Webstart Client connects to a single AppMon Server and automatically updates with AppMon Server upgrades. Learn how to set up the Webstart Client.

In addition to viewing monitored data directly in the AppMon Client, you can also use dynamic web dashboards for quick data access on mobile devices and desktop browsers.



Menu bar and toolbar (1)

The Client menu bar and toolbar icons provide quick access to commonly performed tasks, such as opening a web dashboard, performing Client and AppMon Server configuration tasks, accessing the Start Center, and choosing a data timeframe.

Start Center icon



Opens the **Start Center**, from which you can access a collection of predefined dashboards for common use cases and administrative tasks. Learn more about the Start Center.

Timeframe selector

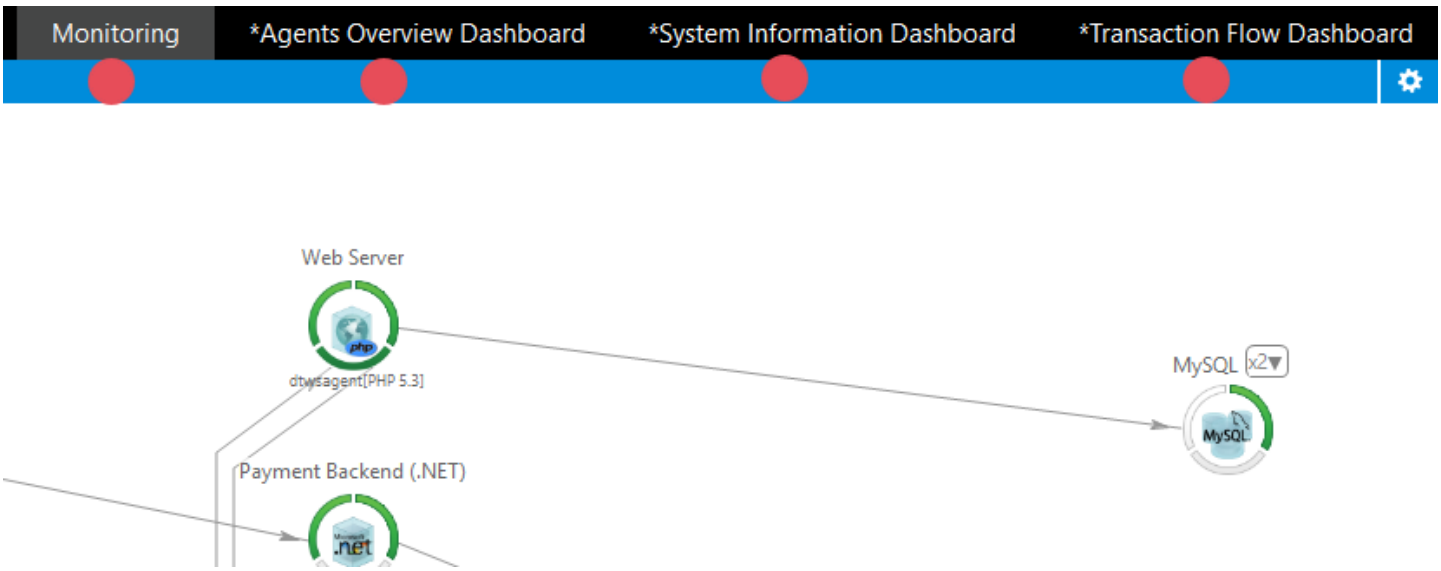
Use the **Timeframe Selector** on the toolbar to choose the timeframes affecting all displayed data and charts.

Default Monitoring Perspective & dashboard tabs (2)

The top of the Client window displays the blue default Monitoring Perspective button and any opened custom dashboard tabs. Click the default Monitoring Perspective button

Monitoring

to display the Monitoring Overview dashboard of a specific System Profile or stored session and related blue breadcrumb navigation buttons. In addition, click a custom dashboard tab to view the dashlets of the selected dashboard.



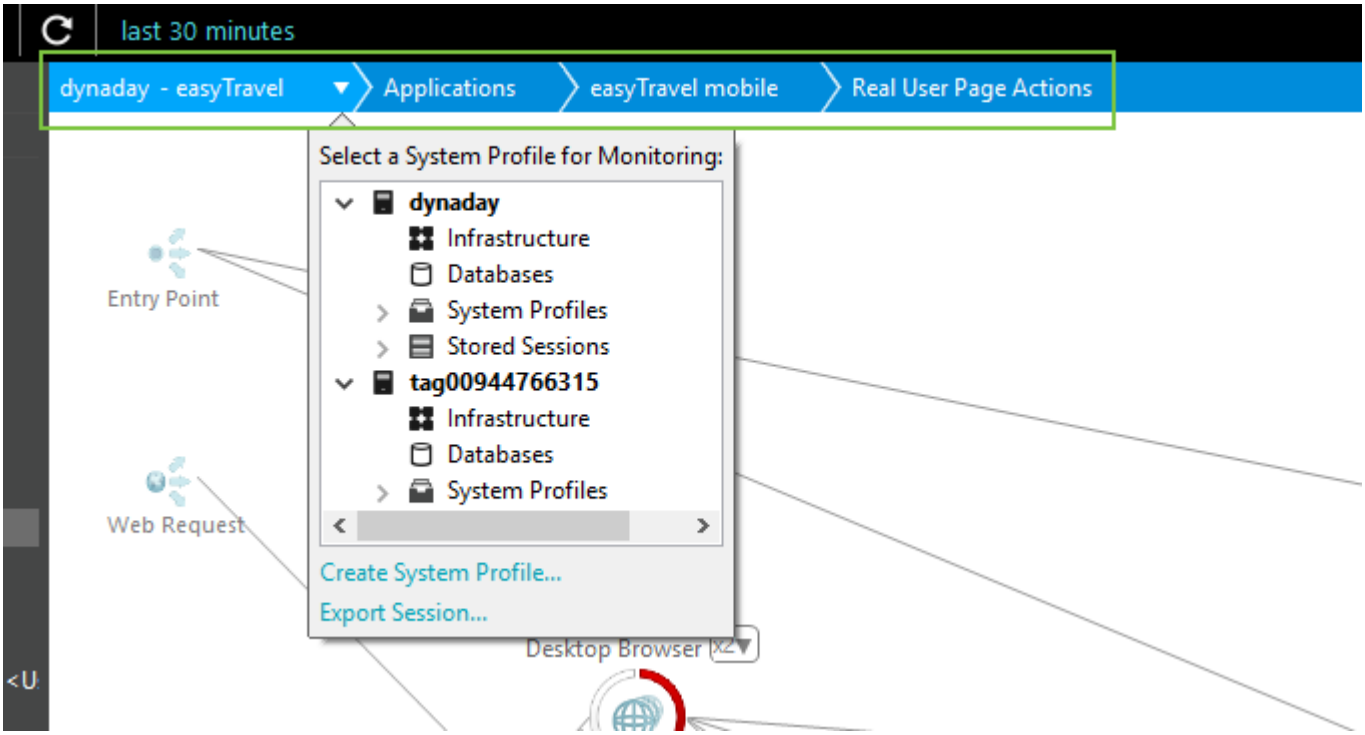
Breadcrumb navigation buttons (3)

Use the blue breadcrumbs that appear under the toolbar after clicking the default Monitoring Perspective button to quickly navigate to the built-in dashboards of selected System Profile or stored session. Breadcrumb items reflect the drill down order from the Monitoring Overview dashboard. The first breadcrumb entry corresponds to the selected System Profile or stored session.

Click the down arrow

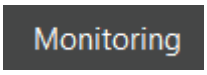


of the first breadcrumb button to select a different System Profile or a stored session. For example, in the screenshot below, the easyTravel System Profile is selected. Clicking the down arrow brings up a menu from which you can select a different System Profile or stored session.



Grey breadcrumb buttons indicate that you've previously drilled down from the Monitoring Overview to a related KPI analysis dashboard, but it's no longer actively displayed. Click a grey breadcrumb button to quickly navigate to a previously accessed dashboard. In the above example, Applications, easyTravel Mobile, and Real User Page Actions are previously accessed dashboards and the blue breadcrumb button labeled *easyTravel* is the currently displayed dashboard.

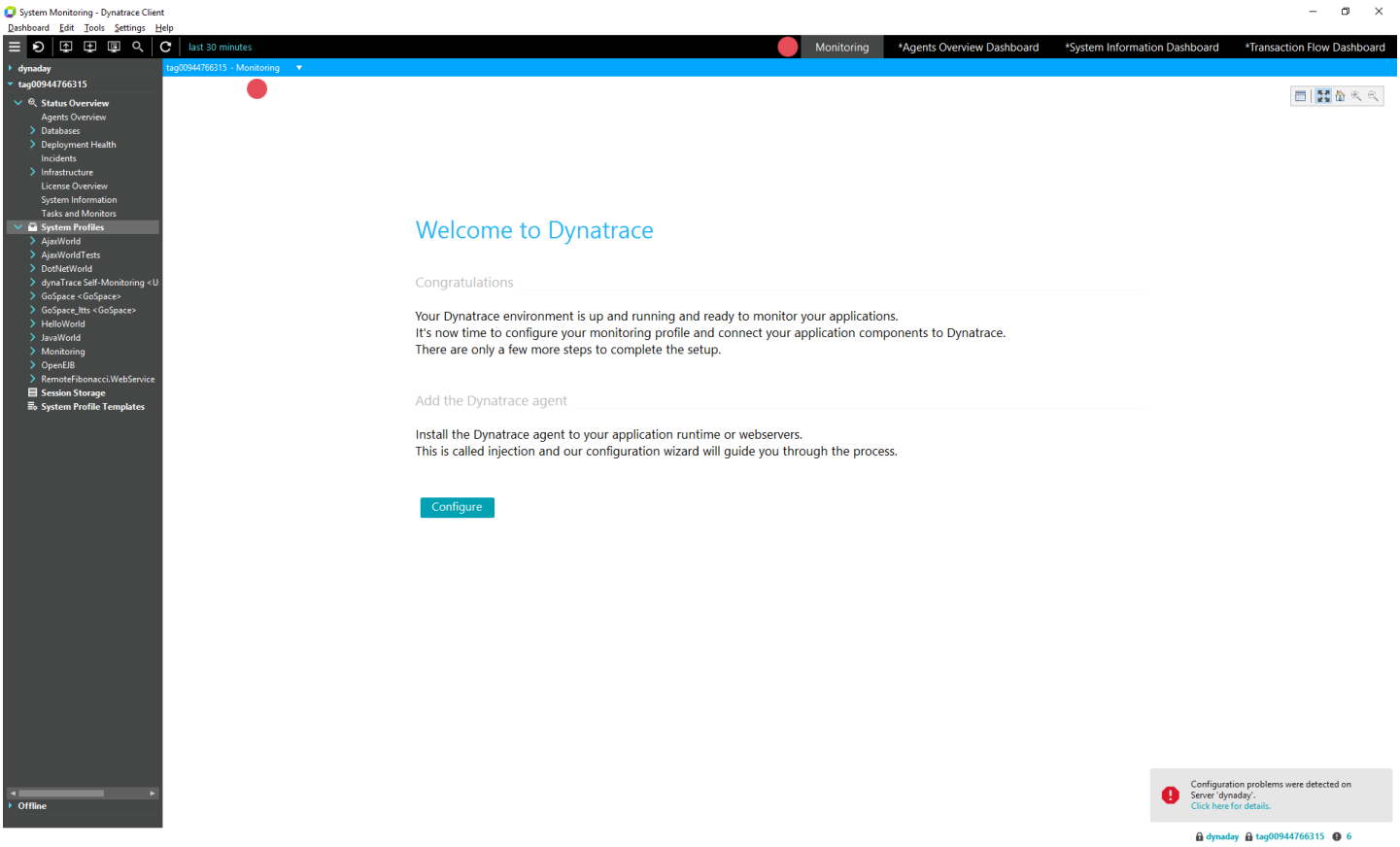
Note the difference between the default Monitoring



perspective button (with the eye) and the first breadcrumb navigation button labeled *Monitoring*



, which is the default System Profile name of a new AppMon installation.



Cockpit sidebar (4)

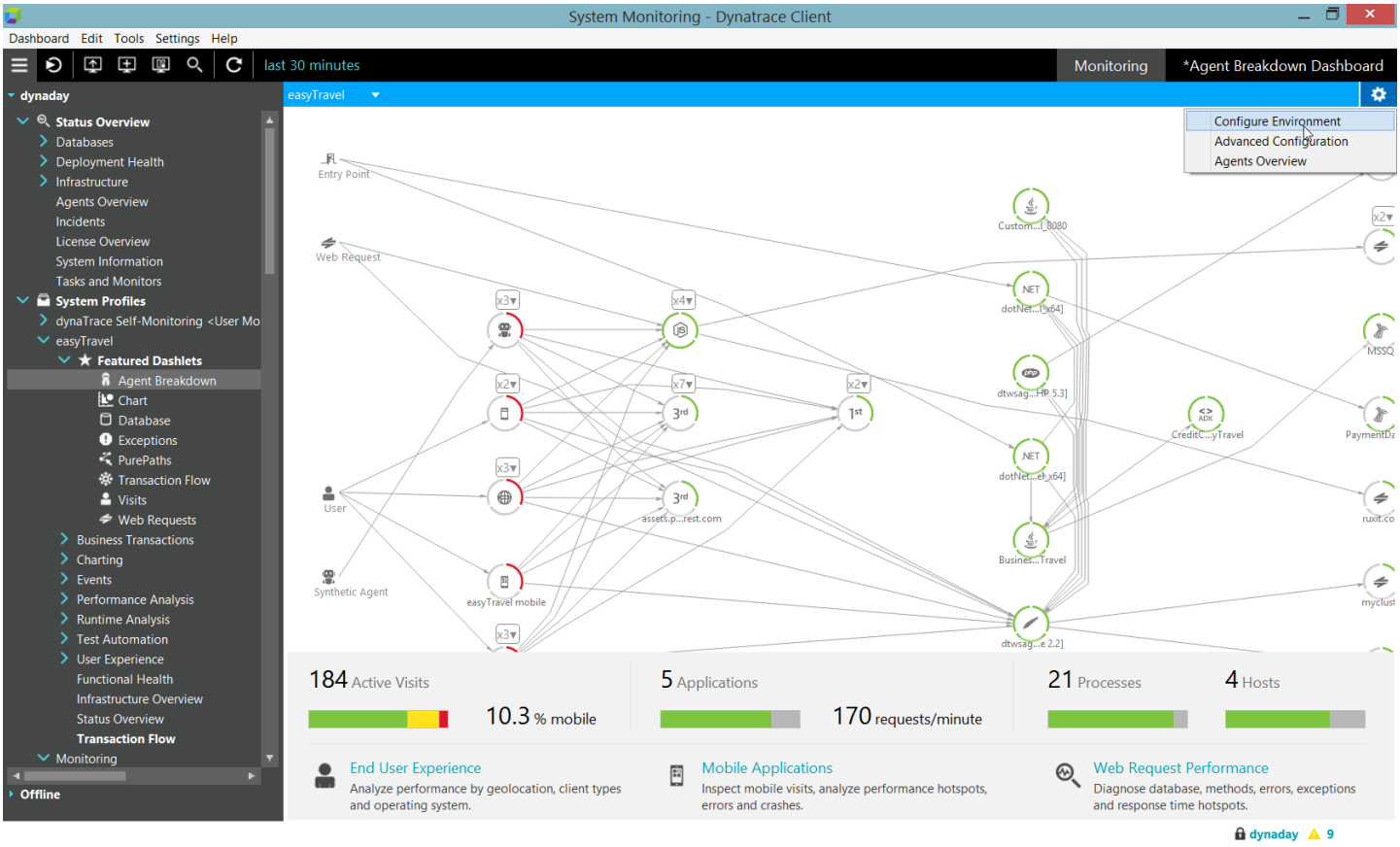
The **Cockpit sidebar** is a sliding and dockable control panel for managing System Profiles, session storage, quickly adding dashlets to dashboards, and for viewing the Status Overview dashlets of a selected System Profile. You can double-click assets appearing in the **Cockpits** tab and **Dashlets** tab to open them or right-click to view a context menu. Learn more about the Cockpit sidebar.

To slide the Cockpit Sidebar in and out, click the menu icon in the top left of the Client UI.

Configuration menu (5)

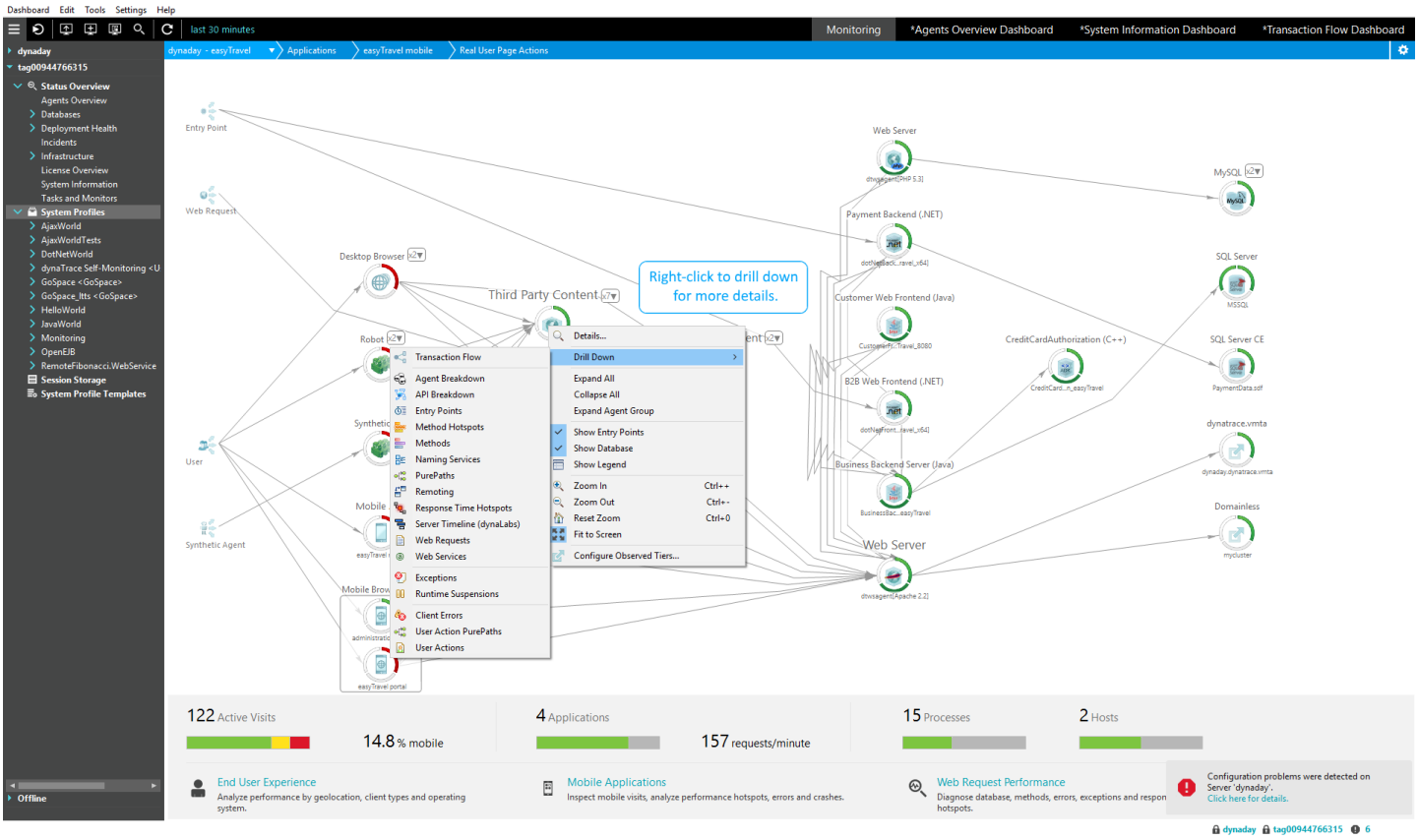
On each of the analysis dashboards of the default Monitoring Perspective, a gear icon

appears on the top right of the window. Click this icon to open a configuration menu that contains options that are specific to each view. For example, the configuration menu of the Application Details dashboard provides options for editing baselines and configuring business transactions, as shown below.



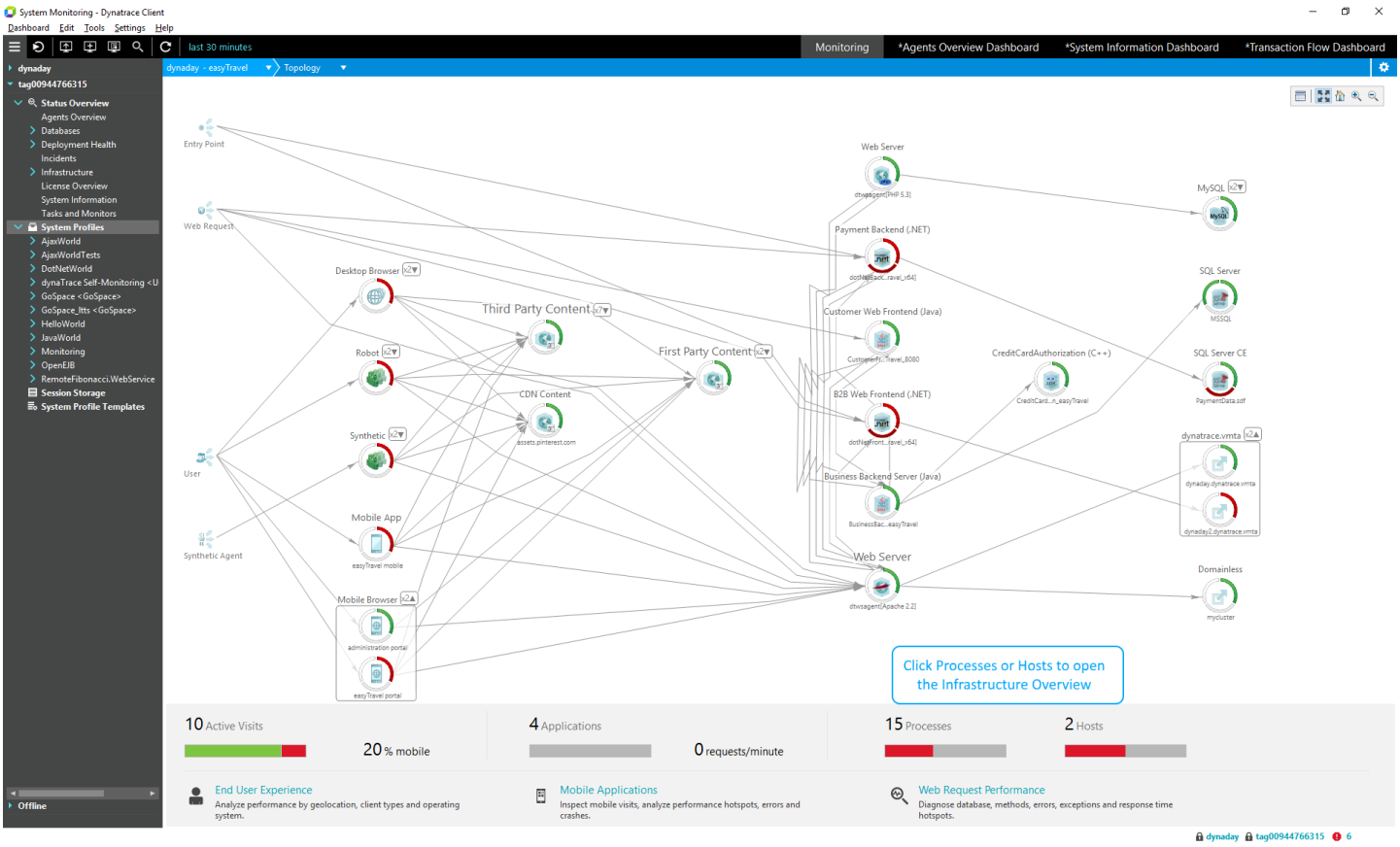
Dashboard content (6)

The main area of the Client displays the dashlets of a selected dashboard or one of the analysis dashboards of the default Monitoring Perspective. Right-click in a dashlet or in a Monitoring Perspective dashboard to open a context menu from which you can drill down to get more information.



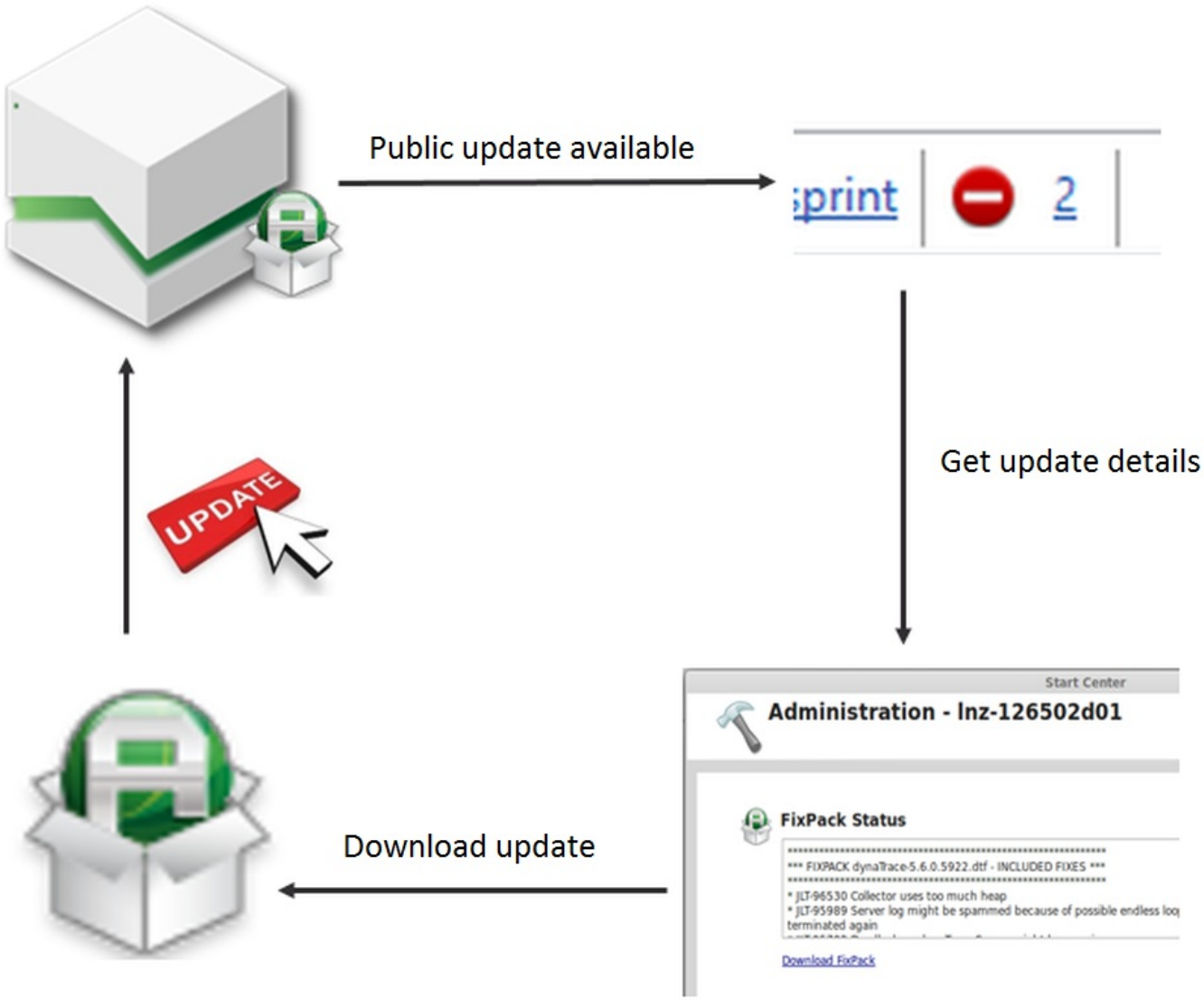
In the Monitoring Perspective dashboards, you can navigate to other dashboards. For example, from the Montoring Overview dashboard, which contains a speed-optimized version of the Transaction Flow Dashlet, you can click a link or section in the dashboard to quickly access the following KPI (Key Performance Indicator) analysis dashboards of the selected System Profile or stored session: Active Visits Dashlet, Applications Overview, Processes and Hosts / Infrastructure Overview, End User Experience / World Map, Mobile Applications and Web Request Performance.

For example, to access the Infrastructure Overview, click anywhere in the Process/Hosts section of the Monitoring Overview dashboard, as shown below.



Product and security notifications (7)

When new security updates are made available or product issues detected, an alert automatically appears on the bottom right of the Client along with a link to get more detailed information. For example, when a new update is released, an alert appears in the Client. Clicking the link takes you to the Administration pane of the Start Center from which you can download the new update.


Public update available
sprint
2
Get update details
Start Center
Administration - lnz-126502d01
FixPack Status
*** FIXPACK dynaTrace-5.6.0.5922.dtf - INCLUDED FIXES ***
* JLT-96530 Collector uses too much heap
* JLT-95989 Server log might be spammed because of possible endless loo
terminated again
Download FixPack
Download update
UPDATE

Start Center

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Administration

Getting started

Related pages

Application Environment Configuration

System Profile

Server administration

Performance Warehouse administration

Client administration

User permissions and authentication

User experience management administration

The **Start Center** lets you access predefined dashboards for common use cases. Use cases and predefined dashboards shown in the Start Center differ depending on your AppMon Edition (Test Center or Production). The Start Center opens by default when you start the AppMon Client. Clear the **Show on Startup** check box in the Start Center window to change this default.

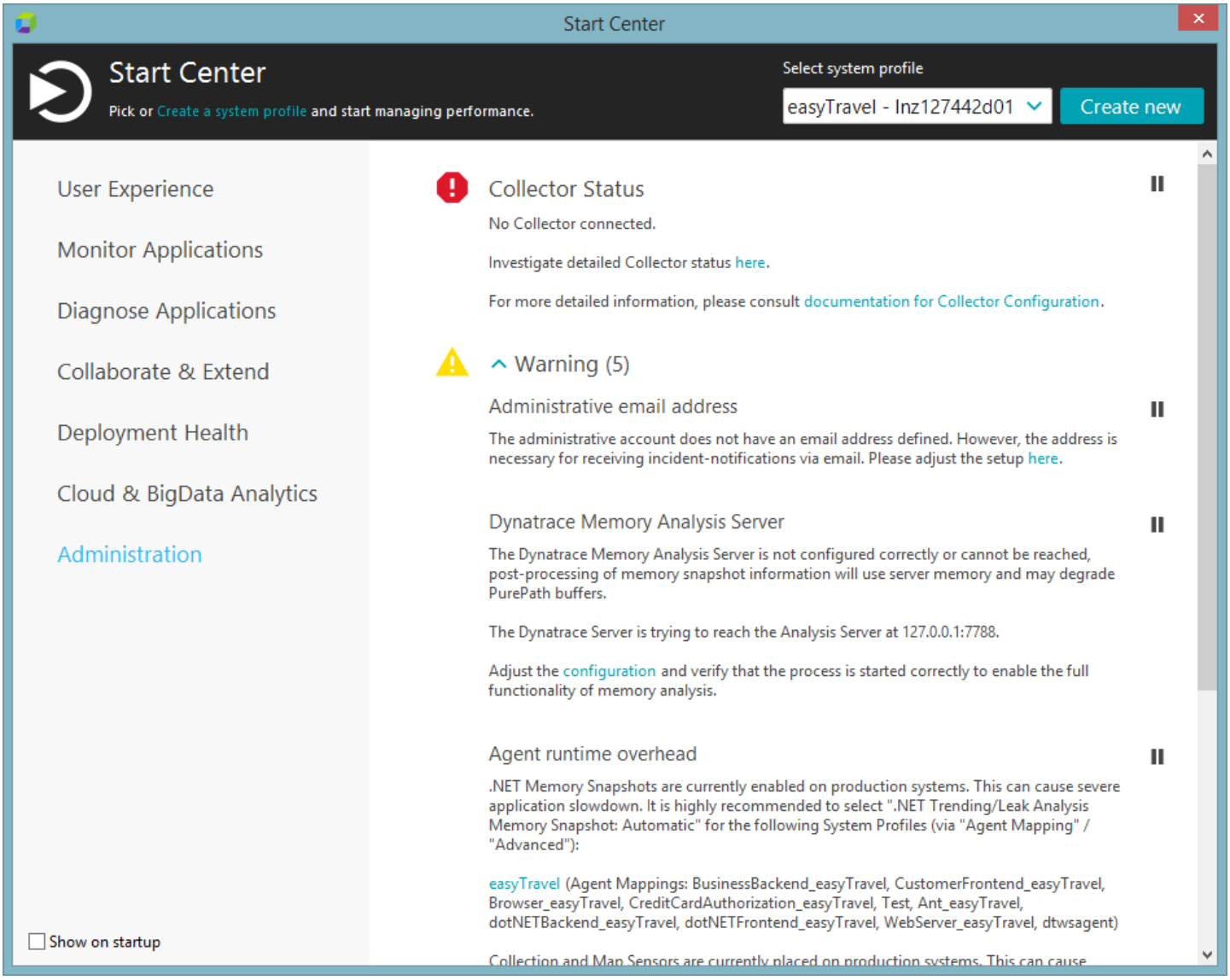
To open the start center manually, click the **Open Start Center**



icon.

Administration

The **Administration** tab gives an overview of common administrative tasks. When you configure AppMon after a fresh installation, you are prompted to perform mandatory tasks such as installing a license, setting up a data warehouse, and configuring your email accounts. Tasks that are successfully completed are indicated with a green check-mark. Tasks that need your immediate attention are flagged with an error icon. Tasks that are less urgent but need to be worked on are flagged as warnings. Make sure you work through all of these administrative tasks before you start to use AppMon. See the related pages links on the right of this page.



Getting started

After completing the Administration tasks, you can close the Start Center and adapt the default Monitoring System Profile as described in Application Environment Configuration or create a new one (**Tools menu > Create System Profile**) and go through the wizard.

Take the AppMon Client wherever you go

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Get the Portable Client

Related pages

Architecture

Client UI overview

Set up the Webstart Client

Web dashboards

Online license activation and deactivation

Watch and learn

Exploring the strategy

Exploring Dynatrace components

Conveniently run the AppMon Client on Windows from your memory stick or any other removable device. No installation is needed. Just run the Portable Client from a memory card, USB flash drive, or any removable media. Using the Portable Client not only provides total mobility, but you can also use it to work with different AppMon releases on one machine.

Get the Portable Client

Simply download the Portable Client. To do this, go to the Dynatrace Application Monitoring download page and choose Windows Client Portable 64-bit. The Portable Client downloads as a `.zip` file. The file name includes the AppMon release number.

Attach your removable drive to the machine on which you downloaded the Portable Client and move the file to your removable drive. Use any Windows zip utility to extract the `.zip` file.

Once extracted, double-click the `dtclient.exe` file in the root to start the Portable Client. The Portable Client is a fully functional client with all AppMon Client features.

Note

License installation and activation for the Portable Client is the same as with the AppMon Client. See information on Launching the AppMon Client for the first time and Activating an Installation for details.

REST interfaces

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Client REST interface

REST filters

Server REST interfaces

Watch and learn

Introducing the REST interface

Talking REST with Dynatrace

Both the AppMon Server and AppMon Client expose management functions via REST interfaces. Use these interfaces to perform tasks such as:

- Start/stop session recording.
- Switch active configuration for a System Profile.
- Trigger memory dumps, thread dumps, and sampling sessions.
- Restart AppMon Server and AppMon Collector.
- Perform Hot Sensor Placements.
- Generate reports.

Note

In Java 7, the support for the pseudo-protocol SSLv2Hello has been disabled by default. This is now also reflected in the REST Interfaces. Applications running with Java 6 or earlier might be affected if they access the REST Interfaces via HTTPS. To be able to access the REST Interfaces using Java 6, SSLv2Hello has to be disabled as shown below:

```
Set<String> protocols = new HashSet<String>(Arrays.asList(socket.getEnabledProt
protocols.remove("SSLv2Hello");
socket.setEnabledProtocols(protocols.toArray(new String[protocols.size()]));
```

Server REST interfaces

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

HTTP response codes

Authentication and authorization

Related pages

Legacy Server REST interfaces

Server REST examples

Web API tests

External tests

Overview

The AppMon Server and AppMon Client expose management and integration functionality via REST interfaces. Through these interfaces, you can start and stop session recording, trigger memory and thread dumps, create reports, and other tasks.

Access all AppMon Server REST interfaces via HTTP 1.1 and HTTPS 1.1 with the default ports of 8020 and 8021. See Set up Communication Connections for information on how to change port setting and how to enable or disable the REST interfaces.

Most AppMon REST interfaces sections require authentication. For AppMon 6.3 and later, all pages and services requiring authentication are by default only available if HTTPS is used. This default setting can be changed in Webserver Management, but is not recommended. Configure the HTTPS certificates in Certificate Management.

The REST interfaces documentation is based on the Swagger specification and accessible via the AppMon Server. By default, an interactive rendering of the documentation is accessible at `https://<YourDTServerHost>:8021/api-docs/current/`.

HTTP response codes

The HTTP status response code shows your request result. The *RFC 2616* standard (W3C, IETF) describes the status codes.

For example, if the request was successfully received, understood, and accepted, the REST interface returns a status code of class 2*xx*. If the request could not be understood by the AppMon Server, a status code of class 4*xx* is returned. A status code from class 5*xx* indicates that a server error prevented the AppMon Server from completing the request.

The following are common response codes:

- **200** – OK: The request has succeeded.
- **201** – Created: The request was fulfilled and resulted in a new resource being created.
- **202** – Accepted: The request was accepted for processing, but the processing has not been completed.
- **204** – No Content: The request has succeeded but does not have an entity-body in the response.
- **400** – Bad Request: Some request parameters are not correct.
- **401** – Unauthorized: A valid authorization header (Basic Authentication) is required but is not available.
- **403** – Forbidden: Execution of request is not allowed, e.g. the user has no privileges.
- **404** – Not Found: Some entities could not be found, e.g. System Profile, Dashboard, Collector, or Agent.
- **500** – Internal Server Error: See response body for details.
- **501** – Not Implemented: The requested service is currently not implemented.

Authentication and authorization

The User Agent must authenticate itself via HTTP Basic Authentication to access AppMon REST interfaces. RFC 2617 describes this authentication technique. The BASE 64 hash key must be calculated based on the concatenated string consisting of the user name, a colon (:), and the password. The string `Basic` plus the hash key must be set as the `Authorization` header to the HTTP request. See the Wikipedia page for more information.

Once successfully authenticated, the AppMon Server checks authorization to access the REST interfaces. If there is no `Web Service Interface Access` permission, the HTTP status code 401 (Unauthorized) is returned. Depending on the requested service, the additional permissions may be needed, for example to access a certain dashboard or System Profile.

Transferring your username and password is only secure, if you use HTTPS. Disabling the HTTP port within Webserver Management or selecting the option to Accept authentication data only with HTTPS is recommended.

Legacy Server REST interfaces

On this page

Overview

HTTP response codes

Authentication and authorization

Related pages

Agent Groups (REST)

Agents and collectors (REST)

Dashboards and reporting (REST)

Incident downtime (REST)

Incidents (REST)

Live sessions (REST)

Memory dumps (REST)

Performance Warehouse (REST)

Plugins and updates (REST)

PureLytics Stream (REST)

Resource dumps (REST)

Server management (REST)

Stored sessions (REST)

System Profile templates (REST)

System Profiles (REST)

Tasks and monitors (REST)

Test automation (REST)

Overview

The AppMon Server and AppMon Client expose management and integration functionality via REST interfaces. Through these interfaces, you can start and stop session recording, trigger memory and thread dumps, create reports, and so on.

You can access all AppMon Server REST interfaces via HTTP 1.1 and HTTPS 1.1 with the default ports of 8020 and 8021. See Set up Communication Connections for information on how to change port setting and how to enable or disable the REST interfaces.

Most sections of the AppMon REST interfaces requires authentication. For AppMon 6.3 and later, all pages and services requiring authentication are by default only available if HTTPS is used. This default setting can be changed within Webserver Management, but it's not recommended. The certificates used for HTTPS can be configured within Certificate Management.

Most interfaces are also exposed via HTML and can be accessed by the browser. By default, the interfaces for HTML are accessible at https://DTSERVER:8021/rest/html/management/server.

See Related Pages right-hand side-bar for a description of the REST services.

HTTP response codes

The HTTP status code of the response shows the result of your request. The *RFC 2616* standard (W3C, IETF) describes the status codes.

For example, if the request was successfully received, understood, and accepted, the REST interface returns a status code of class 2xx. If the request could not be understood by the AppMon Server, a status code of class 4xx is returned. A status code from class 5xx indicates that a server error prevented the AppMon Server from completing the request.

The following are common response codes:

- **200** – OK: The request has succeeded.
- **201** – Created: The request was fulfilled and resulted in a new resource being created.
- **202** – Accepted: The request was accepted for processing, but the processing has not been completed.
- **204** – No Content: The request has succeeded but does not have an entity-body in the response.
- **400** – Bad Request: Some request parameters are not correct.

- **401** – Unauthorized: A valid authorization header (Basic Authentication) is required but is not available.
- **403** – Forbidden: Execution of request is not allowed, e.g. the user has no privileges.
- **404** – Not Found: Some entities could not be found, e.g. System Profile, Dashboard, Collector, or Agent.
- **500** – Internal Server Error: See response body for details.
- **501** – Not Implemented: The requested service is currently not implemented.

Authentication and authorization

To access AppMon REST interfaces, the User Agent must authenticate itself via HTTP Basic Authentication. This authentication technique is described in RFC 2617. The BASE 64 hash key must be calculated based on the concatenated string consisting of the user name, a colon (:), and the password. The string `Basic` plus the hash key must be set as the `Authorization` header to the HTTP request. See the Wikipedia page for more information.

If the user is successfully authenticated, the AppMon Server checks that the user is authorized to access the REST interfaces. If the user does not have `Web Service Interface Access` permission, the HTTP status code 401 (Unauthorized) is returned. Depending on the requested service, the user may need additional permissions (for example to access a certain dashboard or System Profile).

Transferring your username and password is only secure if you use HTTPS. Disable the HTTP port within Webserver Management or keep the option *Accept authentication data only with HTTPS* enabled (default).

Plugins and updates (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Install plugin or update

Plugin or update installation state

Get update confirmation

Create update confirmation

List all update confirmations

Related pages

Install an update

Plugins

Downloads

Latest update

The AppMon Server offers several RESTful interfaces to install AppMon plugins and updates and to monitor the installation progress. The REST interface can deal with the following types:

- Updates (formerly Fixpacks. **Updates** is used in the REST calls)
- System plugins
- User plugins
- Resource packs
- Language packs

Required permissions

Update-related requests such as installing or confirming an update require the **Manage Updates** permission. Plugin installation requires the **Manage Plugin Bundles** permission.

Install plugin or update

Send AppMon Plugins and Updates to the AppMon Server and trigger the installation process. The response Location header contains the URL to the installation job.

POST `https://<server>:8021/rest/management/installjobs` Consumes `multipart/form-data`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Form Data	**file**	The AppMon Plugin archive file (`application/octet-stream`).	yes	
Query Param	**activateInstantly**	Only for updates. Use this parameter to define whether the update to install should be set to **active** after the installation. If set to **false**, the AppMon Server does not load the update after restart.	no	true

Example

Request URL

```
POST https://localhost:8021/rest/management/installjobs
```

HTML form

```
<html>
  <body>
    <h1>Upload Plugin File</h1>
    <form action="https://localhost:8021/rest/management/installjobs" method="
      <p>
        Select the Plugin file:
        <input type="file" name="file" size="45" />
      </p>
      <input type="submit" value="Install Plugin" />
    </form>
  </body>
</html>
```

Response code

```
201 Created
```

Response header

```
Location: https://localhost:8021/rest/management/installjobs/273764e2-a959-4da
```

Plugin or update installation state

Checks the progress and success of a running installation process for a plugin or update. The whole URL including the job ID is contained in the HTTP Location header of the Install Plugin response.

GET `https://<server>:8021/rest/management/installjobs/<jobid>` Produces `applic`

Parameters

Type	Parameter	Description	Mandatory	Default Value

Path	**jobid**	The ID of the installation job.	yes

Example

Request URL

```
GET https://localhost:8021/rest/management/installjobs/273764e2-a959-4da9-a12e-
```

Response content

```
<installjob id="65f566ba-2cb6-4127-936e-8a5feb63dea7">
  <isfinished>true</isfinished>
  <isstarted>true</isstarted>
  <issuccess>true</issuccess>
  <isserverrestartrequired>true</isserverrestartrequired>
  <result />
</installjob>
```

Response code

```
200 OK
```

Get update confirmation

Get an update confirmation. The response contains the name, description and confirmation of the update.

GET `https://<server>:8021/rest/management/fixpackconfirmations/<id>`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**id**	The name of the update confirmation.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/fixpackconfirmations/5.5.1.1234
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<fixpackconfirmation id="5.5.1.1234">
  <name>Update Name</name>
  <description>Update Description</description>
  <isconfirmed>true</isconfirmed>
</fixpackconfirmation>
```

Response code

```
200 OK
```

Create update confirmation

Create an update confirmation. This returns the created resource in the HTTP Location header.

PUT	`https://<server>:8021/rest/management/fixpackconfirmations/<id>`
POST	`https://<server>:8021/rest/management/fixpackconfirmations/<id>?httpMethod=PUT`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**id**	The ID of the update confirmation.	yes	

Example

Request URL

```
PUT https://localhost:8021/rest/management/fixpackconfirmations/5.5.1.1234
```

Request content

```
<fixpackconfirmation>
  <name>Update 5.5.1.1234</name>
  <description>Update Description</description>
</fixpackconfirmation>
```

Response code

```
201 Created
```

Response header

```
Location https://localhost:8021/rest/management/fixpackconfirmations/5.5.1.1234
```

List all update confirmations

Get a list of all update confirmations.

GET	`https://<server>:8021/rest/management/fixpackconfirmations`	Produce `appli`

Example

Request URL

```
GET https://localhost:8021/rest/management/fixpackconfirmations
```

Response content

```
<fixpackconfirmations>
 <fixpackconfirmation id="5.5.1.1234">
   <name>Name</name>
   <description>Description</description>
   <isconfirmed>true</isconfirmed>
 </fixpackconfirmation>
 <fixpackconfirmation id="5.5.1.1233">
   <name>Name</name>
   <description>Description</description>
   <isconfirmed>false</isconfirmed>
 </fixpackconfirmation>
</fixpackconfirmations>
```

Response code

```
200 OK
```

Agent Groups (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Agent Groups

Agent mappings

Agent Groups

The AppMon Server offers several RESTful interfaces for the URL `http://<server>:8020/rest/management/profiles/<profilename>/agentgroups/` to manage Agent Groups of a System Profile. Use HTTP GET to request the details of an Agent Group. Use HTTP PUT to create new Agent Groups or to modify existing Agent Groups. Use the HTTP method DELETE to delete an existing Agent Group. There is also an interface to list all Agent Groups that are configured for a certain System Profile.

List groups

This interface lists all configured Agent Groups of a certain System Profile.

GET `https://<server>:8021/rest/management/profiles/<profilename>/agen`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding system profile.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel/agentgroups
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentgroups>
  <agentgroupreference name="Business Backend Server (Java)" href="http://local
  <agentgroupreference name="Customer Web Frontend (Java)" href="http://localho
  <agentgroupreference name="CreditCardAuthorization (C++)" href="http://localh
  <agentgroupreference name="Payment Backend (.NET)" href="http://localhost:802
  <agentgroupreference name="B2B Web Frontend (.NET)" href="http://localhost:80
</agentgroups>
```

Response code

```
200 OK
```

GET group

Use HTTP GET on a specific Agent Group to get an XML representation describing the Agent Group.

GET `https://<server>:8021/rest/management/profiles/<profilename>/agen`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**groupname**	The name of the corresponding Agent Group.	yes	

Example

This example shows how an XML description of the Agent Group *Customer Web Frontend (Java)* can requested. The resulting XML contains the name of the Agent Group, its description, and references to related Agent Mappings.

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/Cust
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentgroup name="Customer Web Frontend (Java)">
  <description>Web application which provides the customer web pages via JSF/ic
  <agentmappingsreference href="http://localhost:8020/rest/management/profiles/
</agentgroup>
```

Response code

```
200 OK
```

PUT group

Use the HTTP PUT interface to create or overwrite an Agent Group by sending an XML representation as content of the request. The request's expected XML structure is similar corresponding HTTP GET interface response. Permissions are required to write to the System Profile configuration. An overloading POST interface provides the same output as HTTP PUT.

PUT `https://<server>:8021/rest/management/profiles/<profilename>/agen`

POST	`https://<server>:8021/rest/management/profiles/<profilename>/ager` `httpMethod=PUT`

This PUT interface does not evaluate `agentmappingsreference` elements. Use the HTTP PUT interface for Agent Mappings. Renaming an Agent Group is not supported by this interface.

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding system profile.	yes	
Path	**groupname**	The name of the Agent Group to create or update.	yes	

Update example

This example shows how to modify the description of the existing Agent Group **Customer Web Frontend (Java)**. If the request is performed successfully, the status code 204 (No Content) is returned.

Request URL

```
PUT https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/Cust
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentgroup>
  <description>This is a modified description.</description>
</agentgroup>
```

Response code

```
204 No Content
```

Creation example

This example shows how to create a new Agent Group named **New Agent Group**, including a description. Specify the Agent Group name using the path parameter, and specify the description within the XML content. If the request is performed successfully and a response code 201 is returned, then the Location header specifies the URL of the created Agent Group.

Request URL

```
PUT https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/New
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentgroup>
  <description>This is a new agent group.</description>
</agentgroup>
```

Response code

```
201 Created
```

Response header

```
Location: https://localhost:8021/rest/management/profiles/easyTravel/agentgroup
```

DELETE group

Use the HTTP DELETE interface to delete an existing Agent Group. Permissions are required to modify the System Profile. An overloading POST interface provides the same output as HTTP DELETE.

DELETE	`https://<server>:8021/rest/management/profiles/<profilename>/ag`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/ag` `httpMethod=DELETE`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding system profile.	yes	
Path	**groupname**	The name of the Agent Group to delete.	yes	

Example

This example shows how to delete an Agent Group called **Customer Web Frontend (Java)**.

Request URL

```
DELETE https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/(
```

Response code

```
204 No Content
```

Agent mappings

Similar to the interfaces for Agent Groups, the AppMon Server offers several interfaces to manage Agent Mappings. Use HTTP GET to request the details of an Agent Mapping. Use HTTP methods PUT and DELETE to create, modify, and delete Agent Mappings. Use the Lists interface to get a list of configured Agent Mappings for a certain Agent Group.

List mappings

This interface lists all configured Agent Mappings of a certain Agent Group.

GET `https://<server>8021/rest/management/profiles/<profilename>/agent`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**groupname**	The name of the corresponding Agent Group.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/Cust
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentmappings>
  <agentmappingreference alias="CustomerFrontend_easyTravel" href="http://local
  <agentmappingreference alias="easyTravel_CF" href="http://localhost:8020/rest
</agentmappings>
```

Response code

```
200 OK
```

GET mapping

Use HTTP GET on a specific Agent Mapping to get an XML representation describing the Agent Mapping.

GET `https://<server>:8021/rest/management/profiles/<profilename>/agen`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding system profile.	yes	
Path	**groupname**	The name of the corresponding Agent Group.	yes	
Path	**mappingalias**	The alias of the Agent Mapping to request.	yes	

Example

This example shows how to request an XML description of the Agent Mapping **CustomerFrontend_easyTravel**. The returned XML describes which agent names and host names this mapping matches.

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/Cust
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentmapping alias="CustomerFrontend_easyTravel">
  <agentnamepattern>CustomerFrontend_easyTravel</agentnamepattern>
  <agentnamematchtype>starts</agentnamematchtype>
  <hostnamepattern />
  <hostnamematchtype>starts</hostnamematchtype>
</agentmapping>
```

Response code

```
200 OK
```

PUT mapping

Use this HTTP PUT interface to create or overwrite an Agent Mapping. Permissions are required to write to the System Profile configuration. An overloading POST interface provides the same output as HTTP PUT.

PUT	`https://<server>:8021/rest/management/profiles/<profilename>/ager`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/ager?httpMethod=PUT`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**groupname**	The name of the corresponding Agent Group.	yes	
Path	**mappingalias**	The alias of the Agent Mapping to create or modify.	yes	

The request must send an XML representation of the Agent Mapping. Do a HTTP GET request for an Agent Mapping to see the required XML structure. The `agentnamematchtype` and `hostnamematchtype` elements can have one of the following values: `starts` , “ends”, “contains”, “equals” or “regex” - see Agent Group - Agent Mapping. Renaming an Agent Mapping is not supported by this interface.

Agent mapping XML Representation example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentmapping alias="CustomerFrontend_easyTravel">
  <agentnamepattern>CustomerFrontend_easyTravel</agentnamepattern>
  <agentnamematchtype>starts</agentnamematchtype>
  <hostnamepattern />
  <hostnamematchtype>starts</hostnamematchtype>
</agentmapping>
```

Update example

This example shows how to modify the Agent Mapping **CustomerFrontend_easyTravel**. If the request is performed successfully, the status code 204 (No Content) is returned.

Request URL

```
PUT https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/Cust
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentmapping>
  <agentnamepattern>sampleApp</agentnamepattern>
  <agentnamematchtype>contains</agentnamematchtype>
  <hostnamepattern>machineX</hostnamepattern>
  <hostnamematchtype>contains</hostnamematchtype>
</agentmapping>
```

Response code

```
204 No Content
```

Creation example

This example shows how to create a new Agent Mapping with the alias name **New Mapping**. If the request is performed successfully and a response code 201 is returned, the Location header specifies the URL of the created Agent Mapping.

Request URL

```
PUT https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/Cust
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agentmapping>
  <agentnamepattern>someprefix</agentnamepattern>
  <agentnamematchtype>starts</agentnamematchtype>
  <hostnamepattern>machineX</hostnamepattern>
  <hostnamematchtype>ends</hostnamematchtype>
</agentmapping>
```

Response code

```
201 Created
```

Response header

```
Location: https://localhost:8021/rest/management/profiles/easyTravel/agentgroup
```

DELETE mapping

Use this HTTP DELETE interface to delete an existing Agent Mapping. Permissions are required to modify the System Profile. An overloading POST interface produces the same output as HTTP DELETE.

DELETE	`https://<server>:8021/rest/management/profiles/<profilename>/ag`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/ag` `httpMethod=DELETE`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**groupname**	The name of the corresponding Agent Group.	yes	
Path	**mappingalias**	The alias of the Agent Mapping to delete.	yes	

Example

This example shows how to delete an Agent Mapping with the alias **CustomerFrontend_easyTravel**.

Request URL

```
DELETE https://localhost:8021/rest/management/profiles/easyTravel/agentgroups/(
```

Response code

```
204 No Content
```

Dashboards and reporting (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Generate a dashboard report

Dashboard information

Types and formats

Extended XML reporting

Related pages

Reporting

Client reporting

Scheduled reporting

XML reporting (REST)

The AppMon Server provides a set of RESTful interfaces to create dashboard reports via HTTP requests. Use this interface to retrieve the information contained in dashboards in various file formats and with flexible options to choose the source and timeframe of the reported data.

See Reporting for a general overview of the reporting functionality of AppMon.

Generate a dashboard report

Generate a report

The syntax to generate reports via REST is in general:

GET `https://<server>:8021/rest/management/reports/create/<dashboardna`

Parameters

You can add request parameters to refine the resulting report.

> **Note**
>
> Changes are only applied on the dashboard level. Specific settings of the individual dashlets such as the data source are not affected by the parameters.

Parameter	Description	Mandatory	Default Value
type	Specifies the type of output that is requested.	no	HTML
format	Specifies the reporting format to use for the selected report type.	no	Selected internally based on the report type.
orientation	Set the page orientation for PDF reports	no	portrait
filter	Sets the filter for both data sources, but is overridden by individual source filters. Can be specified multiple times.	no	
source	Overrides the dashboard's default data source.	no	
compare	Overrides (and activates if necessary) the comparison default data source.	no	

Parameters in detail

type

This specifies the output type. Supported formats can be found at Reporting or can be listed via REST.

Example

URI https://localhost:8021/rest/management/reports/create/demo?type=CSV

This creates a CSV report of the dashboard named **demo**.

format

Some types offer different formats. Available formats per type can be found at List Supported Formats per Type.

Example

URI https://localhost:8021/rest/management/reports/create/demo?type=HTML&format=HTML%20Paged

This creates a paged HTML report of the dashboard named **demo**.

orientation

PDF reports can have one of either portrait or landscape as page orientation.

Example

URI https://localhost:8021/rest/management/reports/create/demo?type=PDF&orientation=landscape

This creates a PDF report of the dashboard named **demo** with orientation landscape.

filter

Filters can customize the data used for the report. See REST Filters for a detailed explanation of filters.

Example

URI https://localhost:8021/rest/management/reports/create/demo?filter=tf:Last5Min&filter=ag:AgentGroups?Browser

This creates an HTML report of the dashboard **demo** with a timeframe of the last five minutes and only Agents of the Agent Group **Browser**.

source

The source can override the dashboard's source System Profile for the report.

Example

URI https://localhost:8021/rest/management/reports/create/demo?source=live:easyTravel

This create an HTML report of the dashboard **demo** with the live session **easyTravel** as the source.

> **Note**
>
> The value of the `source` parameter is the name of the stored session as it appears in the AppMon Client. You may need to encode some characters to construct a valid URL.

compare

URI https://localhost:8021/rest/management/dashboard/Incident%20Dashboard?compare=stored:easyTravel%20baseline

This compares the default source of the Incident dashboard with the stored session **easyTravel baseline**.

Examples

- Get the dashboard named **demo** reported as PDF with page orientation landscape: https://localhost:8021/rest/management/reports/create/demo?type=PDF&orientation=landscape
- Queries the Incident dashboard on a default configured local Server as tabbed HTML: https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?type=HTML&format=HTML+Paged

- Changes the default dashboard source to the live easyTravel session and generates a regular HTML: https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?source=live:easyTravel&type=HTML
- Applies a web request filter to the default dashboard source: https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?filter=wr:/frontend/userlogin.do
- Sets a baseline session as the compare source and output format to PDF: https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?compare=stored:easyTravel%20baseline&type=PDF
- Filters the UserLogin and UserLogout transaction and creates an XLSX file: https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?filter=bt:UserLogin&filter=bt:UserLogout&type=XLSX
- Requests data from the last 15 minutes as XML. See Extended XML Reporting for extended XML options. https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?filter=tf:Last15Min&type=XML
- Requests data from the previous day in CSV format: https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?filter=tf:Yesterday&type=CSV
- Requests data from the last 15 seconds: https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?filter=tf:OffsetTimeframe?15:SECONDS
- Requests data from 2015-03-14 09:26 GMT to 09:27 GMT: https://localhost:8021/rest/management/reports/create/Incident%20Dashboard?filter=tf:CustomTimeframe?1426339613000:1426339620000

Dashboard information

List available dashboards

To get a list of dashboards available, send the following RESTful request:

GET	`https://<server>:8021/rest/management/dashboards`	produces `text/xml`

The returned XML contains a `dashboards` root node and `dashboard` sub-nodes that list available dashboards. The `href` attribute contains a link for reporting this dashboard. In specific cases, a `dashboard` references an icon that can be requested with the link of the `icon` attribute.

> **Note**
>
> Only dashboards that are available for the logged-in user are listed. Dashboards to which the user does not have access permissions are not listed.

Dashboard icon

Some preconfigured dashboards reference an icon of the type PNG, GIF, or JPEG. Use the following request to download the icon of a specified dashboard:

GET	`https://<server>:8021/rest/management/dashboards/<dashboardname>/`

Types and formats

List supported reporting types

A reporting type is an output that is supported by the reporting functionality of the AppMon Server, such as PDF or HTML. Reporting types can support multiple reporting formats like the different layouts for HTML, although usually only one reporting format is provided per reporting type.

GET	`https://<server>:8021/rest/management/reports/types`	produces `text/xml`

The returning XML contains a `reporttypes` root node and `reporttype` sub-nodes that list available types.

Sample response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<reporttypes href="https://localhost:8021/rest/management/reports/types">
  <reporttype id="HTML"/>
  <reporttype id="PDF"/>
  <reporttype id="CSV"/>
  <reporttype id="XML"/>
  <reporttype id="XLS"/>
  <reporttype id="XLSX"/>
</reporttypes>
```

List supported formats per type

A reporting type is an output supported by the reporting functionality of the AppMon Server, such as PDF or HTML. A reporting type can support multiple reporting formats like the different layout for HTML, although usually only one reporting format is provided per reporting type.

GET	`https://<server>:8021/rest/management/reports/formats`	produces `text/xml`

The returning XML contains a `reporttypes` root node and `reporttype` sub-nodes that list available dashboards.

Example

To get the available formats for HTML reports:

URI	https://localhost:8021/rest/management/reports/formats/HTML

which returns:

Sample response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<reportformats href="https://localhost:8021/rest/management/reports/formats/HTML">
  <reportformat id="HTML Report">
    <description>Produces a report in HTML format</description>
  </reportformat>
  <reportformat id="HTML Paged">
    <description>Produces a report in HTML format with tabs to quickly switch between the
  </reportformat>
</reportformats>
```

Extended XML reporting

XML reports can be created like any other report. There is also a special reporting interface for XML only that supports some additional parameters.

XML reporting (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

Examples

Related pages

Reporting

Client reporting

Scheduled reporting

Dashboards and reporting (REST)

REST filters

Overview

In addition to regular REST Reporting the AppMon Server also supports another REST interface for XML reports. It offers more parameters than the regular reporting interface.

XML Reports can also be generated using Client Reporting.

Note

XML REST Reporting differs from regular REST Reporting. Some parameters that are listed here are not available for other report types and formats.

The XML request interface

The general syntax for the creation of XML reports is:

Request type	HTTP GET
URI	https://DTSERVER:8021/rest/management/dashboard/*DASHBOARDNAME*?*PARAMETERS*

See Set up Communication Connections for further configuration details.

> **Note**
>
> XML Reporting differs from regular REST Reporting, which uses the interface `/rest/management/reports/create/`.

Parameters

The following parameters configure the report and the dashboard itself. Changes are only applied on the dashboard level. Specific settings of the individual dashlets are not altered.

The parameters **source**, **compare**, and **filter** are explained in Dashboards and Reporting (REST).

Parameter	Description	Mandatory
source	Overrides the dashboard's default data source.	no
compare	Overrides the comparison default data source, and activates the data source if necessary.	no
filter	Sets the filter for both data sources, but is overridden by individual source filters. Can be specified multiple times. See REST Filters for details.	no

includeLayout	When set to *true* (default is *false*), additional layout information is added, such as table column order, column display names, or sorting. **Note** This is only available for XML.	no
purePathDetails	Enables reporting of full details including all node-data for one or for all PurePaths in the reported dashboard. This only affects reports of the PurePath dashlet. **Note** This is only available for XML.	no

includeLayout

includeLayout determines whether additional layout information is included in the XML report.

Example

URI https://localhost:8021/rest/management/dashboard/demo?includeLayout=true

This creates an XML report containing a layout element that describes the column order for dashlets with flat tables, as well as the sort column and sorting order. For example:

```
<layout>
  <sort field="exec" order="descending"/>
  <field order="0" name="state" display=""/>
  <field order="1" name="name" display="PurePath"/>
  <field order="5" name="agent" display="Agent"/>
  <field order="4" name="node_count" display="Size"/>
  <field order="8" name="exec" display="Duration [ms]"/>
  <field order="7" name="start" display="Start Time"/>
  <field order="9" name="agentid" display="Agent ID"/>
  <field order="10" name="tagid" display="Tag ID"/>
  <field order="6" name="application" display="Application"/>
  <field order="3" name="breakdown" display="Breakdown"/>
  <field order="2" name="response_time" display="Response Time [ms]"/>
</layout>
```

purePathDetails

The purePathDetails are the equivalent to the PurePath tree shown in the PurePath Dashlet. This parameter only has effect on reports containing the PurePath dashlet. It can either be reported for all PurePaths (purePathDetails=ALL) or for specific paths selected by their PurePath Identifiers.

Example

URI https://localhost:8021/rest/management/dashboard/demo?purePathDetails=ALL

This creates an XML report containing all PurePathDetails, assuming the ‘demo’ dashboard contains a PurePath dashlet.

> **Note**
>
> Requesting all PurePath details by specifying `ALL` may produce a large amount of data. Exporting all PurePaths may take a long time, depending on the number of PurePaths and nodes in the report. The CPU overhead on the AppMon Server may increase while exporting the data for all PurePaths.

PurePath identifiers for XML reports

To select a PurePath for which you want to see the details, identify it with its Agent ID and Tag ID. To get these values, open the PurePath dashlet and choose **Details** on the specific PurePath. The **Agent ID** and **Tag ID** are listed there. You can also show the columns in the PurePath dashlet.

Part	Description	Mandatory	Example
PA	The Agent tag in decimal or hex (starting with '0x') format.	yes	`PA=0x4ae37f41`
PT	The tag number in decimal format.	yes	`PT=0`

Example

The following requests details for one specific PurePath from the 'demo' dashboard:

URI	https://localhost:8021/rest/management/dashboard/demo?purePathDetails=PT=1175239;PA=0xe1bfd3b9

Resulting XML (excerpt)

```
<?xml version="1.0" encoding="UTF-8"?>
<dashboardreport name="demo" version="6.3.0.1222" reportdate="2015-11-30T12:24:12.673+01:
  <source name="easyTravel" filtersummary="last 30 days"/>
  <data>
    <purepathsdashlet name="PurePaths" description="" displaysource="Base">
      <purepaths>
        <purepath state="OK" name="/img/rssbutton.png" agent="CustomerFrontend_easyTravel
          <node method="doFilter(javax.servlet.ServletRequest request, javax.servlet.Serv
            <attachment type="StoredServletNodeAttachment">
              <property key="uri" value="/img/rssbutton.png"/>
              <property key="requestmethod" value="GET"/>
              <property key="servlet" value="Image Servlet"/>
              <property key="application" value="easyTravel Customer Frontend"/>
              <property key="clientip" value="205.126.251.253"/>
              <property key="responsestatus" value="200"/>
              <property key="requestsize" value="496"/>
              <property key="responsesize" value="1610"/>
              <property key="requestheadersize" value="496"/>
```

```
            <property key="responseheadersize" value="-1"/>
            <property key="requestbodysize" value="0"/>
            <property key="responsebodysize" value="1610"/>
            <property key="requestheader_referer" value="http://localhost:8079/orange-b
            <property key="requestheader_user-agent" value="Mozilla/5.0 (Windows; U; Wi
            <property key="sessionattribute_loginBean.getUserContext().getUserName()" v
            <property key="sessionattribute_searchBean.getDestinationName()" value="Zur
          </attachment>
          <attachment type="EuWebRequestNodeAttachment">
            <property key="pagecontextcookie" value="G_9049500559370034284h1"/>
            <property key="referrer" value="http://localhost:8079/orange-booking-finish
            <property key="sessioncookie" value="021D4EB9F88DA3F7829A69B47CA87BC3"/>
            <property key="id" value="8"/>
            <property key="requestid" value="-1102818147"/>
            <property key="responseid" value="-1291503704"/>
          </attachment>
          <node method="doGet(javax.servlet.http.HttpServletRequest request, javax.serv
            <attachment type="StoredServletNodeAttachment">
              <property key="uri" value="/img/rssbutton.png"/>
              <property key="requestmethod" value="GET"/>
              <property key="servlet" value="Image Servlet"/>
              <property key="application" value="easyTravel Customer Frontend"/>
              <property key="clientip" value="205.126.251.253"/>
              <property key="responsestatus" value="200"/>
              <property key="requestsize" value="496"/>
              <property key="responsesize" value="1610"/>
              <property key="requestheadersize" value="496"/>
              <property key="responseheadersize" value="-1"/>
              <property key="requestbodysize" value="0"/>
              <property key="responsebodysize" value="1610"/>
              <property key="requestheader_referer" value="http://localhost:8079/orange
              <property key="requestheader_user-agent" value="Mozilla/5.0 (Windows; U;
              <property key="sessionattribute_loginBean.getUserContext().getUserName()"
              <property key="sessionattribute_searchBean.getDestinationName()" value="Z
            </attachment>
          </node>
        </node>
      </purepath>
...
```

Examples

- Creates the default XML report for the demo dashboard: https://localhost:8021/rest/management/dashboard/demo
- Uses the session 'easyTravel' with the timeframe 'Last 7 days': https://localhost:8021/rest/management/dashboard/demo?source=live:easyTravel!tf:Last7d
- Uses the 'easyTravel Baseline' as compare source: https://localhost:8021/rest/management/dashboard/Incident%20Dashboard?compare=stored:easyTravel%20Baseline
- Includes layout information to the report https://localhost:8021/rest/management/dashboard/demo?includeLayout=true
- Includes the details for the PurePath with Agent ID 0xe1bdf3b9 and Tag ID 1306756: https://localhost:8021/rest/management/dashboard/demo?purePathDetails=PA=0xe1bfd3b9;PT=1306756

XML schema

The response is specified by the XML Schema definition which can be queried live from the AppMon Server at `https://DTSERVER:8021/rest/management/schema/dashboard`.

Incident downtime (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Overview

List all incident downtimes

GET incident downtimes

Add or update incident downtimes

Delete incident downtimes

Questions and answers

Related pages

Incident downtimes

Incidents and alerting

Overview

You can define Incident Rules to perform actions such as sending an email if a specified threshold is violated. In the case of planned actions such as system backups, database mirroring, or server restarts, you can suppress incidents. Learn how to configure this in Incident Downtimes.

Use the AppMon Server RESTful services at `https://<server>:8021/rest/management/profiles/<profilename>/incidentdow` to manage incident suppression.

Examine details with HTTP GET requests and use HTTP PUT / DELETE to add or update / delete a planned incident downtime.

List all incident downtimes

GET `https://<server>:8021/rest/management/profiles/<profilename>/inci`

Lists all existing incident downtimes for a specific profile. The response contains the URL and the name of each downtime.

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel/incidentdowntime
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incidentdowntime>
  <incidentdowntimereference href="https://localhost:8021/rest/management/profi
  <incidentdowntimereference href="https://localhost:8021/rest/management/profi
</incidentdowntime>
```

Response code

```
200 OK
```

GET incident downtimes

GET `https://<server>:8021/rest/management/profiles/<profilename>/inci`

Gets the XML representation of an existing incident downtime.

Parameters

Type	Parameter	Description	Mandatory
Path	**profilename**	The name of the corresponding System Profile.	yes
Path	**downtimename**	The name of the incident downtime which must be unique over all system profiles.	yes

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel/incidentdowntime
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incidentdowntime>
  <downtime>
    <start>2013-04-23T05:30:00+02:00</start>
    <duration unit="MINUTES">60</duration>
  </downtime>
</incidentdowntime>
```

Response code

```
200 OK
```

Add or update incident downtimes

Use the HTTP PUT interface to create or overwrite an incident downtime by sending an XML representation as request content. The expected XML structure of the request content is similar to the response of the corresponding HTTP GET interface. Permissions are required to write to the System Profile configuration. An overloading POST interface provides the same output as HTTP PUT.

PUT	`https://<server>:8021/rest/management/profiles/<profilename>/inci`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/inci` `httpMethod=PUT`

Parameters

Type	Parameter	Description	Mandatory
Path	**profilename**	The name of the corresponding System Profile.	yes
Path	**downtimename**	The name of the incident downtime which must be unique over all system profiles.	yes

Creation example

Request URL

```
PUT https://localhost:8021/rest/management/profiles/easyTravel/incidentdowntime
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incidentdowntime>
  <downtime>
    <start>2013-04-23T05:30:00+02:00</start>
    <duration unit="MINUTES">60</duration>
  </downtime>
</incidentdowntime>
```

Response code

```
201 Created
```

Response header

```
Location: https://localhost:8021/rest/management/profiles/easyTravel/incidentdo
```

Update example

Request URL

```
PUT https://localhost:8021/rest/management/profiles/easyTravel/incidentdowntime
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incidentdowntime>
  <downtime>
    <start>2013-04-23T05:30:00+02:00</start>
    <duration unit="MINUTES">120</duration>
  </downtime>
</incidentdowntime>
```

Response code

```
204 No Content
```

Delete incident downtimes

DELETE	`https://<server>:8021/rest/management/profiles/<profilename>/in`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/in?httpMethod=DELETE`

Deletes an existing incident downtime.

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**downtimename**	The name of the corresponding downtime.	yes	

Example

Request URL

```
DELETE http://localhost:8020/rest/management/profiles/easyTravel/incidentdownt
```

Response code

```
200 OK
```

Questions and answers

Does the name of an incident downtime needs to be unique?

The name of the incident downtime is used as identifier and needs to be unique over all system profiles.

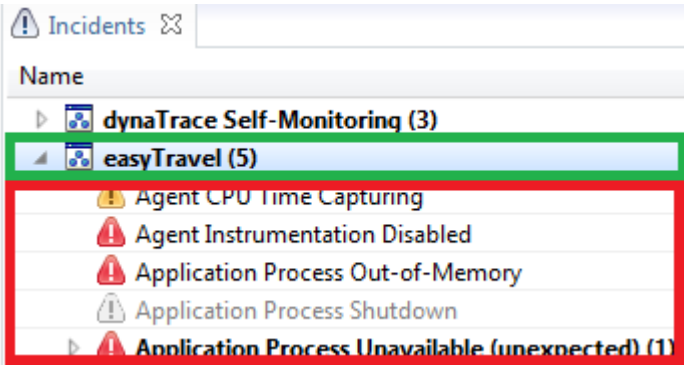
Why does the PUT or DELETE interface return HTTP 403 error message, *Downtime 'X'* must not be modified because it doesn't relate exclusively to system profile 'Y'?

Possible reasons are…

- The incident downtime you are trying to modify relates to a system profile "Y", but you have used a REST URI for another system profile.
- You are trying to create an incident downtime that already exists and belongs to a different system profile.
- The incident downtime you are trying to modify must belong to exactly one system profile. If the incident downtime relates to multiple profiles or doesn't belong to any system profile, you have to use AppMon Client to manage it.

Is it possible to manage incident downtimes on incident rule level?

Within AppMon Client it is possible to manage downtimes on system profile and incident rule level. With the REST interfaces you can manage downtime only on the system profile level (marked green below), but not on the incident rule level.



Incidents (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

XML data description

Get Incident

Get all incidents

Create incident

Update incident

Delete incident permanently

Related pages

Incidents Overview dashlet

User permissions and authentication

The AppMon Server offers several RESTful interfaces to manage incidents. See Incidents Overview Dashlet for more information about incidents.

You must have at least the permission **Analyze Incident** for each of these RESTful interfaces. To modify an incident, you must have the permission **Manage Incident**. See User Permissions and Authentication.

For security reasons, the calls for RESTful interfaces to create and update incidents are limited. By default, 60 calls per minute are allowed. To modify the limit, use the flag `com.dynatrace.diagnostics.rest.incidentCreationLimit`.

XML data description

Describes the XML representation of an incident and lists XML elements with multiple values.

Incident example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incident id="7825229f-039d-4761-95e1-01cb79d98a81">
  <message>Test Message</message>
  <description>Test Description</description>
  <severity>warning</severity>
  <state>Created</state>
  <start>2013-12-18T04:31:12.772+01:00</start>
  <end>2013-12-19T04:30:00.000+01:00</end>
</incident>
```

Element	Values	Description
severity	`informational | warning | severe`	The severity of the incident.
state	`Created | InProgress | Confirmed`	The state of the incident.
start, end	ISO 8601 compatible date/time of format “yyyy-MM-dd’T’HH:mm:ss.SSSXXX”	Start and end of the incident.

Get Incident

Get the XML representation of an incident.

GET `https://<server>:8021/rest/management/profiles/<profilename>/inci`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile or “Infrastructure”, if you only know that the incident is a host incident but don’t know which system profile in which it occurred.	yes	
Path	**incidentruleid**	The ID of the corresponding incident rule.	yes	

Path	**id**	The ID of the incident.	yes

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel/incidentrules/Cu
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incident id="7825229f-039d-4761-95e1-01cb79d98a81">
  <message>Test Message</message>
  <description>Test Description</description>
  <severity>warning</severity>
  <state>Created</state>
  <start>2013-12-18T04:31:12.772+01:00</start>
  <end>2013-12-19T04:30:00.000+01:00</end>
</incident>
```

Response code

```
200 OK
```

Get all incidents

Get a list of all incidents for the specified System Profile and incident rule.

GET	`https://<server>:8021/rest/management/profiles/<profilename>/inci`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile or “Infrastructure” if you want to retrieve all host incidents for a certain rule across all system profiles.	yes	
Path	**incidentruleid**	The ID of the corresponding incident rule.	yes	
Query	**from**	The start date of the incident.	no	If no `from` and `to` date is specified, a default time frame of three days is specified.
Query	**to**	The end date of the incident.	no	If no `from` and `to` date is specified, a default time frame of three days is specified.
Query	**state**	The state of the incident.	no	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel/incidentrules/Cu
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incidents>
  <incidentreference id="9cb9d5b2-59bb-4a08-912c-637ce5f6aee2"href="http://loca
  <incidentreference id="e95ee7d4-0999-4887-90e2-35a3179feaf8"href="http://loca
  <incidentreference id="7825229f-039d-4761-95e1-01cb79d98a8d"href="http://loca
  <incidentreference id="b295aa73-0fc2-44a1-b75a-b279602f5973"href="http://loca
</incidents>
```

Response code

```
200 OK
```

Create incident

Create an incident for a System Profile and incident rule. The request must contain the incident as XML representation. If the request does not contain a start and end date, the creation date is used. The default severity is informational and the default state is `created`. See XML Data Description) for more information.

Note

Incidents with a severity of informational are automatically set to state `Confirmed`. You can update such incidents to other states with a followup **Update Incident** REST call if necessary.

The response contains a reference to the created incident in the HTTP Location Header unless the operation was not successful. See Get Incident for more information.

POST `https://<server>:8021/rest/management/profiles/<profilename>/inci`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**incidentruleid**	The ID of the corresponding incident rule.	yes	

Example

Request URL

```
POST https://localhost:8021/rest/management/profiles/easyTravel/incidentrules/(
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incident>
  <message>Test Message</message>
  <description>Test Description</description>
  <severity>warning</severity>
  <start>2013-12-18T04:31:12.772+01:00</start>
  <end>2013-12-19T04:30:00.000+01:00</end>
</incident>
```

Response code

```
201 Created
Location Header: https://localhost:8021/rest/management/profiles/easyTravel/inc
```

Update incident

Several attributes of an incident can be modified by updating it. For instance, to confirm an incident, you can request the incident via the GET interface, change the state to `Confirmed`, and send the XML representation back to this interface. See XML Data Description for details about the XML representation.

PUT	`https://<server>:8021/rest/management/profiles/<profilename>/inc`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/inc?httpMethod=PUT`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**incidentruleid**	The ID of the corresponding incident rule.	yes	
Path	**id**	The ID of the incident to update.	yes	

Example

Request URL

```
PUT https://localhost:8021/rest/management/profiles/easyTravel/incidentrules/Cu
```

Request content (only the changed values need to be specified)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<incident id="7825229f-039d-4761-95e1-01cb79d98a81">
  <message>Test Message</message>
  <description>Test Description</description>
  <severity>warning</severity>
  <state>Confirmed</state>
  <start>2013-12-18T04:31:12.772+01:00</start>
  <end>2013-12-19T04:30:00.000+01:00</end>
</incident>
```

Response code

```
204 No Content
```

Delete incident permanently

DELETE	`https://<server>:8021/rest/management/profiles/<profilename>/in`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/in` `httpMethod=DELETE`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**incidentruleid**	The ID of the corresponding incident rule.	yes	
Path	**id**	The ID of the incident to delete.	yes	

Example

Request URL

```
DELETE https://localhost:8021/rest/management/profiles/easyTravel/incidentrules
```

Response code

```
204 No Content
```

Live sessions (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Store session

Start session recording

Stop session recording

Deprecated

Clear session

Related pages

Using sessions

As described in Usage of Sessions, there are two different approaches creating a stored PurePath session. If you use the **Store Session** request, all PurePaths of the current live session are dumped to a stored PurePath session.

You can also record all incoming PurePaths by issuing **Start Recording** and **Stop Recording** requests.

Store session

Use the **Store Session** request to store all time series and PurePaths in the Server's memory to a stored session. To limit the data to be stored, specify a start time and end time.

The timeframe parameters must be formatted according to ISO 8601, without a time zone specification, in the form `yyyy-MM-dd'T'HH:mm:ss` or `yyyy-MM-dd'T'HH:mm:ss.S`. See the examples below. The timeframe parameters always relate to Server time. If one of the timeframe parameters is not available or is not formatted as expected, the last 30 minutes will be stored.

GET `https://<server>:8021/rest/management/profiles/<profilename>/store`

Parameters

Parameter	Description	Mandatory	Default Value
appendTimestamp	`true` to append timestamp information to recorded session name, otherwise `false`.	no	`false`
recordingOption	`all` – All PurePaths, including time series. `violations` – Anly PurePaths marked as violated, including time series. `timeseries` – Time series data only.	no	`all`
isSessionLocked	`true` to lock session, otherwise `false`.	no	`false`
label	Descriptive text to mark the stored session (multiple label parameters allowed).	no	
timeframeStart	Start time of the interval during which the data will be stored.	no	last 30 minutes
timeframeEnd	End time of the interval during which the data will be stored.	no	last 30 minutes
storedSessionName	Name of the stored session. If the value is not set, the name will be the System Profile name.	no	`PROFILENAME`

Examples

Requests

The following example HTTP requests will store all PurePaths in the live session of the *easyTravel* System Profile on a local AppMon Server. The second and the third calls include a limit to only return data for the specified timeframe.

```
GET rest/management/profiles/easyTravel/storepurepaths?recordingOption=all HTTP/1.1
```

```
GET rest/management/profiles/easyTravel/storepurepaths?timeframeStart=2012-01-02T10%3A20%
```

```
GET rest/management/profiles/easyTravel/storepurepaths?timeframeStart=2012-01-02T10%3A52%
```

Result

The returned XML contains a `result` root node with a `value` attribute. The value contains the session name that was used to store the session.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="easyTravel"/>
```

Start session recording

Use this request to start session recording for a specific System Profile:

POST `https://<server>:8021/rest/management/profiles/<profilename>/sta`

Parameters

Parameter	Description	Mandatory	Default Value
presentableName	User-readable presentable name for the session to be stored.	no	
description	Description for the session to be stored.	no	
isTimeStampAllowed	`true` to append timestamp information to the recorded session name, otherwise `false`.	no	false
recordingOption	`all` – All PurePaths including time series. `violations` – Only PurePaths marked as violated, including time series. `timeseries` – Time series only.	no	`all`
isSessionLocked	`true` to lock the session that is recorded, otherwise `false`.	no	`false`

label		Descriptive text to mark the stored session (multiple label parameters allowed).	no

Example

Request

The following example HTTP request starts session recording for the System Profile *easyTravel* on a local AppMon Server.

```
POST rest/management/profiles/easyTravel/startrecording HTTP/1.1
Host: localhost:8020
Content-Length: 80
presentableName=easyTravelTestRun1&description=MyTestRun1&isTimeStampAllowed=true
```

Returned XML

The returned XML contains a `result` root node with a `value` attribute. The value contains the session name that was used to store the session.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="easyTravel [2009-07-30 16:57:48]"/>
```

Stop session recording

Use this request to stop session recording:

GET	`https://<server>:8021/rest/management/profiles/<profilename>/stop`

This call completes when until all recorded data is fully processed on the Server. Depending on the environment, it can take a few minutes until an HTTP response message is received.

Example

Request

The following example HTTP request stops any active session recording for the System Profile *easyTravel* on a local AppMon Server.

```
GET rest/management/profiles/easyTravel/stoprecording HTTP/1.1
```

Returned XML

The returned XML contains a `result` root node with a `value` attribute. The value contains the session name that was used to store the session.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="easyTravel [2009-07-30 16:57:48]"/>
```

Deprecated

This section contains deprecated live session REST interfaces, which should not be used.

Clear session

> **Note**
>
> **Deprecated**. To prevent breaking scripts it returns success without an action (as long as the default configuration for continuous recording is active)

GET `https://<server>:8021/rest/management/profiles/<profilename>/clea`

Example

The following example HTTP request clears the live session of the *easyTravel* System Profile on a local running AppMon Server.

```
GET https://localhost:8021/rest/management/profiles/easyTravel/clear HTTP/1.1
```

Memory dumps (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Memory dump

Post-processing

Deprecated

Related pages

Memory diagnostics

The AppMon Server offers several RESTful interfaces to manage memory dumps. See Memory Diagnostics for more information.

XML data description

This section describes XML elements.

Element	Values	Description
`state`	`finished | running | waitingForExecutor | failed | scheduled`	The state of the job.
`storedsessiontype`	`memdump simple | memdump_extended | memdump_selective`	The type of memory dump.

Memory dump

Create Memory Dump job

Put a job to create a memory dump. The Location header contains the reference to the created job. See Get Memory Dump Job, below.

PUT	`https://<server>:8021/rest/management/profiles/<profilename>/memo`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/memo?httpMethod=PUT`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	

The memory dump job to be created must be described by an XML document. This XML document has to be sent to the REST interface within the request content. The structure of the expected XML is described by the Get Memory Dump Job interface. Also, the following optional XML elements can be added to the XMLs root element:

Parameters

Element	Type	Default Value
iscaptureprimitives	boolean	`false`
iscapturestrings	boolean	`false`
isdogc	boolean	`false`

Example

This example demonstrates how to create a memory dump job for the System Profile *easyTravel*.

Request URL

```
PUT https://localhost:8021/rest/management/profiles/easyTravel/memorydumpjob/
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<memorydumpjob>
  <ispostprocessed>true</ispostprocessed>
  <issessionlocked>false</issessionlocked>
  <storedsessiontype>memdump_simple</storedsessiontype>
  <agentpattern>
    <agentname>BusinessBackend_easyTravel</agentname>
    <hostname>lnz123456</hostname>
    <processid>8560</processid>
  </agentpattern>
</memorydumpjob>
```

Response code

```
201 Created
```

Response header

```
Location: https://localhost:8021/rest/management/profiles/easyTravel/memorydumpjobs/Memor
```

Get memory dump job

Get the state for a memory dump job. If job has finished, it contains a reference to the created memory dump. See Get Memory Dump below for more information.

GET `https://<server>:8021/rest/management/profiles/<profilename>/memo`

Parameters

Type	Parameter	Description	Mandatory	Default Value

Path	**profilename**	The name of the corresponding System Profile.	yes
Path	**memorydumpjobid**	The ID of the job.	yes

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/JavaWorld/memorydumpjobs/Me
```

Response content

####### Running

```
<memorydumpjob id="Memory Dump [11524704613770]">
<state>running</state>
<ispostprocessed>false</ispostprocessed>
<issessionlocked>false</issessionlocked>
<iscapturepprimitives>false</iscapturepprimitives>
<iscapturestrings>false</iscapturestrings>
<isdogc>false</isdogc>
<progress unit="percent">57</progress>
</memorydumpjob>
```

####### Finished

```
<memorydumpjob id="Memory Dump [5552484470037]">
<storedsessiontype>memdump_simple</storedsessiontype>
<state>finished</state>
<ispostprocessed>false</ispostprocessed>
<issessionlocked>false</issessionlocked>
<iscaptureprimitives>false</iscaptureprimitives>
<iscapturestrings>false</iscapturestrings>
<isdogc>false</isdogc>
<progress unit="percent">100</progress>
<duration unit="MILLISECONDS">526</duration>
<agentpattern>
  <agentname>JavaWorld</agentname>
  <hostname>lnz124984d01</hostname>
  <processid>9508</processid>
</agentpattern>
<sessionreference href="https://localhost:8021/rest/management/profiles/JavaWo
</memorydumpjob>
```

Response code

```
200 OK
```

Get memory dump

Get information about the created memory dump for the specified System Profile.

GET `https://<server>:8021/rest/management/profiles/<profilename>/memo`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	

Path	**resourceid**	The ID of the dump resource.	yes

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps/2013(
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<memorydump resourceid="20130521074603_0" sessionid="JavaWorld/20130521074603_(
<name>Tue May 21 07:46:03 CEST 2013</name>
<state>finished</state>
<description></description>
<ispostprocessed>false</ispostprocessed>
<issessionlocked>false</issessionlocked>
<iscapturepprimitives>false</iscapturepprimitives>
<iscapturestrings>false</iscapturestrings>
<isdogc>false</isdogc>
<storedsessiontype>memdump_simple</storedsessiontype>
<agentpattern>
  <agentname>JavaWorld</agentname>
  <hostname>lnz123456</hostname>
  <processid>9508</processid>
</agentpattern>
<usedmemory>1788984</usedmemory>
<processmemory>35418112</processmemory>
<classes>909</classes>
<objects>14454</objects>
</memorydump>
```

Response code

```
200 OK
```

Get list of all memory dumps

Get a list of all memory dumps for the specified System Profile.

GET `https://<server>:8021/rest/management/profiles/<profilename>/memo`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps/
```

Response content

```
<sessions>
<sessionreference xsi:type="xmlStoredSessionReference" storedsessiontype="memdu

<sessionreference xsi:type="xmlStoredSessionReference" storedsessiontype="memdu
</sessions>
```

Response code

```
200 OK
```

Get memory dump hotspot

Get memory dump Hotspots. The memory dump must be post-processed and of type `memdump_extended` to get Hotspot data. The result contains the top 10 classes.

GET `https://<server>:8021/rest/management/profiles/<profilename>/memo`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**resourceid**	The ID of the dump resource for which to get the Hotspot.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps/2013(
```

Response content

```
<hotspots>
  <class gcsize="366192" instance="byte[]"/>
  <class gcsize="710144" instance="com.dynatrace.diagnostics.agent.introspectic
  <class gcsize="194576" instance="byte[]"/>
  <class gcsize="74712" instance="byte[]"/>
  <class gcsize="35368" instance="byte[]"/>
  <class gcsize="59528" instance="java.lang.System"/>
  <class gcsize="25744" instance="byte[]"/>
  <class gcsize="25544" instance="com.dynatrace.diagnostics.agent.shared.Consta
  <class gcsize="16400" instance="char[]"/>
  <class gcsize="16400" instance="char[]"/>
</hotspots>
```

Response code

```
200 OK
```

Delete memory dump

Delete a memory dump.

DELETE `https://<server>:8021/rest/management/profiles/<profilename>/me`

POST `https://<server>:8021/rest/management/profiles/<profilename>/me`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**resourceid**	The ID of the dump resource to delete.	yes	

Example

Request URL

```
DELETE https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps/2(
```

Response code

```
200 OK
```

Post-processing

Create post-processed memory dump job

Put a job to post-process an existing memory dump. The response contains a reference to the created job in the Location header.

PUT	`https://<server>:8021/rest/management/profiles/<profilename>/memo`
POST	`https://<server>:8021/rest/management/profiles/<profilename>/memo?httpMethod=PUT`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**resourceid**	The ID of the dump resource to post-process.	yes	

Example

Request URL

```
PUT https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps/2013(
```

Response code

```
201 Created
```

Response header

```
Location: https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps
```

Get post-processed memory dump job

Get a post-processing job. This returns the actual processing state, and a reference to the post-processed resource if the job is finished. See Get Memory Dump Post-Processed Result, below.

GET `https://<server>:8021/rest/management/profiles/<profilename>/memo`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**resourceid**	The ID of the dump resource to get the post-processed job.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps/2013(
```

Response content

####### Running

```
<memorydumppostprocessjob id="Memory Dump Postprocessing[8265196351026]">
<state>running</state>
</memorydumppostprocessjob>
```

####### Finished

```
<memorydumppostprocessjob id="Memory Dump Postprocessing[8265196351026]">
<state>finished</state>
<sessionreference href="https://localhost:8021/rest/management/profiles/JavaWo
</memorydumppostprocessjob>
```

Response code

```
200 OK
```

Get memory dump post-processed result

Get the result for post processing. The result contains a reference to the available memory dump resources.

GET `https://<server>:8021/rest/management/profiles/<profilename>/memo`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**resourceid**	The ID of the dump resource to get the post-processed result.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps/2013(
```

Response content

```
<postprocession>
  <resourcereference href="https://localhost:8021/rest/management/profiles/easy
  <resourcereference href="https://localhost:8021/rest/management/profiles/easy
 </postprocession>
```

Response code

```
200 OK
```

Get memory dump post-processed content

Get the processed classes for a post-processed memory dump. This includes the Garbage Collection size, the shallow class size, and the number of instances for the classes.

GET `https://<server>:8021/rest/management/profiles/<profilename>/memo`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**resourceid**	The ID of the dump resource to get post-processed content.	yes	

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/JavaWorld/memorydumps/2013(
```

Response content

```
<memorydumpclasses>
  <class instancecount="1" shallowsize="0" gcsize="106145144" instance="<VMRoot
  <class instancecount="1" shallowsize="0" gcsize="39663536" instance="<Garbage
</memorydumpclasses>
```

Response code

```
200 OK
```

Deprecated

This section contains deprecated REST interfaces to manage memory dumps. These interfaces have been replaced and should not be used.

Creating memory dumps

Note

Use Create Memory Dump Job to create a memory dump job.

POST `https://<server>:8021/rest/management/profiles/<profilename>/memo`

Parameters

Parameter	Description	Mandatory	Default Value
agentName	The name of the Agent for which the memory dump is created.	yes	
hostName	The host name of the Agent for which the memory dump is created.	yes	
processId	The process ID of the Agent for which the memory dump is created.	yes	
type	`simple` – Simple Total Memory Dashlet Dump `extended` – Extended Total Memory Dashlet Dump `selective` – Selective Memory Dashlet Dump	no	`simple`
isSessionLocked	`true` to lock session, `false` otherwise	no	`false`

Example

The following example HTTP request triggers a simple memory dump for a specific agent that is connected to the local running AppMon Server. The specific agent identification values can be retrieved from the REST service that provides the list of agents.

```
POST rest/management/profiles/easyTravel/memorydump HTTP/1.1
Host: localhost:8021
Content-Length: 88
agentName=easyTravelBackend&hostName=GRABS&processId=6612&type=simple&isSessionLocked=tru
```

The returned XML contains a `result` root node with a schedule ID in the `value` attribute.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="Memory Dump [1250108367765]"/>
```

This schedule ID can be used to query the status of the memory dump.

Memory dump status

> **Note**
>
> Use Get Memory Dump Job to get the status of a memory dump.

The following sample HTTP GET request queries the status of a previously triggered Memory Dump.

```
GET https://localhost:8021/rest/management/profiles/easyTravel/memorydumpcreated/Memory%2
```

The returned XML contains a `result` root node with a `value` attribute of `true` if the memory dump was created, or `false` if creation is still in progress. The `success` element indicates whether the memory dump was created successfully. If creation of the dump fails, `message` elements describe the reason. The dump status is kept for 15 minutes after the dump task was finished.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="true">
  <success>true</success>
  <message language="en"><text>SUCCESS</text></message>
</result>
```

Performance Warehouse (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Connection status

Connection configuration

The AppMon Server offers several RESTful interfaces to mange the Performance Warehouse connection settings and connection status. Use HTTP GET to get information about Performance Warehouse or HTTP PUT to modify the connection configuration and the connection status.

Connection status

GET connection status

Check if the Performance Warehouse is currently connected or not.

GET `https://<server>:8021/rest/management/pwhconnection/status`

GET `https://<server>:8021/rest/management/pwhconnection/status.xml`

GET `https://<server>:8021/rest/management/pwhconnection/status.json`

Example for XML

Request URL

```
GET https://localhost:8021/rest/management/pwhconnection/status.xml
```

Response content (Performance Warehouse is connected)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<pwhconnectionstatus>
  <isconnected>false</isconnected>
</pwhconnectionstatus>
```

Response code

```
200 OK
```

Example for JSON

Request URL

```
GET https://localhost:8021/rest/management/pwhconnection/status.json
```

Response content (Performance Warehouse is connected)

```
{
    "pwhconnectionstatus":{
        "isconnected":true
    }
}
```

Response code

```
200 OK
```

PUT connection status

Use the HTTP PUT interface to connect or disconnect the Performance Warehouse. The expected request content XML structure is similar to the response of the corresponding HTTP GET interface. Permissions are required to write to Repository configuration. An overloading POST interface provides the same output as HTTP PUT.

PUT	`https://<server>:8021/rest/management/pwhconnection/status`
PUT	`https://<server>:8021/rest/management/pwhconnection/status.xml`
PUT	`https://<server>:8021/rest/management/pwhconnection/status.json`

More overloading POST interfaces

POST	`https://<server>:8021/rest/management/pwhconnection/status?httpMethod=PUT`
POST	`https://<server>:8021/rest/management/pwhconnection/status.xml?httpMethod=PUT`
POST	`https://<server>:8021/rest/management/pwhconnection/status.json?httpMethod=PUT`

Example for XML

This example shows how to connect the performance warehouse.

Request URL

```
PUT https://localhost:8021/rest/management/pwhconnection/status.xml
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<pwhconnectionstatus>
  <isconnected>true</isconnected>
</pwhconnectionstatus>
```

Response code

```
200 OK
```

Example for JSON

This example shows how to disconnect the performance warehouse.

Request URL

```
PUT https://localhost:8021/rest/management/pwhconnection/status.json
```

Request content

```
{
  "isconnected":false
}
```

Response code

```
200 OK
```

Connection configuration

GET connection configuration

Get a description of the current configuration of the Performance Warehouse connection.

```
GET   https://<server>:8021/rest/management/pwhconnection/config
```

```
GET   https://<server>:8021/rest/management/pwhconnection/config.xml
```

```
GET   https://<server>:8021/rest/management/pwhconnection/config.json
```

Example for XML

Request URL

```
GET https://localhost:8021/rest/management/pwhconnection/config.xml
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<pwhconnectionconfiguration>
  <user>SCOTT</user>
  <password>*******</password>
  <dbname>dynadb</dbname>
  <dbms>sqlserver</dbms>
  <host>localhost</host>
  <port>1433</port>
  <useurl>false</useurl>
  <usessl>false</usessl>
  <url xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:nil="true"/>
</pwhconnectionconfiguration>
```

Response code

```
200 OK
```

Example for JSON

Request URL

```
GET https://localhost:8021/rest/management/pwhconnection/config.json
```

Response content

```
{
    "pwhconnectionconfiguration":{
        "user":"SCOTT",
        "password":"***********",
        "dbname":"dynadb",
        "dbms":"sqlserver",
        "host":"localhost",
        "port":"1433",
        "useurl":false,
        "usessl":false,
        "url":null
    }
}
```

Response code

```
200 OK
```

PUT connection configuration

Use the HTTP PUT interface to set the performance warehouse connection settings by sending an XML or JSON representation as content of the request. The performance warehouse will connect/reconnect automatically in background. The expected XML/JSON structure of the request content is similar to the response of the corresponding HTTP GET interface. Permissions are required to write to Repository configuration.

PUT	`https://<server>:8021/rest/management/pwhconnection/config`
PUT	`https://<server>:8021/rest/management/pwhconnection/config.xml`
PUT	`https://<server>:8021/rest/management/pwhconnection/config.json`

More overloading POST interfaces

POST	`https://<server>:8021/rest/management/pwhconnection/config?httpMethod=PUT`	Consumes `application/xml`
POST	`https://<server>:8021/rest/management/pwhconnection/config.xml?httpMethod=PUT`	Consumes `application/xml`
POST	`https://<server>:8021/rest/management/pwhconnection/config.json?httpMethod=PUT`	Consumes `application/json`

For the `dbms` element within the connection configuration description the following values are valid: `oracle`, `sqlserver`, `db2`, `postgresql`, `sqlazure`, or `embedded`.

Example for XML

Request URL

```
PUT https://localhost:8021/rest/management/pwhconnection/config.xml
```

Request content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<pwhconnectionconfiguration>
  <user>SCOTT</user>
  <password>whatagreatday</password>
  <dbname>dynadb</dbname>
  <dbms>sqlserver</dbms>
  <host>localhost</host>
  <port>1433</port>
  <useurl>false</useurl>
  <usessl>false</usessl>
  <url xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:nil="true"/>
</pwhconnectionconfiguration>
```

Response code

```
200 OK
```

Example for JSON

Request URL

```
PUT https://localhost:8021/rest/management/pwhconnection/config.json
```

Request content

```
{
    "user":"testaccount",
    "password":"testaccount",
    "dbname":"dynatest620",
    "dbms":"sqlserver",
    "host":"localhost",
    "port":"1433",
    "useurl":false,
    "usessl":false,
    "url":null
}
```

Response code

```
200 OK
```

Agents and collectors (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Agents

Collectors

Deprecated

Agents

The AppMon Server offers two RESTful interfaces to interact with AppMon Agents. Use the List Agents interface to gather information about AppMon Agents. Use the Hot Sensor Placement interface to perform a Hot Sensor Placement for a specific AppMon Agent.

List agents

This interface lists all known AppMon Agents and provides information such as which AppMon Collector they are related to and other metadata.

GET `https://<server>:8021/rest/management/agents` produces `text/xml`

Example

Request URL

```
GET https://localhost:8021/rest/management/agents
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<agents>
  <agentinformation>
    <agentGroup>easyTravelBackend</agentGroup>
    <agentId>-91443089</agentId>
    <capture>true</capture>
    <classLoadCount>4783</classLoadCount>
    <collectorinformation>
      <name>Embedded dynaTrace Collector</name>
      <host>GRABS</host>
      <connected>true</connected>
      <embedded>true</embedded>
    </collectorinformation>
    <configuration>easyTravel</configuration>
    <connected>true</connected>
    <eventCount>70</eventCount>
    <host>GRABS</host>
    <licenseInformation>License OK</licenseInformation>
    <licenseOk>true</licenseOk>
    <name>easyTravelBackend</name>
    <processId>6612</processId>
    <skippedEvents>0</skippedEvents>
    <skippedPurePaths>0</skippedPurePaths>
    <startupTimeUTC>1250107304703</startupTimeUTC>
    <supportsHotSensorPlacement>true</supportsHotSensorPlacement>
    <systemProfile>easyTravel</systemProfile>
    <technologyType>Java</technologyType>
    <totalClassLoadCount>0</totalClassLoadCount>
    <totalCpuTime>0.0</totalCpuTime>
    <totalExecutionTime>0.0</totalExecutionTime>
    <totalPurePathCount>0</totalPurePathCount>
  </agentinformation>
</agents>
```

Response code

```
200 OK
```

Hot Sensor Placement

This interface performs a Hot Sensor Placement on an AppMon Agent with ID `<agent_id>`.

GET `https://<server>:8021/rest/management/agents/<agent_id>/hotsensor`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Path	**agent_id**	The ID of the corresponding AppMon Agent.	yes	

Example

The following example triggers a Hot Sensor Placement on AppMon Agent, which has the ID `-86650354`. The XML response describes that the request was executed successfully.

Request URL

```
GET https://localhost:8021/rest/management/agents/-86650354/hotsensorplacement
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<result value="true"/>
```

Response code

```
200 OK
```

Collectors

List of collectors

GET `https://<server>:8021/rest/management/collectors` produces `text/xml`

The returned XML contains a `collectors` root node and `collectorinformation` sub-nodes with detailed information about connected collectors.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<collectors href="https://localhost:8021/rest/management/collectors">
  <collectorinformation href="https://localhost:8021/rest/management/collectors/Embedded%
    <connected>true</connected>
    <embedded>true</embedded>
    <host>GRABS</host>
    <name>Embedded dynaTrace Collector</name>
  </collectorinformation>
</collectors>
```

Collector details

GET `https://<server>:8021/rest/management/collectors/<collectorname>`

The returned XML contains a `collectorinformation` root node with detailed information about the collector.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<collectorinformation href="https://localhost:8021/rest/management/collectors/Embedded%20
  <connected>true</connected>
  <embedded>true</embedded>
  <host>GRABS</host>
  <name>Embedded dynaTrace Collector</name>
</collectorinformation>
```

Restart and shut down a collector

The AppMon Server offers two REST interfaces to restart and shut down connected AppMon Collectors. Both are HTTP POST requests. The following two example calls show how to restart or stop collectors connected to a local AppMon Server.

```
POST https://<server>:8021/rest/management/collector/<collectorname>/restart HTTP/1.1
POST https://<server>:8021/rest/management/collector/<collectorname>/shutdown HTTP/1.1
```

Deprecated

This section contains deprecated REST interfaces for server management. These interfaces have been replaced and should not be used.

They are disabled by default.

If, however, you need to enable these interfaces, use the debug flag `com.dynatrace.diagnostics.rest.depricatedinterfaces.enabled=true`.

Restart and shut down a collector

> **Deprecated**
>
> Use POST interfaces to shut down or restart an AppMon Collector.

The AppMon Server offers two REST interfaces to stop and restart connected AppMon Collectors. Both are simple HTTP GET requests. The following two example calls show how to restart or stop collectors connected to a local AppMon Server.

```
GET https://<server>:8021/rest/management/collector/<collectorname>/restart HTTP/1.1
GET https://<server>:8021/rest/management/collector/<collectorname>/shutdown HTTP/1.1
```

PureLytics Stream (REST)

Element	Values	Description
duration	Milliseconds	The job duration up to now.
jobId	UUID	The job UUID.
progress	0-100	The progress of the job up to now.
startTime	Milliseconds	The start time of the job in milliseconds
status	unknown | refused | queued | scheduled | running | finished | failed | configured | waiting for executor	The current status of the job.
systemProfile	System Profile	The system profile that the job runs on.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<analyticsstreamjob>
  <duration>0</duration>
  <jobId>d81c297e-d2f2-4892-b645-89c2db5ecd35</jobId>
  <progress>0</progress>
  <startTime>1453883116322</startTime>
  <status>running</status>
  <systemProfile>easyTravel</systemProfile>
</analyticsstreamjob>
```

```
200 OK
```

```
GET https://localhost:8021/rest/management/profiles/easyTravel/analyticsstreamjob/d81c297
```

Assuming that a job UUID was obtained from the location header or response content of a previously

created job, this example demonstrates how to get the PureLytics job info of said job.

Example

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	
Path	**jobId**	The UUID of the PureLytics Job.	yes	

Parameters

GET `https://<server>:8021/rest/management/profiles/<profilename>/anal`

Get the state of a PureLytics Job with a UUID.

Get PureLytics job

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<analyticsstreamjob>
  <duration>0</duration>
  <jobId>d81c297e-d2f2-4892-b645-89c2db5ecd35</jobId>
  <progress>0</progress>
  <startTime>1453883116322</startTime>
  <status>running</status>
  <systemProfile>easyTravel</systemProfile>
</analyticsstreamjob>
```

```
Location: https://localhost:8021/rest/management/profiles/easyTravel/analyticsstreamjob/c
```

```
201 Created
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<analyticsstreamjob>
  <timeframeStart>2016-01-26T12:00:00</timeframeStart>
  <timeframeEnd>2016-01-26T15:00:00</timeframeEnd>
  <exportType>ALL</exportType>
    <target>
      <elasticsearch>
        <url>https://my-es-server:9200</url>
        <username></username>
        <password></password>
        <maxQueueSize>100</maxQueueSize>
        <flushInterval>1</flushInterval>
        <untrustedSSL>true</untrustedSSL>
        <basicHttpAuthentication>false</basicHttpAuthentication>
        <index>test</index>
      </elasticsearch>
    </target>
</analyticsstreamjob>
```

```
POST https://localhost:8021/rest/management/profiles/easyTravel/analyticsstreamjob
```

This example demonstrates how to create a PureLytics job that streams all data on the *easyTravel* System Profile for 26 January 2016 between 12:00:00 and 15:00:00 to the Elasticsearch server located at https://my-es-server:9200. The job streams to an Elasticsearch instance which can only be set in the AppMon Client. A successful request returns the PureLytics job along with a location header that can be used to request and monitor the job. See Get PureLytics Job for more information.

Example

Element	Values	Description
`timeframeStart`	`ISO 8601 timestamp`	A start date and time for the data that should be exported.
`timeframeEnd`	`ISO 8601 timestamp`	An end date and time for the data that should be exported.
`exportType`	`ALL | USERACTION | VISIT`	The type of data that should be exported.

target	`String`	Configure the target to which the data should be exported. Valid values include “elasticsearch” or “generichttp”
url	`String`	The URL of the Elasticsearch cluster (REST port) or the generic HTTP endpoint.
username	`String`	The username for basic HTTP authentication.
password	`String`	The password for basic HTTP authentication.
maxQueueSize	`number`	The Maximum number of documents per request.
untrustedSSL	`true | false`	Allow untrusted SSL certificates.
basicHttpAuthentication	`true | false`	Enable basic HTTP authentication.
index	`String`	Index identifier.

```
<analyticsstreamjob>
<timeframeStart>2016-01-26T12:00:00</timeframeStart>
<timeframeEnd>2016-01-26T15:00:00</timeframeEnd>
<exportType>ALL</exportType>
  <target>
    <elasticsearch>
      <url>http://my-elasticsearch-server:9200</url>
      <username></username>
      <password></password>
      <maxQueueSize>100</maxQueueSize>
      <untrustedSSL>true</untrustedSSL>
      <basicHttpAuthentication>false</basicHttpAuthentication>
      <index>test</index>
    </elasticsearch>
  </target>
</analyticsstreamjob>
```

```
{
```

```
    "timeframeStart": "2016-01-26T12:00:00",
    "timeframeEnd": "2016-01-26T15:00:00",
    "exportType": "ALL",
    "target": {
        "elasticsearch": {
            "url": "http://my-elasticsearch-server:9200",
            "username": "",
            "password": "",
            "maxQueueSize": 100,
            "untrustedSSL": true,
            "basicHttpAuthentication": false,
            "index": "test"
        }
}
```

Request content

Type	Parameter	Description	Mandatory	Default Value
Path	**profilename**	The name of the corresponding System Profile.	yes	

Parameters

POST `https://<server>:8021/rest/management/profiles/<profilename>/ana`

The PureLytics job is described by either a JSON or an XML document and exports data. This document is sent to the REST interface within the request content. The expected JSON and XML structure is described below. All XML elements or JSON name/value pairs are required in order to create a PureLytics job.

Post a request containing a JSON or XML body to create a PureLytics job. The location header of the response contains a reference to the created job.

Create PureLytics job

The on-demand export is limited to one job per server, this includes jobs that are started from the Client.

On-demand export

The AppMon Server offers RESTful interfaces to start and monitor an on-demand export with the PureLytics Stream. You can export data within a specific timeframe and type to an Elasticsearch instance. See PureLytics Stream for more information.

Resource dumps (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Create a thread dump

Get thread dump status

Create a sampling session

Get sampling session status

Use the Thread Dump REST interface to request a thread dump. You can then use a REST interface call to query the status of the requested dump.

Create a thread dump

POST `https://<server>:8021/rest/management/profiles/<profilename>/thre`

Parameters

Parameter	Description	Mandatory	Default Value
agentName	The name of the Agent for which the thread dump is created.	yes	
hostName	The host name of the Agent for which the thread dump is created.	yes	
processId	The process ID of the Agent for which the thread dump is created.	yes	
isSessionLocked	Valid values are `true` to lock the session, otherwise `false`.	no	`false`

Example

The following HTTP Request example requests a thread dump for a specific Agent connected to the local AppMon Server. You can retrieve the specific Agent identification values from the REST service that provides the Agents list.

Example request

```
POST rest/management/profiles/easyTravel/threaddump HTTP/1.1
Host: localhost:8021
Content-Length: 76
agentName=easyTravelBackend&hostName=GRABS&processId=6612&isSessionLocked=true
```

Returned XML

The returned XML contains a `result` root node with a schedule ID in the `value` attribute.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="Thread Dump [1250108367765]"/>
```

Use this schedule ID to query the status of the thread dump.

Get thread dump status

You can request the thread dump creation status based on the schedule ID.

GET `https://<server>:8021/rest/management/profiles/<profilename>/threa`
`id>`

Example

Example request

The following example HTTP GET request queries the status of a previously triggered thread dump:

```
GET https://localhost:8021/rest/management/profiles/easyTravel/threaddumpcreated/Thread%2
```

Returned XML request

The returned XML contains a `result` root node with a `value` attribute of `true` if the thread dump was created or `false` if creation is still in progress. The `success` element indicates whether the thread dump was created successfully. If creation of the dump fails, `message` elements describe the reason. The dump status is kept for 15 minutes after the dump task was finished.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="true">
  <success>true</success>
  <message language="en"><text>SUCCESS</text></message>
</result>
```

Create a sampling session

POST `https://<server>:8021/rest/management/profiles/<profilename>/samp`

Parameters

Parameter	Description	Mandatory	Default Value
agentName	The name of the Agent to sample.	yes	
hostName	The Agent host name to sample.	yes	
processId	The process ID of the Agent to sample.	yes	
isSessionLocked	Valid values are `true` to lock the session, otherwise `false`.	no	`false`
duration	The duration of the sampling session, in seconds.	no	`120`
samplingRate	The sampling rate. Valid values are `lowest`, `low`, `medium`, `high`, `highest`.	no	high

Example

The following HTTP request example requests a sampling session for a specific Agent connected to the local AppMon Server. You can retrieve specific Agent identification values from the REST service that provides the Agents list.

```
POST rest/management/profiles/easyTravel/sampling HTTP/1.1
Host: localhost:8021
Content-Length: 76
agentName=easyTravelBackend&hostName=GRABS&processId=6612&isSessionLocked=true
```

Returned XML

The returned XML contains a `result` root node with a schedule ID in the `value` attribute.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="Sampling [1250108367765]"/>
```

This schedule ID can be used to query the status of the sampling session.

Get sampling session status

Based on the schedule ID the creation status of the sampling can be requested.

GET `https://<server>:8021/rest/management/profiles/<profilename>/samp`
`id>`

Example

Example HTTP request

The following HTTP GET example queries the status of a previously triggered sampling session:

```
GET https://localhost:8021/rest/management/profiles/easyTravel/samplingcreated/Sampling%2
```

Returned XML

The returned XML contains a `result` root node with a `value` attribute of `true` if sampling is done or `false` if sampling is still in progress. The `success` element indicates whether sampling was successful. If sampling fails, `message` elements describe the reason. The sampling status is kept for 15 minutes after sampling was finished.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="true">
  <success>true</success>
  <message language="en"><text>SUCCESS</text></message>
</result>
```

Server management (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Version

License information

License import

License volume import

Restart server

Shut down server

Support archives

Deprecated

Version

Request the version of the AppMon Server.

GET `https://<server>:8021/rest/management/version` Produces `text/xml`

Example

Request URL

```
GET https://localhost:8021/rest/management/version
```

Response content

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result value="4.1.0.2888"/>
```

Response code

```
200 OK
```

License information

Request license information.

GET `https://<server>:8021/rest/management/server/license` Produces `application`

Example

Request URL

```
GET https://localhost:8021/rest/management/server/license
```

Response content

```
<licenseinformation>
  <licensedto>dynaTrace SaaS Production Edition for internal use only</licensed
  <licensenumber>201305151008</licensenumber>
  <licenseedition>SaaS - Production Edition</licenseedition>
  <usedvolumepercentage>0.0</usedvolumepercentage>
  <nextvolumerenewaldate />
  <validfrom>2011-01-10</validfrom>
  <expiredate>2014-10-05</expiredate>
  <currentuemtransactions>-1</currentuemtransactions>
  <maximaluemtransactions>-1</maximaluemtransactions>
  <licensedagents>
    <agent name="Java">3</agent>
    <agent name=".NET">-1</agent>
    <agent name="Message Broker">-1</agent>
    <agent name="Browser">-1</agent>
    <agent name="PHP">-1</agent>
    <agent name="Web server">-1</agent>
    <agent name="z/OS">-1</agent>
    <agent name="Host Monitoring">-1</agent>
    <agent name="Java (NoSQL)">-1</agent>
    <agent name="ADK">-1</agent>
    <agent name="Windows Instances">-1</agent>
    <agent name="Web Server Developer">-1</agent>
  </licensedagents>
</licenseinformation>
```

Response code

```
200 OK
```

License import

Send a license file to the AppMon Server and trigger a license import.

POST	`https://<server>:8021/rest/management/server/license`	Consumes `multipart/data`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Form Data	**file**	The AppMon license file (`application/octet-stream`).	yes	

Example

Request URL

```
POST https://localhost:8021/rest/management/server/license
```

HTML form

```
<html>
  <body>
    <h1>Upload License File</h1>
    <form action="https://localhost:8021/rest/management/server/license" metho
      <p>
        Select the license file:
        <input type="file" name="file" size="45" />
      </p>
      <input type="submit" value="Upload License" />
    </form>
  </body>
</html>
```

Response code

```
200 OK
```

License volume import

Send a volume file to the AppMon Server and trigger a volume import.

		Cons
POST	`https://<server>:8021/rest/management/server/license/volume`	`mul` `dat`

Parameters

Type	Parameter	Description	Mandatory	Default Value
Form Data	**file**	The AppMon license file (`application/octet-stream`).	yes	

Example

Request URL

```
POST https://localhost:8021/rest/management/server/license/volume
```

HTML form

```
<html>
  <body>
    <h1>Upload Volume File</h1>
    <form action="https://localhost:8021/rest/management/server/license/volume"
      <p>
        Select the volume file:
        <input type="file" name="file" size="45" />
      </p>
      <input type="submit" value="Upload Volume" />
    </form>
  </body>
</html>
```

Response code

```
200 OK
```

Restart server

Restart the AppMon Server. You must have permissions to shut down and restart the Server.

POST `https://<server>:8021/rest/management/server/restart` Produces `applicatio`

Shut down server

Shut down the AppMon Server without restarting. You must have permissions to shut down the Server.

POST `https://<server>:8021/rest/management/server/shutdown` Produces `applicati`

Support archives

AppMon has a REST interface to download and stream Support Archives, which are also called SysInfo Packages. The compressed bytes stream directly to the REST Client instead of creating a complete file on the Server, providing a faster response.

GET `https://<server>:8021/rest/management/sysinfo/package`

Parameters

Parameter	Description	Mandatory	Default Value
timeout	Timeout, in seconds, for fetching large files.	no	60 seconds
include	Comma-separated list of file types to be included.	no	See Supported File Types.

exclude	Comma-separated list of file types to be excluded.	no	See Supported File Types.

Supported file types

File type	Description	Default included
logfiles	Log files	yes
configfiles	Configuration files	yes
dashboards	Dashboards	yes
profiles	System Profiles	yes
agentrecords	Agent records	yes
sessions	All stored sessions, including the self-monitoring session	**no**
interactionlogs	Interaction logs	yes
debugfiles	Debug files	yes
protocoldump	Protocol dumps	yes
licensefile	License (activation) file	yes
sensorrecords	Sensor records	yes
componentproperties	Component properties	yes
serversession	The self-monitoring session	yes

agentusagelogs	Agent usage log files	yes

Examples

URI	Description
https://localhost:8021/rest/management/sysinfo/package?include=sessions	Include sessions.
https://localhost:8021/rest/management/sysinfo/package?include=all	Include everything
https://localhost:8021/rest/management/sysinfo/package?exclude=configfiles	Exclude configuration files.
https://localhost:8021/rest/management/sysinfo/package?exclude=all&include=logfiles	Include only log files.
https://localhost:8021/rest/management/sysinfo/package?exclude=all&include=logfiles,configfiles	Include log files and configuration files.

Error handling

If you specify an `include` or `exclude` parameter that is not one of the file types listed above, an error occurs and the REST interface displays information about supported file types.

Deprecated

This section describes deprecated REST interfaces for Server management. They are replaced by the interfaces described above, and should not be used.

These deprecated interfaces are disabled by default. If, however, you need to enable them, use the debug flag `com.dynatrace.diagnostics.rest.deprecatedinterfaces.enabled=true`.

Restart server

Deprecated

Use the POST interface to restart the AppMon Server.

GET `https://<server>:8021/rest/management/server/restart`

Shut down server

Deprecated

Use the POST interface to shut down the AppMon Server.

GET `https://<server>:8021/rest/management/server/shutdown`

License

Deprecated

Use the new GET interface to request the license information.

GET `https://<server>:8021/rest/management/server/license/information`

Stored sessions (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

List stored sessions

Metadata

Import and export

Delete

Reanalyze

Related pages

Live sessions (REST)

The AppMon Server offers REST interfaces to manage stored sessions. Using this interfaces you can list them, get their metadata, export, import and delete them.

List stored sessions

This request lists all stored sessions that are available to the user that requested the call.

GET `https://<server>:8021/rest/management/storedsessions` Produces `application`

Example

The following example HTTP request retrieves an XML document that lists all available stored sessions:

```
GET rest/management/storedsessions HTTP/1.1
```

Returned XML

The returned XML contains a `sessions` root node with a `sessionreference` node for each stored session. The `href` or the `id` of the stored session can be used to access other REST interfaces. For example, `GET management/storedsessions/{id}`.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sessions>
  <sessionreference id="JavaWorld/20150316095824_0.threaddump" href="http://localhost:802
    <profile>JavaWorld</profile>
    <sessiontype>stored</sessiontype>
    <storedsessiontype>threaddump</storedsessiontype>
    <name>Mobile_Test</name>
  </sessionreference>
  <sessionreference id="JavaWorld/20150312081037_0.sampling" href="http://localhost:8020/
    <profile>JavaWorld</profile>
    <sessiontype>stored</sessiontype>
    <storedsessiontype>sampling</storedsessiontype>
    <name>easyTravel</name>
  </sessionreference>
  <sessionreference id="JavaWorld/20150317104017_0.memdump" href="http://localhost:8020/r
    <profile>JavaWorld</profile>
    <sessiontype>stored</sessiontype>
    <storedsessiontype>memdump_selective</storedsessiontype>
    <name>JavaWorld</name>
  </sessionreference>
...
</sessions>
```

Deprecated interface to list stored session IDs

> **Deprecated**
>
> Please see List Stored Sessions for more information.

This request lists IDs of all stored sessions that are available to the user who requested the call.

GET `https://<server>:8021/rest/management/sessions` Produces `text/xml`

Parameters

Parameter	Description	Mandatory	Default Value
type	Type of stored session. The type can be one of `purepath`, `memdump_simple`, `memdump_extended`, `memdump_selective`, `threaddump`, `sampling`, or empty to retrieve all session types.	no	

Example

The following example HTTP request retrieves an XML document that all available stored sessions:

```
GET rest/management/sessions HTTP/1.1
```

Returned XML

The returned XML contains a `sessionids` root node with a `sessionid` node for each stored session. The name of the stored session can be used in other REST interfaces, such as `sessions/export`.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sessionids>
  <sessionid sessiontype="purepath">AjaxWorld [2011-10-07 12:10:57]</sessionid>
  <sessionid sessiontype="purepath">AjaxWorld [2011-10-21 10:08:50]</sessionid>
  <sessionid sessiontype="purepath">AmazonAccount [2011-10-21 10:08:50]</sessionid>
  <sessionid sessiontype="purepath">BrowserDiagnostics [2011-10-21 10:08:50]</sessionid>
  ...
</sessionids>
```

Metadata

This request shows comprehensive details of a stored session. Check at the returned XML for more information.

> **Note**
>
> Ensure to encode the `id` beforehand as it may otherwise lead to routing errors due to possible slashes in the `id`.

GET `https://<server>:8021/rest/management/sessions/storedsessions/<id`

Example

The following example HTTP request retrieves an XML document showing comprehensive information of a stored session.

```
GET rest/management/storedsessions/JavaWorld%252F20150317104017_0.memdump HTTP/1.1
```

Returned XML

The returned XML contains a `sessiondetail` root node with several nodes for data.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<sessiondetail id="JavaWorld/20150316100010_0.threaddump" href="http://localhost:8020/res
  <profile>JavaWorld</profile>
  <sessiontype>stored</sessiontype>
  <storedsessiontype>threaddump</storedsessiontype>
  <agent>JavaWorld@lnz127525d01:4952</agent>
  <capturingduration>0</capturingduration>
  <capturingstart>1426496410106</capturingstart>
  <description>Session exported via REST download</description>
  <directoryPath>C:\workspaces\dynatrace\trunk-classic\jloadtrace\server\sessions\stored\
  <deletionlocked>true</deletionlocked>
  <labels>
    <label>labelC</label>
    <label>labelB</label>
    <label>labelA</label>
  </labels>
  <name>JavaWorld/20150316100010_0.threaddump</name>
  <numberofpurepaths>0</numberofpurepaths>
  <recordingtype>manual</recordingtype>
  <size>22563</size>
  <state>finished</state>
</sessiondetail>
```

Import and export

Import stored sessions

Use this request to upload and import a stored session.

POST `https://<server>:8021/rest/management/storedsessions`

Parameters

Parameter	Description	Mandatory	Default Value

labels	List of labels for the imported session.	no	
locked	`true` to activate deletion lock, `false` to deactivate it.	no	true

Example

```
POST rest/management/storedsessions?labels=labelA&labels=labelB&labels=labelC&locked=true
```

Export stored sessions

Use this request to export a stored session. For large stored sessions, the size of the downloaded file can be very large. Make sure the download machine has enough free space to ensure that the download is successful.

> **Note**
>
> Encode the id beforehand as it may otherwise lead to routing errors due to possible slashes in the id.

To retrieve the live data, first store a session via `rest/management/profiles/<SESSIONNAME>/storepurepaths`.

Concurrent session downloads is limited to two to prevent AppMon Server overload. Additional download requests are delayed until the first requests are completed.

GET `https://<server>:8021/rest/management/storedsessions/<id>.dts`

Parameters

Parameter	Description	Mandatory	Default Value
removeconfidentialstrings	`true` to remove confidential strings from the exported session, `false` to keep them included.	no	true

Example

The following example HTTP request exports the data for the stored session `JavaWorld/20150317104017_0.memdump`.

```
GET rest/management/storedsessions/JavaWorld%252F20150317104017_0.memdump.dts HTTP/1.1
```

The data for the session is streamed as binary data with the file extension `.dts`. You can import the session data on the same or a different AppMon installation by using the AppMon Client or the REST interface.

Deprecated interface to export stored sessions

> **Deprecated**
>
> This interface will be removed in future releases. Use the Export stored sessions interface instead.

GET `https://<server>:8021/rest/management/sessions/<sessionname>/expo`

Parameters

Parameter	Description	Mandatory	Default Value
removeConfidentialStrings	`true` to remove confidential strings from the exported session, `false` to keep them included.	no	true

Delete

Delete individual stored session

Use this request to delete a stored session which is not continuously recorded. Trying to delete a continuously recorded session will return an error.

Note

Encode the id beforehand as it may otherwise lead to routing errors due to possible slashes in the id.

DELETE `https://<server>:8021/rest/management/storedsessions/<id>`

Example

```
DELETE rest/management/storedsessions/JavaWorld%252F20150317104017_0.memdump
```

Delete multiple stored sessions

Use this request to delete every non-continuous stored session that matches a specified session type and is older than a specified number of minutes. Additionally the deletion request can be restricted to stored sessions of a specific system profile.

DELETE `https://<server>:8021/rest/management/storedsessions?type=<sessiontype>&olderthanminutes=<age-in-minutes>`

Parameters

Parameter	Description
type	Type of sessions which shall be deleted. The type can be one of `purepath`, `memdump_simple`,`memdump_extended`, `memdump_selective` or `sampling` ; or `all` to delete all session types.
olderthanminutes	The minimum age in minutes a stored session must have to be deleted. The value `0` deletes regardless of their age.
profile	If specified, the deletion only affects stored sessions of this system profile. Otherwise the sys irrelevant for deletion.

Example

```
DELETE rest/management/storedsessions?type=all&olderthanminutes=42&profile=JavaWorld HTTP
```

This request deletes any profile of any stored session type which is at least `42` minutes old and belongs to the system profile `JavaWorld`.

Returned XML

The returned XML contains a `deletedumps` root node which contains 3 child nodes. `successfullydeleted` contains all stored sessions which have been deleted due to this request, `totaldeletedbytes` contains the summarized byte sizes of the deleted stored sessions.

`failedtodelete` contains all stored sessions which could not be deleted. Each of these nodes has a `href` attribute which can be used to acquire more information via the `GET /storedsessions/<id>` interface.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<deletestoredsessions>
  <failedtodelete>
    <session id="JavaWorld/20150319122600_0.memdump" href="http://localhost:8020/rest/mar
      <profile>JavaWorld</profile>
      <sessiontype>stored</sessiontype>
      <storedsessiontype>memdump_extended</storedsessiontype>
    </session>
    <session id="JavaWorld/20150319122623_0.threaddump" href="http://localhost:8020/rest/
      <profile>JavaWorld</profile>
      <sessiontype>stored</sessiontype>
      <storedsessiontype>threaddump</storedsessiontype>
    </session>
  </failedtodelete>
  <successfullydeleted>
    <session type="memdump_extended" id="JavaWorld/20150319122600_0.memdump" profile="Jav
    <session type="threaddump" id="JavaWorld/20150319122623_0.threaddump" profile="JavaWo
    <session type="sampling" id="JavaWorld/20150319122547_0.sampling" profile="JavaWorld"
    <session type="memdump_selective" id="JavaWorld/20150319122632_0.memdump" profile="Ja
    <session type="threaddump" id="JavaWorld/20150319122625_0.threaddump" profile="JavaWo
    <session type="threaddump" id="JavaWorld/20150319122556_0.threaddump" profile="JavaWo
    <session type="sampling" id="JavaWorld/20150319122620_0.sampling" profile="JavaWorld"
    <session type="threaddump" id="JavaWorld/20150319122629_0.threaddump" profile="JavaWo
    <session type="memdump_selective" id="JavaWorld/20150319122551_0.memdump" profile="Ja
  </successfullydeleted>
  <totaldeletedbytes>1673642</totaldeletedbytes>
</deletestoredsessions>
```

Reanalyze

Start session reanalysis

GET `https://<server>:8021/rest/management/sessions/<sessionname>/reana`

Example

The following example HTTP request reanalyzes a session with the name *SampleSession*:

```
GET rest/management/sessions/SampleSession/reanalyze HTTP/1.1
```

Returned XML

The returned XML contains a `result` root node with a `value` attribute of `true` or `false`. If `true`, the specified session could be found and session reanalysis started successfully.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><result
value="true"/>
```

Check reanalysis status

GET `https://<server>:8021/rest/management/sessions/<sessionname>/reana`

Example

The following example HTTP GET request queries whether the session named *SampleSession* is currently being reanalyzed:

```
GET rest/management/sessions/SampleSession/reanalyze/finished HTTP/1.1
```

Returned XML

The returned XML contains a `result` root node with a `value` attribute of `true` or `false`. If `true`, the last session reanalysis is finished.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><result
value="true"/>
```

System Profile templates (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Create new System Profile based on template

Copy agent group to existing system profile

On the System Profile Templates page you can find out how to configure System Profile Templates within the AppMon Client.

Create new System Profile based on template

You can create a system profile based on a template using following REST call:

PUT `https://<server>:8021/rest/management/profiles/<newprofilename>`

Parameters

Parameter	Description	Mandatory
templateprofile	The name of the system profile template.	yes

Example

Request URL

```
PUT https://localhost:8021/rest/management/profiles/myNewSystemProfile?templat
```

Response code

```
201 Created
```

Response header

```
Location: https://localhost:8021/rest/management/profiles/myNewSystemProfile
```

Limitation

A high number of system profiles may degrade the performance of your AppMon Server. Thus, the creation of new system profiles is limited to 30 per hour.

Copy agent group to existing system profile

You can copy an agent group defined in a system profile template to an existing system profile using following REST call:

PUT `https://<server>:8021/rest/management/profiles/<profilename>/agent`

Parameters

Parameter	Description	Mandatory
templateprofile	The name of the system profile template.	yes
sourceagentgroup	The name of the agent group defined in the system profile template.	yes

Example

Request URL

```
https://localhost:8021/rest/management/profiles/myNewSystemProfile/agentgroups/
```

Response Code

```
201 Created
```

Response Header

```
https://localhost:8021/rest/management/profiles/myNewSystemProfile/agentgroups/
```

To define an agent mapping for the agent group, use the PUT Mapping REST call that is described in the section Agent Group Mapping.

System Profiles (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Profile list

Profile meta data

Enable or disable System Profile

System Profile configurations

Related pages

System Profile

Profile list

This interface lists all System Profiles of the AppMon Server and is able to respond with different media types. Besides the default media type XML, also JSON or JSONP can be produced. Just add the appropriate extension (`.xml`, `.json`, or `.jsonp`) to the URL or alternatively set the appropriate media type at the requests `Accept` header.

GET	`https://<server>:8021/rest/management/profiles`	Produces `appli`
GET	`https://<server>:8021/rest/management/profiles.xml`	Produces `appli`
GET	`https://<server>:8021/rest/management/profiles.json`	Produces `application`
GET	`https://<server>:8021/rest/management/profiles.jsonp`	Produces `application`

XML example

Request

```
GET https://localhost:8021/rest/management/profiles
```

Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<profiles href="https://localhost:8021/rest/management/profiles">
  <systemprofile isrecording="false" id="dynaTrace Self-Monitoring" href="https
  <systemprofile isrecording="true" id="easyTravel" href="http://localhost:802
</profiles>
```

The returned XML contains a `profiles` root node with `systemprofile` sub-nodes with `id` and `href` attributes. `id` refers to the System Profile name. `href` is the link to System Profile specific REST URI.

JSON example

Request

```
GET https://localhost:8021/rest/management/profiles.json
```

Response

```
{
    "href": "http://localhost:8020/rest/management/profiles",
    "systemProfiles": [
        {
            "href": "http://localhost:8020/rest/management/profiles/dynaTrace%
            "id": "dynaTrace Self-Monitoring",
            "isRecording": false
        },
        {
            "href": "http://localhost:8020/rest/management/profiles/easyTravel'
            "id": "easyTravel",
            "isRecording": false
        }
    ]
}
```


The returned JSON contains an array of all `systemProfiles` with their `id` and `href`, where `id` refers to the System Profile name and `href` is the link to the System Profile specific REST URI.

JSONP example

Request

```
GET https://localhost:8021/rest/management/profiles

Headers:
Accept: application/javascript
```

Response


```
callback({"href":"https://localhost:8021/rest/management/profiles","systemProf
```

The returned JSON contains an array of all `systemProfiles` with their `id` and `href`, where `id` refers to the System Profile name and `href` is the link to the System Profile specific REST URI.

The default function name is `callback`, but it can be easily changed: add the `jsonp` URL parameter.

Request

```
GET https://localhost:8021/rest/management/profiles?jsonp=something

Headers:
Accept: application/javascript
```

Response

```
something({"href":"https://localhost:8021/rest/management/profiles","systemPro
```

Profile meta data

Use HTTP GET on a specific System Profile to get an XML representation describing the System Profile and its meta data.

GET	`https://<server>:8021/rest/management/profiles/<profilename>`	Prod `tex`

Example

Request URL

```
GET https://localhost:8021/rest/management/profiles/easyTravel
```

Response content

```
<systemprofile enabled="true" isInteractiveLicensed="false" isrecording="false"
  <agentgroupsreference href="https://localhost:8021/rest/management/profiles/e
  <description>Profile for the easyTravel demo application.</description>
</systemprofile>
```

Response code

```
200 OK
```

Enable or disable System Profile

GET `https://<server>:8021/rest/management/profiles/<profilename>/enabl`

GET `https://<server>:8021/rest/management/profiles/<profilename>/disal`

Example

Request

```
GET https://localhost:8021/rest/management/profiles/easyTravel/enable
```

Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<result value="true"/>
```

System Profile configurations

Profile configuration list

This request retrieves all configurations of the specified System Profile.

GET `https://<server>:8021/rest/management/profiles/<profilename>/conf`

Example

Request

```
GET https://localhost:8021/rest/management/profiles/easyTravel/configurations
```

Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<configurations href="https://localhost:8021/rest/management/profiles/easyTrave
  <configuration isactive="true" id="easyTravel" href="https://localhost:8021/
  <configuration isactive="false" id="easyTravel light" href="https://localhost
</configurations>
```

The returned XML contains a `configurations` root node with `configuration` sub-nodes that contain an `id` and `href` attribute. The `id` refers to the configuration name. The `href` is the link to a configuration-specific REST URI.

Activate profile configuration

A System Profile can have several configurations, but only one *active* configuration. Use this request to activate a System Profile configuration, so that all others are set to inactive:

GET `https://<server>:8021/rest/management/profiles/<profilename>/conf`

Example

Request

```
GET https://localhost:8021/rest/management/profiles/easyTravel/configurations/I
```

Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<result value="true"/>
```

Tasks and monitors (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Tasks

Monitors

Execution of tasks and monitors can be triggered by REST services. Some REST services also provide information if execution of specific tasks and monitors is currently ongoing.

Tasks

Run a task

GET `https://<server>:8021/rest/management/profiles/<profilename>/task`

The resulting XML document contains a `result` root node with a `value` attribute of `true` or `false`. A value of `true` indicates that execution of the specified task was successfully triggered. HTTP response code 404 (Not Found) returns if the task is not available.

Example

The following sample HTTP GET request triggers the execution of task `task1` in the system profile easyTravel:

```
GET https://localhost:8021/rest/management/profiles/easyTravel/tasks/task1/run
```

Check running state

```
GET  https://<server>:8021/rest/management/profiles/<profilename>/task
```

The resulting XML document contains a `result` root node with a `value` attribute of `true` or `false`, which indicates whether execution of the specified task or monitor is currently ongoing. HTTP response code 404 (Not Found) is returned if the task is not available.

Monitors

Similar to the interfaces running a Task and check its running state, also Monitorscan be ran and checked.

Run a monitor

```
GET  https://<server>:8021/rest/management/profiles/<profilename>/moni
```

Check running state

```
GET  https://<server>:8021/rest/management/profiles/<profilename>/moni
```

Test automation (REST)

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Test automation explained

Test Automation REST API endpoints allow the following operations on test automation data:

- Create a test run.
- Read a test run with its test results. Various filters can be applied.
- Update a test run state, such as finish, invalidate, or mark as failed.
- Add test results to an external test run.

Detailed REST API descriptions, examples, and interactive testing capability can be found on your local AppMon Server at the following location:

```
https://<server>:8021/api-docs/current/index.html#/Test_Automation
```

Test Automation

Show/Hide | List Operations | Expand Operations

GET	/profiles/{profileid}/testruns	List test runs by System Profile
POST	/profiles/{profileid}/testruns	Register test run
POST	/profiles/{profileid}/testruns/search	Search and filter test runs by System Profile
GET	/profiles/{profileid}/testruns/{testrunid}	Get test run by id
POST	/profiles/{profileid}/testruns/{testrunid}	Post test result

Implementation Notes

Post a test result for a registered test run of category 'external'.

Parameters

Parameter	Value	Description	Parameter Type	Data Type
profileid	(required)	**System profile id**	path	string
testrunid	(required)	**UUID of the test run**	path	string
body	(required)	**Test result**	body	Model | Example Value

Parameter content type: application/json ▾

```
{
  "testName": "string",
  "timestamp": "2016-05-11T11:35:31.170+02:00",
  "measures": [
    {
      "name": "string",
      "value": 0,
      "timestamp": "2016-05-11T11:35:31.170+02:00",
      "unit": "string",
      "minValue": 0,
      "maxValue": 0,
```

Server REST examples

Available Guides

Alert suppression based on Business Hours (REST)

Monitor creation (REST)

User management REST

On the sub pages you can find guides and examples for logical subsets of the AppMon Server REST API.

Alert suppression based on Business Hours (REST)

On this page

Step 1: Create Business Hours definition

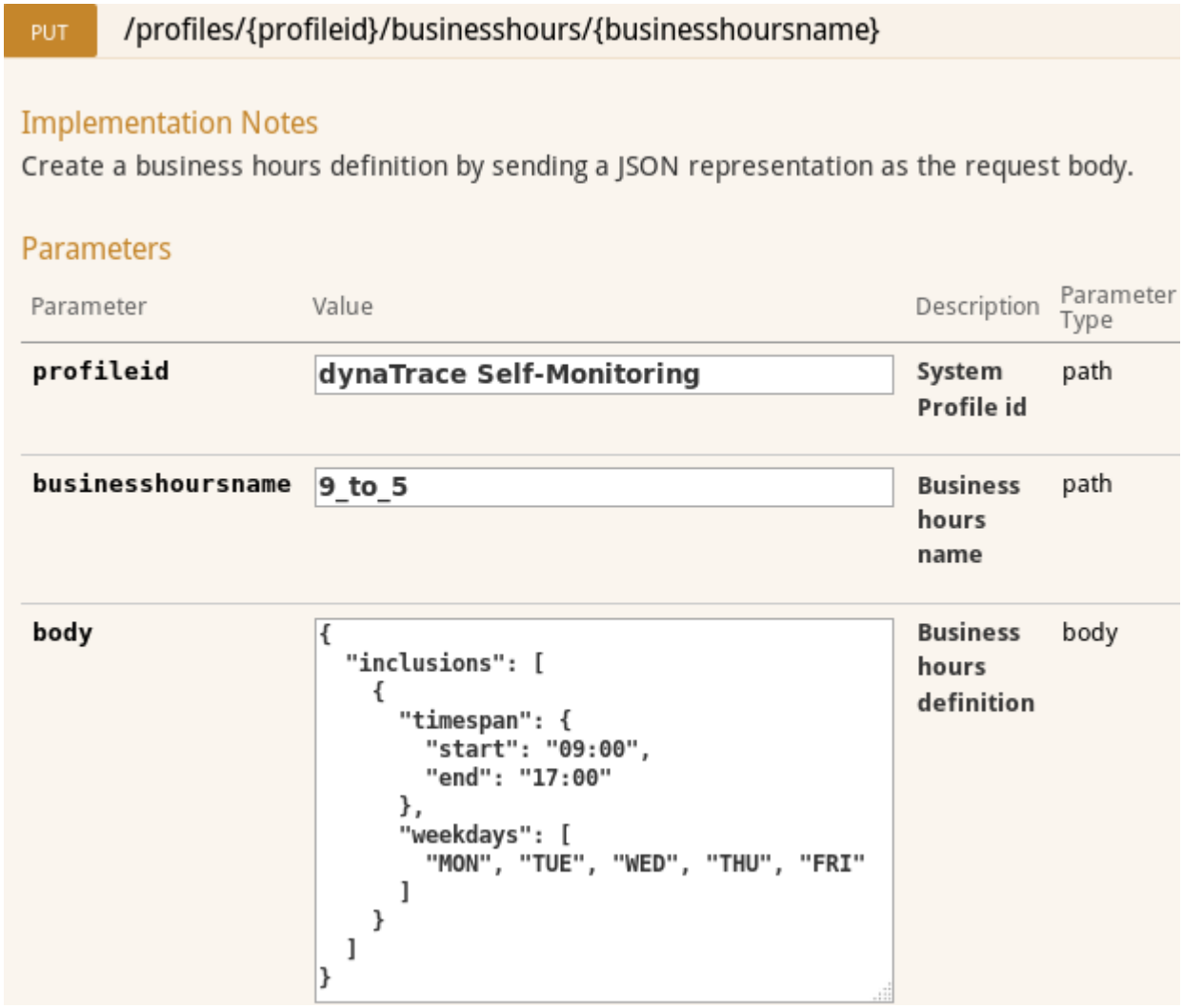
Step 2: Create alert suppression

This is a step-by-step workflow guide for how to create alert suppressions (Incident Downtimes) based on Business Hours. As a prerequisite, you should be familiar with Incident Downtimes and Business Hours.

The individual steps can be tried out interactively in our REST documentation that comes with the AppMon Server at https://DTSERVER:8021/api-docs/current/.

Step 1: Create Business Hours definition

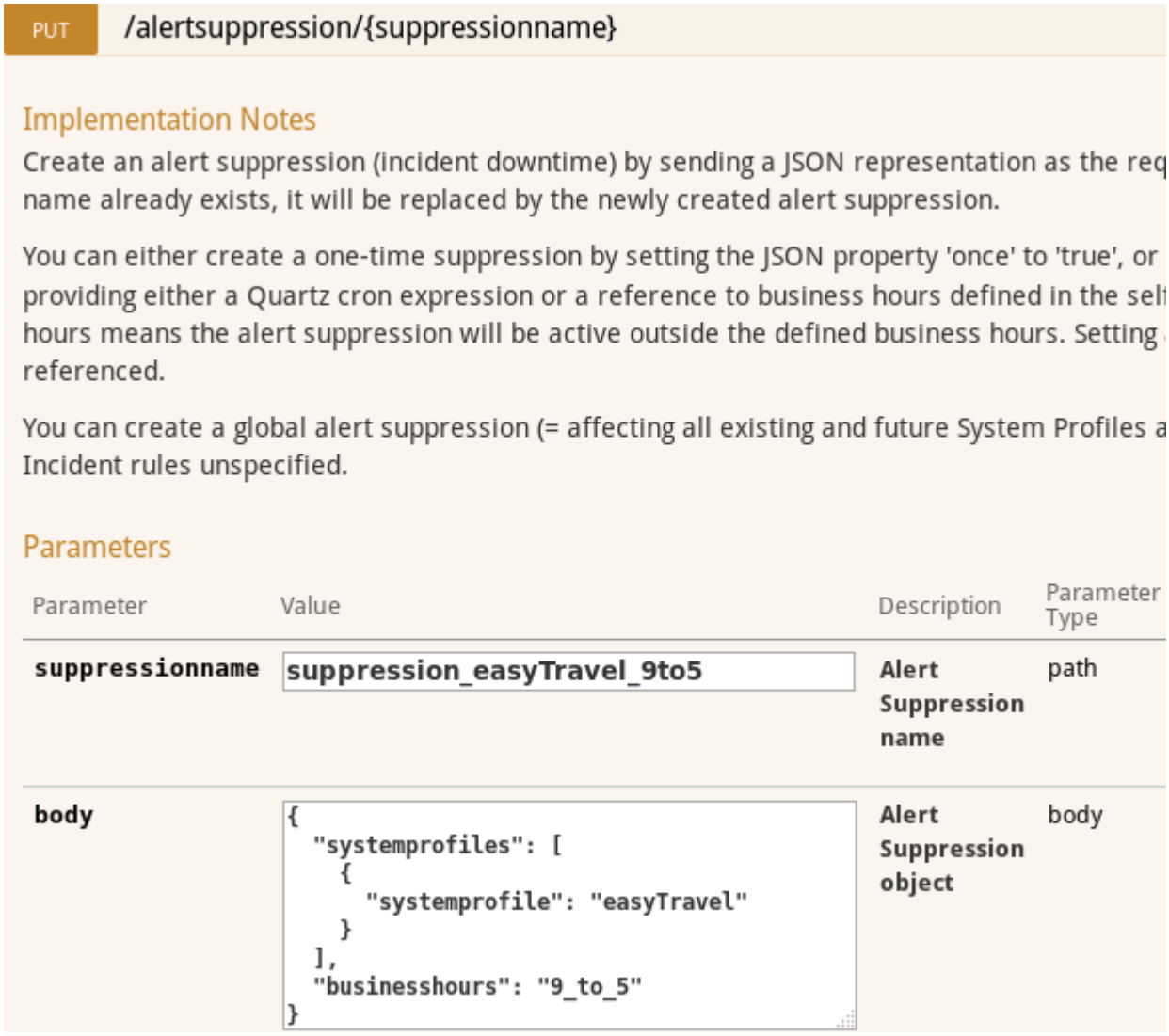
To create an alert suppression based on Business Hours, first create a Business Hours definition in the Dynatrace Self-Monitoring System Profile using the call shown in the following image:



Link to REST documentation for this endpoint

Step 2: Create alert suppression

Create an alert suppression for all Incident Rules of the System Profile ‘easyTravel’ based on the ‘9 to 5’ Business Hours previously created. This suppresses alerts all weekend, and from Monday to Friday 0:00 to 9:00 and 17:00 to 24:00. This can be achieved using the call shown in the following image:



Link to REST documentation for this endpoint

Monitor creation (REST)

On this page

Step 1: Create default Monitor for a specific type

Step 2: View available Monitor task parameters

Step 3: Update task parameters and execution target

Step 4: Combine step 1 and 3 for automation scripts

Step 5: Add target hosts

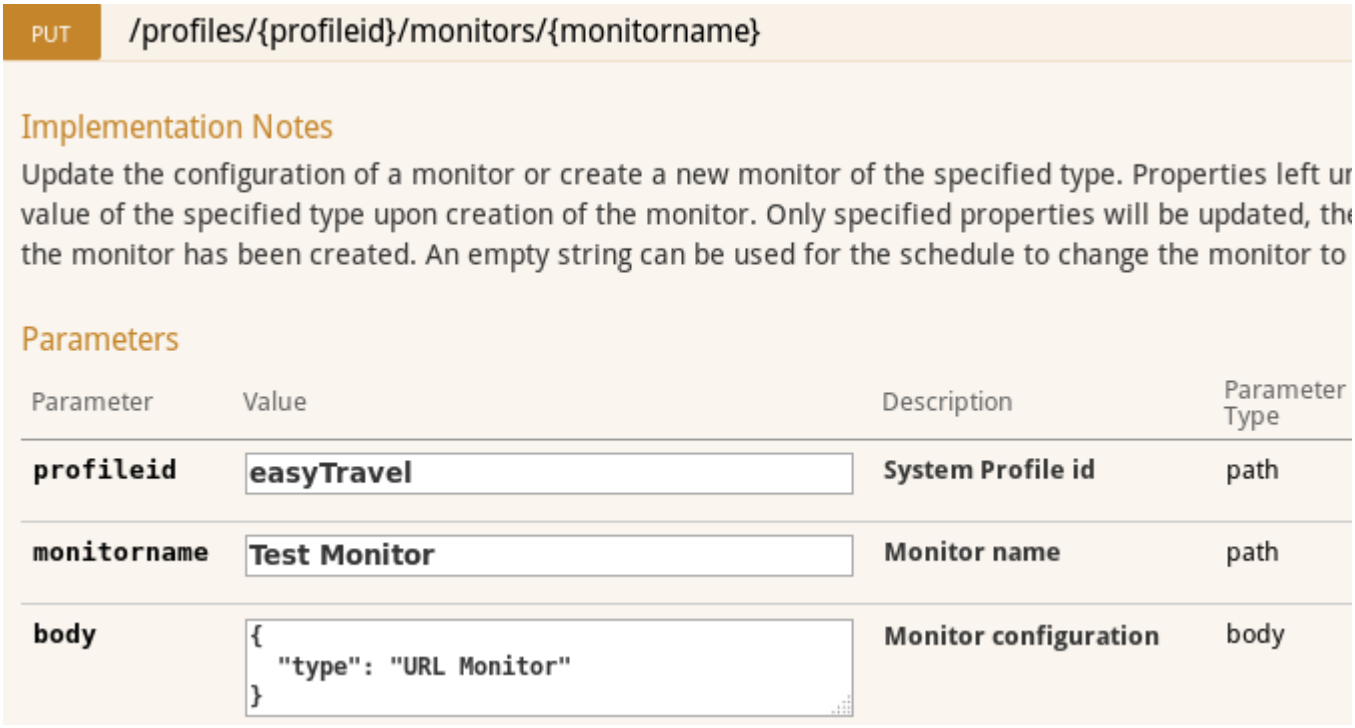
This is a step-by-step workflow guide for how to create Monitors via REST. As a prerequisite, you should be familiar with Monitor configuration in the AppMon client, where you also get an overview over available monitor types, execution targets and schedules during the creation dialog.

The individual steps can be tried out interactively in our REST documentation that comes with the AppMon Server at https://DTSERVER:8021/api-docs/current/.

Steps 1 to 3 are preparatory to make you familiar with the configuration options for a specific Monitor type. In an automation script you would usually only want to use steps 4 and 5.

Step 1: Create default Monitor for a specific type

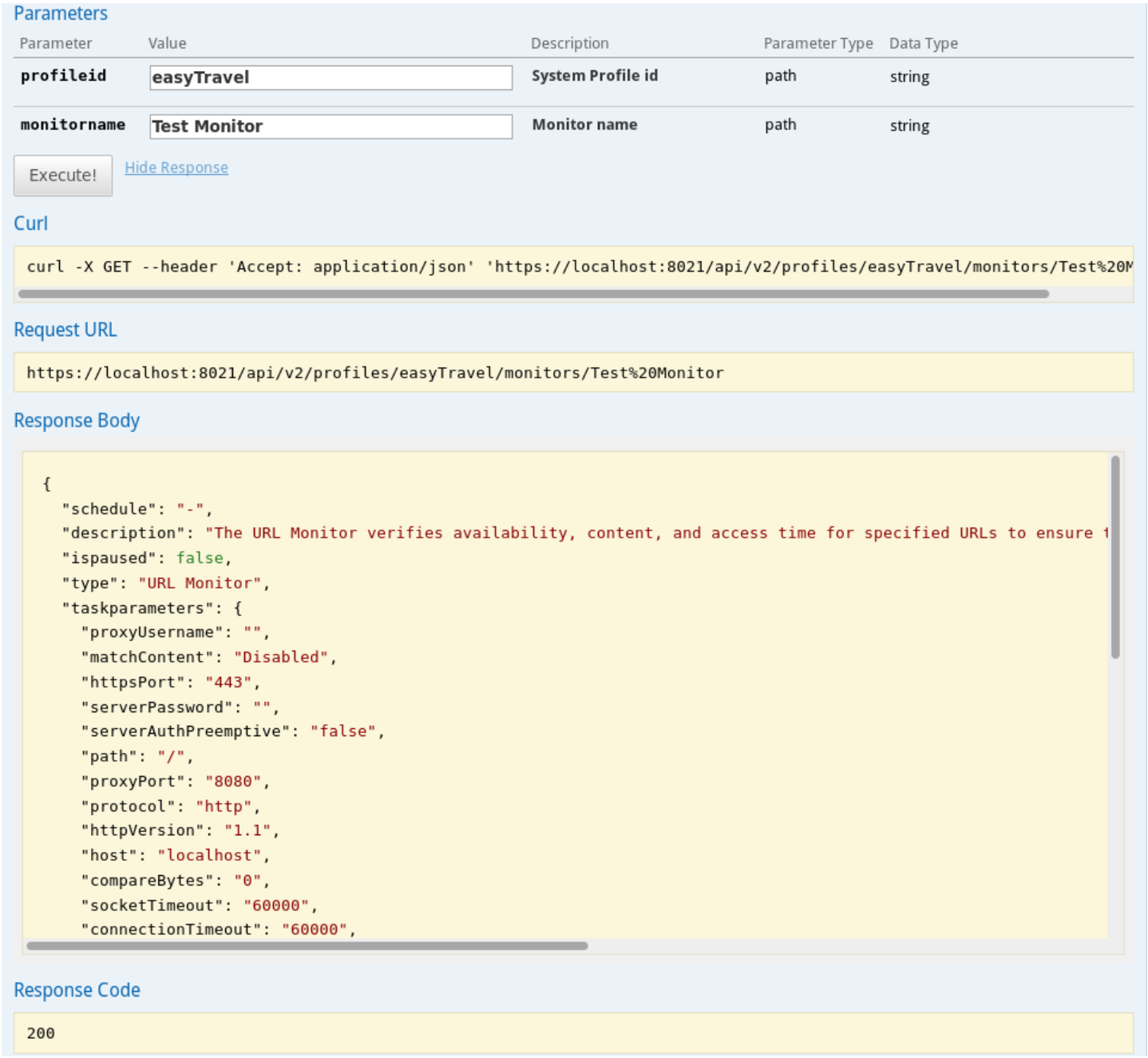
To create a Monitor, first create the default setting for a specific Monitor type (for example `URL Monitor`). The only required JSON field is the `type`. The available Monitor types depend on your installed Monitor plug-ins, try to configure a Monitor through the AppMon client to view the available types in the **Monitor Type Chooser**.



Link to REST documentation for this endpoint

Step 2: View available Monitor task parameters

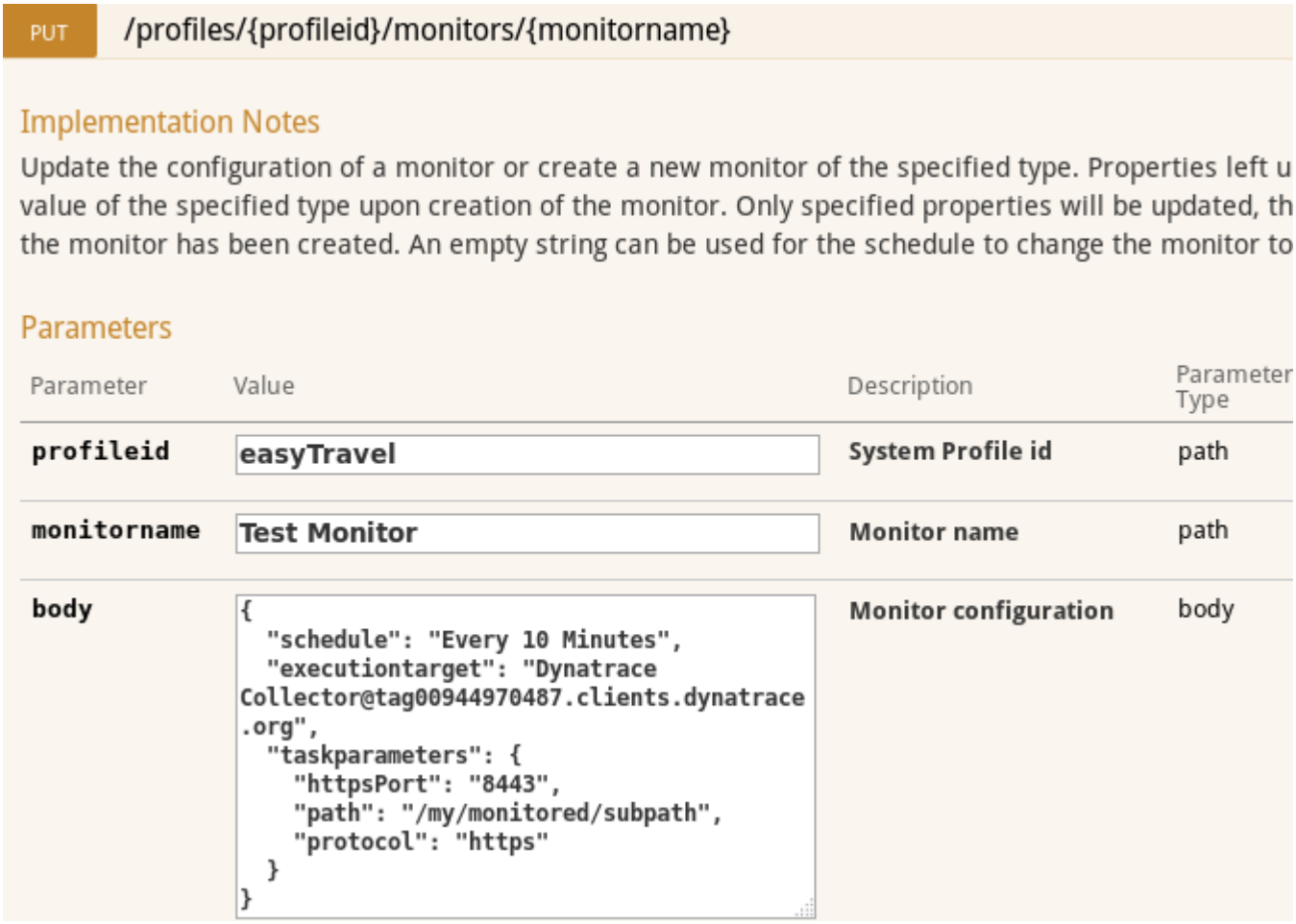
Monitors can have any number of task parameters, which are defined in the respective Monitor plug-in. To get an overview over the supported parameters for the specific Monitor type used, you can query the details of the newly created Monitor. Any properties which need to be further adapted can be updated in the next step.



Link to REST documentation for this endpoint

Step 3: Update task parameters and execution target

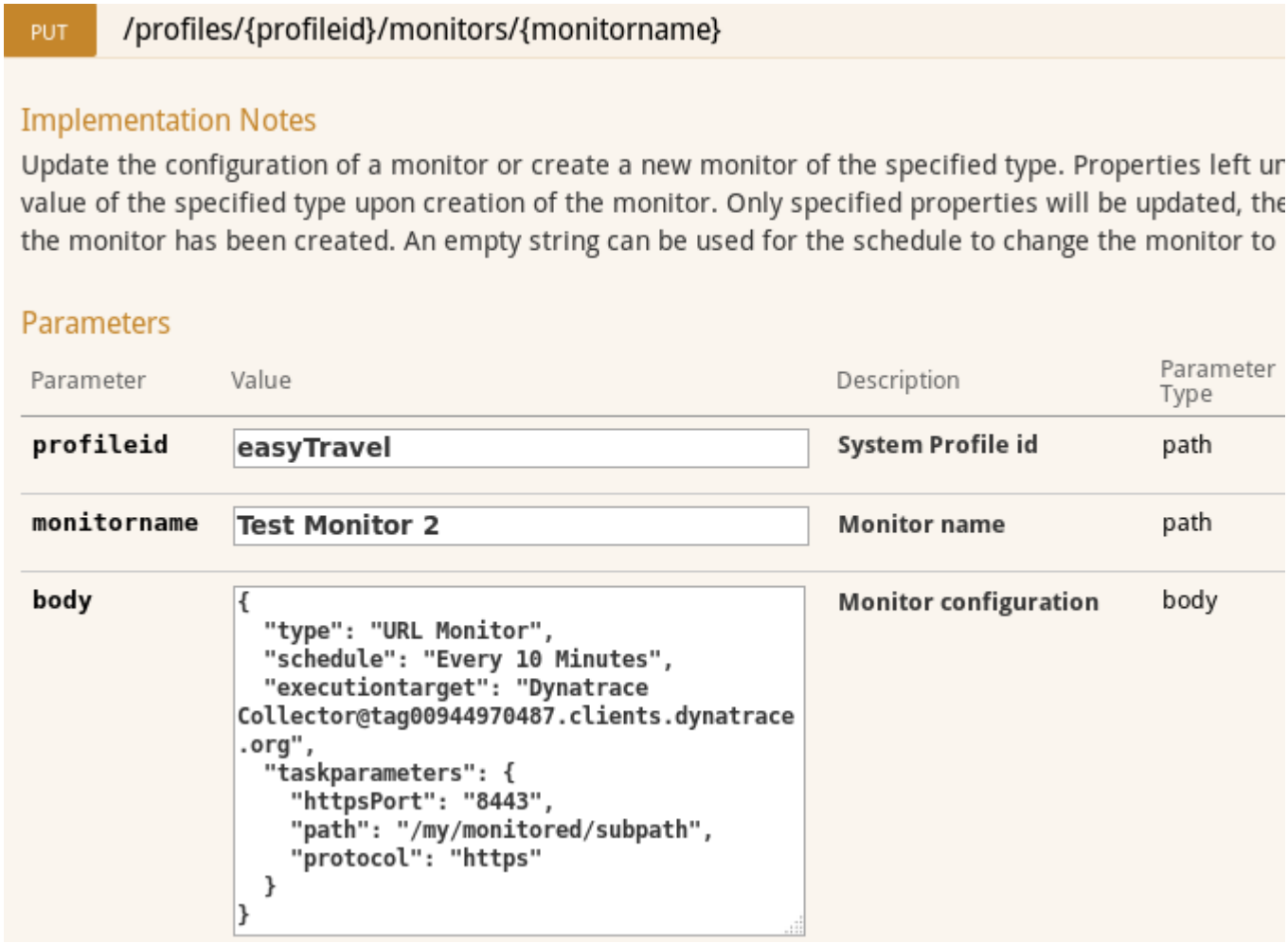
In order to selectively update certain properties of the Monitor, you would use the same endpoint as was used for the creation of the new Monitor. In this example we set the execution target to a specific Collector, schedule the Monitor task to run periodically, and change some of the task parameters. The Collector has to be specified in the format `collectorname@collectorhost` (you can view the available execution targets in the expected format in the AppMon client if you have any doubts - configure a Monitor and switch to the **Schedule** tab). Note that you cannot modify the type of a Monitor.



Link to REST documentation for this endpoint

Step 4: Combine step 1 and 3 for automation scripts

Once you know the available task parameters for a specific Monitor type, you can skip the previous three steps and configure any new Monitor with only one call instead, just use the JSON body from step 3 and add the `type` property from step 1.



Link to REST documentation for this endpoint

Step 5: Add target hosts

Monitors are executed separately for each specified host, therefore at least one host has to be configured for the Monitor to be executed at all. Hosts can be selected via host expressions and intersections of host expressions (for example a combination of different labels). The following call can be used to add a host expressions to the specified Monitor. The host expression is the name of any configured host, site or label (the latter corresponds to the type `hostGroup` in the REST call). The host expression will be evaluated during runtime against the infrastructure configuration, which has to be adapted manually through the AppMon client if a yet unknown host should be monitored.

POST /profiles/{profileid}/monitors/{monitorname}/hosts

Implementation Notes

Add a new host expression to the monitor. If more than one subexpression is defined, a host has to m
the host expression.

Parameters

Parameter	Value	Description	Parameter Type
profileid	**easyTravel**	**System Profile id**	path
monitorname	**Test Monitor**	**Monitor name**	path
body	`{ "hostintersection": [ { "type": "host", "expression": "tag00944970487" } ] }`	**Host expression**	body

Link to REST documentation for this endpoint

User management REST

On this page

Step 1: Query available permissions

Step 2: Create a new role

Step 3: Assign permissions to role

Step 4: Create new user group

Step 5: Add System Profile permissions to user group

Step 6: Add dashboard permissions to user group

Step 7: Add user

Step 8: Add user to user groups

This is a step-by-step workflow guide for how to use the user management REST interface. As a prerequisite, you should be familiar with the user permission management.

The individual steps can be tried out interactively in our REST documentation that comes with the AppMon Server at https://DTSERVER:8021/api-docs/current/. The following overview figure shows where to find the individual endpoints used in each step:

User Management

Show/Hide | List Operations | Expand Operations

GET	/usermanagement/groups	List user groups	4a
DELETE	/usermanagement/groups/{groupid}	Delete user group	
GET	/usermanagement/groups/{groupid}	Get user group	
PUT	/usermanagement/groups/{groupid}	Create or update user group	4
DELETE	/usermanagement/groups/{groupid}/dashboardmappings	Clear dashboard permissions of user group	6a
GET	/usermanagement/groups/{groupid}/dashboardmappings	List dashboard permissions of user group	
POST	/usermanagement/groups/{groupid}/dashboardmappings	Add dashboard permission to user group	6
DELETE	/usermanagement/groups/{groupid}/profilemappings	Clear System Profile permissions of user group	5a
GET	/usermanagement/groups/{groupid}/profilemappings	List System Profile permissions of user group	
POST	/usermanagement/groups/{groupid}/profilemappings	Add System Profile permission to user group	5
GET	/usermanagement/ldap	Get LDAP configuration	
PUT	/usermanagement/ldap	Replace LDAP configuration	
PUT	/usermanagement/ldap/certificate	Update LDAP server certificate	
GET	/usermanagement/ldap/status	Activation status of LDAP configuration	
PUT	/usermanagement/ldap/status	Activate/Deactivate LDAP	
GET	/usermanagement/permissions	List permissions	1
GET	/usermanagement/permissions/{permissionid}	Get permission	
GET	/usermanagement/roles	List roles	2a
DELETE	/usermanagement/roles/{roleid}	Delete role	
GET	/usermanagement/roles/{roleid}	Get role	
PUT	/usermanagement/roles/{roleid}	Create or update role	2
DELETE	/usermanagement/roles/{roleid}/permissions	Clear permissions of role	
GET	/usermanagement/roles/{roleid}/permissions	List role permissions	
POST	/usermanagement/roles/{roleid}/permissions	Add permissions to role	3
PUT	/usermanagement/roles/{roleid}/permissions	Replace permissions of role	
GET	/usermanagement/users	List users	7a
DELETE	/usermanagement/users/{userid}	Delete user	
GET	/usermanagement/users/{userid}	Get user	
PUT	/usermanagement/users/{userid}	Create or update user	7
DELETE	/usermanagement/users/{userid}/groups	Clear group memberships of user	
GET	/usermanagement/users/{userid}/groups	List groups of user	
POST	/usermanagement/users/{userid}/groups	Add user to groups	8
PUT	/usermanagement/users/{userid}/groups	Replace group memberships of user	

Step 1: Query available permissions

Permissions need to be referred to by their respective ID, the call in the following image lists all available permissions:

Curl

```
curl -X GET --header 'Accept: application/json' 'https://localhost:8021/api/v2/usermanagement/permissions'
```

Request URL

```
https://localhost:8021/api/v2/usermanagement/permissions
```

Response Body


```
{
  "permissions": [
    {
      "id": "TestBundle",
      "href": "https://localhost:8021/api/v2/usermanagement/permissions/TestBundle"
    },
    {
      "id": "AnalyzeHeap",
      "href": "https://localhost:8021/api/v2/usermanagement/permissions/AnalyzeHeap"
    },
    {
      "id": "WriteInfrastructureConfig",
      "href": "https://localhost:8021/api/v2/usermanagement/permissions/WriteInfrastructureConfig"
    },
    {
      "id": "ShowInfrastructure",
      "href": "https://localhost:8021/api/v2/usermanagement/permissions/ShowInfrastructure"
    },
    {
```

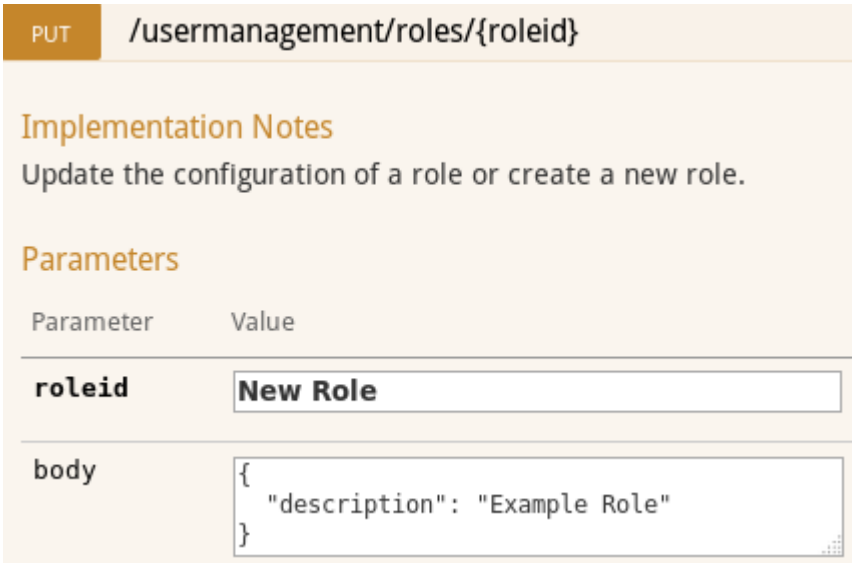

Response Code

```
200
```

Link to REST documentation for this endpoint

Step 2: Create a new role

A new role requires a unique name and optionally a description. Permissions can be assigned with a different call (see step 3).



Link to REST documentation for this endpoint

Step 2a: (Alternatively) Query available roles

To find out the IDs and URLs of already existing roles, they can be listed using the call shown in the following image:

Curl

```
curl -X GET --header 'Accept: application/json' 'https://localhost:8021/api/v2/usermanagement/roles'
```

Request URL

```
https://localhost:8021/api/v2/usermanagement/roles
```

Response Body


```
{
  "roles": [
    {
      "id": "Administrator",
      "href": "https://localhost:8021/api/v2/usermanagement/roles/Administrator"
    },
    {
      "id": "No Permission",
      "href": "https://localhost:8021/api/v2/usermanagement/roles/No%20Permission"
    },
    {
      "id": "PowerUser",
      "href": "https://localhost:8021/api/v2/usermanagement/roles/PowerUser"
    },
    {
      "id": "User",
      "href": "https://localhost:8021/api/v2/usermanagement/roles/User"
    },
    {
      "id": "Guest"
```

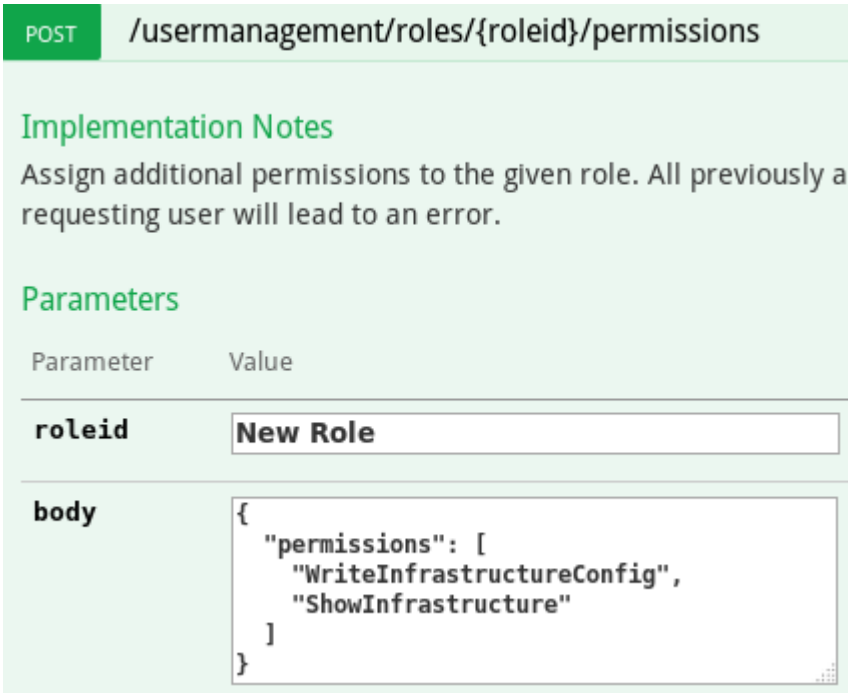

Response Code

```
200
```

Link to REST documentation for this endpoint

Step 3: Assign permissions to role

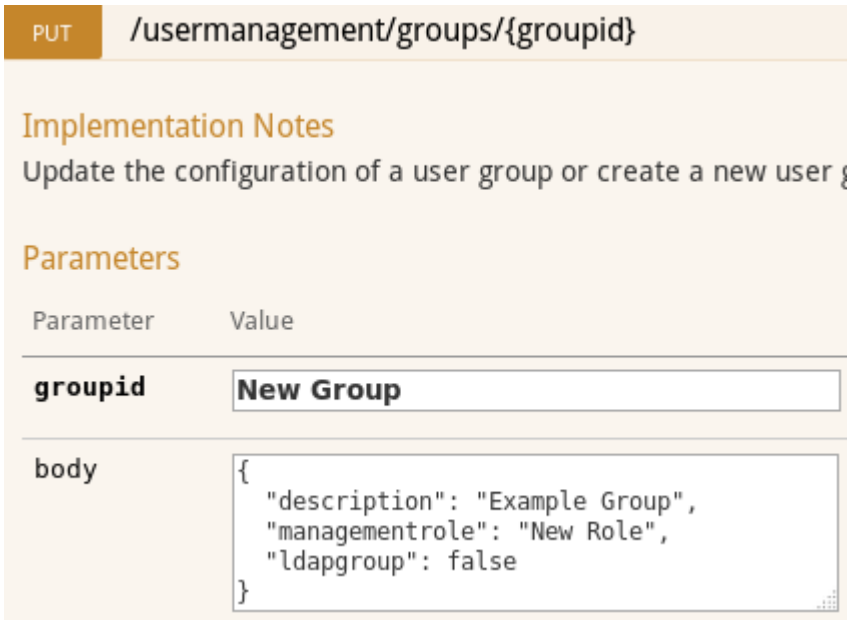
Additional permissions can be assigned to a role by sending their respective IDs via the call shown in the following image:



Link to REST documentation for this endpoint

Step 4: Create new user group

A new user group can be assigned a management role already at creation time by referring to the role ID. Additional permissions for System Profiles and dashboards can be granted with different calls (see steps 5 and 6).



Link to REST documentation for this endpoint

Step 4a: (Alternatively) Query available user groups

List the IDs and URLs of already existing user groups using the call shown in the following image:

Curl

```
curl -X GET --header 'Accept: application/json' 'https://localhost:8021/api/v2/usermanagement/groups'
```

Request URL

```
https://localhost:8021/api/v2/usermanagement/groups
```

Response Body


```
{
  "groups": [
    {
      "id": "New Group",
      "href": "https://localhost:8021/api/v2/usermanagement/groups/New%20Group"
    },
    {
      "id": "Business Users",
      "href": "https://localhost:8021/api/v2/usermanagement/groups/Business%20Users"
    },
    {
      "id": "Test",
      "href": "https://localhost:8021/api/v2/usermanagement/groups/Test"
    },
    {
      "id": "Incident Email Group",
      "href": "https://localhost:8021/api/v2/usermanagement/groups/Incident%20Email%20Group"
    },
    {
      "id": "Personal System Profile Access"
```

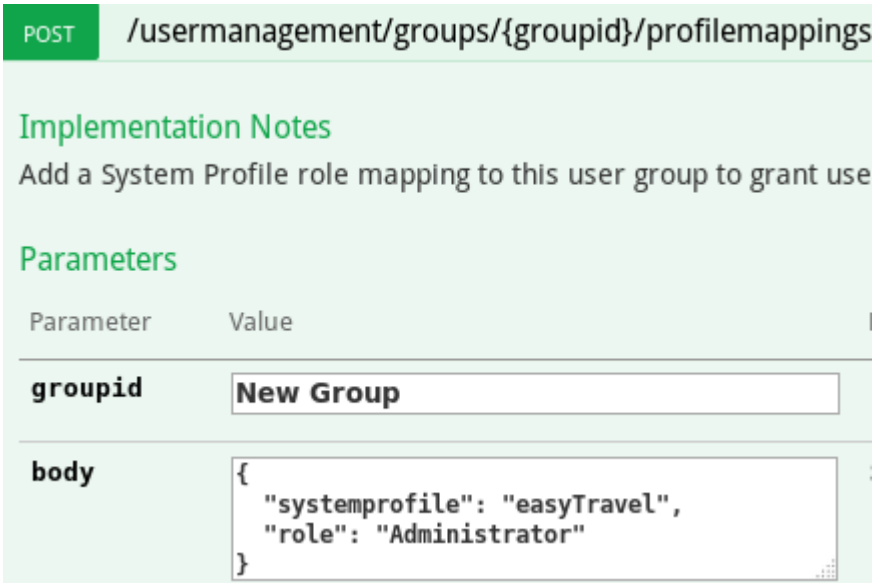

Response Code

```
200
```

Link to REST documentation for this endpoint

Step 5: Add System Profile permissions to user group

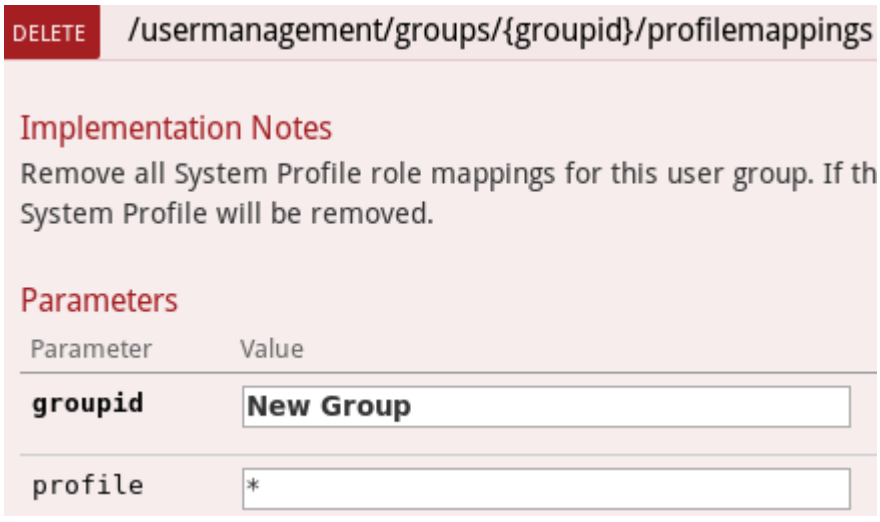
By default, a newly created user group grants the permissions of the Guest role to all System Profiles via a wildcard entry. To override this permission for an individual System Profile, a different role can be assigned using the call shown in the following image:



Link to REST documentation for this endpoint

Step 5a: Delete System Profile permissions

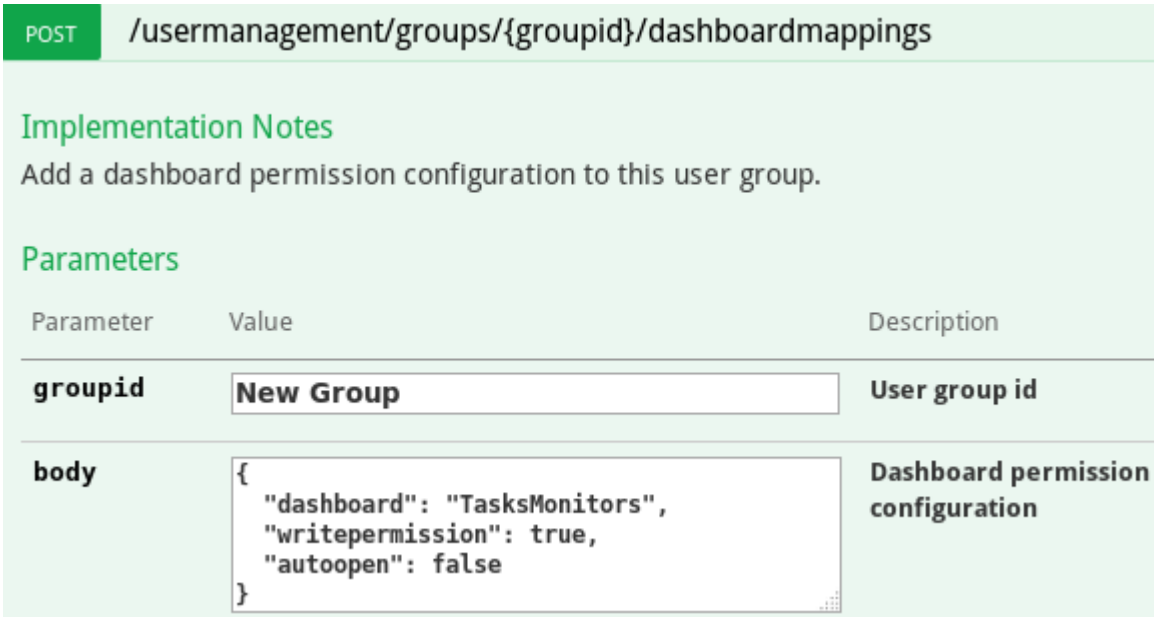
The default wildcard entry can be deleted using the following call in order to revoke all permissions from System Profiles not explicitly added in step 5. This call can also be used to delete the role assignments for other System Profiles.



Link to REST documentation for this endpoint

Step 6: Add dashboard permissions to user group

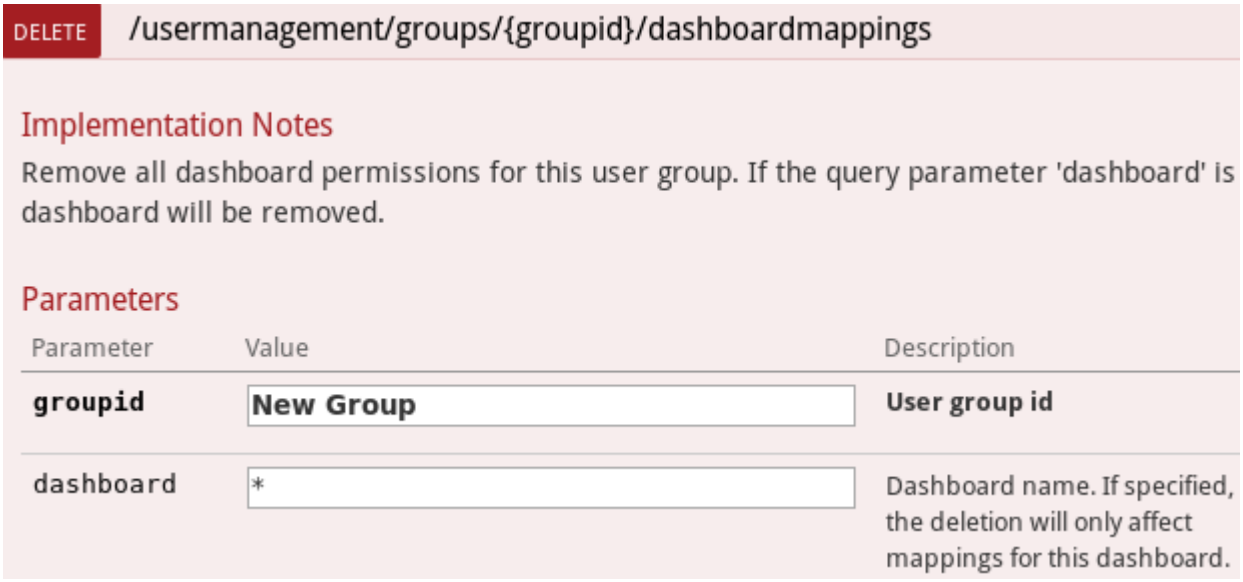
By default, a newly created user group grants reading permissions to all dashboards via a wildcard entry. To customize permissions (e.g. add writing permissions) for a specific dashboard, use the call shown in the following image:



Link to REST documentation for this endpoint

Step 6a: Delete dashboard permissions

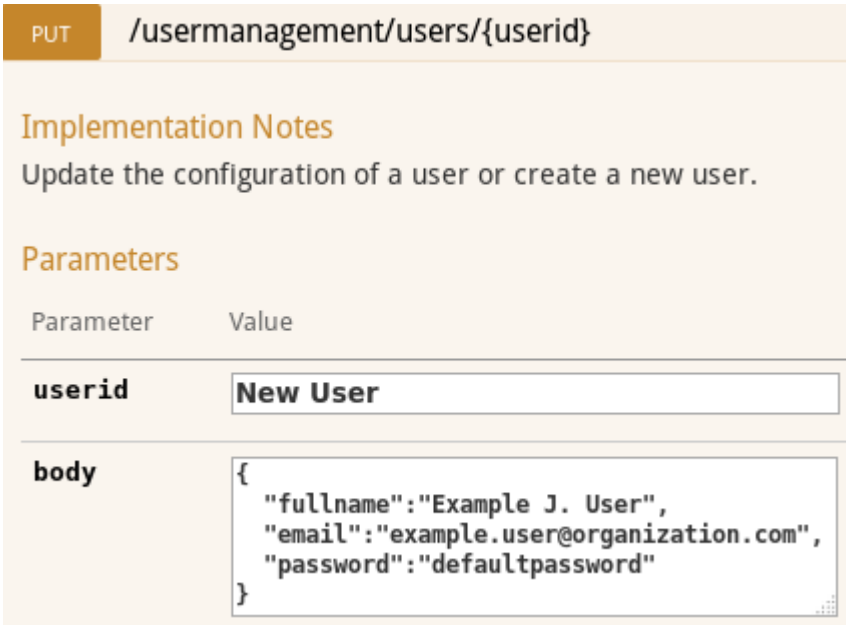
The default wildcard entry can be deleted using the following call in order to revoke even reading permissions from dashboards not explicitly added in step 6. This call can also be used to delete the permission configuration of other dashboards.



Link to REST documentation for this endpoint

Step 7: Add user

In order to add a user, an initial password needs to be set using the following call. User groups can later be assigned with a different call (see step 8).



Link to REST documentation for this endpoint

Step 7a: (Alternatively) Query available users

List the IDs and URLs of already existing users using the call shown in the following image:

Curl

```
curl -X GET --header 'Accept: application/json' 'https://localhost:8021/api/v2/usermanagement/users'
```

Request URL

```
https://localhost:8021/api/v2/usermanagement/users
```

Response Body


```
{
  "users": [
    {
      "id": "New User1",
      "href": "https://localhost:8021/api/v2/usermanagement/users/New%20User1"
    },
    {
      "id": "business",
      "href": "https://localhost:8021/api/v2/usermanagement/users/business"
    },
    {
      "id": "test",
      "href": "https://localhost:8021/api/v2/usermanagement/users/test"
    },
    {
      "id": "New User",
      "href": "https://localhost:8021/api/v2/usermanagement/users/New%20User"
    },
    {
      "id": "admin"
```

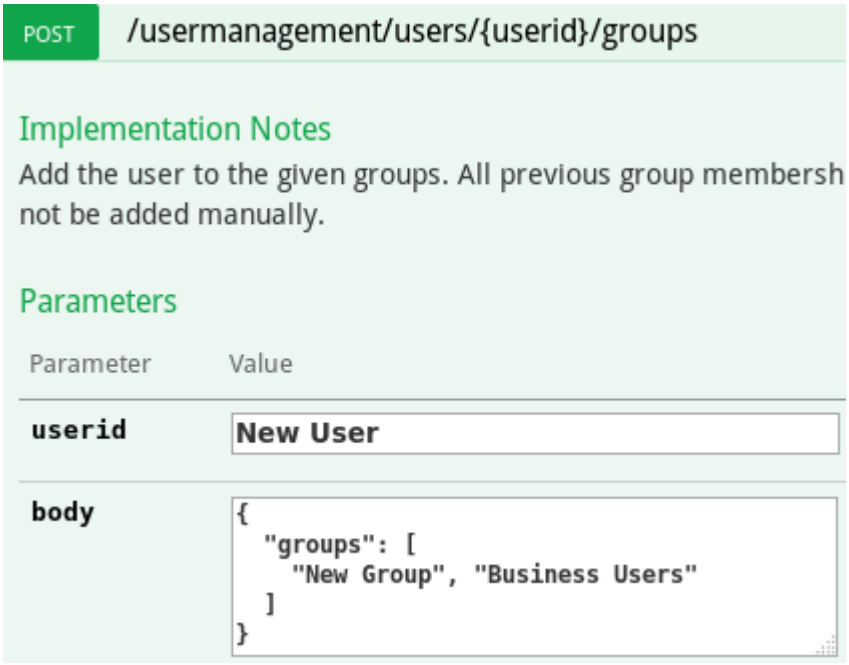

Response Code

```
200
```

Link to REST documentation for this endpoint

Step 8: Add user to user groups

A user can be added to existing user groups by referring to their respective IDs in the call shown in the following image:



Link to REST documentation for this endpoint

Documentation downloads

- Download deliverables are manually generated periodically and may not include the latest updates. To ensure you are viewing the most current documentation, use the online documentation.
- The HTML download must be unzipped to work. The start page is `support/doc/appmon/index.html`.

Download notes

Documentation Deliverable	Download Format
Complete 7.0 (May 2017) Documentation Set (v. 20170623)	

For offline access to the documentation, download a documentation deliverable listed below.

Keyboard shortcuts

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Windows & Linux shortcuts

Mac shortcuts

If a shortcut does not work, it may be overridden by a system wide shortcut, such as those used for virtual desktops.

Windows & Linux shortcuts

Standard keys

Modifiers	Key	Function
Alt	F4	Close application
	F5	Refresh all active dashlets
Ctrl	F5	Refresh dashlet
	F2	Edit dashlet name and description or Freeze tooltip
	F11	Toggle full screen mode

Ctrl	C	Copy
Ctrl	V	Paste
	Enter	OK button default action in dialog box
	Return	OK button default action in dialog box
	Del	Delete
	Space	Toggle (visibility, placement)
Ctrl+Shift	X	Toggle sidebar

Dashboard

Modifiers	Key	Function
Ctrl	Tab	Activate next dashlet in active folder
Ctrl+Shift	Tab	Activate previous dashlet in active folder
Ctrl Ctrl	F4 W Middle mouse click	Close dashlet
Ctrl+Shift Ctrl+Shift	F4 W	Close dashboard
Ctrl	N	New dashboard
Ctrl	O	Open dashboard
Ctrl	F6	Activate next dashboard

Ctrl+Shift	F6	Activate previous dashboard
Ctrl	T	New dashlet
	F6	Activate next folder
Shift	F6	Activate previous folder
Ctrl+Shift	H	Toggle headers (for all folders)

Active dashlet

Modifiers	Key	Function
	Right Mouse	Context menu
Shift	Right Mouse	Override default drilldown target within drilldown submenu
Alt	Return	Properties
Ctrl	H	Toggle header
Ctrl+Shift	E	Edit filter*
Ctrl	P	Report*

- only for reportable/filterable dashlets

Chart dashlet

Modifiers	Key	Function

Ctrl	Insert	Add series, measures, etc.
Ctrl	E	Edit series, measures, etc. (for live sessions only)
Ctrl	Arrow Up/Down	Change drawing order of selected series. Only in the **Series** configuration tab of the dashlets' **Properties** dialog boxes.
Ctrl+Shift	T	Show threshold
Ctrl+Shift	D	Show dynamic measures
Ctrl+Shift	C	Configure chart series
Ctrl+Shift	#	Toggle table visibility
Ctrl+Shift	+	Zoom into time frame
Ctrl+Shift	-	Zoom out time frame
	Space (Windows) or Shift+V (Linux)	Toggle series visibility

Window, control and tab navigation

Modifiers	Key	Function
	Tab	Move focus to next control
Shift	Tab	Move focus to last control
Ctrl	A	Mark all elements
	Home	First element in list

	End	Last element in list

Sortable lists

Modifiers	Key	Function
Ctrl	Up	Move current line up
Ctrl	Down	Move current line down

Tree control navigation

Modifiers	Key	Function
Ctrl	B	Adjusts column width to minimum
	NUMPAD *	Expands the subtree of current node
	NUMPAD + Right Arrow	Expands the current node (Linux: Shift + right arrow)
	NUMPAD - Left Arrow	Collapses the current node (Linux: Shift + left arrow)
	Up Arrow	Move one line up
	Down Arrow	Move one line down
	Home	First element in list
	End	Last element in list

Mac shortcuts

If the function keys on your Mac trigger special features like system volume or display brightness, do one of the following:

- Use the function keys together with the *fn* key.
- Change the function key behavior in the **Keyboard & Mouse Preference** panel in the System Preferences of your Mac.

Standard keys

Modifiers	Key	Function
Command	Q	Close application
Command	R	Refresh all dashlets
Command+ Shift	R	Refresh dashlet
	F2	Edit dashlet name and description or Freeze tooltip
Command+ Shift	F	Toggle fullscreen mode
Command	C	Copy
Command	V	Paste
	Enter	OK button default action in dialog box
	Return	OK button default action in dialog box
	Del	Delete
	Space	Toggle (visibility, placement)
Command+ Shift	X	Toggle sidebar

Dashboard

Modifiers	Key	Function
Ctrl	Tab	Activate next dashlet in active folder
Ctrl+Shift	Tab	Activate previous dashlet in active Folder
Command	W Middle mouse click	Close dashlet
Command+Shift	W	Close dashboard
Command	F6	Activate next dashboard
Command+Shift	F6	Activate previous dashboard
Command	N	New dashboard
Command	O	Open dashboard
Command	T	New dashlet
	F6	Activate next folder
Shift	F6	Activate previous folder
Ctrl+Shift	H	Toggle headers (for all folders)

Active dashlet

Modifiers	Key	Function
	Right Mouse	Context menu

Shift	Right Mouse	Override default drilldown target within drilldown sub menu
Alt	Return	Properties
Ctrl	H	Toggle header
Command+Shift	E	Edit filter*
Command	P	Report*

- only for reportable/filterable dashlets

Chart dashlet

Modifiers	Key	Function
Command	Insert	Add series, measures, etc.
Command	E	Edit series, measures, etc. (for live sessions only)
Command+Shift	T	Show threshold
Command+Shift	D	Show dynamic measures
Command+Shift	C	Configure chart series
Command+Shift	#	Toggle table visibility
Command+Shift	+	Zoom into time frame
Command+Shift	-	Zoom out time frame

	Space	Toggle series visibility

Window, control and tab navigation

Modifiers	Key	Function
	Tab	Move focus to next control
Shift	Tab	Move focus to last control
Command	A	Mark all elements
Alt	Up	First element in list
Alt	Down	Last element in list
	Home	Scroll to top
	End	Scroll to bottom

Sortable lists

Modifiers	Key	Function
Ctrl	Up	Move current line up
Ctrl	Down	Move current line down

Tree control navigation

Modifiers	Key	Function
Command	B	Adjusts column width to minimum

	Right Arrow	Expands the current node
	Left Arrow	Collapses the current node
	Up Arrow	Move one line up
	Down Arrow	Move one line down
Alt	Up	First element in list
Alt	Down	Last element in list
	Home	Scroll to top
	End	Scroll to bottom

Glossary

A visit is a group of user actions performed by the same user within a certain time period. It is created with the first user action and ends after the user is inactive for a specified period of time. See System Profile > User Experience for more information. This user inactivity time setting defaults to 30 minutes. You can define it on an application-level (System Profile - Applications). A visit represents the user's click-path and helps to analyze problems. A visit duration is calculated from the start of the first user action to the end of the last user action.

Visit

V

A user action's duration is called *response time*. It represents the time the user waits until it is possible to continue with the use case. A low response time indicates better performance than a high one.

A user action contains all client activity during the transition from one state to another. Every user action contains at least one web request. For example, when a user navigates to a page, the browser downloads many resources and calls the JavaScript `onload` handler, which in turn might perform XHR requests, or other actions.

User action

U

Splitting can be used to segment business transactions according to splitting criteria. By default, all newly created business transactions are split by one criterion: application.

Transaction splitting

The **Transaction Flow** dashlet graphically represents the tiers and backends of a monitored application environment as it is modeled by AppMon Agent Groups and AppMon Agent mappings defined in a System Profile.

Transaction Flow

AppMon provides built-in *tasks* to automate repeating tasks such as generating reports or storing sessions.

Task

T

The *System under Diagnosis* is the application where you inserted the AppMon Agent.

System under diagnosis (SUD)

This page describes how to set up a System Profile, which is the “workspace” where you administer a certain application environment.

System Profile

The shallow size of an object is the flat size, without children. The shallow size is defined by the sizes of all fields defined within the corresponding class and all fields defined in super-classes. It does not include the size of referenced children objects.

Shallow size

- **PurePath Session**: Combined information about PurePaths and Time Series within a given time period.
- **Memory Dump**: Analysis data about the number, size, and class of allocated objects and their references.
- **Thread Dump**: Collection of data for thread analysis, including CPU time information.
- **Sampling**: Statistical data for all threads, which can help to find Rules and entry points.

Examples of a session are:

A *session* is a set of diagnosis data related to a specific System Profile.

Session

S

PureStack Technology® directly correlates system infrastructure health data from every transaction tier in a monitored application environment with individual transactions and affected end users in real-time. PureStack is the vertical view of infrastructure in a monitored application environment and is the basis for *bottom up analysis*, which is defined by analyzing problems in the infrastructure and assessing and correlating their impact on application performance and end-user experience.

PureStack technology

PurePath Technology® is a complete multi-dimensional trace of every transaction in a monitored application providing comprehensive visibility across web browsers, web apps, mobile apps, web servers, Java, .NET, PHP, databases, middleware, and mainframe apps.

PurePath technology

PureModel Technology® combines the PurePath and PureStack views to provide the complete context of business application performance and end-user impact.

PureModel technology

PureLytics is the AppMon **big data analytics engine** that streamlines data analysis by pulling relevant information from millions of PurePath and PureStack data points in real time to enable multi-dimensional analytics.

PureLytics

P

Data retrieved by Monitors are stored as Measures, which can then be combined and correlated to metrics captured from within the application. For example, transaction response times, number of exceptions and executed database statements.

A *Monitor* is scheduled for repetitive execution on a Collector to deliver additional performance metrics to the AppMon Server.
Typical examples are windows performance counters that are not specific to a process but host-wide, and SNMP data sources.

Monitor

Memory snapshot post-processing extracts optimization classes, instances, and references from the raw data and performs calculations on the raw data to produce data for analysis.

Memory snapshot post-processing

Measures in AppMon are metrics, or data points, that are collected and recorded in the AppMon Performance Warehouse either periodically or based on transactions. Measures are used for long-term

charting, trend analysis, and as the basis for configuring incidents.

Measure

M

A *(logical) application* lets you manage more than one application within a single System Profile. Monitor and manage transaction performance for separate applications, or make a logical separation automatically by host / web application ID, or manually by using entities such as URI pattern or PurePath entry point.

Logical application

L

The **Keep Alive Set** dashlet is the set of instances that are “held” by an object instance. If the corresponding instance is garbage-collected, all instances held by this object can also be garbage-collected.

Keep Alive Set Dashlet

The Keep Alive graph shows how an instance is kept alive on the heap.

Keep Alive Graph

K

- Network latency if the agents are physically separated
- Time added by other tiers that are between the measure points but are not instrumented
- Messaging queues

In the *Transaction Flow* visualization, **Inter Tier Time** is the time between the last measure point in a PurePath on one agent and the first measure point in the same PurePath on another agent. This measurement may include the following factors:

Inter tier time

Instrumentation in AppMon is when AppMon injects bytecode into applications to make it possible to monitor a PurePath from end to end.

Instrumentation

An *incident rule* in AppMon is a mapping between measured thresholds and the actions to be taken when such thresholds are violated. If all the thresholds defined in the incident rule are violated, this is an *incident* , even if no actions have been configured for the incident rule.

Incident

I

Hot Sensor Placement is a technology used to commit changes to an application’s instrumentation during run time without the need to restart the application or application server.

Hot Sensor placement

A *host* in AppMon is a system under analyis. A host is instrumented so that that measures can be monitored and reported on when thresholds ae exceeded.

Host

H

The *Garbage Collection size* (GC size) of an instance is the amount of memory that is freed if the corresponding instance is garbage collected.

Garbage collection size

G

An *error* in AppMon can be any kind of unexpected event that happens while a user interacts with a system. Some examples of errors that can occur are resources that cannot be found on a web server, HTTP 500 response codes sent by a web service, or exceptions that occur within an application.

Error

The starting point of a PurePath is its entry point. It is the start of the method call chain that makes up the PurePath.

Entry point

E

It originally refereed to the environment variable that is **optional now** and can be used for convenience and then should point to the AppMon installation root, e.g. C:\Program Files\dynaTrace\dynaTrace 7.0 for a default 64 bit AppMon installation on Windows.

DT_HOME refers to the AppMon installation directory.

DT_HOME

The *discovery run* is the first run of an application after the agent injection that covers all or most of the application functions. AppMon requires it to be aware of all classes that the application uses, for further analysis and instrumentation.

Discovery run

D

The AppMon Collector performs bytecode modifications when instrumented applications start up. AppMon Agents therefore need to send the bytecode of your applications to the AppMon Collector. To keep start-up times low, it is necessary to deploy at least one AppMon Collector in every location/data center that is connected to an AppMon Server.

Bytecode injection

A *business transaction* is a set of automated interactions between IT systems that execute a business task or business process. In AppMon, a business transaction is a categorization of PurePaths defined by filter or grouping parameters. When a PurePath matches those parameters, one or more evaluation metrics are calculated to create the result set of the business transaction.

Business Transaction

AppMon provides automatic, “smart” baselines. The smart alerting mechanism reduces alerts on false positives, which can occur with traditional statistical approaches that look only at averages and violations based on standard deviation.

Baseline

R

B

Auto Sensors (aka extended PurePath information) automatically retrieve information (transactional snapshots) provided by PurePath technology without explicitly instrumenting methods. It includes class and method names, parameter and return types, access modifiers, and call hierarchy information. Auto Sensors complement AppMon out-of-the-box Sensors which explicitly instrument methods to capture context information for web requests, database statements or exceptions, and other information. See Auto Sensors on the Sensors page for more information.

Auto Sensor

Use Agent Groups to logically group certain Agents for common configuration and easier maintenance. Examples are separate groups for front-end and back-end servers. For more information, see Usage of Agent Groups.

Agent Group

An Agent is a software that is either running on the host or injected into the application process. It gathers performance data and sends them to the AppMon server.

Agent

An active visit is a visit in progress, which means that the representative user is actively using your application. If the user has left your application or is inactive for a certain time, the visit is finished.

Active visit

A

*NIX is used to refer to all Unix-like systems. Here, they typically include Linux, Solaris, AIX, HP-UX and Mac OS X.

*NIX

*

Learn about the most common AppMon concepts and features. For the full definition, click the term name in the list below.

AppMon Vocabulary

AppMon Vocabulary

Child glossary pages should be deleted with terse definitions added to the parent Glossary parent page.

Measure

You can also create custom Agent measures based on an out-of-the-box measure type and specify settings that are specific to this new measure. You can then use these measures for charting and split the measure data by agent.

- Agent-based measures such as Host and Agent.
- Monitor-based measures for hardware, operating systems, servers, and databases.

The following types of measures provide insight into your environment and infrastructure:

PurePath and transaction-based measures provide insight into your application and transactions. *Measures* in AppMon are metrics, or data points, that are collected and recorded in the AppMon Performance Warehouse either periodically or based on transactions. Measures are used for long-term charting, trend analysis, and as the basis for configuring incidents.

Agent Group

Use Agent Groups to logically group certain Agents for common configuration and easier maintenance. Examples are separate groups for front-end and back-end servers. For more information, see Usage of Agent Groups.

Auto Sensor

Auto Sensors (aka extended PurePath information) automatically retrieve information (transactional snapshots) provided by PurePath technology without explicitly instrumenting methods. It includes class and method names, parameter and return types, access modifiers, and call hierarchy information. Auto Sensors complement AppMon out-of-the-box Sensors which explicitly instrument methods to capture context information for web requests, database statements or exceptions, and other information. See Auto Sensors on the Sensors page for more information.

Baseline

- For every splitting value or business transaction that is selected for the Application Overview.
- For Response Time (median and slowest 10%), Failure Rate, and Throughput.

A baseline is calculated:
AppMon provides automatic, “smart” baselines. The smart alerting mechanism reduces alerts on false positives, which can occur with traditional statistical approaches that look only at averages and violations based on standard deviation.

Business Transaction

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Business Transactions

Glossary

Watch and learn

What are Business Transactions?

Business Transaction Management Webinar

Creating server-side Business Transactions

Creating User Action Business Transactions

Creating visit-based Business Transactions

A *business transaction* is a set of automated interactions between IT systems that execute a business task or business process. In AppMon, a business transaction is a categorization of PurePaths defined by filter or grouping parameters. When a PurePath matches those parameters, one or more evaluation metrics are calculated to create the result set of the business transaction.

Bytecode injection

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Glossary

The AppMon Collector performs bytecode modifications when instrumented applications start up. AppMon Agents therefore need to send the bytecode of your applications to the AppMon Collector. To keep start-up times low, it is necessary to deploy at least one AppMon Collector in every location/data center that is connected to an AppMon Server.

Discovery run

1. AppMon discovers the internal structure of the application, e.g. which classes derive from interfaces like `java.sql.Statement` . This is required to execute the use cases of the monitored application.
2. AppMon places Sensors on the specific classes of the inheritance structure via a restart of the application, because Hot Sensor Placement is not available for all sensors.

The discovery run is necessary for sensors that AppMon defines on interfaces or super classes. It consists of two steps:
The *discovery run* is the first run of an application after the agent injection that covers all or most of the application functions. AppMon requires it to be aware of all classes that the application uses, for further analysis and instrumentation.

DT_HOME

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Glossary

DT_HOME refers to the AppMon installation directory.

It originally refereed to the environment variable that is **optional now** and can be used for convenience and then should point to the AppMon installation root, e.g. C:\Program Files\dynaTrace\dynaTrace 7.0 for a default 64 bit AppMon installation on Windows.

DT_HOME (referenced as %DT_HOME% on Windows or $DT_HOME on *NIX or generically <DT_HOME>) may have played a more important role in earlier versions of AppMon, but is more of a convenience factor now and only referenced as a fall-back in some places.

Fall-back: The bootstrap part of an Agent tries to save the core into a subdirectory `agent/downloads` for the following directories in the following order:

`DT_STORAGE`, if not set: `DT_HOME`, if not set: `DTD_HOME` (precursor of `DT_HOME`), if not set: path of the Agent, `tmp`, current directory

So, if no environment variable is set you are perfectly fine, because write permissions to `agent/downloads` were set at install time. If all fails, updates are not available.

Convenience factor:

When you set DT_HOME you can use it to construct the path in the Agent connection string, e.g. instead of `-agentpath:/opt/dynatrace-7.0/agent/lib64/dtagent.dll=name=Java,server=localhost`
put
`-agentpath:$DT_HOME/agent/lib64/dtagent.dll=name=Java,server=localhost`

Entry point

The starting point of a PurePath is its entry point. It is the start of the method call chain that makes up the PurePath. The Entry Points Dashlet dashlet shows an aggregated representation of PurePaths that started at the same entry point.

Error

See Error Analysis for a full discussion of detecting and analyzing errors.

AppMon evaluates error detection rules on all PurePaths that arrive at the AppMon Server, and applies the defined consequences.

AppMon enables error detection by defining error detection rules that are based on captured HTTP response codes, Java/.NET exceptions, Java/.NET log messages, or JavaScript errors (browser errors).
An *error* in AppMon can be any kind of unexpected event that happens while a user interacts with a system. Some examples of errors that can occur are resources that cannot be found on a web server, HTTP 500 response codes sent by a web service, or exceptions that occur within an application.

Garbage collection size

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

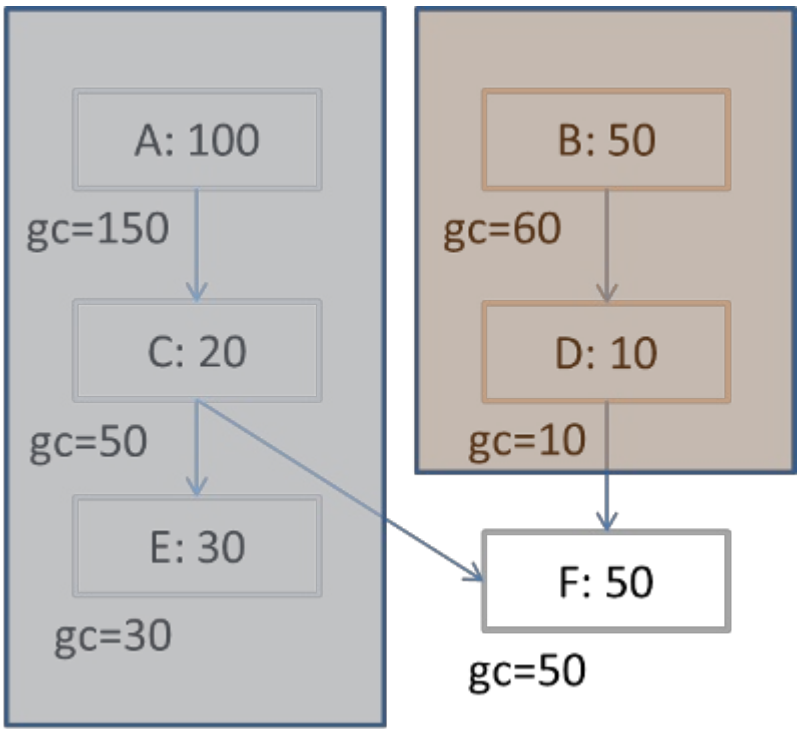
Total Memory dashlet

Glossary

The *Garbage Collection size* (GC size) of an instance is the amount of memory that is freed if the corresponding instance is garbage collected.

It is calculated by the Shallow Size of the instance plus the shallow size of all instances that are kept alive by this Instance. The following graph depicts a simplified Heap snapshot with 6 Instances.

Calculating Garbage Collection Size



Calculating garbage collection size

The GC size of instance A is calculated by summing up the shallow size of all instances that it keeps alive. This is A + C + E = 150.

Although instance F is also referenced by A, it is not garbage-collected because it is still referenced by B. For B the GC size is B + D = 60. Again, F is not garbage-collected if only B is garbage-collected.

The figure above depicts the GC sizes of all other instances.

The two colored shapes also represent the Keep Alive Set Dashlet of A and B, respectively.

Host

A *host* in AppMon is a system under analyis. A host is instrumented so that that measures can be monitored and reported on when thresholds ae exceeded.

Incident

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Incidents and alerting

Incidents dashlet

Glossary

An *incident rule* in AppMon is a mapping between measured thresholds and the actions to be taken when such thresholds are violated. If all the thresholds defined in the incident rule are violated, this is an *incident* , even if no actions have been configured for the incident rule.

Each incident is shown in the Incidents dashlet.

Incident rules can be based on multiple measures and different severity levels of threshold violations to trigger notifications. Threshold checks are made every 10 seconds.

Instrumentation

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Glossary

Instrumentation in AppMon is when AppMon injects bytecode into applications to make it possible to monitor a PurePath from end to end.

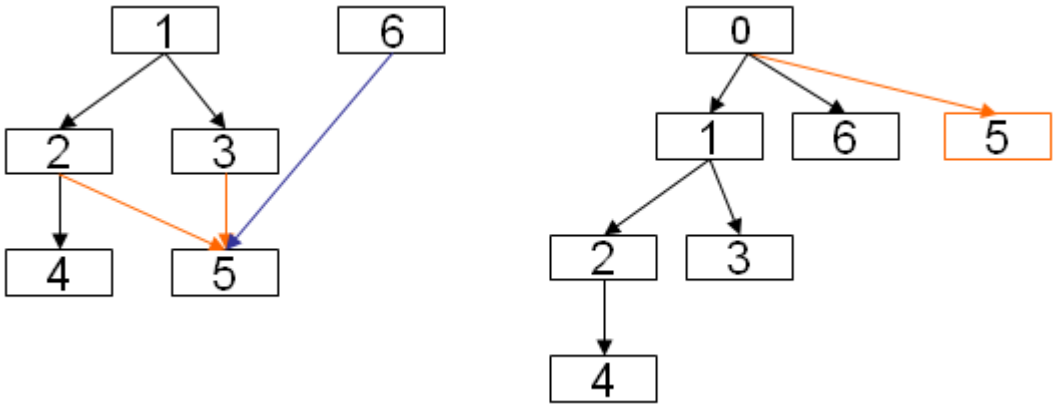
The AppMon Collector performs byte-code modifications when *instrumented* applications start up. The AppMon Agents therefore need to send the bytecode of your applications to the AppMon Collector. To keep start-up times low, it is necessary to deploy at least one AppMon Collector in every location/data center that is connected to an AppMon Server.

Inter tier time

- Network latency if the agents are physically separated
- Time added by other tiers that are between the measure points but are not instrumented
- Messaging queues

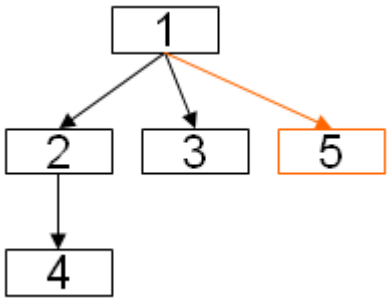
In the *Transaction Flow* visualization, **Inter Tier Time** is the time between the last measure point in a PurePath on one agent and the first measure point in the same PurePath on another agent. This measurement may include the following factors:

Keep alive graph



Example 2 – Direct Reference Graph and Keep Alive Graph with Virtual Root Node

When we add another reference to instance 5, we get a slightly different picture. Instance 6 has no referrer (i.e. it is a heap-root instance) and it references instance 5. Because we now have 2 heap-root instances, we need a virtual root (the <VMRoot> instance). Because instance 5 is referenced by instance 6, it cannot be garbage collected when instance 1 is removed. Therefore, the keep-alive referrer is the virtual root node (instance 0).



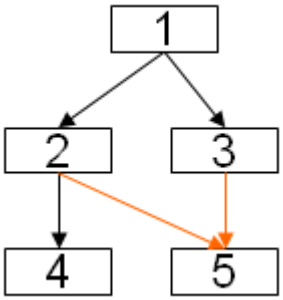
Example 1 - keep alive graph

Example 1 – Keep Alive Graph

The following figure shows the corresponding Keep Alive graph of the reference graph above. The referrer that is keeping instance 5 alive is instance 1. If instance 1 is removed, 2 and 3 are also removed, so that 5 has no more references and can be garbage collected.

The Keep Alive graph converts the reference graph into a graph in which each instance has exactly one

parent. If an instance has multiple referrers, the keep-alive referrer is shifted upwards.



Example 1 - direct reference graph

Example 1 – Direct Reference Graph

The following diagram shows a simplified direct reference graph. Instance 1 references instance 2 and 3, which in turn reference 4 and 5 respectively. However, instance 5 is referenced by instance 2 *and* 3.

It is usually not the same as the direct reference graph on the heap, which shows which instances reference another instance. If an instance is referenced by multiple instances, the *Keep-Alive instance* is the first instance on the upper branches of the tree that exclusively references the instance.
The Keep Alive graph shows how an instance is kept alive on the heap.

Logical application

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

System Profile - Applications

A *(logical) application* lets you manage more than one application within a single System Profile. Monitor and manage transaction performance for separate applications, or make a logical separation automatically by host / web application ID, or manually by using entities such as URI pattern or PurePath entry point.

For example, if you host `blog.example.com` and `community.example.com` on the same servers, you can separate data by defining logical applications by sub-domain. See System Profile - Applications for more information.

Agent

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

OneAgent

Install agents

Glossary

An Agent is a software that is either running on the host or injected into the application process. It gathers performance data and sends them to the AppMon server.

There are different agents that are specifically designed for the following technologies and application layers:

- The Java Agent.
- The .NET Agent.
- The Node.js Agent.
- The Apache Agent.
- The IIS Agent.
- The NGINX Agent
- The PHP Agent.
- The JavaScript Agent.
- The zOS Agent.
- The Host Monitoring Agent.
- The Database Agent.
- The OneAgent.
- Custom Agents created using the Agent Development Kit (ADK).

Memory snapshot post-processing

AppMon usually requires approximately 75% of the amount of heap of the application from which the snapshot is taken to run with optimum speed. Because the size of the heap depends on the number of instances and references in a snapshot, the required percentage may be higher. While a leak analysis snapshot is being post-processed, the progress dialog box displays the percentage of cache in use. This value should always be greater than 90%.

1. Transfer the raw data of the Snapshot from the AppMon Server to the Memory Analysis Server.
2. Extract the optimization classes, instances and references from the raw data and write them into separate files for optimized access.
3. Calculate the Keep Alive Graph, which is the base data for calculating the garbage collection size of each instance as well as for all other Keep Alive analysis functionality.
4. Calculate the garbage collection size for every instance contained in the snapshot.
5. Calculate the garbage collection size for classes. This is the sum of the garbage collection size for every instance of the class, with shared children taken into consideration. See also Total Memory Dashlet.
6. Calculate the HotSpots. See also Total Memory Dashlet.
7. If string capturing is enabled, calculate string duplicates.
8. Persist the resulting data to disk,
9. Transfer the post-processed Snapshot back from the Memory Analysis Server to the AppMon Server.

Post-processing a leak analysis snapshot consists of the following steps:

Post-processing trending memory snapshots or selective snapshots requires only a few seconds, because they contain much less data than leak analysis snapshots. The post-processing of leak analysis snapshots can require a considerable amount of time, memory, and disk I/O, depending on the number of instances and references in your application. To prevent the purging of PurePaths from AppMon Server memory, the AppMon Memory Analysis Server is used for post-processing of leak analysis snapshots. In a production environment it is mandatory to have a Memory Analysis Server installed and configured. (see Set up a Memory Analysis Server)

To minimize the application stall time while creating a memory snapshot, data is initially stored in raw format on the AppMon Server. Before memory snapshots are analyzed, they must be post-processed. Memory snapshot post-processing extracts optimization classes, instances, and references from the raw data and performs calculations on the raw data to produce data for analysis.

Monitor

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

System Profile - Monitors

Glossary

Watch and learn

Configuring monitors

A *Monitor* is scheduled for repetitive execution on a Collector to deliver additional performance metrics to the AppMon Server.
Typical examples are windows performance counters that are not specific to a process but host-wide, and SNMP data sources.

Data retrieved by Monitors are stored as Measures, which can then be combined and correlated to metrics captured from within the application. For example, transaction response times, number of exceptions and executed database statements.

*NIX

*NIX is used to refer to all Unix-like systems. Here, they typically include Linux, Solaris, AIX, HP-UX and Mac OS X.

PureLytics

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Glossary

PurePath technology

PureStack technology

PureLytics is the AppMon **big data analytics engine** that streamlines data analysis by pulling relevant information from millions of PurePath and PureStack data points in real time to enable multi-dimensional analytics. PureLytics provides the following features and benefits:

- **Ad-hoc & explorative** , PureLytics delivers just the information sets you need from user interaction and 3rd party services to supporting apps and infrastructure to quickly identify patterns, predict problems and make informed decisions.
- **Out-of-the-box analytics use cases** provide pre-configured, meaningful data sets for the most common analytics scenarios such as user experience, app version and channel.
- **Eliminates analytic's blind spots** with call to and response data from "black box code".

PureModel technology

Analysis can either start top down by analyzing how an application or transaction is impacted by the underlying infrastructure (PurePath) or bottom up by analyzing problems in the infrastructure and assessing and correlating their impact on application performance and end-user experience (PureStack).
PureModel Technology® combines the PurePath and PureStack views to provide the complete context of business application performance and end-user impact.

PurePath technology

PurePath Technology® is a complete multi-dimensional trace of every transaction in a monitored application providing comprehensive visibility across web browsers, web apps, mobile apps, web servers, Java, .NET, PHP, databases, middleware, and mainframe apps.

PureStack technology

PureStack Technology® directly correlates system infrastructure health data from every transaction tier in a monitored application environment with individual transactions and affected end users in real-time. PureStack is the vertical view of infrastructure in a monitored application environment and is the basis for *bottom up analysis*, which is defined by analyzing problems in the infrastructure and assessing and correlating their impact on application performance and end-user experience.

Session

- **PurePath Session**: Combined information about PurePaths and Time Series within a given time period.
- **Memory Dump**: Analysis data about the number, size, and class of allocated objects and their references.
- **Thread Dump**: Collection of data for thread analysis, including CPU time information.
- **Sampling**: Statistical data for all threads, which can help to find Rules and entry points.

Examples of a session are:
A *session* is a set of diagnosis data related to a specific System Profile.

Shallow size

- X bytes for the **object itself** (8 or 16 bytes, depending on VM).
- 4 bytes for **fField** defined in A.
- 1 byte for **bField**.
- 4 bytes for **iField**.
- 4 or 8 bytes for **baField**, depending on whether the VM is 32-bit or 64-bit. Arrays are objects.
- 4 or 8 bytes for **sField**, depending on whether the VM is 32-bit or 64-bit.
- 1 byte for **boolField**. A single Boolean field requires 1 byte.
- Y bytes for alignment. The VM internally aligns objects to 8-byte borders.

The shallow size for an instance of B would be calculated as follows:

```
class A {
   float   fField;
}

class B extends A {
  byte   bField;
  int    iField;
  byte[] baField;
  String sField;
  boolean boolField;
}
```

An object without any fields has a shallow size, because the object itself requires memory. The following example illustrates shallow size.

The shallow size of an object is the flat size, without children. The shallow size is defined by the sizes of all fields defined within the corresponding class and all fields defined in super-classes. It does not include the size of referenced children objects.

System under diagnosis (SUD)

The *System under Diagnosis* is the application where you inserted the AppMon Agent.

Task

See Tasks for a complete discussion of task configuration.

A task can be scheduled for repetitive execution on the AppMon Server or on a connected AppMon Collector. Arbitrary execution schedules can be defined and shared between different tasks and monitors. On execution, a task gets a set of configuration parameter values, and after execution the task provides an execution status containing data about the execution.
AppMon provides built-in *tasks* to automate repeating tasks such as generating reports or storing sessions.

Transaction Flow

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Transaction Flow dashlet

Glossary

The **Transaction Flow** dashlet graphically represents the tiers and backends of a monitored application environment as it is modeled by AppMon Agent Groups and AppMon Agent mappings defined in a System Profile. Different visualization modes provide different levels of graphical representation and detail. The Topology view identifies which application components communicate with each other, how many calls they make, the call rate, protocols, and the latency impact.

The Transaction Response Time view analyzes single transactions and their flow through the application environment. Use it to trace a single type of transaction and identify which components contribute to the execution of that transaction type. Use this information to identify bottlenecks and resources used.

Transaction splitting

Splitting can be used to segment business transactions according to splitting criteria. By default, all newly created business transactions are split by one criterion: application.The maximum number of splittings per business transaction is set to 5,000 by default. You can add more than one splitting criterion. For example, you can split by user and by URI.

UEM

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

User experience management

Glossary

AppMon User Experience Management (UEM) lets you track user behavior on web pages. It helps you identify user problems before your user complains. It also lets you receive business analytics data like the geographic distribution of your web page visitors and average session duration.

Update

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Updates

Glossary

Updates are deployed to a running AppMon Server via the AppMon Client UI and then rolled out to the Collectors and Agents. Updates are always supplied by the vendor, never the user. For more information, see Details on Update Installation.

The following are the two types of updates:

- **Public** updates are available to all registered customers at the download site. You can install these at your discretion.
- **Private** updates address customer and installation-specific problems. AppMon support hands these out in coordination with release management. They are not publicly available and you must contactsupport to deploy them.

Update versions must match the current system's version except for the build-version and revision number (see Versioning Scheme). For example, you can apply an update with the version 6.5.1.2030 to an AppMon Server with the version 6.5.0.1050, but **not** to version 6.**3**.0.1050. You can only have **one** update active on an AppMon system at any time.

All updates, except **private updates**, are cumulative. That means that a private update 6.5.0.1010 does **not necessarily** include all fixes from a private update with a lower number, e.g. 6.5.0.1008. On the other hand, each publicly available update with a higher build-number does include all fixes from all previous public updates. A public update does **not** necessarily include all fixes from private updates with lower numbers. If a public update does not contain all fixes from the currently running version, no notification will be displayed in the AppMon Client and if installing a manually downloaded file, a warning "You will lose fixes X" will be displayed. If you want to install the public, in such a case, contact support.

The build-version number for custom updates therefore only serves to uniquely identify different updates.

Update files have the extension `.dtf` . AppMon provides these as `.zip` files, with a checksum and release notes.

Versioning scheme

- **Update**: Smaller changes in the software, usually bug fixes or small enhancements, are indicated by a different build number or revision number (e.g. from 6.5.0.1269 to 6.5.0.1289). See Updates for more information.

> **Note**
>
> The revision does not appear in the default installation path to accommodate for in-place updates on the revision level.

When the software differs on major, components such as the Server or Collector are normally compatible, but this is not guaranteed. Major and minor version are part of the default installation path, signaling that software differing therein must be installed into different directories.

- **Upgrade**: Significant changes in the software, including new features and enhancements, are indicated by a different major or minor number. For example, from 6.3.0.1305 to 6.5.0.1289).

When referring to changes made to previously installed software, the following terms describe the significance of the changes:

- major = 6
- minor = 5
- revision = 14
- build number = 1012

For example, Version 6.5.14.1012:

`major.minor.revision.build`

Version numbers assigned to AppMon components follow this format:

AppMon platform overview

Purepath explained

Features and capabilities

Editions
Architecture

Features and capabilities

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Comprehensive & continuous application monitoring

User experience management

Real-time business transaction monitoring

Host and system monitoring

Visualize and share performance data and problem context

Smart baselines and alert notifications

Extensibility and systems integration

Designed for continuous delivery processes

Extensive technology and application environment support

AppMon SaaS

Related pages

Architecture

PurePath explained

Client UI overview

Editions

Watch and learn

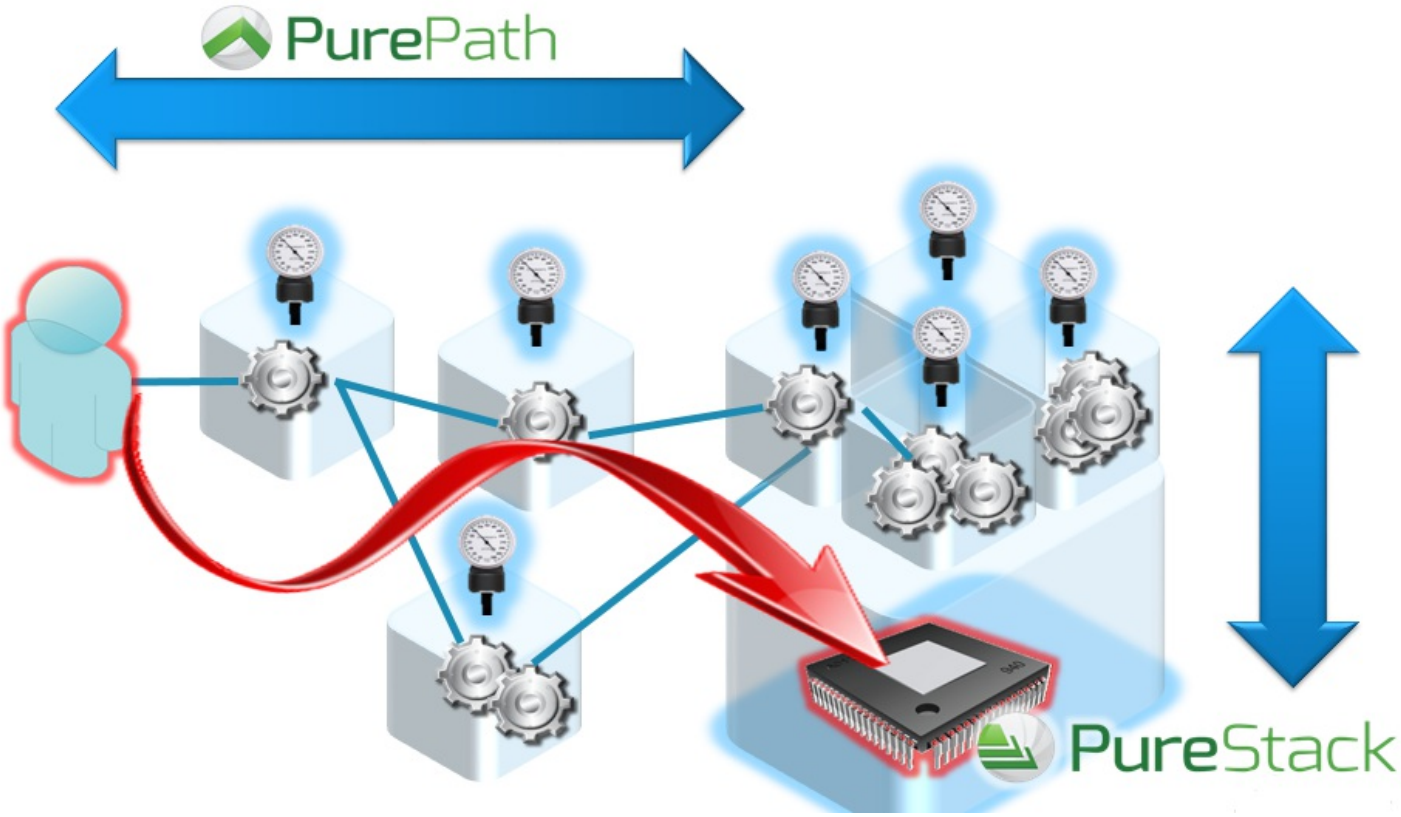
User experience management

Introducing Business Transactions

This page provides a high-level overview of the features and capabilities of AppMon Application Monitoring and User Experience Management.

Comprehensive & continuous application monitoring

Application Monitoring continuously discovers and monitors all processing in your application environment using patented PurePath Technology®. PurePath provides a complete multi-dimensional trace of every transaction in a monitored application, giving comprehensive visibility across mobile apps, web apps, web browsers, web servers, Java, .NET, Node.js, PHP, databases, middleware and mainframe apps. When combined with PureStack Technology®, the complete context of business application performance and end-user impact is visible enabling you to quickly drill-down into any problematic transaction to see code-level details.



User experience management

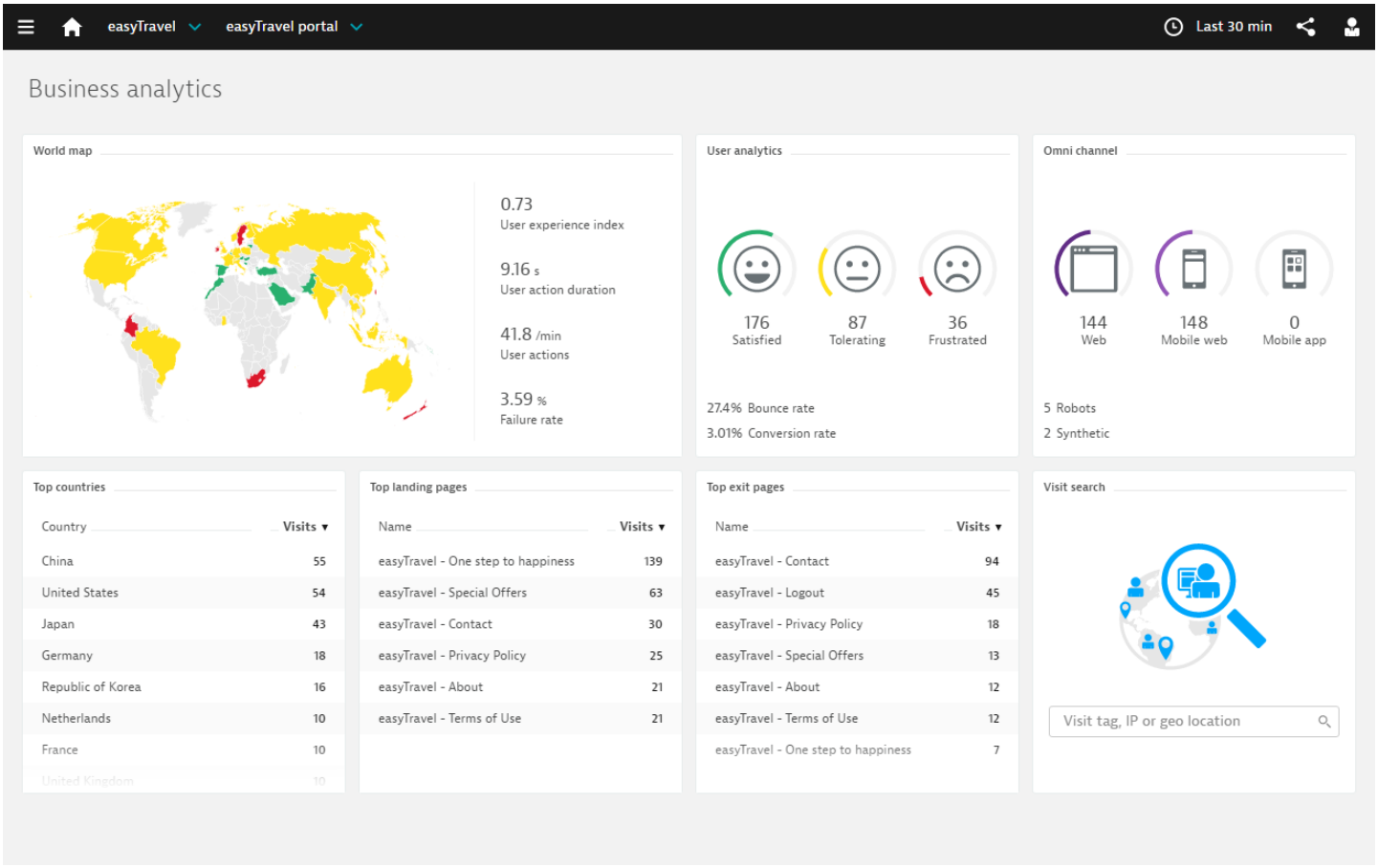
AppMon User Experience Management (UEM) lets you track user behavior on web pages. It helps you identify user problems before your user complains. It also lets you receive business analytics data like the geographic distribution of your web page visitors and average session duration.

Automatically detected and captured UEM data is visible in various User Experience dashlets, which display the following types of data:

- User actions converted into business contexts such as orders, downloads, views, searches, and more.
- Usage statistics across browsers, devices, geographies, carriers and connections that let you see every page visited, every click made, individually or in aggregate, and whether performance issues are detected.
- Slow performance data on site-usage or revenue flow, minute-by-minute, real-time, in production.
- User's click-path and along this path they tend to leave.
- User stratification metrics based on response times and any functional problems along their execution path.

In addition, UEM data correlates with server-side performance data to identify slow requests in a specific user visit, and then drill down from the individual visit into your entire server-side infrastructure (Java, .NET, PHP) down to a single method invocation or database call. For any user visit, see call count, bind values, round trip and execution times on any individual transaction. See User Experience Management for more information.

No manual changes to your Web pages or JavaScript are required. Once agents are deployed to your Web server, your HTML pages and JavaScript code is instrumented automatically. For native mobile and hybrid mobile applications, all the monitoring functionality is added by an auto-instrumentation component, which can be fully integrated into the lifecycle of your mobile application development. See JavaScript Agent and Web Server Agent Configuration. See Mobile Agent for mobile details.



Real-time business transaction monitoring

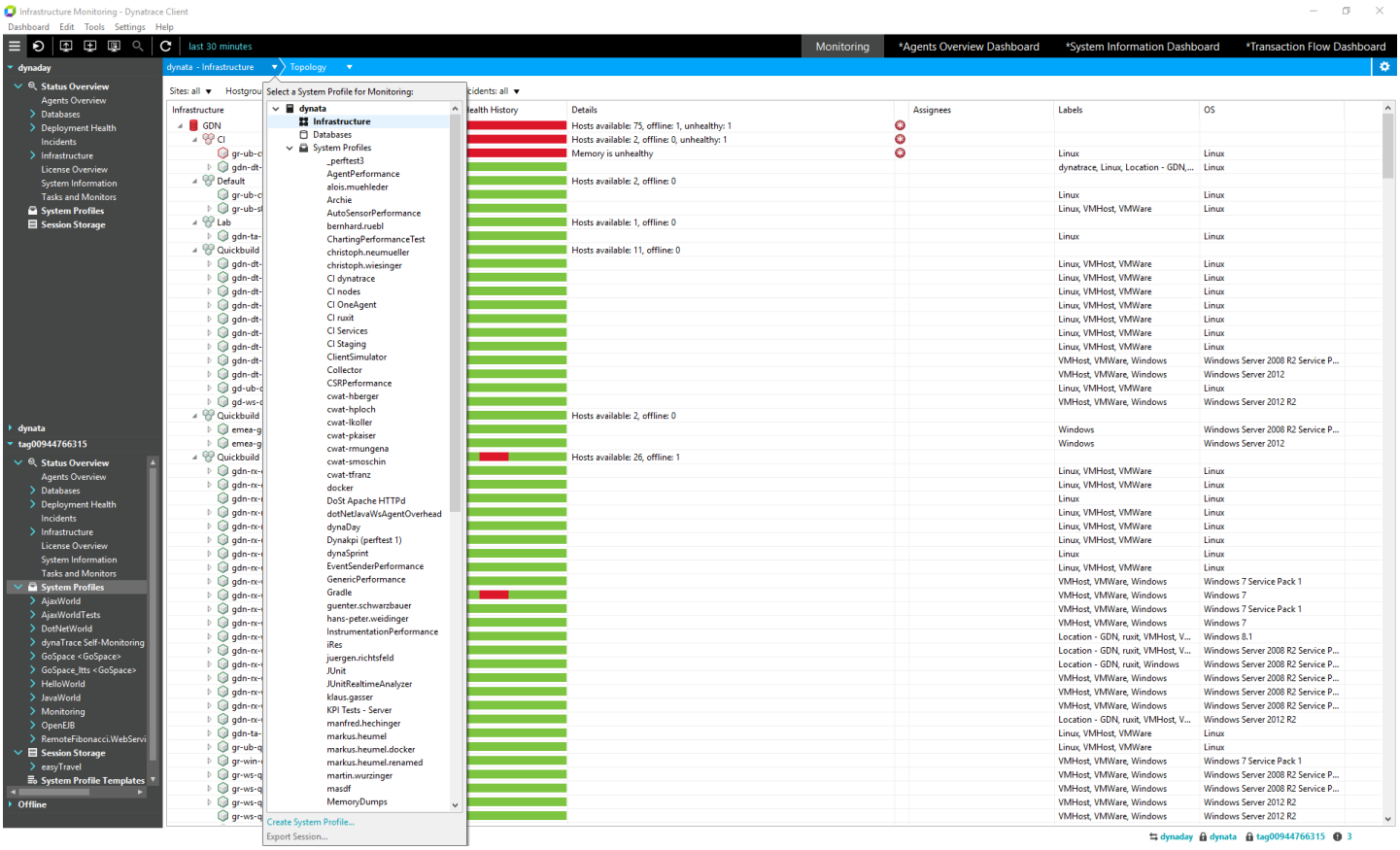
Business transactions let you organize and identify specific types of user requests such as logging in or adding items to a cart, and translate the technical data captured by AppMon's user experience, application, and infrastructure monitoring capabilities into relevant information for your business. This provides real-time visibility of correlated application performance and business performance data, helping all team members better understand applications, infrastructure, and the resulting impact on your business.



Host and system monitoring

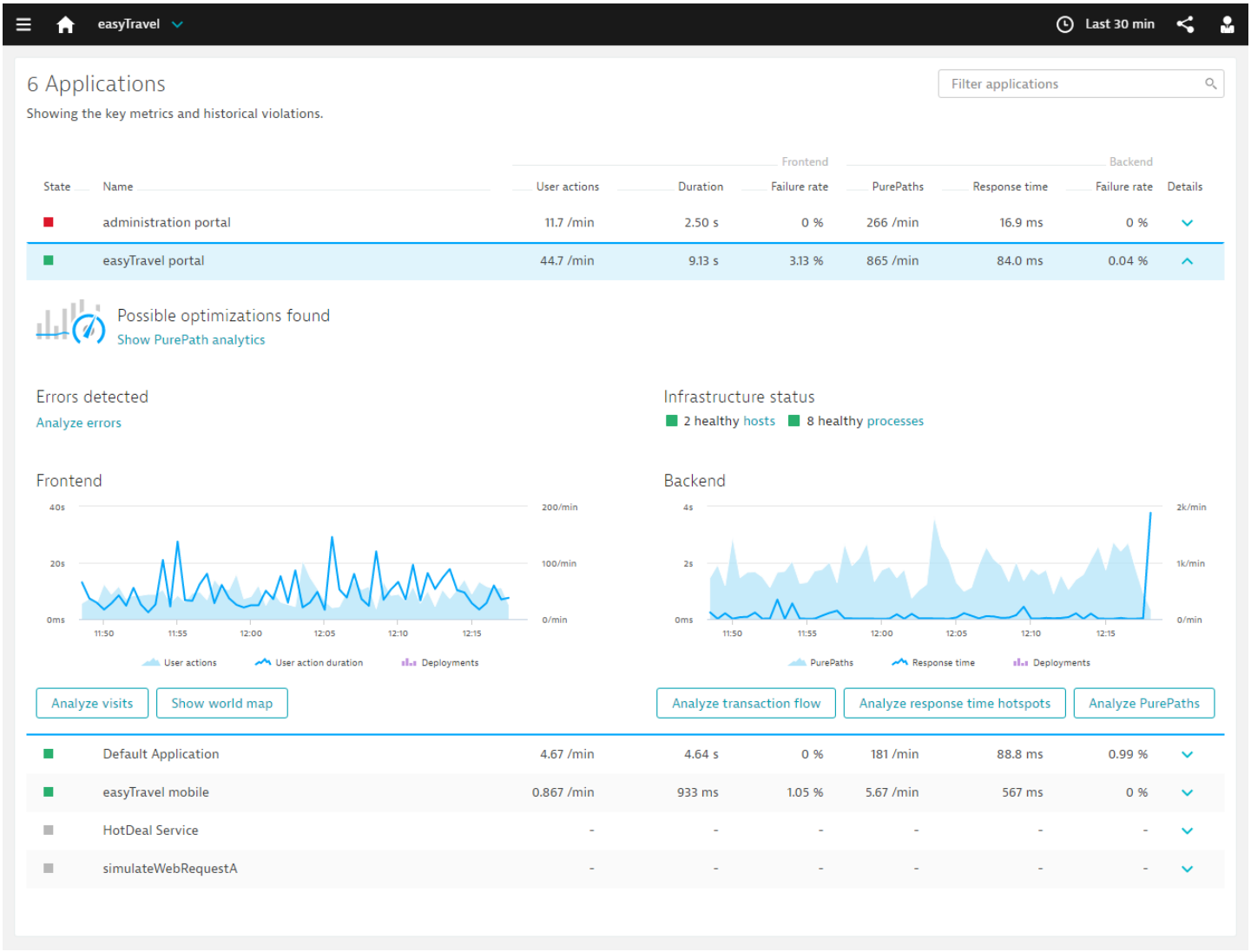
AppMon provides built-in host and system monitoring across all supported operating systems and machines in your application environment. Every type of AppMon Agent, in addition to the standalone Host Monitoring Agent, provides host metrics for CPU, memory, network, and disk. Automatic health indicators are calculated to warn about host-specific problems that may also affect your application or specific business transactions.

To supplement the built-in Host Monitoring capabilities, AppMon offers Monitors for Windows, Linux, SNMP, Oracle v$ Stats, and others that can periodically pull data from different types of data sources.

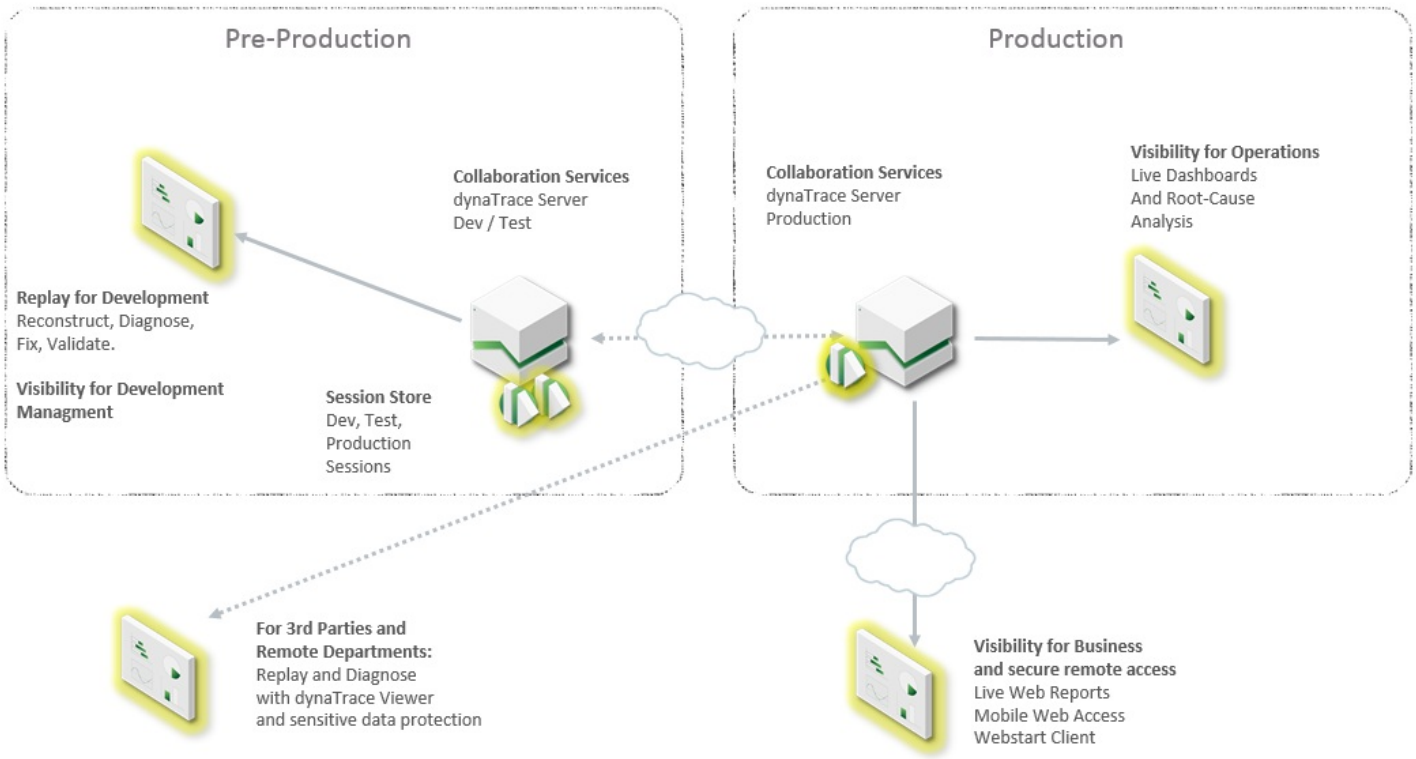


Visualize and share performance data and problem context

AppMon dashboards are the window through which you view captured application monitoring and diagnostic data. Use preconfigured dashboards to quickly access data such as system health, application problems, and response times experienced by end users. You can also create custom dashboards and specify which data to display in a specific format and which users to share the data with. View dashboards in the AppMon Client, a web browser, or as a generated PDF, XML or Excel report. See Client UI Overview, Dashboards and Reporting for more information.

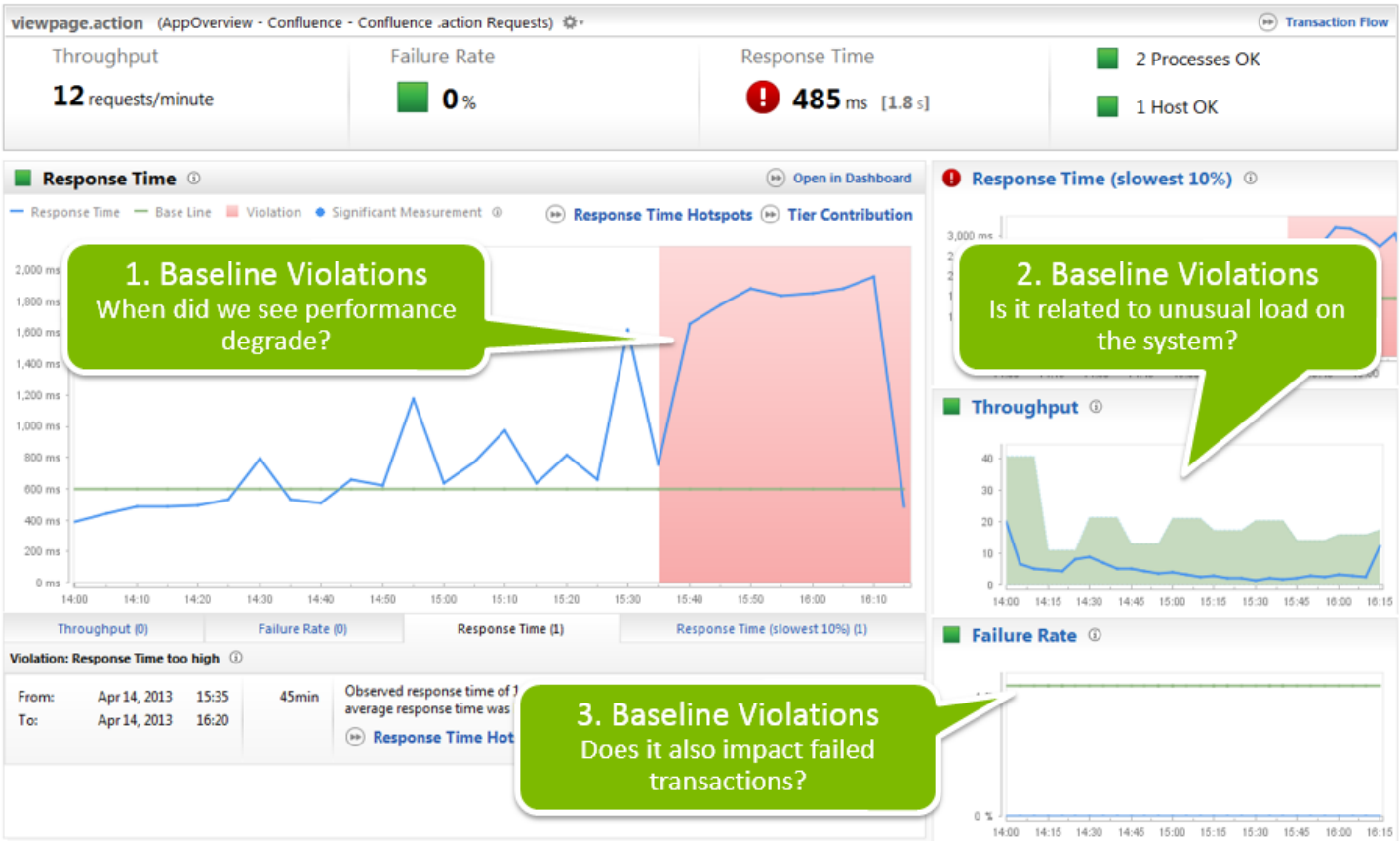


To isolate and share exact problem context, AppMon records diagnostic *sessions*, which are distributable files that capture the full transactional context contained in one or more PurePaths. Sessions promote efficient collaboration by providing a common language for problem resolution that can be shared across development, test, and operations teams.



Smart baselines and alert notifications

To avoid reporting false positives and generating unnecessary alerts that can occur with traditional statistical approaches that look only at averages and violations based on standard deviation, AppMon's smart baseline technology uses the statistical characteristics of different application metrics and employs advanced statistical models to identify only relevant application performance anomalies. Smart baselining combined with AppMon's complex alert notification capability help you detect and troubleshoot problems before users are impacted. See Incidents and Alerting for more information.



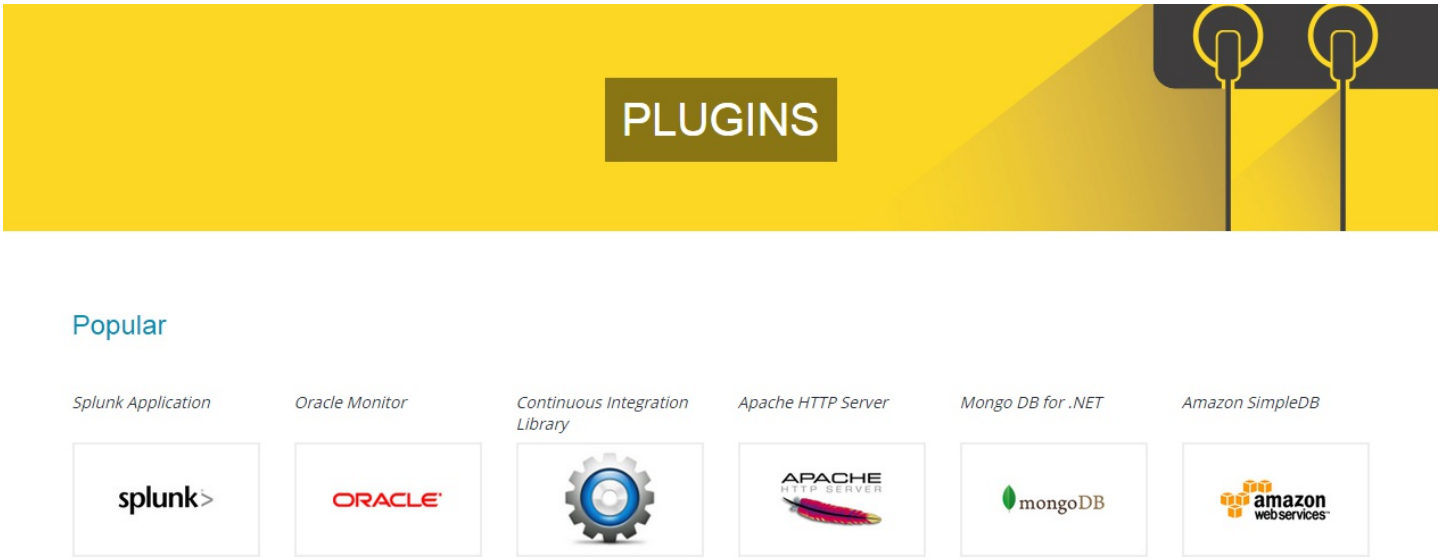
Extensibility and systems integration

AppMon includes integration points for other systems in your organization, such as development IDEs (Eclipse & Visual Studio), load testing (SilkPerformer & LoadRunner), and other monitoring tools (DC RUM and Synthetic Monitoring).

In addition, you can extend AppMon capabilities by installing plugins that provide new features and functions for system monitoring, application monitoring, custom actions, and lifecycle automation.

Use the REST interface to automate commonly performed tasks, such as starting and stopping session recording, and triggering memory dumps.

With the Agent Development Kit, JavaScript ADK, and Mobile app ADK, you can extend the default AppMon Agent monitoring capabilities by integrating technologies such as native C or C++ applications, custom communication protocols, custom JavaScript frameworks, and native mobile applications.

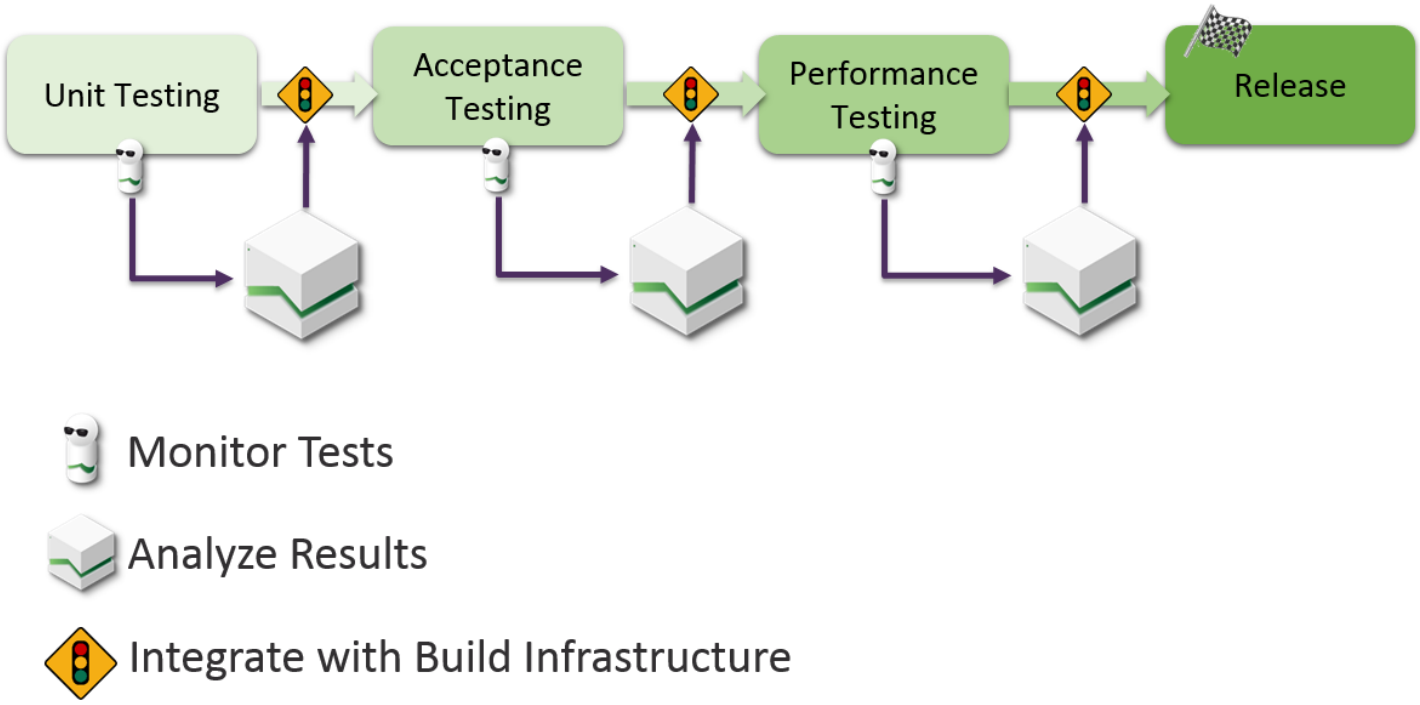


Designed for continuous delivery processes

AppMon supports continuous delivery processes by:

- Promoting performance data sharing across development, test, and operations.
- Providing reliable metrics along your delivery pipeline through automation interfaces for CI/CD systems, such as Jenkins, Bamboo, and TeamCity.
- Delivering more stable applications in production faster through automatic analysis of regressions and the comparison of response times, structural differences, code executions, errors, exceptions and database performance.

See Continuous Delivery & Test Automation for more information.



Extensive technology and application environment support

AppMon provides extensive end-user, application and infrastructure platform and technology support that includes mobile apps, web apps, web browsers, web servers, Java, .NET, Node.js, PHP, databases, middleware and mainframe apps. See System Requirements and Supported Technologies for a comprehensive listing.

AppMon SaaS

As an alternative to managing the deployment of AppMon components in your data center, you can use the cloud-based capabilities of AppMon SaaS to simplify the deployment and component management required in an on-premises approach. See Differences Between On-premises and SaaS Deployment for a comparative description.

Architecture

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

AppMon Server
AppMon Frontend Server
AppMon Memory Analysis Server
AppMon Collector
AppMon Performance Warehouse
AppMon Client

Related pages

Features and capabilities
PurePath explained
Client UI overview
Editions
Web dashboards
Take the AppMon Client wherever you go

Watch and learn

Platform overview
Exploring core components

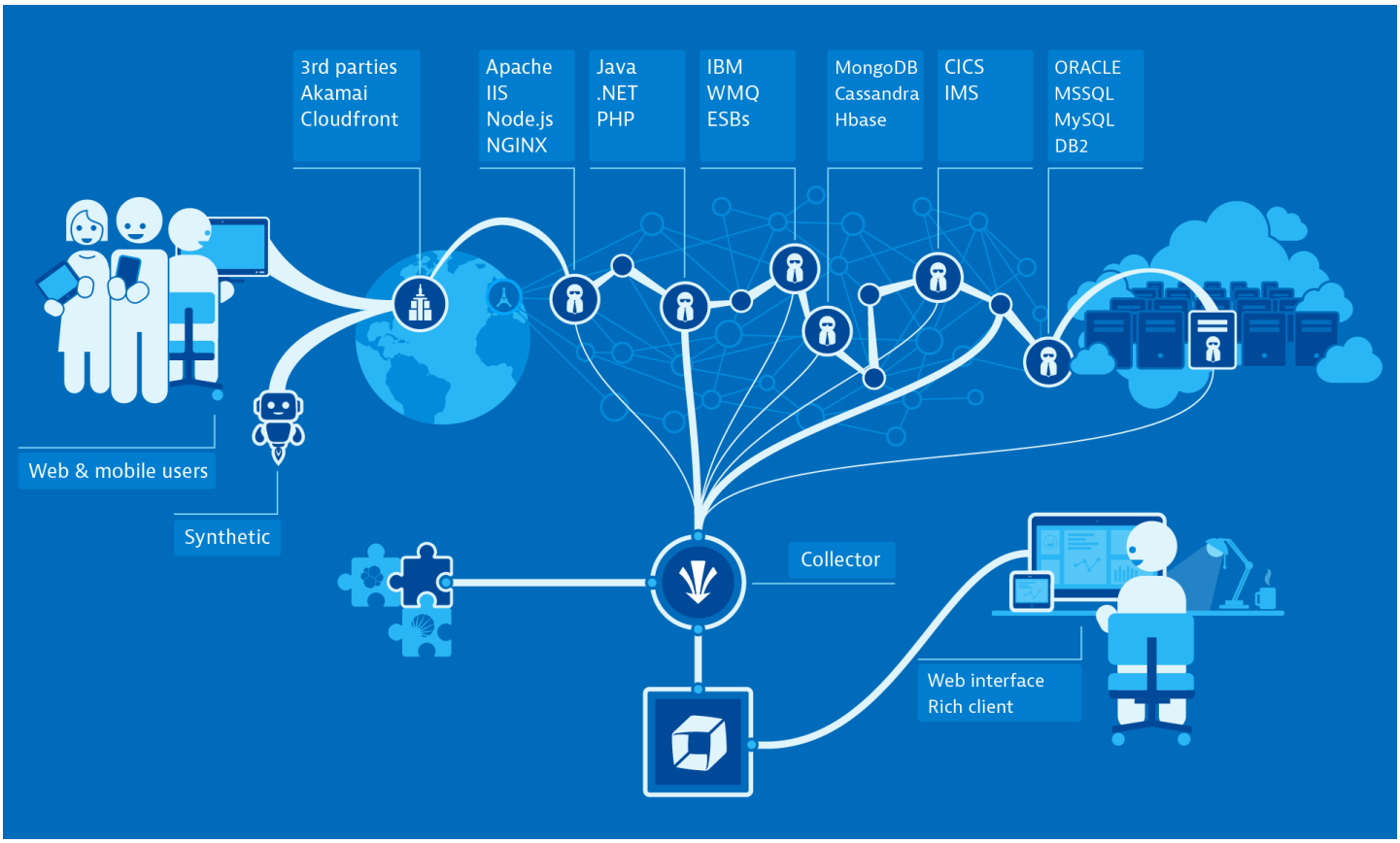
The core components of a typical AppMon deployment in a live environment are the AppMon Server,

Memory Analysis Server, Frontend Server, Collector, application Agents and a Performance Warehouse database.

Just like a Performance Warehouse database server, the AppMon Server and Collector components are processes (as Windows services or *NIX daemons). They can all run on a single virtual machine or be dispersed over many machines, depending on the size of your installation and the amount of traffic they handle from your instrumented applications.

Web Server Master Agents are services as well. Core Agents run in-process with the application.

See System requirements, the Deployment Guide, or contact sales if you do not have a proper sizing for AppMon components yet.



AppMon Server

The AppMon Server backend process is the core component of an AppMon environment and, in combination with the AppMon Client, the central configuration and administration unit. It correlates PurePath data received from Agents by way of the AppMon Collectors, and stores these along with thread and memory dumps in the file system. This file system is known as the session store. It also handles measures and their subscriptions.

AppMon Frontend Server

The Frontend Server frontend process handles data analysis tasks (except big memory snapshots - see below) that AppMon Clients trigger. The AppMon Clients connect to the Frontend Server.
The AppMon Frontend Server must be on the same machine as the AppMon Server.

AppMon Memory Analysis Server

The Memory Analysis Server performs post-processing memory snapshots to offload resource usage from the AppMon Server. With the same goal, the AppMon Frontend Server serves all the other analysis data requested by the AppMon Clients.

The Memory Analysis Server can run on the same machine as the AppMon Server, if you have sufficient memory and processing resources. You may need a separate machine for huge memory snapshots or if the Memory Analysis Server is shared between AppMon Servers.

AppMon Collector

The AppMon Collector processes (up to dozens) collect and bundle data from the AppMon Agents. AppMon Collectors must be relatively close in the network to their Agents (same LAN), but can be far from the AppMon Server. Laid out redundantly, they provide load balancing and fail-over (Agents can fail-over to a different Collector). Collectors post-process the data to both, off-load from Agent and Server side and send the payload to the AppMon Server. They also execute monitoring plugins that do not require an Agent.

AppMon Performance Warehouse

The AppMon Performance Warehouse (SQL database server) stores processed measure and time-series data that the Agents deliver. Depending on the complexity and data volume this can be in a local database on the same machine or a high performance DB server.

As opposed to time-series data, PurePaths and thread and memory dump BLOBs are directly stored in the session store.

AppMon Client

The AppMon Client (UI) is the presentation layer installed on users' machines or started with the Webstart Client. There is also a portable Client (for portable media like USB-sticks) that does not need installation (just unzip) and AppMon Web – the browser-based solution (as opposed to the Webstart Client that starts using a browser and loaded from the Server).

PurePath explained

Looking for the 6.5 version of this page? Open in 6.5 documentation

Related pages

Features and capabilities

Architecture

Client UI overview

Editions

Problem pattern detection

Watch and learn

What is a PurePath?

What is a PureStack?

What is a PureModel?

PurePath technology

PurePath Technology® is a complete multi-dimensional trace of every transaction in a monitored application providing comprehensive visibility across web browsers, web apps, mobile apps, web servers, Java, .NET, PHP, databases, middleware, and mainframe apps.

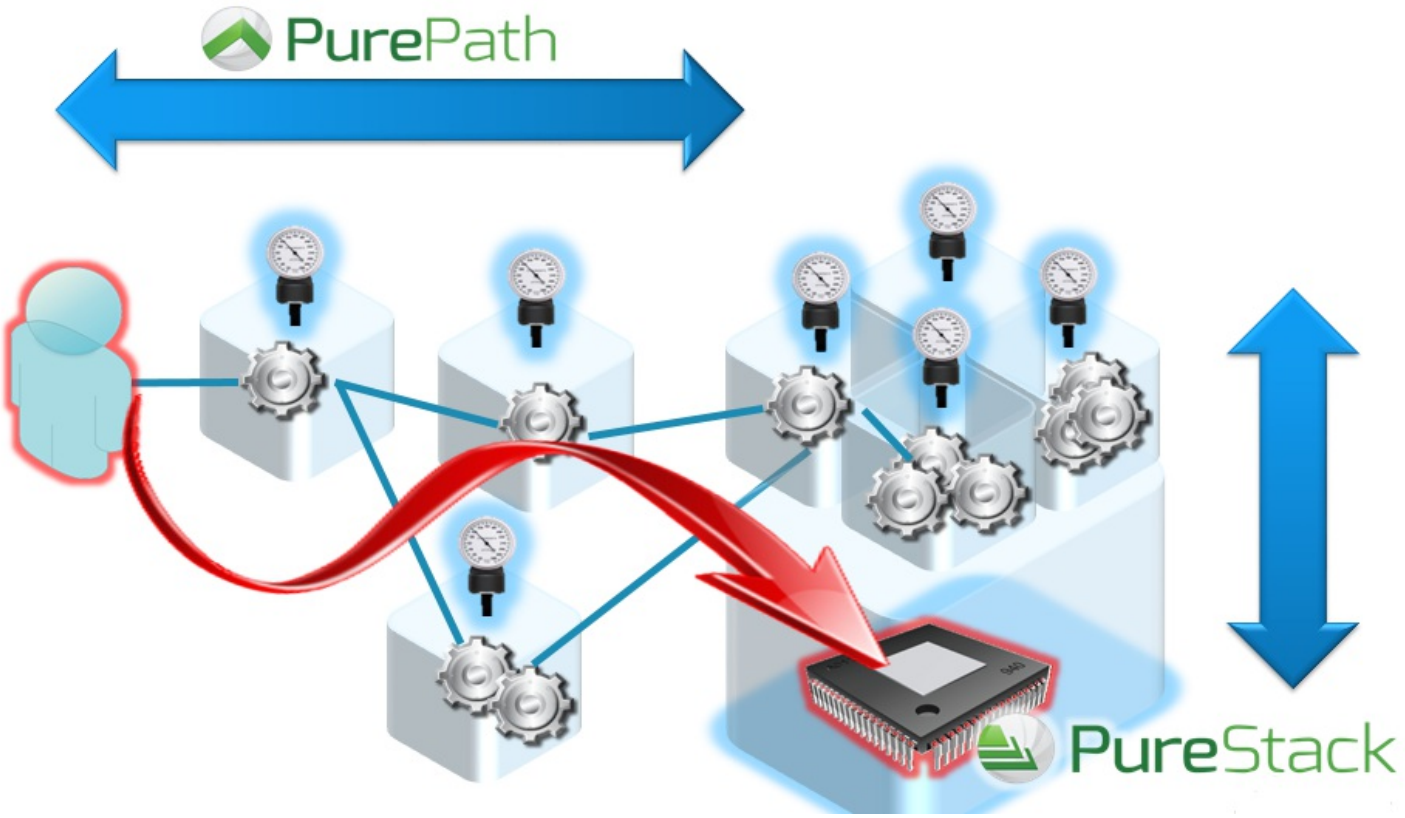
With PurePath technology, you get complete diagnostics, such as timing data and code-level context, across the entire user click path spanning device, operating system, page actions, methods, code, and bind values. This enables more accurate reporting, granular business transaction grouping, precise SLA management, and fast root cause resolution.

PurePath transactional tracing is always on, 24/7, to accurately monitor dynamic executing paths in cloud and virtualized environments. PurePath recordings can be quickly shared with team members. With one-click you can drill down to the root cause without having to reproduce the problem.

A PurePath is the horizontal view of a transaction in a monitored application environment and is the basis for *top down analysis*, which is defined by analyzing how an application or transaction is impacted by the underlying infrastructure.

Typically, a PurePath is composed of data sent asynchronously by many different AppMon Agents, which then send the data to the AppMon Collector. The Collector buffers and enriches the data and passes it on to the AppMon Server, which is responsible for PurePath construction, analysis, and offline storage. PurePaths provide the superset of data for creating specific Business Transactions and high-level Measures.

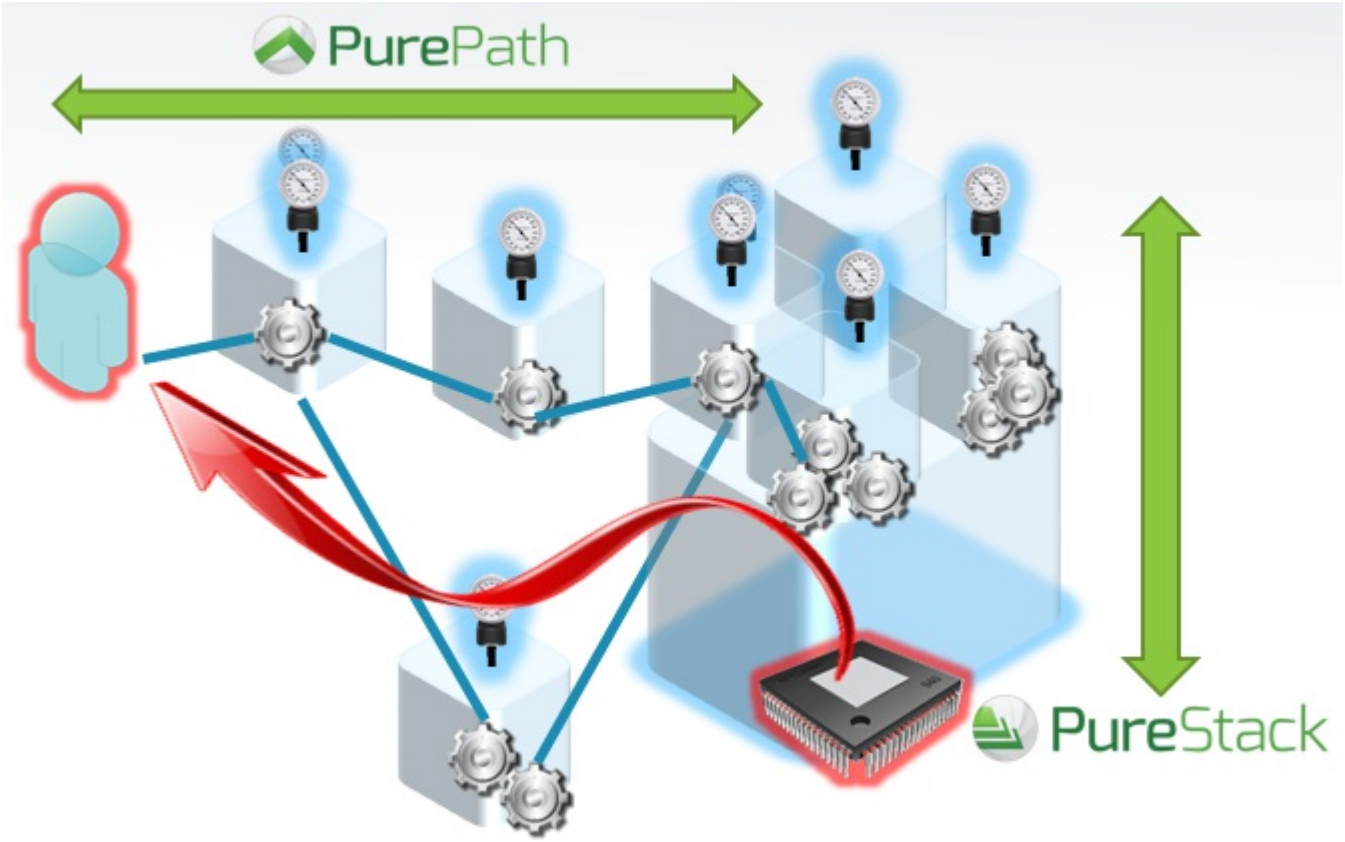
PurePath Technology - Top Down Analysis



PureStack technology

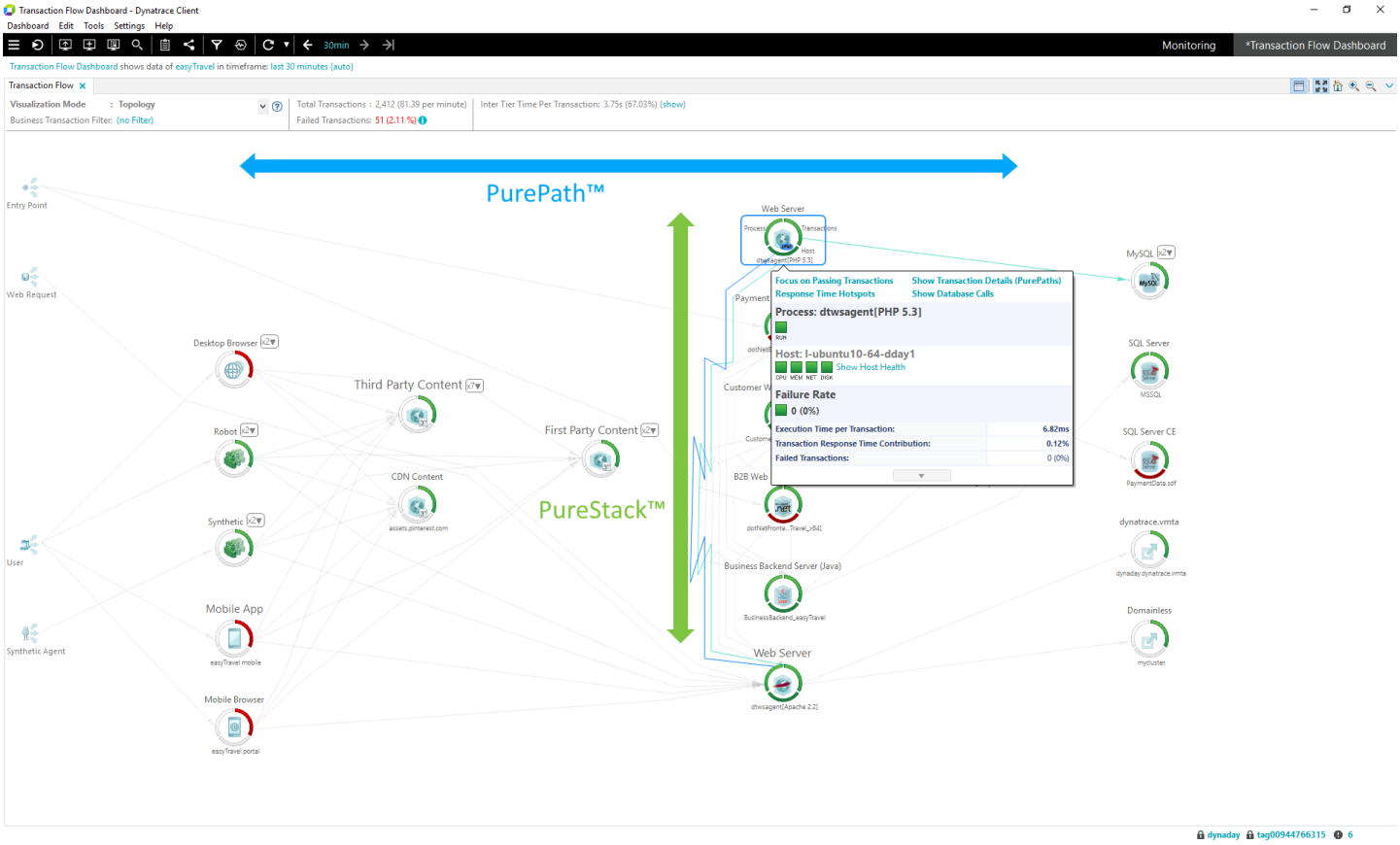
PureStack Technology® directly correlates system infrastructure health data from every transaction tier in a monitored application environment with individual transactions and affected end users in real-time. PureStack is the vertical view of infrastructure in a monitored application environment and is the basis for *bottom up analysis*, which is defined by analyzing problems in the infrastructure and assessing and correlating their impact on application performance and end-user experience.

PureStack Technology - Bottom Up Analysis



PureModel technology

Analysis can either start top down by analyzing how an application or transaction is impacted by the underlying infrastructure (PurePath) or bottom up by analyzing problems in the infrastructure and assessing and correlating their impact on application performance and end-user experience (PureStack).


Transaction Flow Dashboard - Dynatrace Client
Dashboard Edit Tools Settings Help
Monitoring
*Transaction Flow Dashboard
Transaction Flow Dashboard shows data of easyTravel in timeframe: last 30 minutes (auto)
Transaction Flow
Visualization Mode : Topology
Business Transaction Filter: (no Filter)
Total Transactions : 2,412 (81.39 per minute)
Failed Transactions: 51 (2.11 %)
Inter Tier Time Per Transaction: 3.75s (67.03%) (show)
PurePath™
PureStack™
Entry Point
Web Request
User
Synthetic Agent
Desktop Browser
Robot
Synthetic
Mobile App
Mobile Browser
Third Party Content
CDN Content
First Party Content
Web Server
Business Backend Server (Java)
Web Server
MySQL
SQL Server
SQL Server CE
dynatrace.vmta
Domainless
Focus on Passing Transactions
Response Time Hotspots
Show Transaction Details (PurePaths)
Show Database Calls
Process: dtwsagent[PHP 5.3]
Host: l-ubuntu10-64-dday1
Show Host Health
Failure Rate
0 (0%)
Execution Time per Transaction: 6.82ms
Transaction Response Time Contribution: 0.12%
Failed Transactions: 0 (0%)

Editions

Looking for the 6.5 version of this page? Open in 6.5 documentation

On this page

Production edition

Pre-Production edition

Detailed feature comparison

Related pages

Features and capabilities

Architecture

PurePath explained

Client UI overview

Watch and learn

Selecting a Dynatrace edition

AppMon supports the entire application lifecycle in a single, integrated platform supporting the three core areas of the lifecycle: development, test, and production. The AppMon platform comes in production or pre-production/TestCenter editions.

Each edition is packaged with specific features and default settings, such as out-of-the-box dashboards, integrations, zero-configuration, and auto-adapting instrumentation to accelerate time to value for developers, testers, and operations personnel respectively. The specific features and default settings available for each edition are determined by the type of license applied to the product.

Production edition

The AppMon Production edition enables advanced application performance management for rapidly identifying and diagnosing problems in staging and production, which reduces the mean time to repair by as much as 90%. Identifying and resolving production performance issues with mission-critical applications can take days and could cost thousands of dollars per hour.

With the Production edition, you can manage mission-critical application performance in a 24x7 environment, capture critical diagnostics information at all times, provide exact root-cause analysis, and resolve Java/.NET issues in minutes instead of days. The AppMon always-on PurePath transaction tracing makes this all possible by letting you manage business transactions end-to-end and map application dependencies to quickly understand issue priorities. Performance diagnostics down to the code level provides rapid diagnosis of the root cause of issues. You can share PurePath information with all stakeholders to quickly reach consensus on issues and resolutions.

Pre-Production edition

The AppMon Pre-Production edition integrates with major commercial and open-source load testing solutions such as LoadRunner, SilkPerformer, and Visual Studio Team System to eliminate manual problem reproduction and cut lost productivity time by as much as 90%. With this edition, developers and test professionals can eliminate the multiple days it often takes to find, document, reproduce, diagnose, and resolve performance issues.

Using PurePath transaction tracing, AppMon provides in-depth code-level visibility into the performance of each individual transaction of all virtual users in large load tests of highly distributed applications. With AppMon, developers and test professionals can now automate and accelerate the process of documenting, diagnosing, and resolving performance issues in Java/.NET applications, allowing more time to be spent on strategic development and test activities.

Detailed feature comparison

Key:


= supported


= limited support

= not supported

Feature (down)	Production	Pre-Production (Test Center)
Transaction Tracing with PurePath Technology	✓	✓
Tracing of Each Transaction	✓	✓
Auto Sensors for Hotspot Detection with Zero Configuration	✓	✓
Tracing at Code Level and Capturing of Contextual Data	✓	✓
Tracing with Low Overhead and Small Memory Footprint	✓	✓
No Source-Code Changes Required (Java, .NET, JavaScript)	✓	✓
Tracing Across Web Applications	✓	✓
Tracing Across Java, .NET, and Native Applications	✓	✓
Tracing Across Multiple Tiers/Servers	✓	✓
Tracing Across Thread Boundaries	✓	✓
Auto-Discovery of Application Components	✓	✓
Support for Proprietary Remoting/Messaging Technologies	✓	✓
Custom Sensors and Knowledge Sensor Packs	✓	✓
Visual Class Browser and Sensor Placement	✓	✓

Diagnostics	Production	Pre-Production
Real-time Diagnostics while Applications Run	✓	✓
Unicode Support	✓	✓
Manual Session Capturing	✓	✓
Offline Diagnostics and Session Export	✓	✓
Insight into Third-Party Libraries without Source Code	✓	✓
Continuous, 24x7 Session Capturing	✓	✓
Automated and Centralized Session Storage	✓	✓
Memory Snapshot Analysis (Hotspots, Collections, Maps, Http Sessions, etc.)	✓	✓
Auto-Triggered Memory and Thread Snapshots (on OOM or Scheduled)	✓	✓
Centralized, High-Speed Memory Snapshot Analysis	✓	✓
Thread Snapshot Analysis (Hotspots, Locking, Monitors, etc.)	✓	✓
Profiling of CPU and Synchronization Hotspots	✓	✓
UML Sequence Diagrams	✓	✓
Auto-Generated, Real-Time Transaction Flow Map	✓	✓
Automated Architecture Validation	✓	✓

Manual Diffing, Comparison, and Regression Analysis	✅	✅
CodeLink IDE Integration for Eclipse and Visual Studio	✅	✅

Monitoring	Production	Pre-Production
Application Monitoring (JMX, PMI, Performance Counter, JVM, CLR, etc.)	✅	✅
System Monitoring (Windows, Linux, Solaris, AIX, z/OS, etc.)	✅	✅
Scheduled and Pluggable Synthetic End-User Monitors	✅	✅
Alerting/Notifications on Incidents	✅	✅
Percentiles for SLA Engineering Beyond Averages	✅	✅
Business Transaction Management	✅	✅
Long Term Performance Charts	✅	✅
24x7 Monitoring of Live Applications and Systems	✅	⚠
Enterprise Management System Integration and Automation	✅	✅
Self Monitoring Capabilities to Ensure 99.9% Uptime	✅	✅
Out-of-the-box Monitoring Dashboards and Reports	✅	✅

Load testing/Build integration and automation	Production	Pre-Production
Open Automation Interfaces (REST, JMX)	✅	✅

Load Testing Integration (Loadrunner, JMeter, Gomez 360, VSTS, etc.)	✓	✓
Out-of-the-box Load Testing Dashboards and Reports	✗	✓
Build System Integration (Ant, NAnt, Quickbuild, etc.)	✗	✓
Include Custom Events and Build Metadata (Version, etc.)	✗	✓
Test Automation and Regression Reports (Selenium, Watir, QTP, etc.)	✗	✓
Test Framework Integration (JUnit, NUnit, TestNG, etc.)	✗	✓
Automated Regression Reports	✓	✓

Collaboration across the Lifecycle	Production	Pre-Production
Web-enabled AppMon Client (Java Webstart)	✓	✓
Customizable Dashboards	✓	✓
Manually Triggered Reports (PDF, XLS, HTML, CSV, etc.)	✓	✓
Dashboard Sharing	✓	✓
Scheduled Reports and Delivery (Online, eMail, etc.)	✓	✓
Browser-access to Dashboards and Reports	✓	✓
Protection and Omission of Sensitive Data	✓	✓
Secure User Authentication	✓	✓

Flexible Role and Permission System	✓	✓
User Groups and LDAP integration	✓	✓

Deployment	Production	Pre-Production
Central Agent Deployment and Configuration	✓	✓
Free {dt} Viewer	✓	✓
Cluster and Virtualization Support	✓	✓
Recovery and Watchdog (24x7 Support)	✓	✓
WAN Support with Encryption/Compression	✓	✓
Built-in Performance Warehouse	✗	⚠
High-performance, External Performance Warehouse (SQL, DB2, Oracle, PostgreSQL)	✓	✓
Schedule Downtimes for Maintanance Windows	✓	✓
Fault Tolerance (JVM Options, Network Overhead, Smart Sensors)	✓	✓

Community and Support	Production	Pre-Production
Community Downloads (Fast Packs, Plugins, etc.)	✓	✓
Extensive Self-Learning Content on dynaLearn	✓	✓
Upgrade Path to New Product Versions	✓	✓

24x7 Access to {dt} Support	✓	✓
Open Source Agent Development Kit	✓	✓

Dynatrace AppMon 2017 May Release Notes

- The following features will not be available in the .NET OneAgnet:
  - GC/memory metrics for .NET < 4
  - .NET component sensor
  - .NET Winforms sensor
- Apache Cassandra 1.2/2.0 (EoL by Apache)
- IBM WebSphere MQ 7.1 (EoL by IBM Apr 30, 2017)
- IBM WebSphere MQ 7.1 for z/OS (EoL by IBM Nov 06, 2017)
- IBM WebSphere Message Broker 8 (EoL by IBM Apr 30, 2017)
- IBM WebSphere Message Broker 8 for z/OS (EoL by IBM Apr 30, 2017)
- JBoss AS CE 6.x/7.x ([EoL by JBoss]http://jbossas.jboss.org/downloads)

With the **next Dynatrace AppMon release**, support for the following platforms **will be discontinued:**

- Dynatrace Client 32-bit (including Webstart Client)
- Apache Cassandra 1.0/1.1 (EoL by Apache)
- Apache Active MQ 4.x (EoL by Apache)
- Apache HTTP Server 2.0 (EoL by Apache)
- Apache Tomcat 5.x (EoL by Apache)
- IBM AIX 6.1 (EoL by IBM Apr 30, 2017)
- IBM HTTP Server 6 (based on Apache HTTP Server 2)
- IBM WebSphere Message Broker 7 (EoL by IBM Sep 30, 2015)
- IBM WebSphere MQ 6 (EoL by IBM Sep 30, 2012)
- IBM zLinux RHEL 5.11 (EoL by RedHat Mar 31,2017)
- IBM zLinux SLES 10 (EoL by Suse Jul 31, 2016)
- IBM zOS 1.13 (EoL by IBM Sep 30, 2016)
- Mulesoft Mule 1.x and 2.x (EoL by Mule)
- Node.js 0.10/0.12
- Oracle Application Server 9i, 10g (EoL by Oracle Dec 2009)
- Oracle HTTP Server 10 (based on Apache HTTP Server 2)
- Oracle Weblogic Application Server 10 (JVM 5, EoL by Oracle May 31, 2015)
- Pivotal RabbitMQ 3.3 and 3.4 (EoD by Pivotal Feb 11th 2015)
- PostgreSQL 9.1 (EoL by Postgre Sep 2016)
- RedHat 5.11 (EoL by RedHat Mar 31,2017)
- RedHat 6.6 (EoL by RedHat Oct 31, 2016)
- SUSE Linux Enterprise Server 10 (EoL by Suse Jul 31, 2016)

The Dynatrace AppMon 2017 May release **discontinues support** for the following platforms and

components:

Platform support changes

[1] Premium support required (severe production problems of monitored applications only)

Product version	Release date	Support status	Full support until	Limited support[1] until
7.0	May 31, 2017	✓	November 30, 2018	Jun 1, 2019
6.5	September 26, 2016	✓	March 30, 2018	September 30, 2018
6.3	February 29, 2016	✓	August 31,2017	February 28, 2018
6.2	June 29, 2015	✗	December 31, 2016	June 30, 2017
6.1	December 9, 2014	✗		February 28, 2017
6	July 17, 2014	✗		October 31, 2015
5.6	January 15, 2014	✗		June 1st, 2017
5.5 and earlier		✗		

Support level and end of life overview

Dynatrace is fully supporting new AppMon versions for 18 months. 6 month extended support for critical production issues is available as part of the premium support offering. To better facilitate planning for updates and migrations the End of Life dates are now announced together with release dates.

End of Life policy

Support levels and end of life policy

[1] CTG for zOS: EXCI and IPIC CTG for Multiplatforms: IPIC (no support for TCPIP or SNA) CTG local mode: IPIC (no support for EXCI on z/OS)

[2] Inbound only

Technology	Versions	Support level
z/OS	1.12 / 1.13	Limited support
IMS	11 / 12	Limited support
DB2v9		Limited support
CICS TransactionGateway[1]	8.0	Limited support
IBM WebSphere MQ	7.0.1	Limited support
IBM IMS Soap Gateway[2]	2.2 / 3.1	Limited support
IBM IMS Connect API[2]	2.2 / 3.1	Limited support

Releases that are expected to work but are no longer supported by IBM

[1] CTG for zOS: EXCI and IPIC CTG for Multiplatforms: IPIC (no support for TCPIP or SNA) CTG local mode: IPIC (no support for EXCI on z/OS)

[2] Requires CTS 4.2 or above.

[3] Inbound only

Technology	Versions	Support level
z/OS	2.1 / 2.2	Certified
WebSphere MQ	7.1/8.0	Certified
WebSphere MQ	9.0	Supported

DB2	v10 / v11	Certified
DB2	v12	Supported
CICS Transaction Server	4.1 / 4.2 / 5.1 / 5.2 / 5.3 / 5.4 open beta	Certified
CICS Transaction Gateway[1]	8.1 / 9.0 / 9.1 / 9.2	Certified
CICS SOAP over HTTP		Certified
CICS DLI		Certified
CICS MQ Bridge[2]		Certified
IMS	13 / 14 / 15 QPP	Certified
IMS MQ Bridge[3]		Certified
IMS MQ Trigger Monitor		Certified
IMS MSC		Certified
IMS Shared Queues		Certified
IMS Soap Gateway[3]	3.2	Certified
IMS Enterprise Suite[3]	3.2	Certified
IMS Connect API[3]	3.2	Certified

IMS TM Resource Adapter	13 / 14 / 15 QPP	Certified

Mainframe environments

[1] *Supported for Android Auto Instrumentation.*

Technology	Versions	Support level
PhoneGap	2.9.1+	Supported
Apache Cordova	3.x	Supported
Custom WebViews		Supported
KonyOne on Android		Supported
OkHttp	3 (and OkHttp 3 based libraries like Retrofit 2)[1]	Supported

Frameworks

Technology	Versions	Support level
Apple iOS	6 - 10	Certified
Android	2.3 - 7	Certified

Supported OS for mobile

Technology	Versions	Support level
Internet Explorer	10+	Supported
Chrome	25+	Supported

Firefox	35+	Supported
Opera	15+	Supported
Android Webkit	4.4+	Supported
Internet Explorer Mobile	10+	Supported

W3C resource timing support

Technology	Versions	Support level
Safari	4+	Supported
Internet Explorer	8+	Supported
Firefox	3.5+	Supported
Chrome	10+	Supported
Opera	15+	Supported

CORS support

Technology	Versions	Support level
Mobile Safari	4+	Supported
Android Webkit	*2.3 - 3.x*	*Limited support*
Android Webkit	4.0+	Supported

Internet Explorer Mobile	Supported

Mobile browsers

[1] only load actions, no special frameworks.

Technology	Versions	Support level
Internet Explorer	11	Supported
Internet Explorer	*8, 9, 10*	*Limited support*
Firefox[1]	*3.5 - 27*	*Limited support*
Firefox	28+	Supported
Safari	5+	Supported
Chrome	10+	Supported
Opera	*12*	*Limited support*
Opera	15+	Supported
Embedded Browser		*Limited support*
Microsoft Edge		Supported

Browsers

Highlighted entries are compatible with any JavaScript enabled browser.

Browser support for UEM

Technology	Versions	Support level
backbone.js		Supported

The following frameworks are covered with the jQuery sensor:

Technology	Versions	Support level
GWT		Supported
zk		Supported
RichFaces		Supported
ASP.net AJAX		Supported
YUI		Supported
Ember.js		Supported

The following frameworks are covered with the basic XHR detection:

[1] Other versions of these frameworks still work, but certain measures might be missing. The JavaScript ADK can be used to capture the missing data.

[2] In certain cases the JS agent does not capture the correct end time for extjs 6 user actions if the promise API is used.

Technology	Versions	Support level
dojo	1.6.1 – 1.12.2[1]	Supported
extjs	3.4.0 – 6.2.0[1 2]	Supported
icefaces	2 & 3[1]	Supported
jquery	1.6 – 1.12.4, 2.0 – 2.4 & 3.0.0 – 3.2[1]	Supported

jquery	1.3 – 1.12.4, 2.0 - 2.4 & 3.0-3.2[1]	Supported
prototype	1.6.0 – 1.7.3[1]	Supported
angular	1.0 – 1.5.8, 2.0.0[1]	Supported
mootools	1.4.5 – 1.6.0[1]	Supported

UEM is compatible with all JavaScript frameworks. Automatic click detection is available for:

AJAX frameworks for UEM

Technology	Versions	Support level
Java Logging API		Certified
Log4J	2	Certified
Log4NET		Certified
Logback		Certified

Logging

Technology	Versions	Support level
JDK Http API		Certified
Apache HttpClient	3.1/4.x	Certified
Apache HttpAsyncClient	4.0.1	Certified
Jetty Http Client	8/9	Certified

NING Http Client	1.6.x / 1.7x / 2.x	Certified

HTTP Frameworks

[1] Java only, no Scala support

Technology	Versions	Support level
Hibernate		Supported
Struts	1.x/2.x	Supported
AWT/Swing/RCP/WPF/WinForms		Supported
Spring		Supported
Applets		Supported
Play[1]	1.3/2.2/2.3/2.4/2.5	Certified

Frameworks

Technology	Versions	Support level
IBM WebSphere	7/8/8.5.5	Certified
IBM WebSphere	9	Supported
Oracle WebLogic	8.1/9/9.1/9.2/10/11g/12	Certified
.NET WebServices		Certified
AXIS	1.4	Certified

AXIS2	1.3 / 1.4 / 1.5 / 1.6	Certified
Hessian	2.1/3/3.1/3.2/4.0	Certified
Oracle WebServices		Certified
JAX WebServices		Certified
Codehaus XFire	1.2	Certified
Apache CXF	2.2/2.3/2.4/2.5/2.6/3.0	Certified
Apache Wink	1.2.0 - 1.4.0	Certified
Restlet	2.1.0 - 2.3.7	Certified
RestEasy	2.3.5 - 3.0.19	Certified
Jersey	1.15/2.x	Certified
Spring Web Services	2.0	Certified
WebMethods Glue		Supported

Web Service Stacks

- *Place JMS sensor on sender and receiver*
- *No agent needed on TIBCO EMS itself*

[1] On Glassfish 4 and WildFly 8

[2] Client only

[3] Only for JMS 1.1

Technology	Versions	Support level
JMS	1.1/2.0[1]	Certified
IBM WebSphere MQ	7	Certified
Apache Active MQ	5.x	Certified
Microsoft MQ	3.0	Certified
RabbitMQ[2] Java & .NET	3.5.x/3.6.x	Certified
TIBCO EMS[3]		Certified
TIBCO Rendezvous	8.3.2 - 8.4.3	Certified

Messaging

Technology	Versions	Support level
RMI / IIOP		Certified
RMI / JRMP		Certified
RMI / T3		Certified
.NET Remoting		Certified
ORMI (OC4J 10g R3)		Certified

JBoss Remoting	Certified
WCF	Certified
HTTP Remoting	Certified
XML-RPC.NET	Certified
MS Service Fabric Remoting	Certified

Remoting

[1] On AIX, Windows & Linux 64bit

[2] HTTP communication only

Technology	Versions	Support level
IBM WebSphere Message Broker	8[1]	Certified
IBM Integration Bus[1]	9/10	Certified
Oracle Service Bus	10.3.4 / 12c (12.1.3)	Certified
JBoss ESB	4.12	Certified
WSO2	4.6[2]	Certified
Mule	3.0 - 3.8	Certified
TIBCO ActiveMatrix BusinessWorks Service Engine	5.8.2/5.9/5.10	Certified
Apache OpenEJB	3.1.4	Certified
SOA Software Service Manager 5.2/6.1		Limited support

SOA Software Service Manager 5.2/6.1 Limited support

ESBs and SOA

[1] *x86 and x64*

[2] *7 and 8 architectures*

[3] *Solaris SPARC*

[4] *No host health information available*

[5] *OS390x*

Technology	Versions	Support level
Windows[1]	10/8.1/8/7/Vista	Certified
Windows[1] Server	2008/2012/2016	Certified
Windows[1] Server	2008 R2 > 64 logical processors	Limited support
Suse Linux Enterprise Server[1]	11/12	Certified
RedHat Enterprise Linux AS[1]	6.7/7.0/7.1/7.2	Certified
RedHat Enterprise Linux AS Big Endian[2]	6.7/7.0/7.1/7.2	Certified
AIX[2]	7.1/7.2	Certified
Solaris[1, 3]	10/11/11.1/11.2/11.3	Certified
IBM zOS[4]	2.1/2.2	Certified
zLinux[5] SLES	11/12	Certified
zLinux[5] RHEL	6.6/6.7/7.0/7.1/7.2	Certified

On latest patch/service pack level

Platforms and operating systems

[1] Java Thrift based Clients (Hector 1.0/1.1, Astyanax 1.x) and Datastax 1.0.2/2.1.0 - 2.1.3, 2.1.5-2.19, 3.0.0, 3.0.2

[2] Java drivers only, v. 2.3 - 2.13.x/3.0.0-3.0.2

Technology	Versions	Support level
Cassandra	1.2.x[1], 2.0.1[1], 2.1.13[1], 3.0.3[1], 3.3.0[1], 3.4.0[1], 3.7.0[1]	Certified
MongoDB Server Versions	2.0.x, 2.2.x, 2.4.x, 2.6.x ,3.0[2]	Certified
Solr		Supported

NoSQL & big-data solutions

Technology	Versions	Support level
WebSphere Connection Pool	7 - 8.5.5	Certified
WebSphere Connection Pool	9	Supported
WebLogic Connection Pool	10-12.1	Certified
JBoss Connection Pool	5.x-8.0	Certified
Apache DBCP	1.0-1.4/2	Certified
Tomcat Connection Pool	1.0.6 - 1.1.0	Certified
c3p0	0.8.5 - 0.9.5.0	Certified

GlassFish	3.x-4.x	Certified
Hybris		Certified
generic support is available for all J2EE connection pools		Limited support

Connection pools

[1] JDBC Type4 drivers
[2] ADO.NET data providers

Technology	Versions	Support level
IBM DB2	8/9[1]	Certified
Oracle DB[1,2]	9i/10g/11g/12c	Certified
Microsoft SQL Server[1,2]	2008/2012/2014/2016	Certified
PostgreSQL[1]	9.2/9.3/9.4/9.5/9.6	Certified
Derby[1]		Certified
generic support is available for ADO.NET data providers		Limited support
generic support is available for any Type4 JDBC driver		Limited support

Databases

For DB Agents go to AppMon Agent section

Databases, connection pools, and big-data solutions

Technology	Versions	Support level
WebMethods Broker & Integration Server	9/9.5 - 9.10	Certified
Sun Java Web Server	6.1	Certified
Microsoft IIS/ASP.NET	7/7.5/8	Certified
Microsoft IIS/ASP.NET	8.5/10	Supported
ASP.NET Core	1.0/1.1	Supported
Microsoft SharePoint		Certified
MS COM+ Component Services for .NET		Supported
Adobe LiveCycle		Supported
other/proprietary Java/.NET Servers		Limited support

Others

[1] *only IBM JVM 7 on Windows, Linux & AIX*
[2] *only IBM JVM 8 on Windows, Linux & AIX*
[3] *not all JMX measures available*
[4] *only Fujitsu JVM 6/7 64bit*

Technology	Versions	Support level
WebSphere Application Server	7/8/8.5.5[1]	Certified
WebSphere Application Server	9[2]	Supported

WebSphere Liberty profile	8.5.5.5 - 16.0.0.2	Certified
WebLogic Application Server	11g/12	Certified
JBoss AS CE	6.0.0/7.1.1[3]	Certified
Wildfly	8.x/9.x/10.0	Certified
JBoss EAP	6.x/7.0.0	Supported
SAP NetWeaver	7.3 AS Java	Certified
Fujitsu Interstage	11[4]	Limited support
Glassfish	3.1/4 Open Source	Certified
Virgo/SpringSource dm Server	2.x	Supported

J2EE 6/7/8 compliant app servers

Technology	Versions	Support level
Apache Tomcat	6/7/8	Certified
Jetty	8/9	Certified

Servlet containers

Servlet containers & application servers

Other supported application environments and

technologies

Operating system	vSphere 4.0	vSphere 4.1	vSphere 5.0	vSphere 5.1	vSphere 5.5	vSphere 6.0	vSph
Windows	✓	✓	✓	✓	✓	✗	✗
Linux	✓	✓	✓	✓	✓	✗	✗

Supported Virtualization Monitoring

- http, https, Connect, Express, Restify
- Errors
- Code-level visibility, memory metrics, CPU metrics

Supported frameworks:

Operating system	Node.js 4.x	Node.js 5.x	Node.js 6.x	Node.js 7.x
Node.js 8.x	Linux	✓		✓
✓	(soon per update)	Windows	✓	
✓	✓	(soon per update)		

Supported Node.js environments

[1] mod_php (for Apache Web Server) and PHP CGI (for IIS).

[2] PHP FPM (fast process manager; for Nginx)

[3] PHP threaded MPM is not supported as it is not production setup safe. (reason: PHP_TS)

[4] Supported only via PHP OneAgent [5] As of 7.0.3

Operating system	PHP 5.3	PHP 5.4	PHP 5.5	PHP 5.6	PHP 7.0
PHP 7.1[5]	Windows (x86)	✓ 1,3	✓ 1,3	✓ 1,3	✓ 1,3
✗	✗	Linux (x86/x64)	✓ 1,2,3	✓ 1,2,3	✓ 1,2,3

✓ 1,2,3	✓ 1,2,3,4	✓ 1,2,3,4

Supported PHP versions

[1] x86 and x64 only.

[2] Solaris SPARC only.

[3] The NGINX agent is only available for builds where the debug symbols are available.

[4] Apache Web Server 2.2 only.

Operating system	Apache Web Server 2.2/2.4	Microsoft IIS 7/7.5/8/8.5/10	IBM HTTP Server (IHS) 7/8/8.5	Or
Windows	✓	✓	✓	✓
Linux[1]	✓		✓	✓
Solaris[2]	✓		✓	✓
AIX	✓		✓	✓
zLinux	✓ 4			

Supported web servers for UEM

[1] 32 and 64bit.

Operating system	.NET 3.5[1]	.NET 4.0[1]	.NET 4.5[1]	.NET 4.6[1]	.NET 4.7[1]	NET Core 1.0[1]	NET Core 1
Windows	✓	✓	✓	✓	✓	✓	✓

Supported .NET Platforms

[1] Only bytecode created by the Java Programming Language Compilers (javac) from supported

JVM vendors.

[2] 64bit only.

[3] x86 and x64 architecture.

[4] PPC 32 and 64 bit.

[5] Only SAP Java Server VM 6.1.038 1.6-b04 on Windows and SAP Java Server VM 6.1.018 / 6.1.034 19.1-b02 on Linux.

Operating system	Oracle JVM 8/7/6	IBM JVM 8/7/6	Oracle JRockit 6	SAP JVM 6.0/6.1[2,5] 64bit	OpenJDK
Windows	✓	✓	✓	✓	
Linux[3]	✓	✓[4]	✓	✓	✓
Solaris	✓		✓		
AIX		✓			
zLinux		✓			
zOS		✓			

Supported Java virtual machines[1]

For third-party software, AppMon follows the End of Life (EOL) timelines announced by respective software vendors related to specific platform and version support. This means that AppMon announces EOL for support of specific platforms and versions once the respective software vendors announce EOL.

Application environments and technologies

- Windows x64 2008 R2 / 2012 R2

- Linux x64

The zRemote Agent is supported on the following operating systems:

AppMon zRemote Agent

To determine disk-space requirements for the AppMon repository upfront, factor 15GB per 1,000 persistent measures (of all system profiles within the AppMon repository). You can find more details in the Deployment Guide.

AppMon supports partitioned tables for high-load scenarios for all supported databases. For all AppMon installations **greater than Large**, a partitioned database **is mandatory**. You can find more details in Performance Warehouse Partitioning for Dynatrace.

- Oracle 10g/11g/12c
- Microsoft SQL Server 2008 / 2012 / 2014 / 2016
- IBM DB2 Version 9.7 / 9.8 / 10.1 / 10.5 / 11.1
- PostgreSQL 9.2 / 9.3 / 9.4 / 9.5 / 9.6

The AppMon Performance Warehouse uses a relational database to store long-term historical data. By default, AppMon installs and uses an embedded database for demo/testing purposes. Any production installation of AppMon **must use one of the following** database management systems **with the specified version** to host the Performance Warehouse database:

AppMon Performance Warehouse

Drilldowns to the **AppMon Client** require a running AppMon Client. See the AppMon Client requirements for more details.

- Edge 14+
- Firefox 50+
- Chrome 55+
- Safari 10+

AppMon Web is supported and tested with the following browsers:

AppMon Web is built for viewing on **tablet devices** and **desktop machines** that have a minimum browser resolution of 1024 x 768 pixels (e.g., iPad mini physical resolution 2048 x 1536 vs. browser resolution 1024 x 768). More details for your device can be found here.

AppMon Web

- **Windows x64** 10/8.1/8/7/Vista or Server 2008/2012 with latest service pack (certified)
- **Linux x64** SUSE / RedHat Linux with GTK (certified)
  The Linux Client package requires a JRE 1.6 or later for installation. (A private JRE 1.8 for runtime is included!)
  Mac: OS X 10.8/10.9/10.10/10.11 64bit (supported)
- 500MB disk space
- 2GB memory
- Minimum screen resolution: 1024x768px
- The Java **Webstart Client** can be started with any Webstart compatible browser.
  - Internet Explorer 8/9/10/11 and Firefox 38-48 are certified by AppMon.
  - The Java Webstart Client requires a Oracle JRE >= 1.8.0_45.

AppMon Client

[1] Required memory for the Collector. If there are other processes running, then additional heap is required.

[2] No virtual cores (Hyper-Threading).

[3] Small = Default shipped Collector size

Deployment Size	Recommended Disk Space	Required Memory[1]	Physical Cores[2]
Small[3] 100 Java / 25 .NET Agents	30GB	2GB	1
Medium 200 Java / 100 .NET Agents	50GB	4GB	2
Large 400 Java / 250 .NET Agents	50GB	8GB	4

AppMon also provides different Collector sizes. For more information, see the Deployment Guide. For a proper configuration of the Collector(s) environment, see Set up Collectors.

Compatible JVMs for Collector runtime: Oracle (Sun) JVM 1.8+ (included with Windows and Linux installers), IBM JVM 1.8+.

- **Windows** x86/x64 10/8.1/8/7/Vista or Server 2008/2012/2016 with latest service pack

- **Windows** x86/x64 10/8.1/8/7/Vista or Server 2008/2012/2016 with latest service pack
- **Linux** x86/x64 with a recent kernel version 2.6 or higher
- **Solaris** SPARC or x86/x64 10/11/11.1/11.2/11.3
- **AIX** Power 7.1/7.2
- **zOS Unix** 2.1/2.2
- **zOS z/Linux** SLES 11/12 & RHEL 6.6/6.7/7.0/7.1/7.2

AppMon Collector

- No over-commitment of resources (CPU, memory, network, disk) and no virtual-only resources that are not backed by dedicated physical resources.
- Large and XLarge only. VM is equivalent to a physical box in all performance metrics. For more details see Guidelines for running AppMon in VMWare.
- XLarge only. Database partitioning is mandatory.

All deployment sizes listed above may be deployed on a virtual machine. However, you must consider the following:

VM support

> **Note**
>
> To improve performance, AppMon always uses the maximum available CPU and memory resources.
> If you have more CPU cores or memory than required for the selected size, then AppMon uses them as well.

[1] The maximum agent numbers consider a typical combination of different agents types (Java, .NET, host, WebServer, PHP, etc.).
In special circumstances, significant more agents are supported (e. g. mainly host agents). Please contact support for guidance and confirmation.

[2] No virtual cores (Hyper-Threading), for Medium and above a clock speed of 2.6 GHz or higher is required.

[3] Required memory for the AppMon Server and operating system. If there are other processes running, then additional heap is required.

Deployment size[1]	Physical cores required[2]	Recommended physical memory[3]	Also available memory options	Recommended disk space	A
Demo 5 agents | 150 transactions/sec	2	2GB	4GB, 8GB	1.5GB + 100GB Continuous session storage/week	N
Small 10 agents | 300 transactions/sec	4	4GB	8GB, 16GB	3GB + 200GB Continuous session storage/week	Y
Medium 250 agents | 1,500 transactions/sec	8	8GB	16GB, 32GB	5GB + 1TB Continuous session storage/week	Y
Large 750 agents | 3,500 transactions/sec	16	16GB	32GB, 64GB	10GB + 2TB Continuous session storage/week	Y
XLarge **32GB** | 1600 agents | 5,000 transactions/sec	24	32GB	-	20GB + >2TB Continuous session storage/week	N
64GB | 1800 agents | 5,800 transactions/sec	24	64GB	-	20GB + >2TB Continuous session storage/week	N
96GB | 2000 agents | 6,500 transactions/sec	24	96GB	-	20GB + >2TB Continuous session storage/week	N

See the Deployment Guide for detailed hardware recommendations specific to your load.

Hardware requirements

- **Windows x64**: 10/8.1/8/7/Vista or Server 2008/2012/2016 with latest service pack; installer (.msi) does not need a pre-installed JRE; package contains a private Java Runtime Edition v8.
- **Linux x64**: With a recent kernel version 2.6 or higher.
  Installation requires a pre-installed Java Runtime Edition v8.
- **LDAP**: For AppMon user authentication.
  - Active Directory
  - openLDAP

The AppMon Server (backend process / Windows service / *NIX daemon) is only supported on operating systems that support Java 8. Only the included Java Runtime Edition is supported.

AppMon Server

- Supported agent platforms and operating systems
- Oracle (Sun) JVM, IBM JVM, JRockit JVM, Azul JVM

- 10MB disk space
- Dynatrace AppMon bootstrap agent 6.x
- DB Agent supports Oracle Database 11g/12c, Microsoft SQL Server 2012/2014/2016 and MySQL 5.6/5.7

AppMon Agent

Dynatrace AppMon components

System requirements & supported technologies

There are currently no known issues. See also our Knowledge Base for workarounds and tips.

Known issues

Later versions will be included in following 2017 May public updates. If you are missing required changes, contact support before scheduling the upgrade to clarify and resolve.

- Public update 6.5.18 and earlier (not affected by JLT-192329)
- Public update 6.3.25 and earlier
- All updates released for versions 6.2 and earlier

This release together with its latest public update contains all changes of:

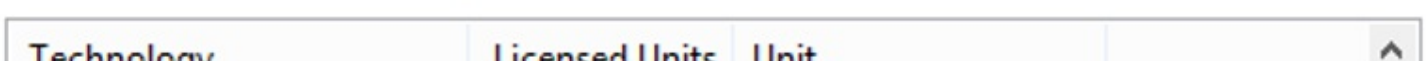
2017 May resolved issues

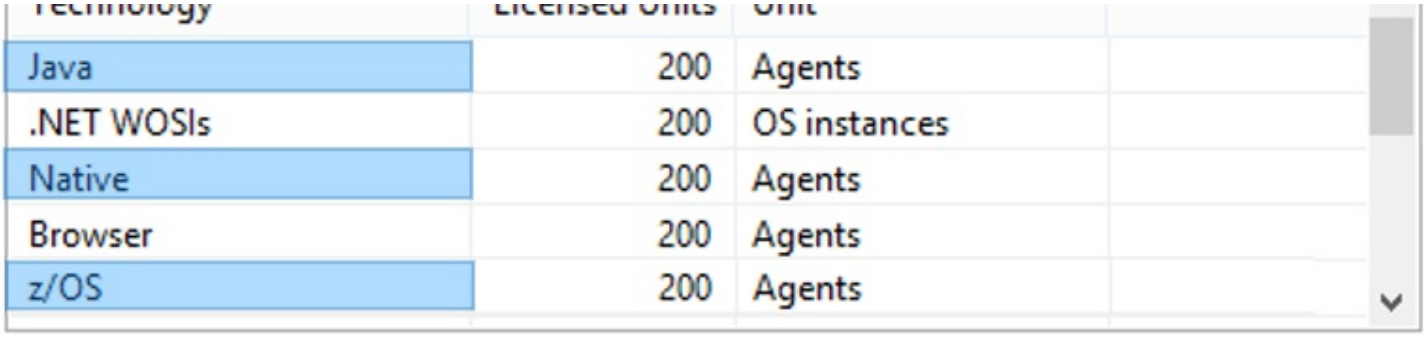
Resolved issues in 2017 May

This release contains a large number of quality improvements. Most of them are also back-ported to older versions and delivered through updates.

Resolved issues

Resolved and known issues





Technology	Licensed Units	Unit
Java	200	Agents
.NET WOSIs	200	OS instances
Native	200	Agents
Browser	200	Agents
z/OS	200	Agents

AppMon for z/OS still uses a MIPS-based licensing model, but **Compuware's License Management System (LMS) has been deprecated**. Beginning with the 6.3 release, z/OS licenses are now only available through eServices. IMS and CICS regions (sub agents) need to be listed as z/OS agents. Java and native agents running on z/OS are now also licensed through the Dynatrace AppMon server.

Following a license upgrade, ensure that your current license is deactivated. If you're using UEM, ensure that the correct amount of UEM volume is reissued in the form of UEM vouchers.

Due to the extensive list of new features and improvements in AppMon 2017 May, it's **necessary that you obtain a new license** to migrate from a previous version. You can upgrade an existing license on your own using our eServices license management portal. For full details, see the license upgrade documentation. If you require support or have questions, our licensing management team is happy to help you with your current license update.

General information

- All-in-one: Visits and hours can be part of the license now! No need for additional volume voucher any more.
- The license file extension is now *.lic* instead of *.key*, which is often blocked by mail servers.
- If your contract is already based on perpetual agent units which aren't restricted to editions or technologies, you are candidate for our 100% licensing self-service. Please contact your CSM or sales representative if you have any questions. This self-service approach provides many advantages, including:
  - Move agent units between editions and technologies with ease!
  - Create, assign, clone or terminate licenses on your own.
  - Choose either a technology bound license (such as 7 x Java, 3 x .NET, or 5 x web server) or a technology independent online agent unit based license (for example, 13 agent units).

Starting with this release, the following improvements and simplifications are available:

Licensing redefined: Simpler, more flexible and 100% self-service

Licensing redefined: Simpler, more flexible and 100% self-service

Licensing update

- New to Dynatrace AppMon and UEM? See the Getting Started Task Overview.
- Advanced user with specific use cases? Quick links offer guidance for specific roles and use cases: developers, testers, business, operations, continuous delivery, and user experience.
- Free trial user looking for a basic install? See our Free Trial and Evaluation Guide.
- Setting up a complete installation in a live environment? See the new Installation Overview.
- Upgrading from a previous version? See our rebuilt Upgrade and Migration Guide.

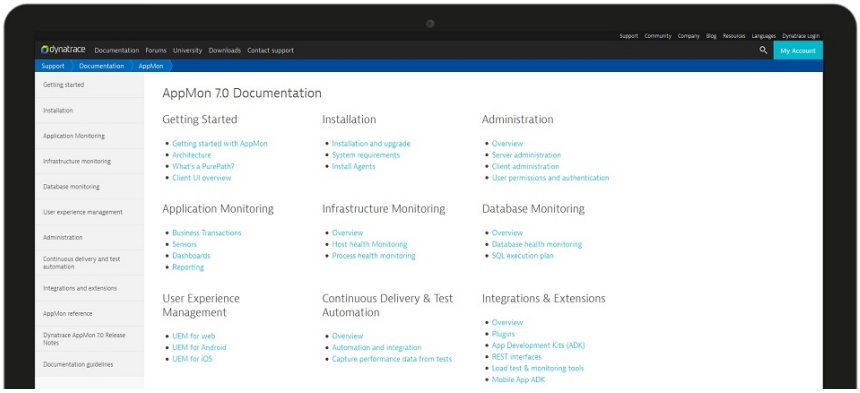
No matter your role or use case, the Getting Started guide has you covered:

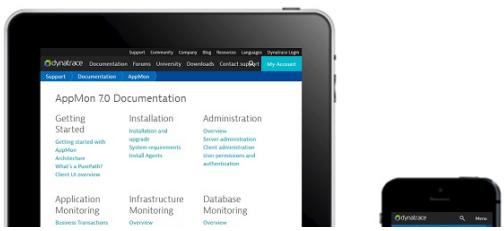
Upgrading and getting started with this release

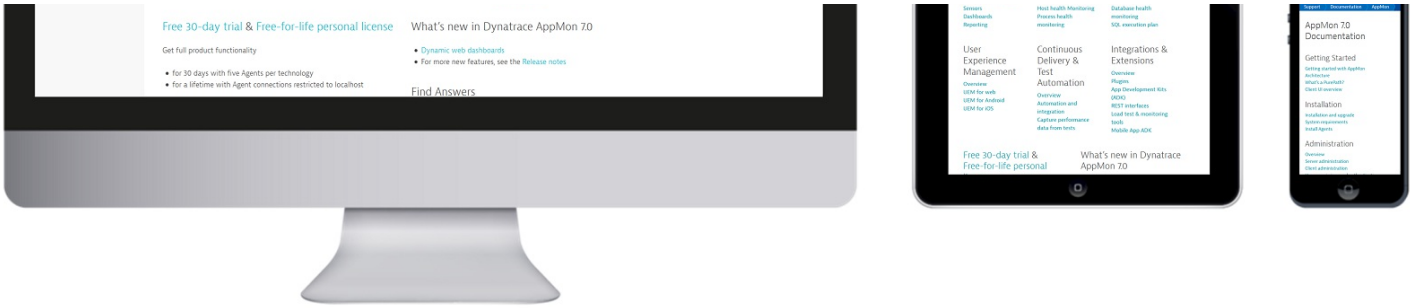
- **Salesforce Commerce Cloud Fastpack**: Provides deep insights into Salesforce Commerce Cloud.
- **Team Foundation Server Plugin**: Integrates AppMon with Team Foundation Server and Visual Studio Team Services enabling you to view AppMon test automation results in your TFS/VSTS builds.
- **Gradle Plugin**: Enables full automation of AppMon with Gradle tasks.
- **REST Monitoring Plugin**: Parses the result of an HTTP call and extracts one or more measures from the XML or JSON response.

For deeper insights, install **Community Plugins** for the technologies you use:

- Documentation has not only improved a lot in layout and navigation but also in speed. Have a look at our **super fast documentation**.
- New **responsive design** makes it possible to access the documentation on the way.
- Check out **AppMon & UEM Support** where you'll find answers to your AppMon questions.



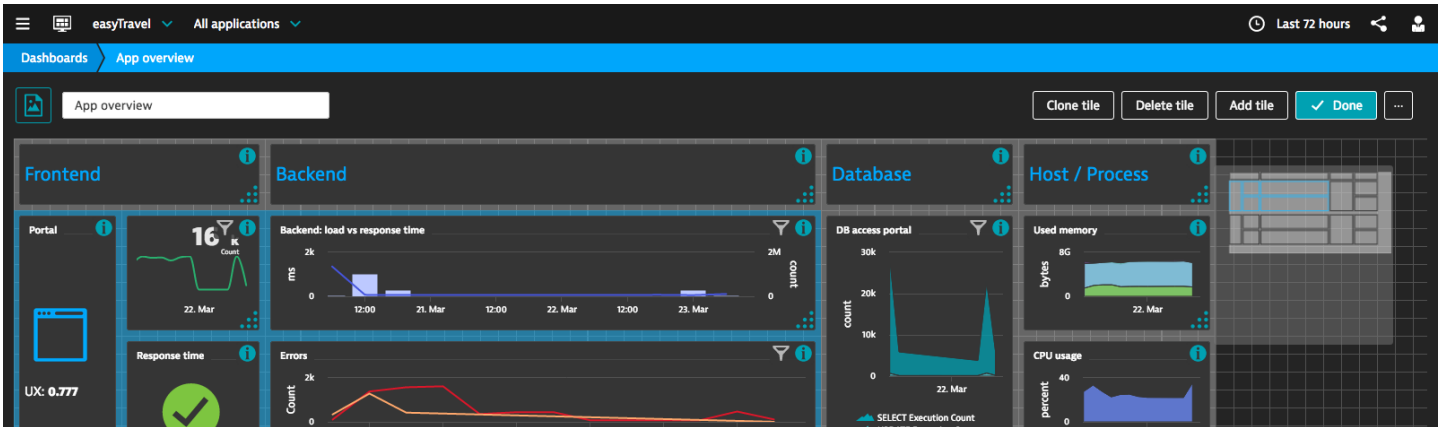


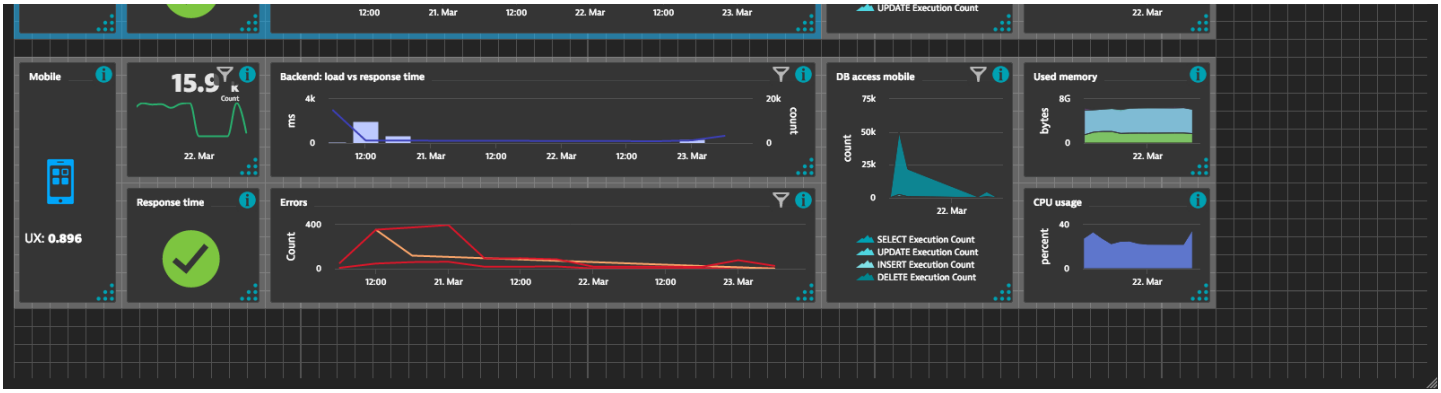


Fast and mobile friendly documentation!

- More structure in web dashboard measure picker
- Markdown and measurement tile (similar to text and measurement tile in AppMon Client)
- More Permissions for dashboards (share with specific users)
- Increase tile and splitting limit (max. 40 tiles/dashboards and max. 40 splittings/tile)
- PurePath detail view
- Waterfall view for user actions
- Database view
- Text and measure dashlet enhancements
- Ability to display measure data points in a table
- Transaction flow as diagram in report
- Drill down capabilities to text and measure dashlet
- Dynatrace AppMon with Angular JS
- Ignore long running (polling) web requests that might impact the user action response time
- Restrict UEM reporting user actions when an action request target URL matches a predefined pattern

Based on forum suggestions and discussions, and feedback gathered at user group events and customer meetups, you steadily contribute to our common success. This release contains 100+ improvements, like:





Why your feedback matters

- Application server: WebSphere AS 8.5.5/9 on AIX, WebMethods 9.7-9.10
- Web service stacks: RestEasy 3.0.7-3.0.19, Restlet 2.3.2-2.3.7
- Messaging: RabbitMQ 3.5/3.6 .NET client
- UEM: AngularJS, Angular 2 and React
- Windows Server 2016, IBM DB2 LUW, zOS IMS Enterprise Suite 3.2 and Mule 3.6/3.7

Restlet React REST Easy ANGULARJS

z/OS IMS Enterprise Suite

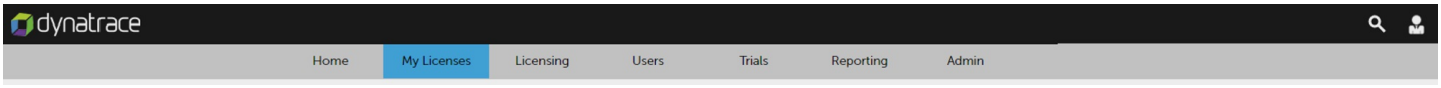
WebSphere Application Server

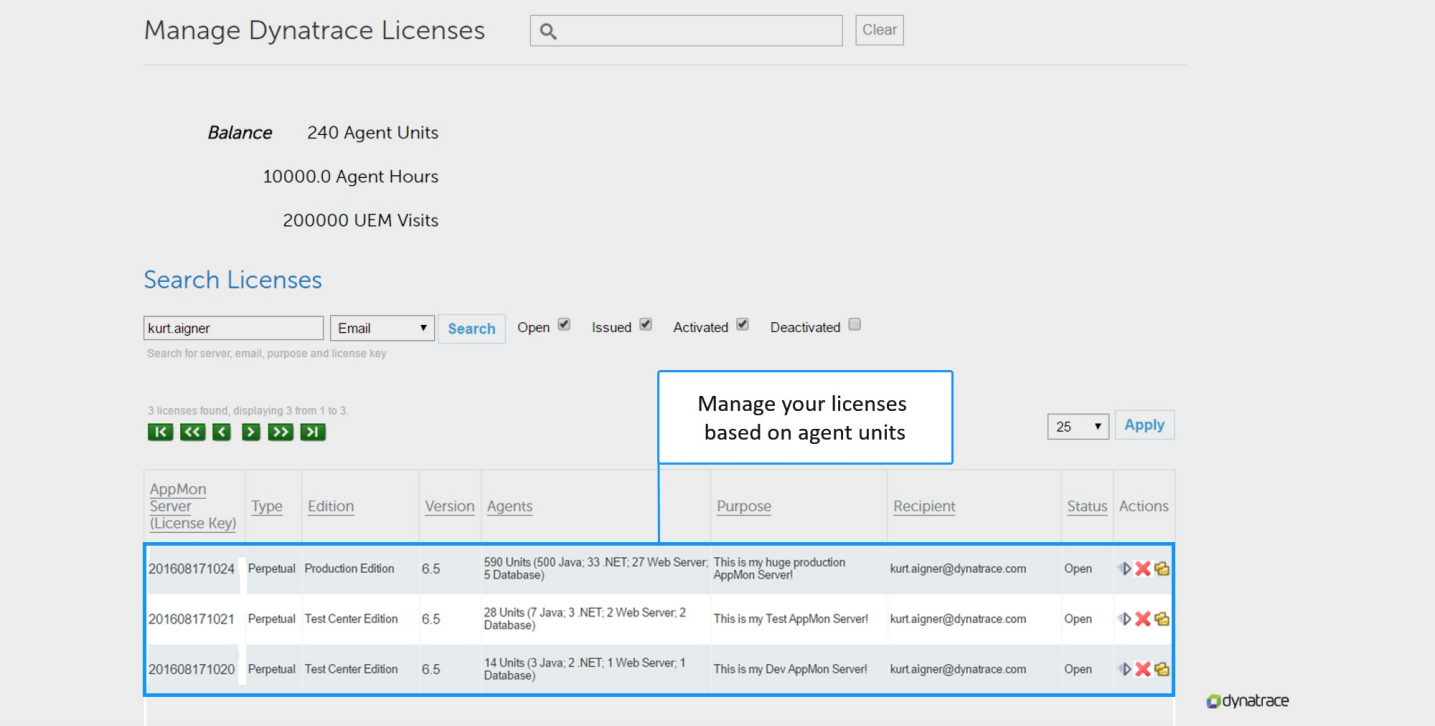
Windows Server 2016

IBM DB2

play RabbitMQ jetty://

AngularJS, IBM DB2, React and much more

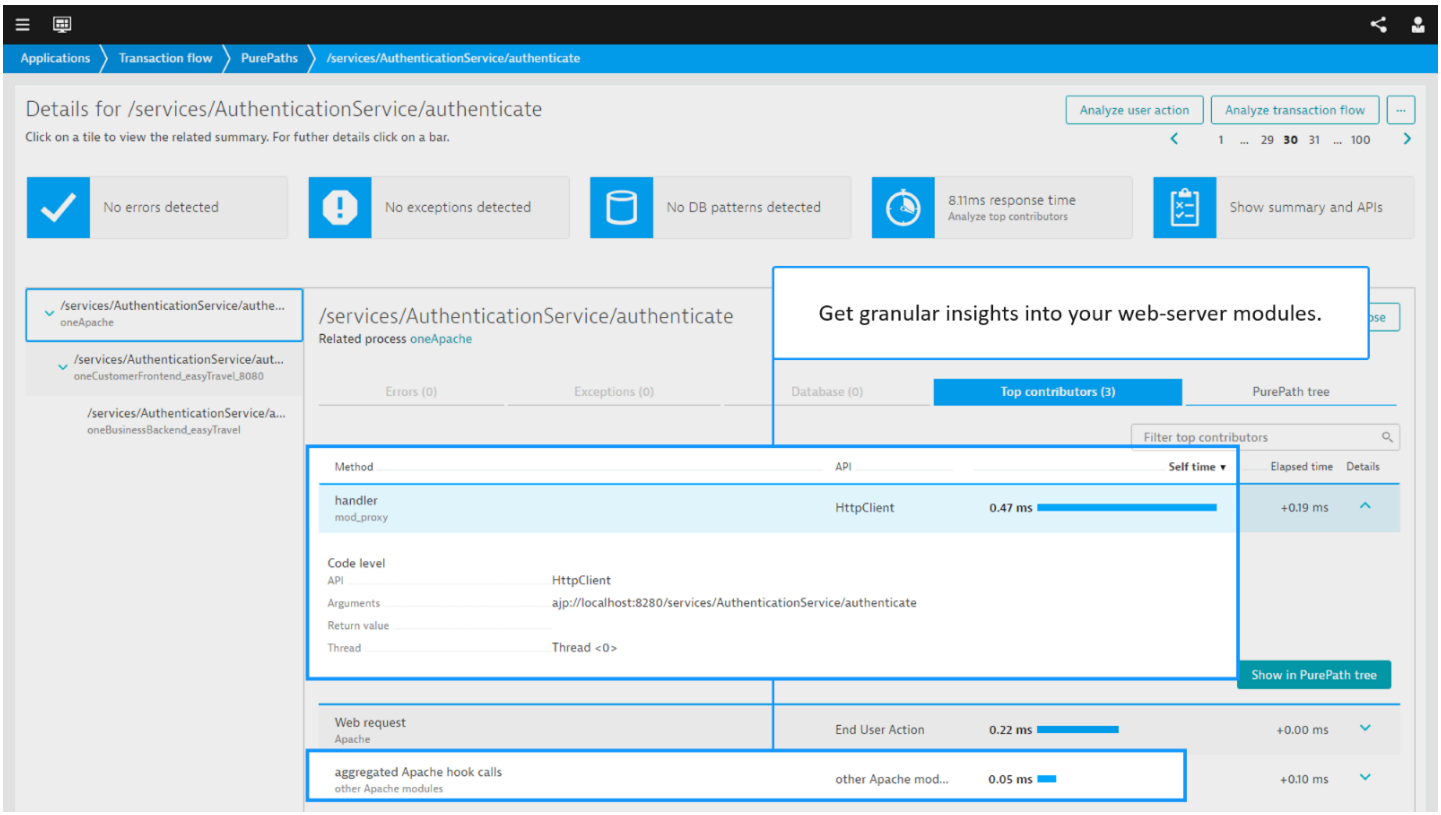


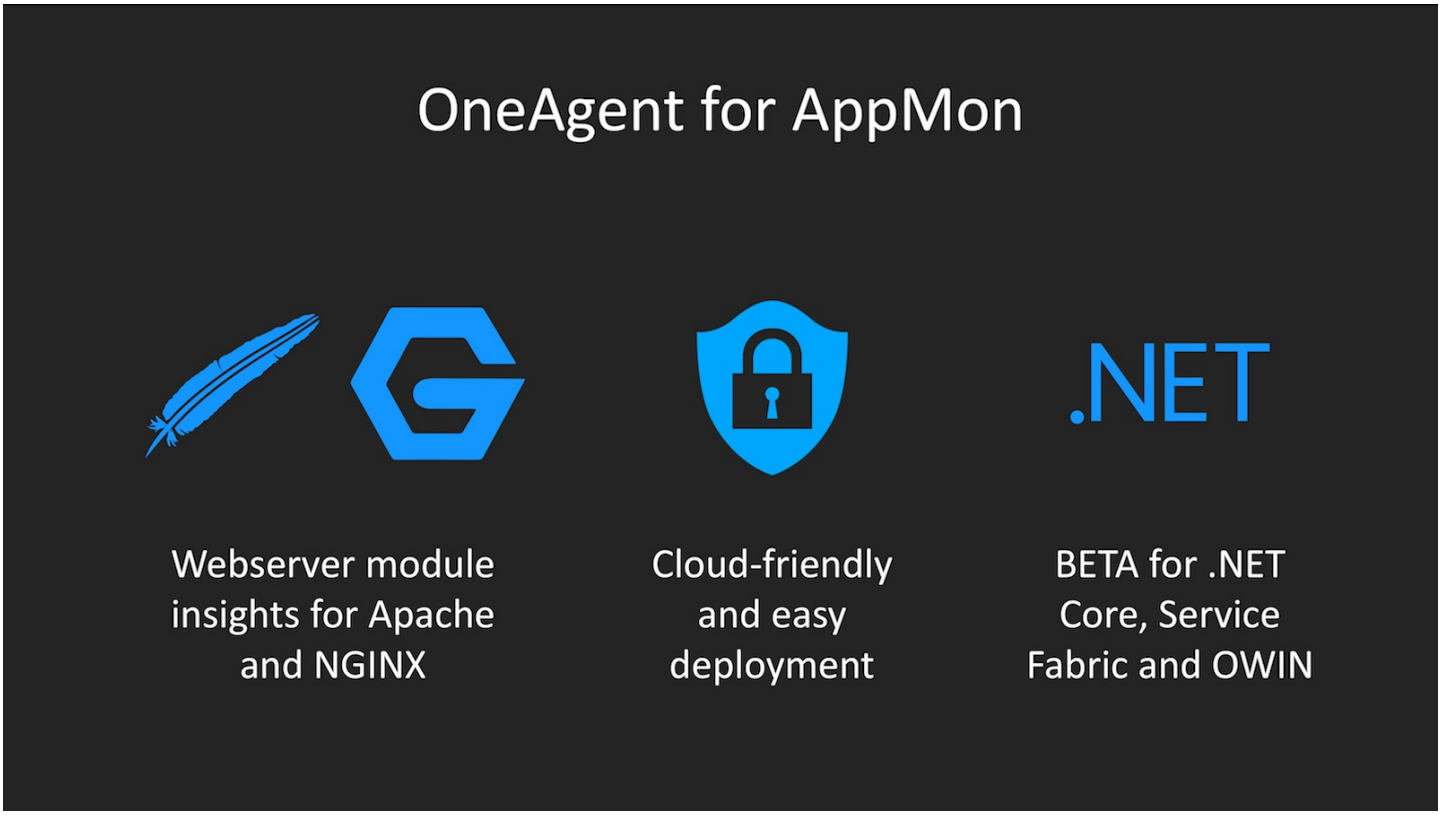




- Starting with this release, encryption between Agent and Collector is available!
- AppMon also provides now a SAML V2 based SSO[beta] (single sign on) authentication for the AppMon Client and the WebUI.
- Easily move agent units, agent hours or UEM visits between AppMon 2017 May servers, editions and technologies with self-service licensing.

Enterprise ready end-to-end encryption, SSO and Licensing





- .NET Core on Windows

- ASP.NET sensor for OWIN/Katana
- Microsoft .NET Service Fabric sensor

Furthermore, **BETA** support is introduced for:

- Insights into webserver modules to spot performance and functional issues.
- Streamlined deployment. Master agents are not required anymore.
- Cloud and firewall friendly setup with http(s)-based transmission of captured data.

Switching is easy, but not mandatory, and provides:

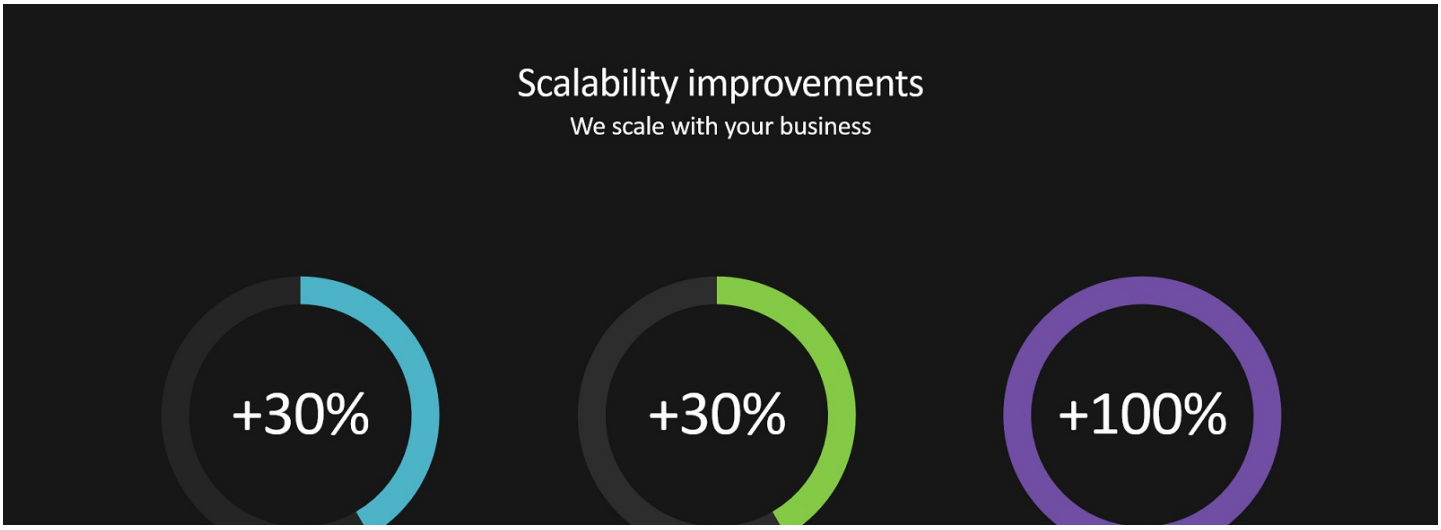
This release introduces two agents fuelled by the new OneAgent-based platform: Apache and NGINX.

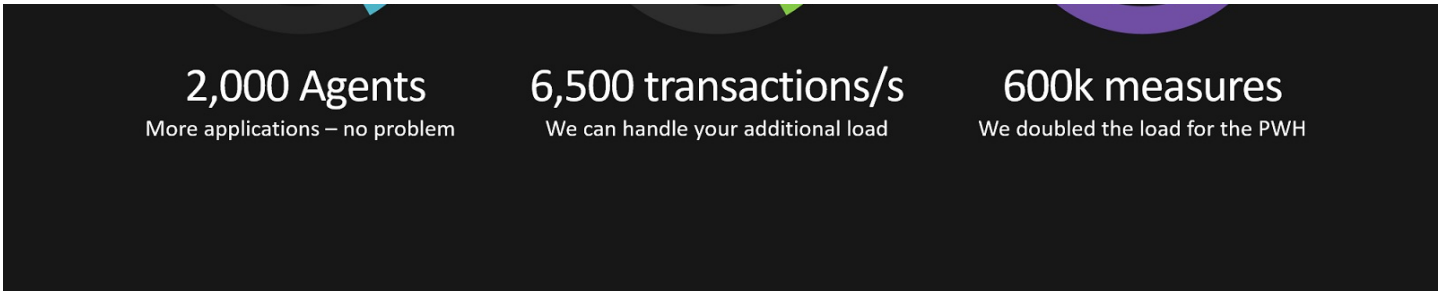
Webserver module insights and BETA for .NET Core, powered by OneAgent

Stay tuned for even more architectural and scalability enhancements in the next release! See the new Deployment Guide for detailed hardware recommendations specific to your load.

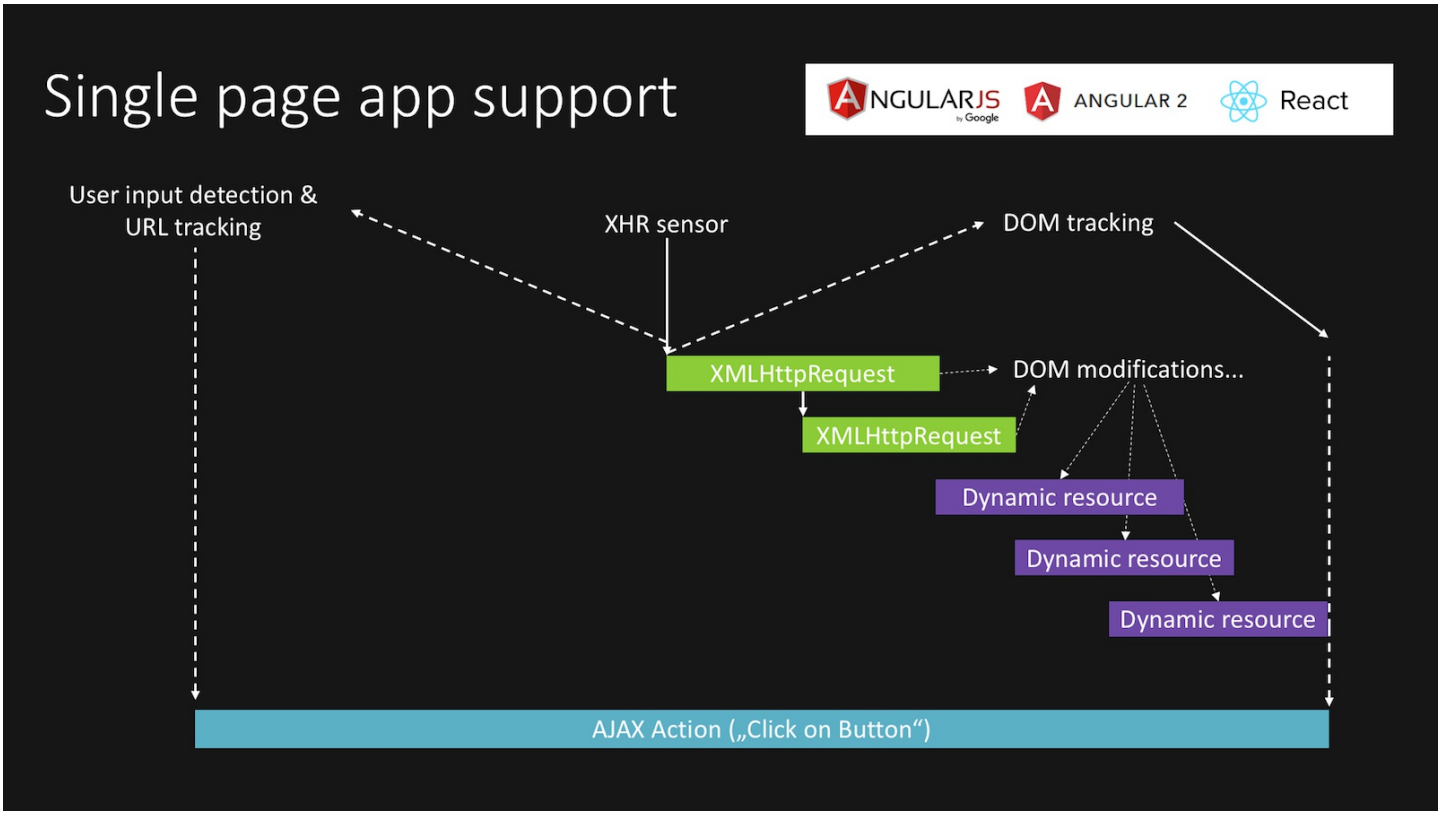
- This release introduces support for **+30% more agents (2,000 agents).** With that you can scale your business and application environment to get the insight you need.
- With this release, we've leveled up the maximum number of transactions per second on a single server that AppMon can support. We've pushed the limit to **+30% transactions/sec**.
- The Performance Warehouse is now able to digest **+100% more measures**.

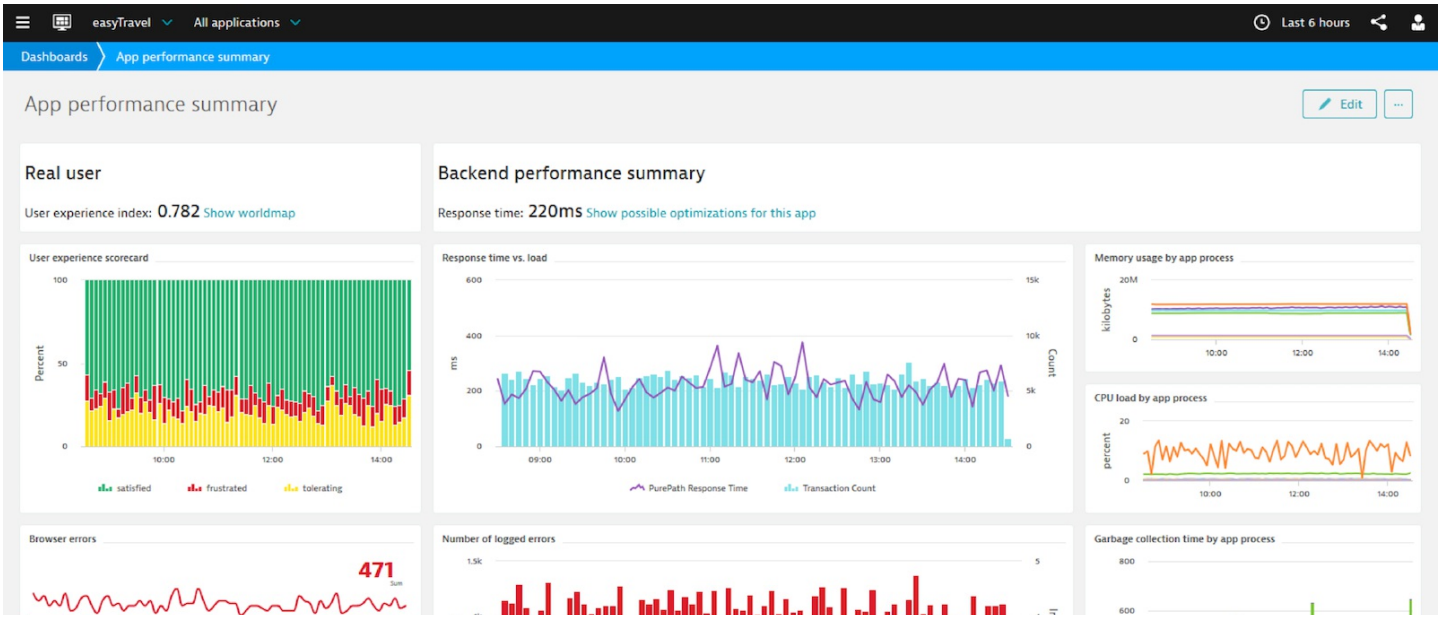
Ready to scale up with Dynatrace AppMon? We're happy to announce that our recent architectural advances are now bearing fruit:



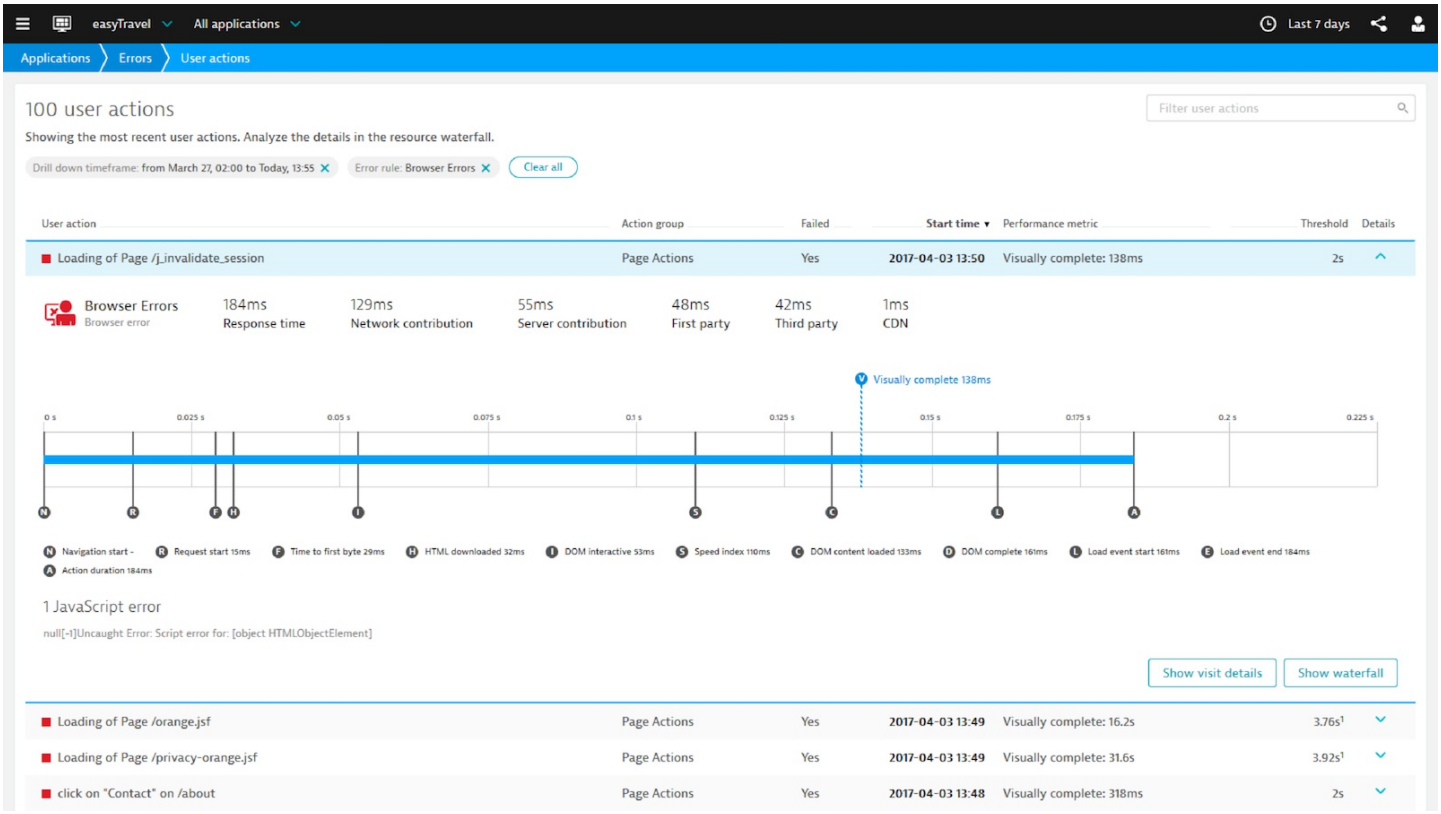


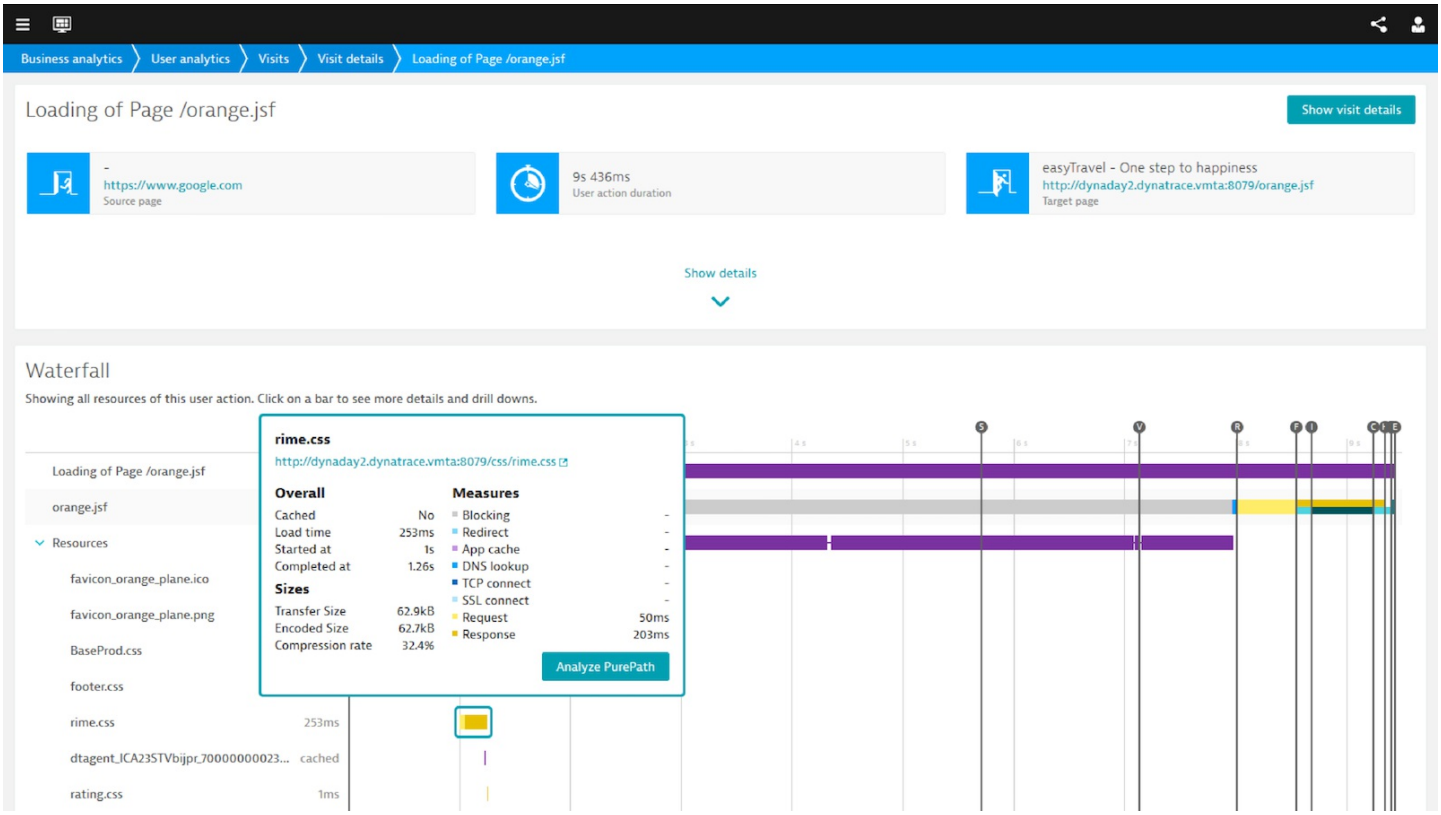
Need more? Don’t worry, we scale with your business





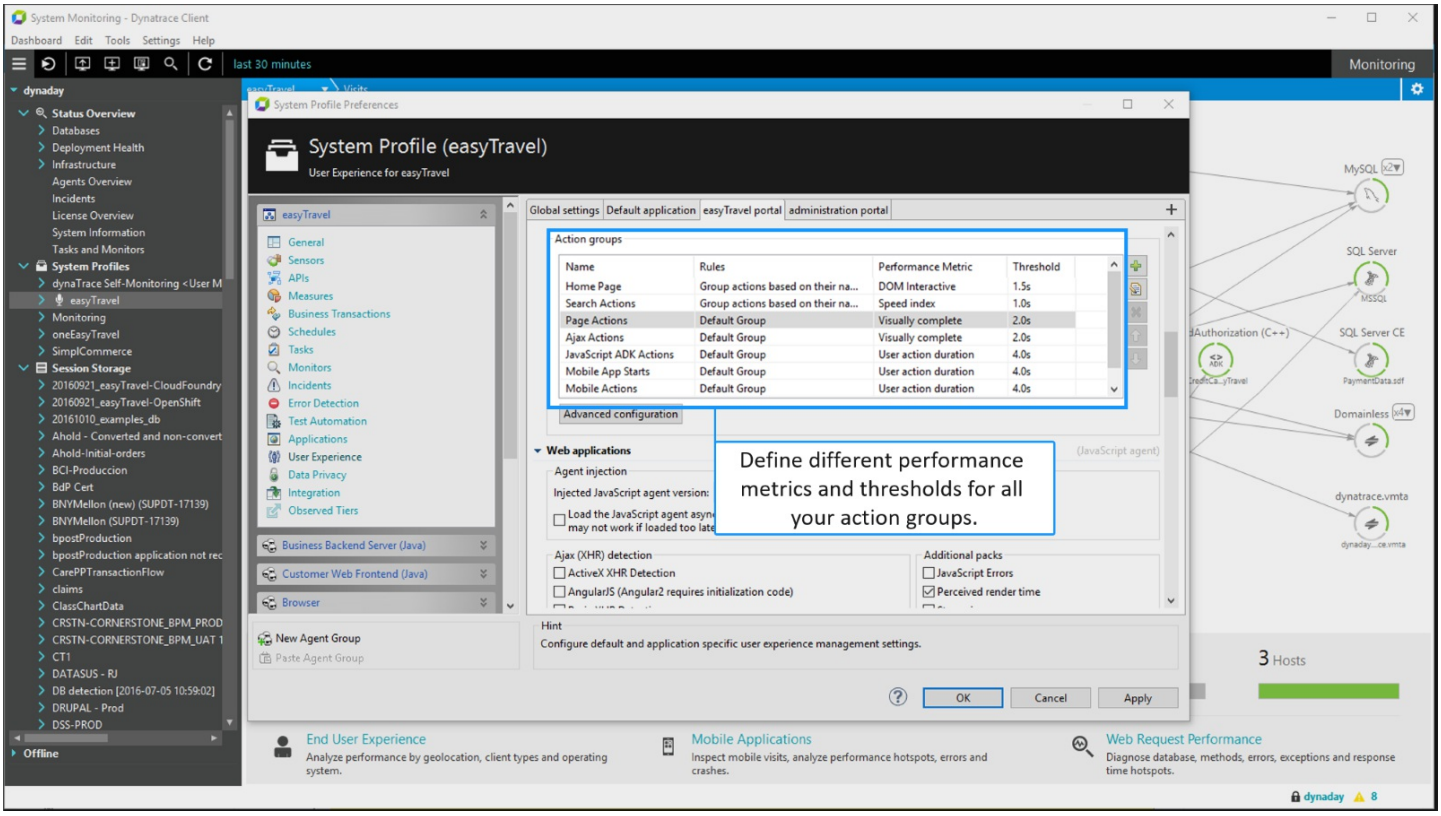

User action fails on HTTP 5xx
125
Count
Exceptions
Important logs
ms

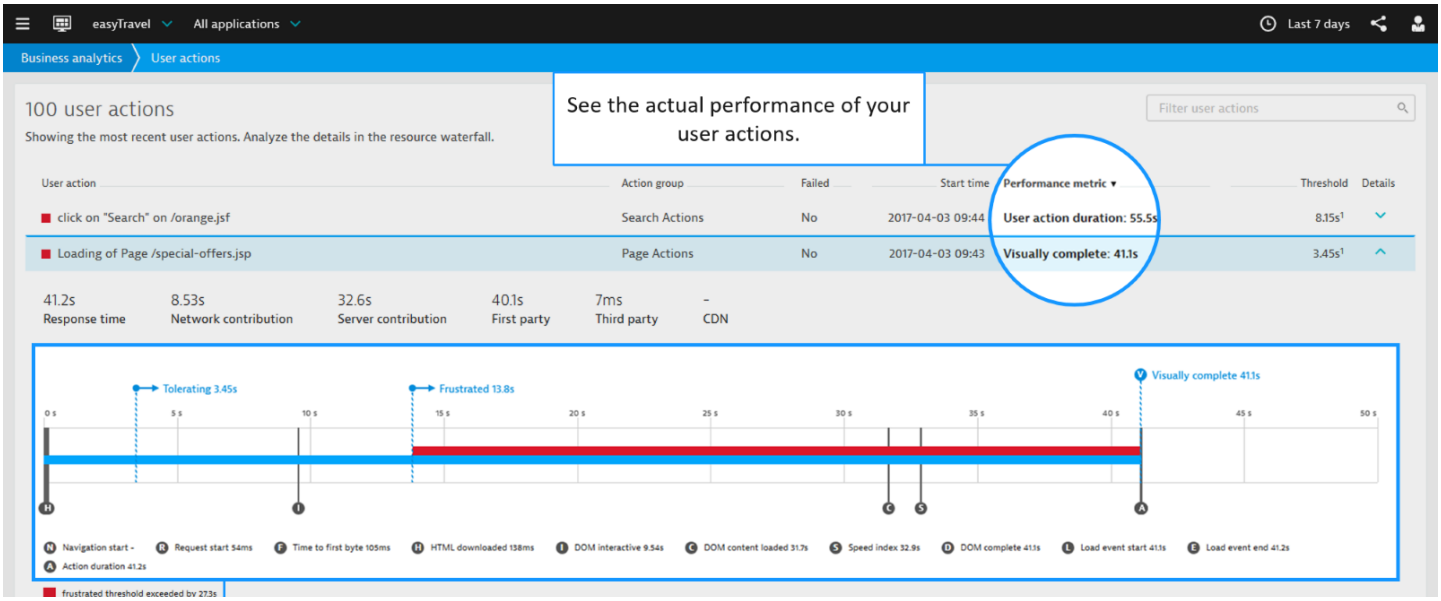

easyTravel
All applications
Last 7 days
Applications
Errors
User actions
100 user actions
Showing the most recent user actions. Analyze the details in the resource waterfall.
Filter user actions
Drill down timeframe: from March 27, 02:00 to Today, 13:55
Error rule: Browser Errors
Clear all
User action
Action group
Failed
Start time
Performance metric
Threshold
Details
Loading of Page /j_invalidate_session
Page Actions
Yes
2017-04-03 13:50
Visually complete: 138ms
2s
Browser Errors
Browser error
184ms
Response time
129ms
Network contribution
55ms
Server contribution
48ms
First party
42ms
Third party
1ms
CDN
Visually complete 138ms
1 JavaScript error
null[-1]Uncaught Error: Script error for: [object HTMLObjectElement]
Show visit details
Show waterfall
Loading of Page /orange.jsf
Page Actions
Yes
2017-04-03 13:49
Visually complete: 16.2s
3.76s
Loading of Page /privacy-orange.jsf
Page Actions
Yes
2017-04-03 13:49
Visually complete: 31.6s
3.92s
click on "Contact" on /about
Page Actions
Yes
2017-04-03 13:48
Visually complete: 318ms
2s

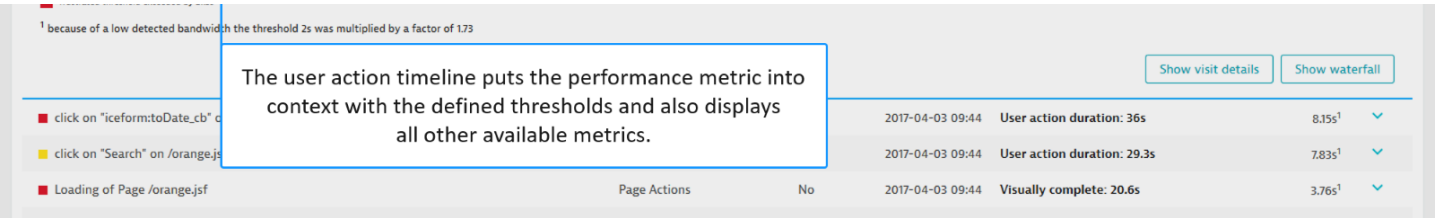

Business analytics
User analytics
Visits
Visit details
Loading of Page /orange.jsf
Loading of Page /orange.jsf
Show visit details
https://www.google.com
Source page
9s 436ms
User action duration
easyTravel - One step to happiness
http://dynaday2.dynatrace.vmta:8079/orange.jsf
Target page
Show details
Waterfall
Showing all resources of this user action. Click on a bar to see more details and drill downs.
Loading of Page /orange.jsf
orange.jsf
Resources
favicon_orange_plane.ico
favicon_orange_plane.png
BaseProd.css
footer.css
rime.css
253ms
dtagent_ICA23STVbijpr_70000000023...
cached
rating.css
1ms
rime.css
http://dynaday2.dynatrace.vmta:8079/css/rime.css
Overall
Cached No
Load time 253ms
Started at 1s
Completed at 1.26s
Sizes
Transfer Size 62.9kB
Encoded Size 62.7kB
Compression rate 32.4%
Measures
Blocking -
Redirect -
App cache -
DNS lookup -
TCP connect -
SSL connect -
Request 50ms
Response 203ms
Analyze PurePath

- Gain valuable insight into your application's performance with the new **waterfall chart**.
- Better **single page application support** than ever delivers spot-on timing and uses route-names for action naming.
- **Beacon signal compression** minimizes the performance impact on your application.
- Start monitoring your mobile apps with Cocoa pods for iOS and Gradle plugin for Android Studio or use our new Dynatrace Cordova Plugin to monitor your hybrid apps.

User experience management redefined



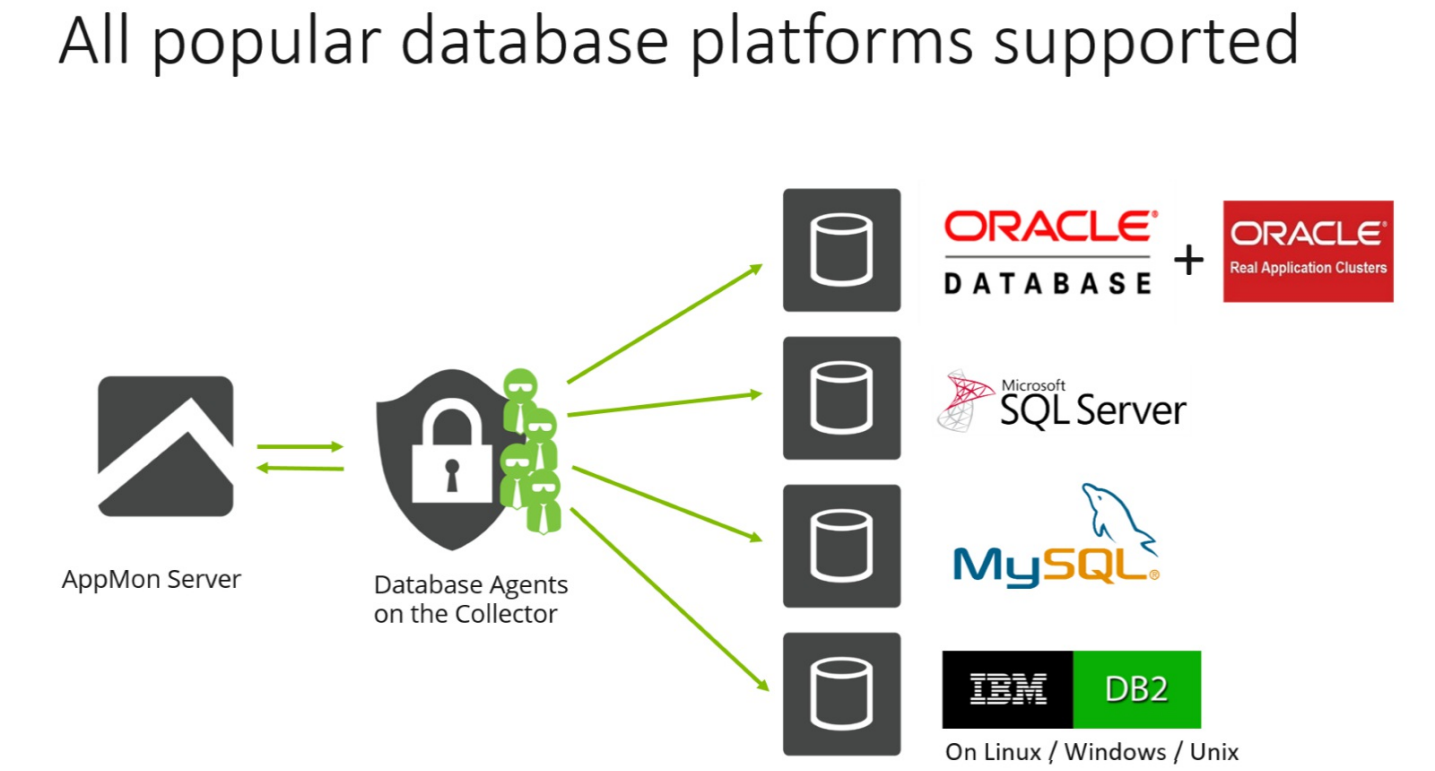


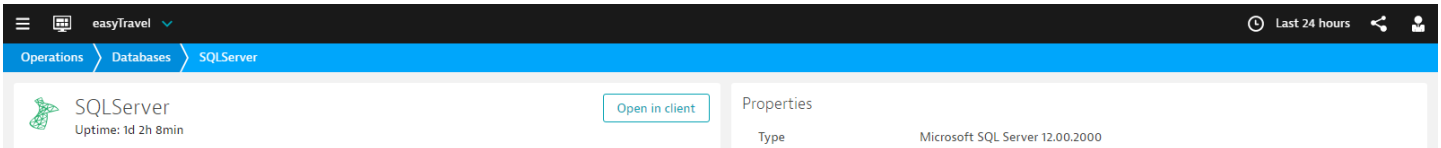


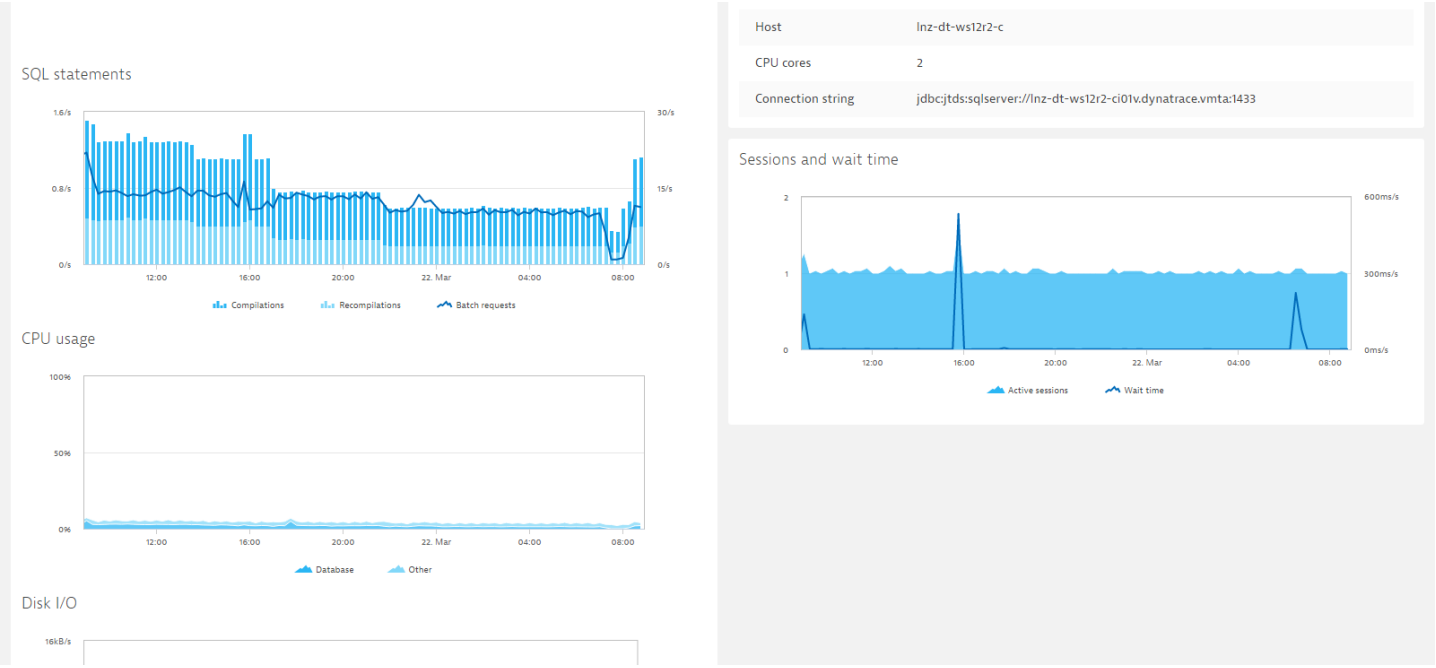
- **Visually complete**: The time at which all content in the browser's visible area is fully rendered.
- **Speed index**: The average time at which visible parts of the page display. A lower speed index means that most parts of the page rendered very quickly.
- Streamlined support for **W3C timings** (navigation, resource, user) across all Dynatrace products.
- Choose the **key performance metric** that reflects your application's performance, whether it's DOM interactive, visually complete, or user action duration.

Traditional web performance optimization is mainly focused load times and resource sizes. While these numbers are important, they do not reflect the actual user experience. This is why we introduced the new metrics ***visually complete*** and ***speed index***. **Learn how they work.**

The best metrics to optimize user experience



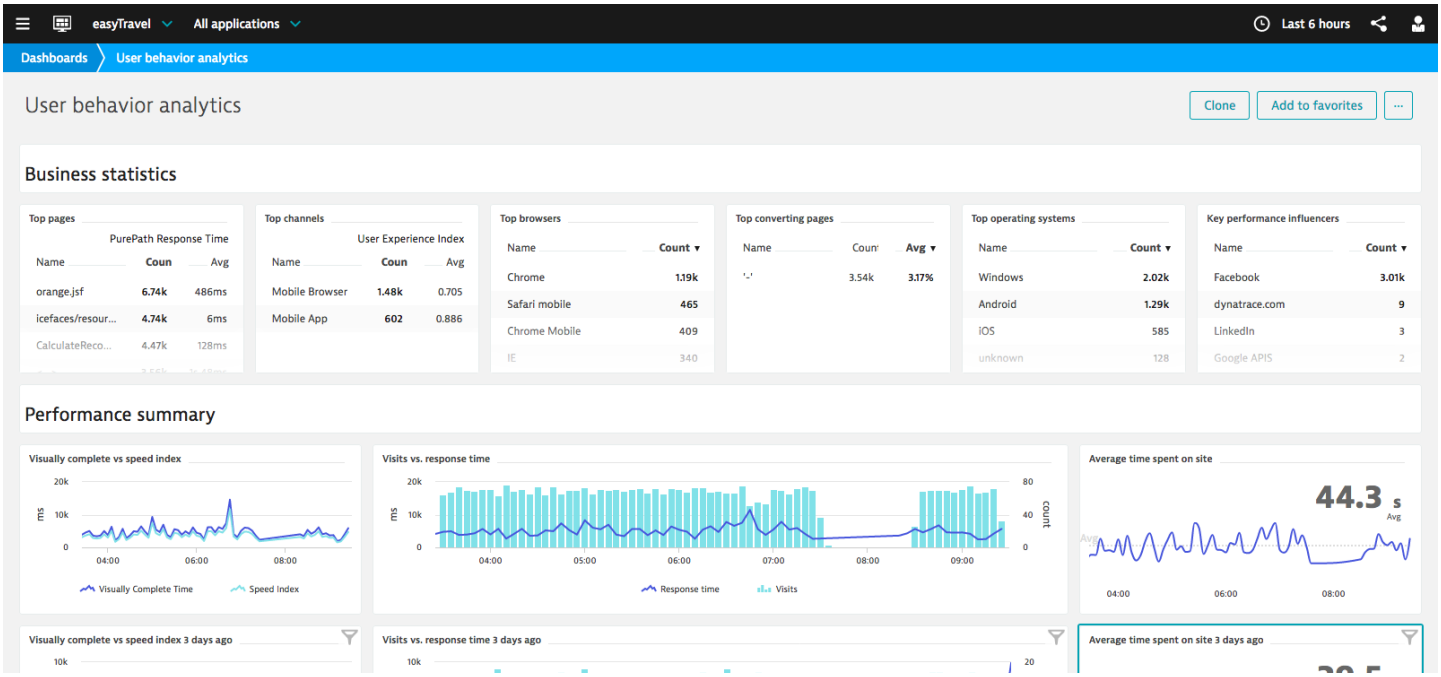


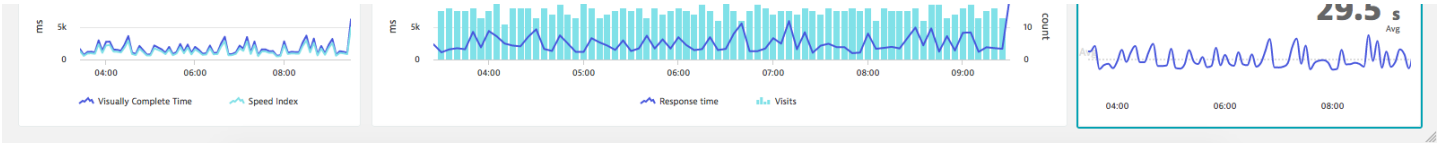


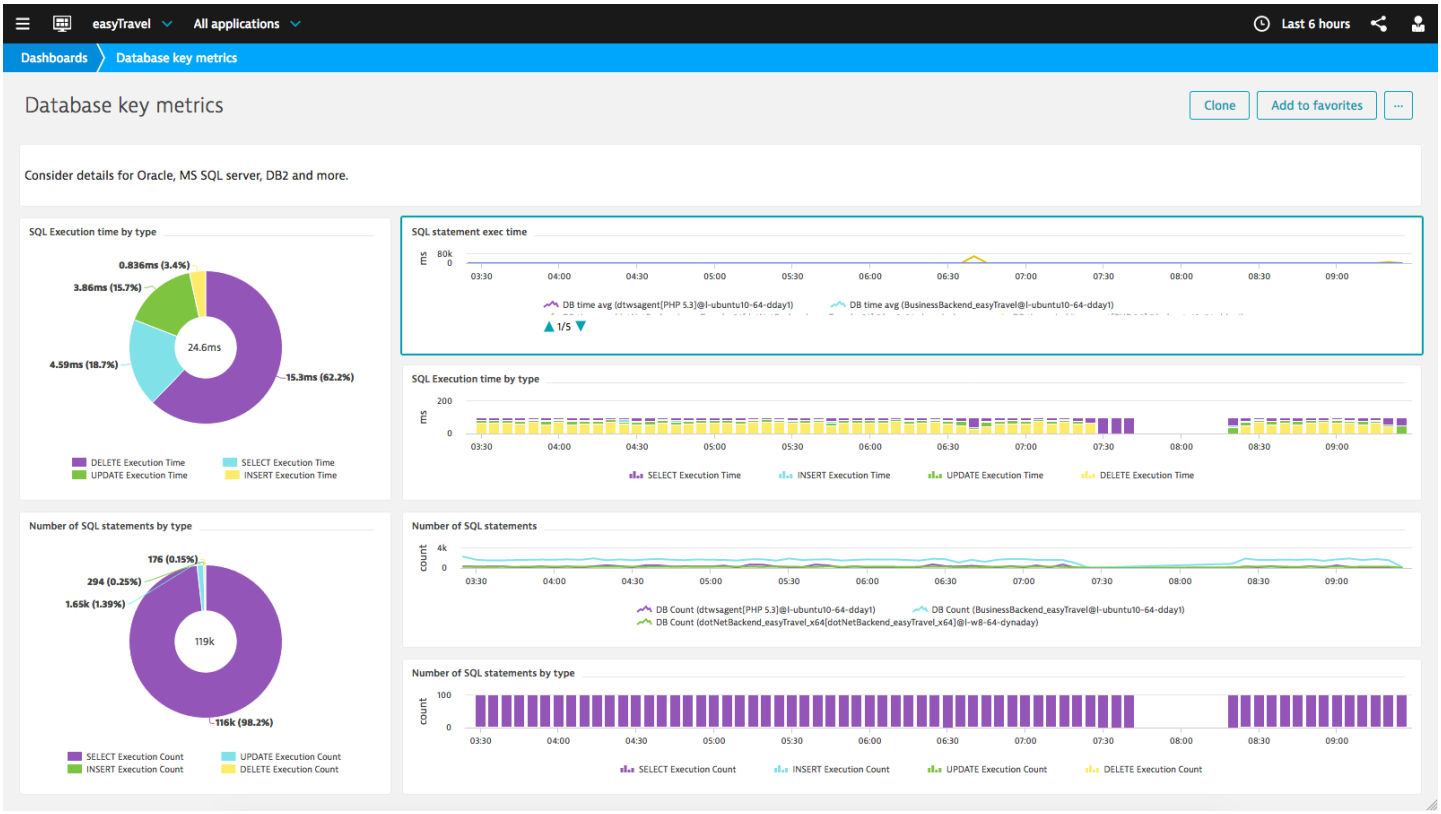
- **IBM DB2 LUW** support and much more
- **More accessible:** All the information your full stack developers need is available in the web for easy access and collaboration.

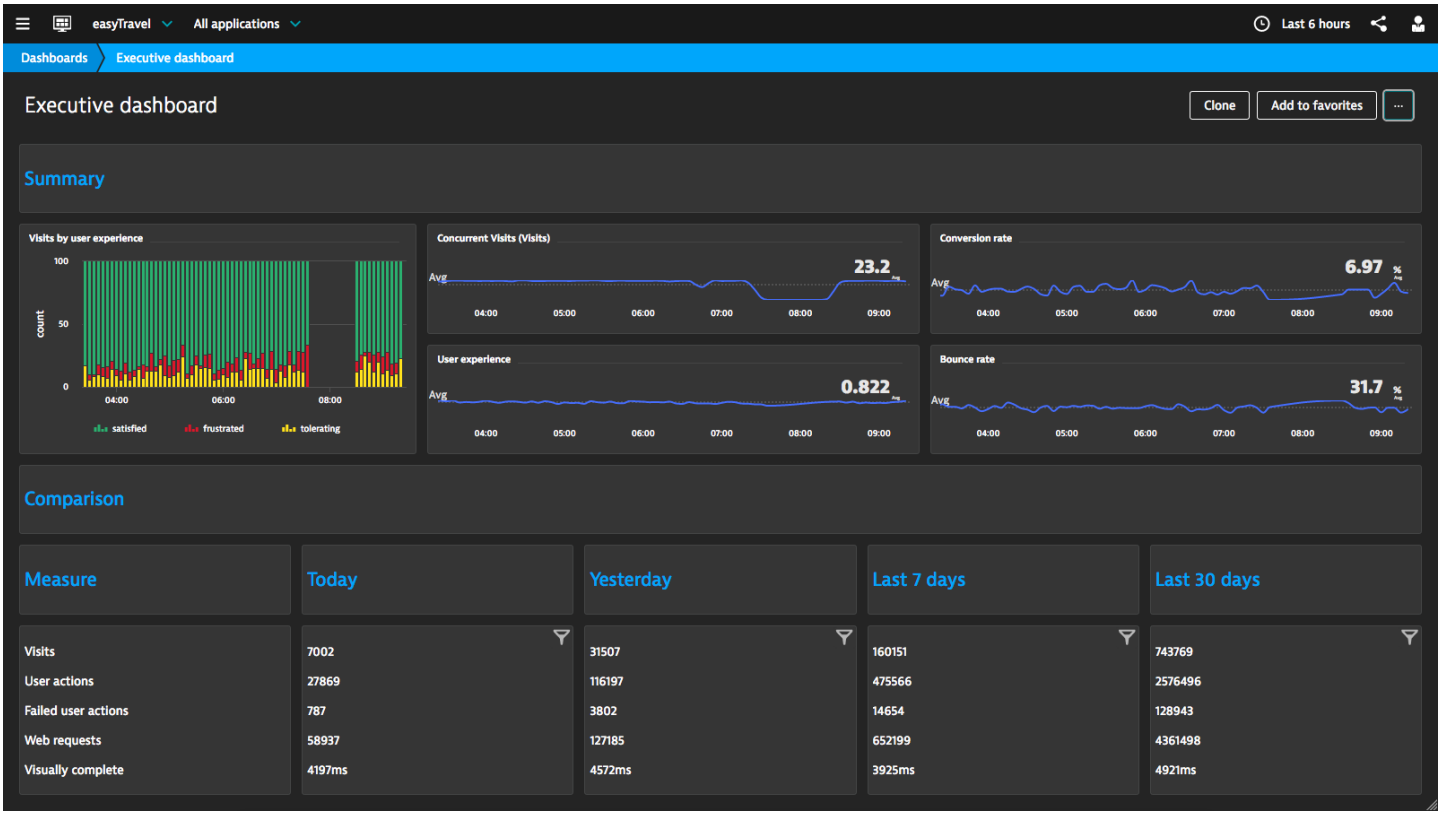
80 percent of typical database problems can be solved by your application teams before the DBAs turn on them. With the broad support for database engines, the database agent provides substantial insight across the application's databases.

Database insight for every team




ms
5k
0
04:00
06:00
08:00
Visually Complete Time
Speed Index
ms
5k
0
10
count
0
04:00
05:00
06:00
07:00
08:00
09:00
Response time
Visits
29.5 s
Avg
04:00
06:00
08:00


easyTravel
All applications
Last 6 hours
Dashboards
Database key metrics
Database key metrics
Clone
Add to favorites
Consider details for Oracle, MS SQL server, DB2 and more.
SQL Execution time by type
0.836ms (3.4%)
3.86ms (15.7%)
24.6ms
4.59ms (18.7%)
15.3ms (62.2%)
DELETE Execution Time
SELECT Execution Time
UPDATE Execution Time
INSERT Execution Time
SQL statement exec time
ms
80k
0
03:30
04:00
04:30
05:00
05:30
06:00
06:30
07:00
07:30
08:00
08:30
09:00
DB time avg (dtwsagent[PHP 5.3]@l-ubuntu10-64-dday1)
DB time avg (BusinessBackend_easyTravel@l-ubuntu10-64-dday1)
1/5
SQL Execution time by type
ms
200
0
SELECT Execution Time
INSERT Execution Time
UPDATE Execution Time
DELETE Execution Time
Number of SQL statements by type
176 (0.15%)
294 (0.25%)
1.65k (1.39%)
119k
116k (98.2%)
SELECT Execution Count
UPDATE Execution Count
INSERT Execution Count
DELETE Execution Count
Number of SQL statements
count
4k
0
DB Count (dtwsagent[PHP 5.3]@l-ubuntu10-64-dday1)
DB Count (BusinessBackend_easyTravel@l-ubuntu10-64-dday1)
DB Count (dotNetBackend_easyTravel_x64[dotNetBackend_easyTravel_x64]@l-w8-64-dynaday)
Number of SQL statements by type
count
100
0
SELECT Execution Count
INSERT Execution Count
UPDATE Execution Count
DELETE Execution Count


easyTravel
All applications
Last 6 hours
Dashboards
Executive dashboard
Executive dashboard
Clone
Add to favorites
Summary
Visits by user experience
count
100
50
0
04:00
06:00
08:00
satisfied
frustrated
tolerating
Concurrent Visits (Visits)
Avg
23.2
04:00
05:00
06:00
07:00
08:00
09:00
Conversion rate
Avg
6.97 %
User experience
Avg
0.822
Bounce rate
Avg
31.7 %
Comparison
Measure
Today
Yesterday
Last 7 days
Last 30 days
Visits
User actions
Failed user actions
Web requests
Visually complete
7002
27869
787
58937
4197ms
31507
116197
3802
127185
4572ms
160151
475566
14654
652199
3925ms
743769
2576496
128943
4361498
4921ms

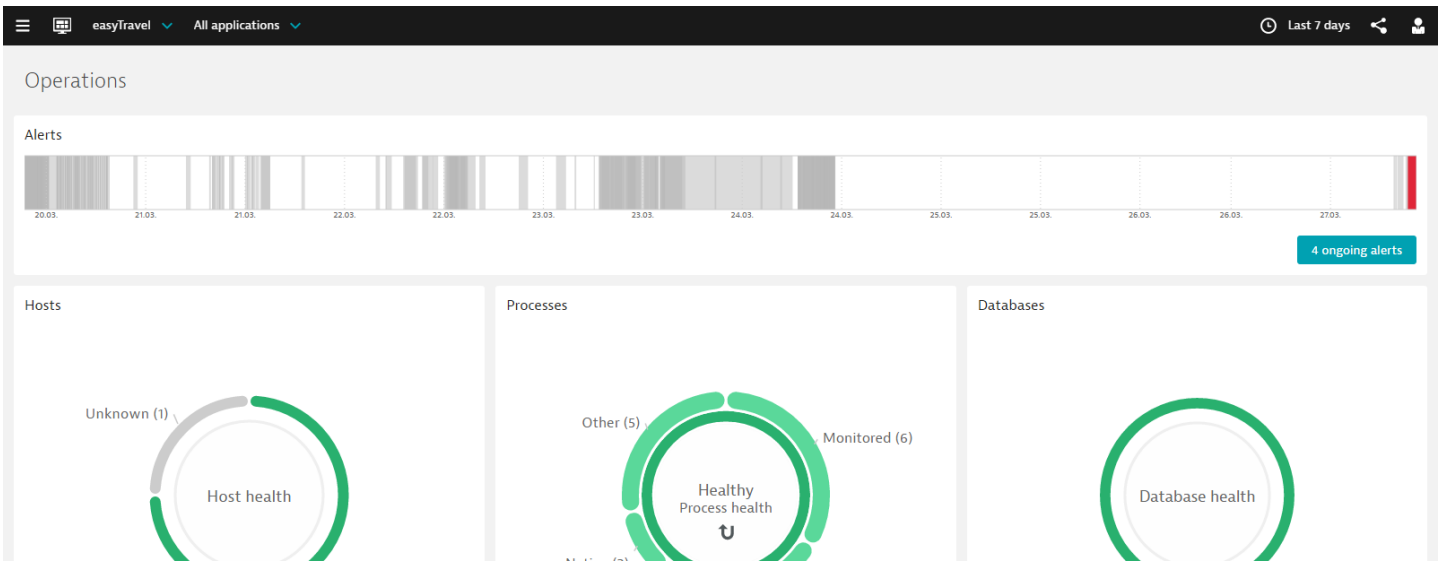

easyTravel
All applications
Last 72 hours

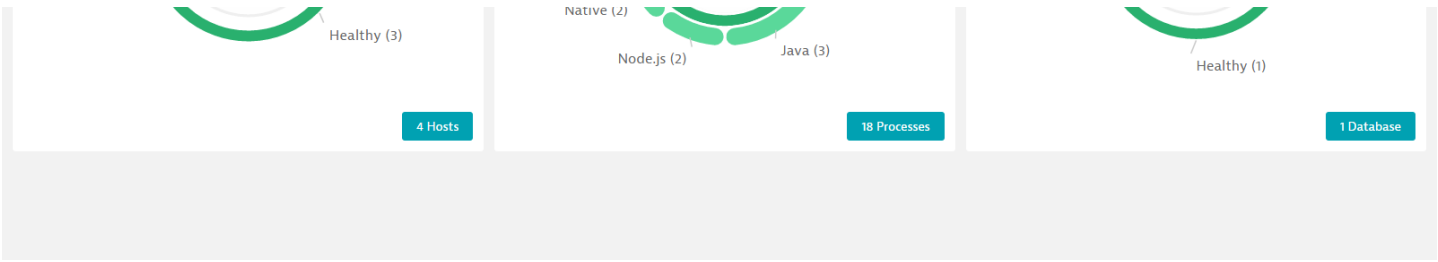


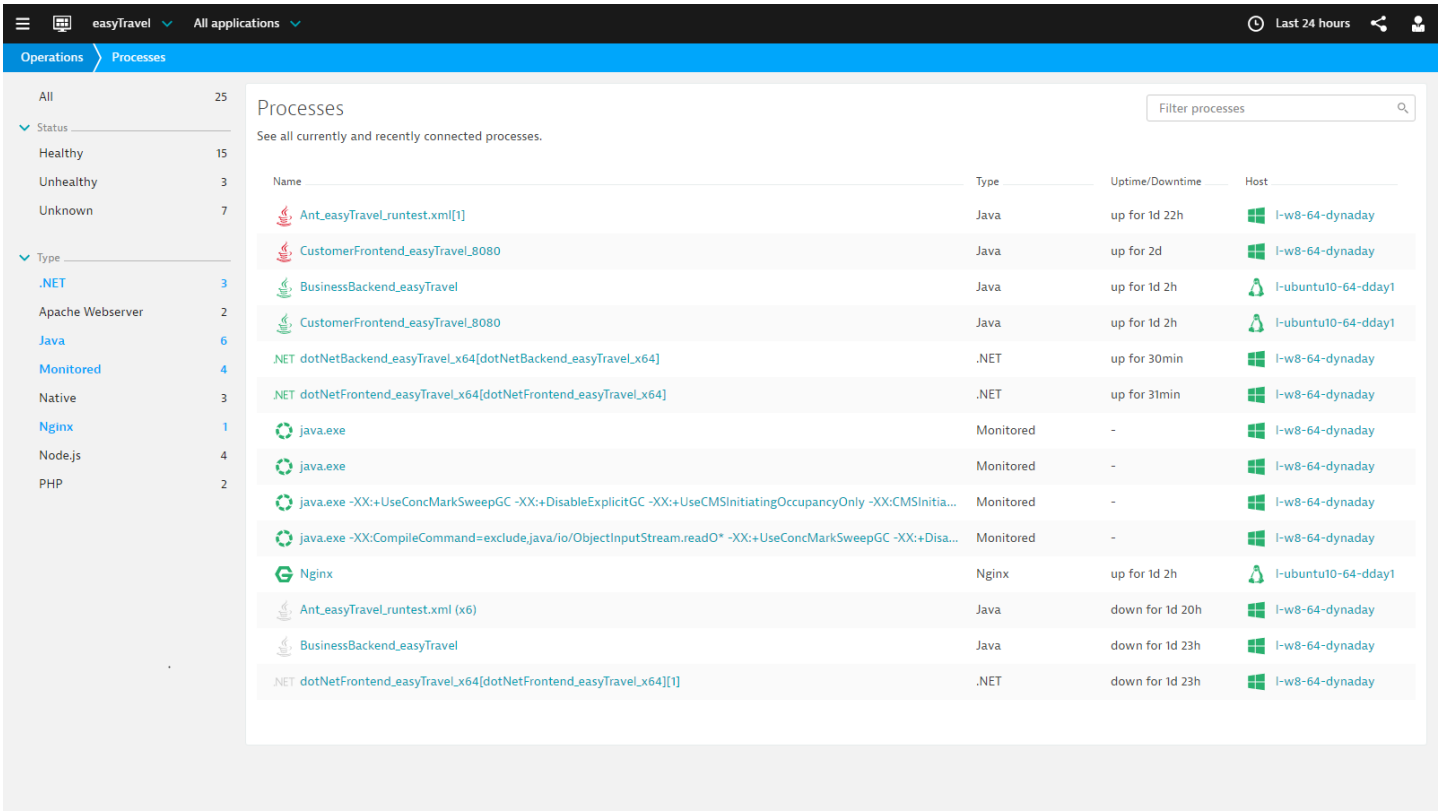
- The **text and measures tile** gives you all the power you need. Create tables with measure values, or compare key business metrics over multiple timeframes.
- **Twice the power**: Up to 40 tiles and 40 measures or splittings, plus application filters. What you need is what you get.
- **Ten times faster**: Select multiple tiles, clone them, scale them. Faster and more beautiful than ever.

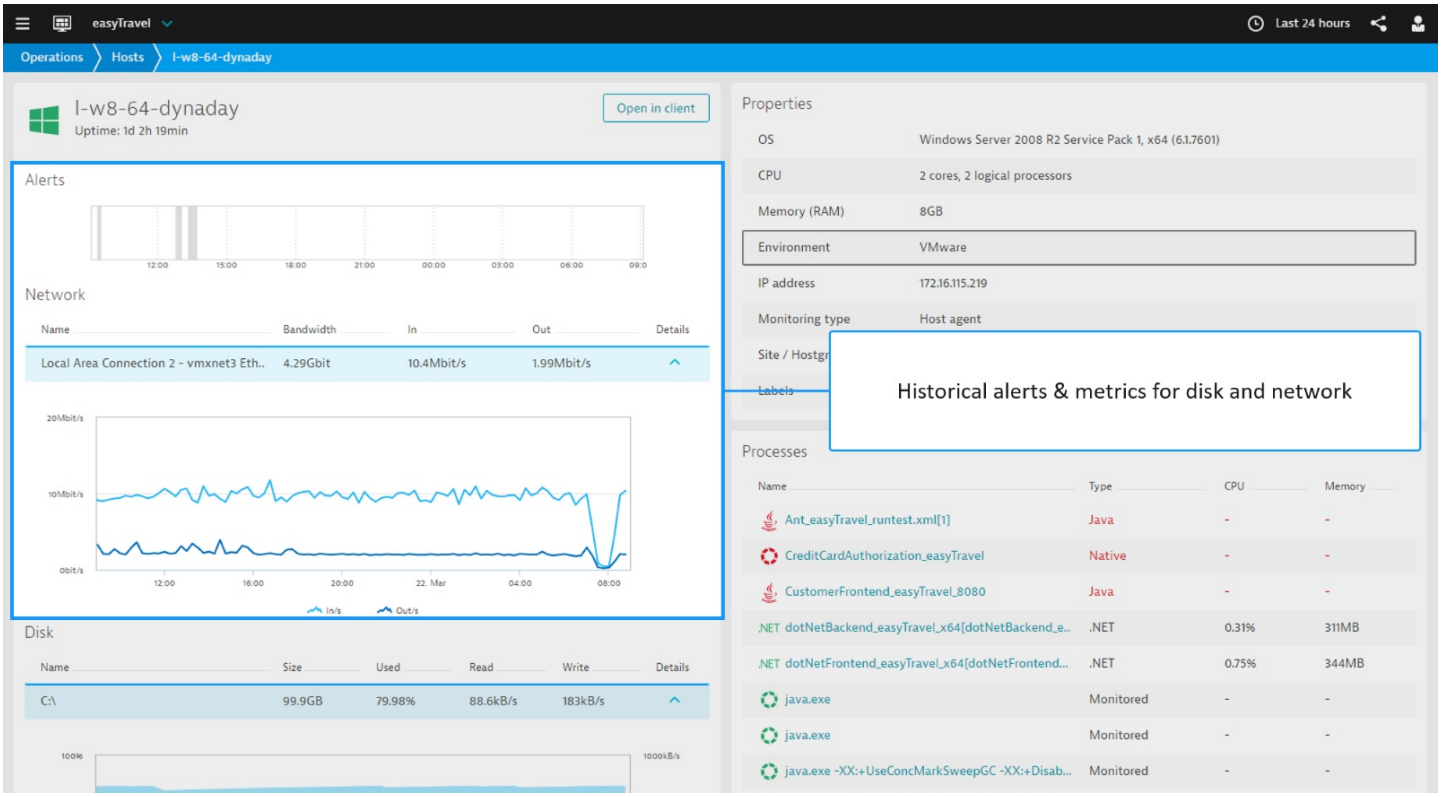
With entirely new layout possibilities, much more power and flexibility, web dashboards are a key factor for a more effective day.

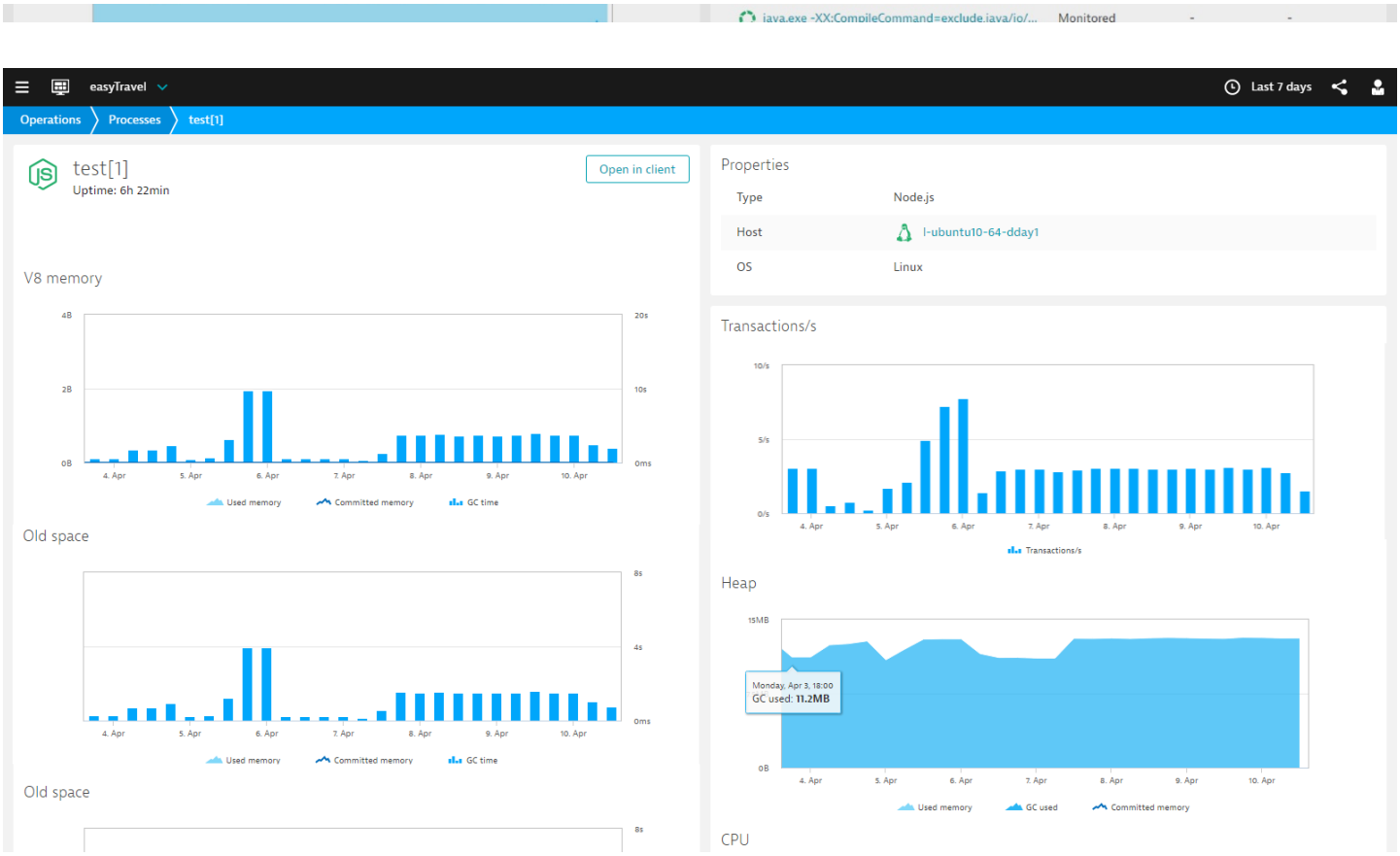
Amazing web dashboards




Healthy (3)
4 Hosts
Native (2)
Node.js (2)
Java (3)
18 Processes
Healthy (1)
1 Database


easyTravel
All applications
Last 24 hours
Operations
Processes
All 25
Status
Healthy 15
Unhealthy 3
Unknown 7
Type
.NET 3
Apache Webserver 2
Java 6
Monitored 4
Native 3
Nginx 1
Node.js 4
PHP 2
Processes
See all currently and recently connected processes.
Filter processes
Name
Type
Uptime/Downtime
Host
Ant_easyTravel_runtest.xml[1] Java up for 1d 22h l-w8-64-dynaday
CustomerFrontend_easyTravel_8080 Java up for 2d l-w8-64-dynaday
BusinessBackend_easyTravel Java up for 1d 2h l-ubuntu10-64-dday1
CustomerFrontend_easyTravel_8080 Java up for 1d 2h l-ubuntu10-64-dday1
dotNetBackend_easyTravel_x64[dotNetBackend_easyTravel_x64] .NET up for 30min l-w8-64-dynaday
dotNetFrontend_easyTravel_x64[dotNetFrontend_easyTravel_x64] .NET up for 31min l-w8-64-dynaday
java.exe Monitored - l-w8-64-dynaday
java.exe Monitored - l-w8-64-dynaday
java.exe -XX:+UseConcMarkSweepGC -XX:+DisableExplicitGC -XX:+UseCMSInitiatingOccupancyOnly -XX:CMSInitia... Monitored - l-w8-64-dynaday
java.exe -XX:CompileCommand=exclude,java/io/ObjectInputStream.readO* -XX:+UseConcMarkSweepGC -XX:+Disa... Monitored - l-w8-64-dynaday
Nginx Nginx up for 1d 2h l-ubuntu10-64-dday1
Ant_easyTravel_runtest.xml (x6) Java down for 1d 20h l-w8-64-dynaday
BusinessBackend_easyTravel Java down for 1d 23h l-w8-64-dynaday
dotNetFrontend_easyTravel_x64[dotNetFrontend_easyTravel_x64][1] .NET down for 1d 23h l-w8-64-dynaday


easyTravel
Last 24 hours
Operations
Hosts
l-w8-64-dynaday
l-w8-64-dynaday
Uptime: 1d 2h 19min
Open in client
Properties
OS Windows Server 2008 R2 Service Pack 1, x64 (6.1.7601)
CPU 2 cores, 2 logical processors
Memory (RAM) 8GB
Environment VMware
IP address 172.16.115.219
Monitoring type Host agent
Alerts
Network
Name
Bandwidth
In
Out
Details
Local Area Connection 2 - vmxnet3 Eth..
4.29Gbit
10.4Mbit/s
1.99Mbit/s
Historical alerts & metrics for disk and network
Disk
Name
Size
Used
Read
Write
Details
C:\
99.9GB
79.98%
88.6kB/s
183kB/s
Processes
Name
Type
CPU
Memory
Ant_easyTravel_runtest.xml[1] Java
CreditCardAuthorization_easyTravel Native
CustomerFrontend_easyTravel_8080 Java
dotNetBackend_easyTravel_x64[dotNetBackend_e.. .NET 0.31% 311MB
dotNetFrontend_easyTravel_x64[dotNetFrontend... .NET 0.75% 344MB
java.exe Monitored
java.exe Monitored
java.exe -XX:+UseConcMarkSweepGC -XX:+Disab... Monitored



- **Everything is connected**: Are your end-users affected by an issue? Is it caused by an overloaded host, or do you need to talk to developers? Find quick answers and determine who can deliver the best support.
- **Database insight** for Oracle, DB2,SQL Server and MySQL.
- **NodeJS process dashboard** with memory and CPU insights.
- **Alert, network & disk history**: Everything you need to see what led up to an infrastructure-related problem is available in a single place.

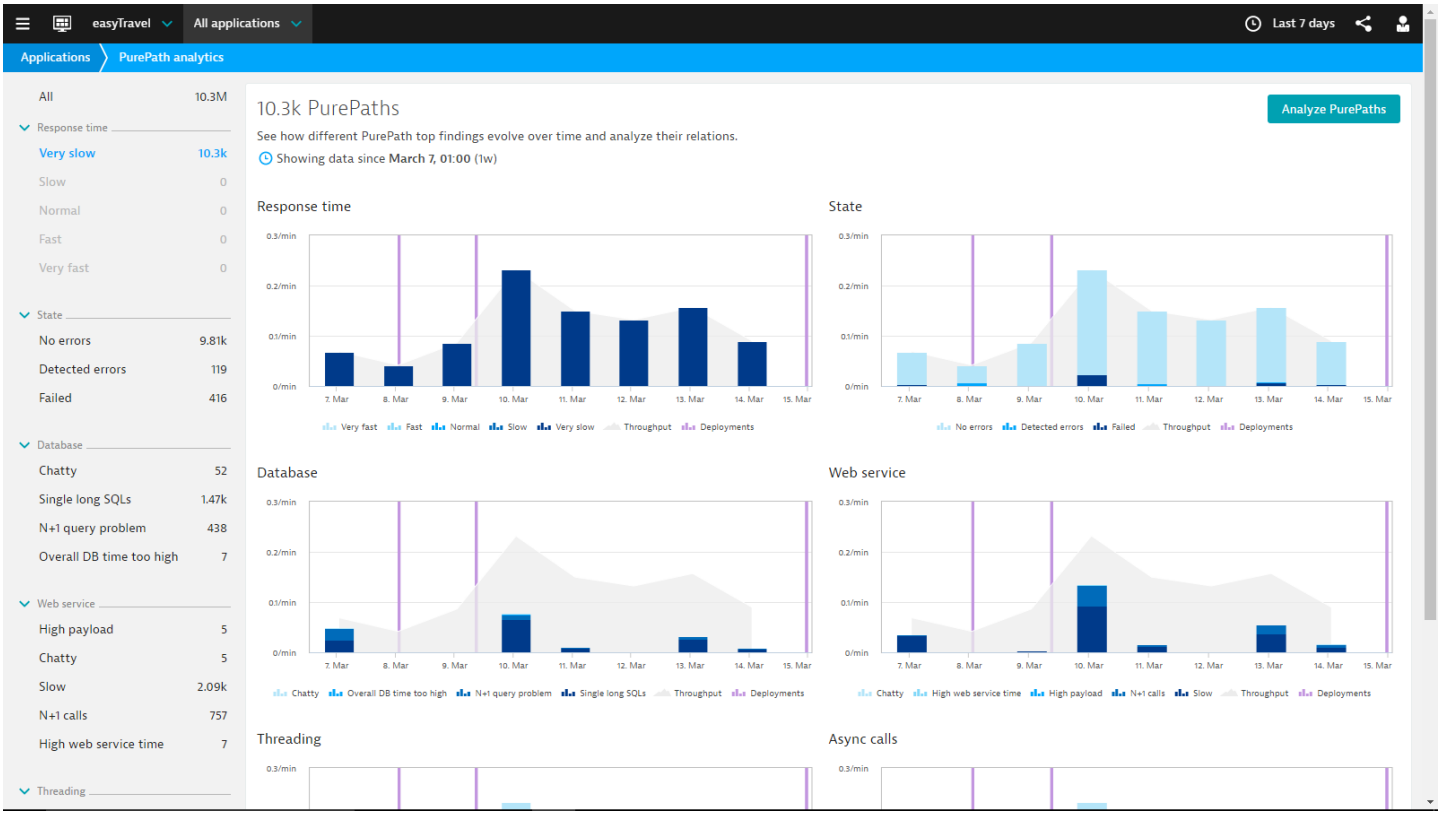
Devops teams have end-to-end insight into all applications and infrastructure to pinpoint issues and improve user experience quicker than before.

Onboard DevOps teams

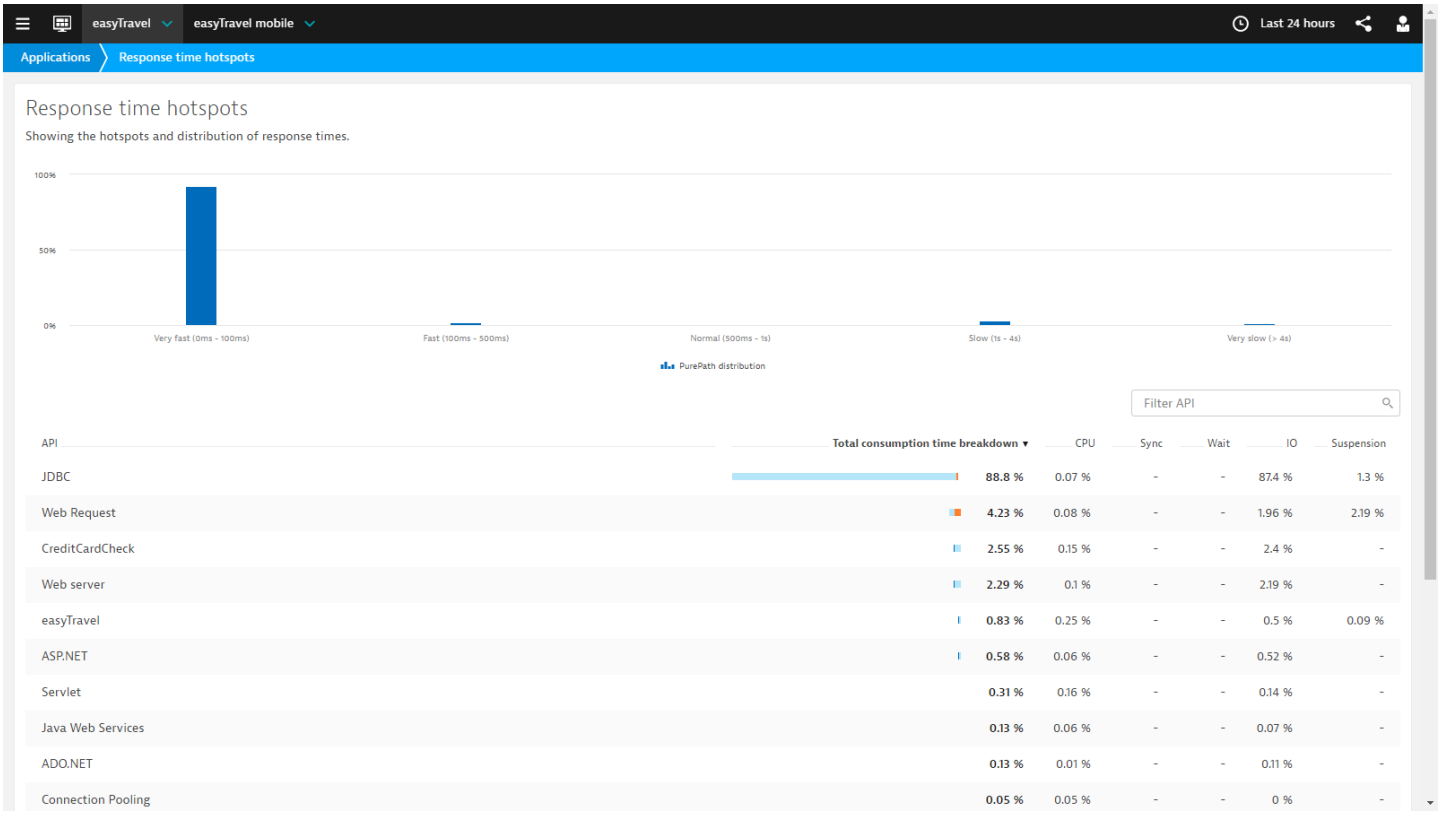
- **Instant feedback:** Deploy your application, see how the PurePath top findings evolve, discover the problem hotspots, and work together with engineering to improve and better deploy your application.
- **Slice and dice millions of PurePaths in real time:** Eliminate quality degradations early and talk to developers in context.

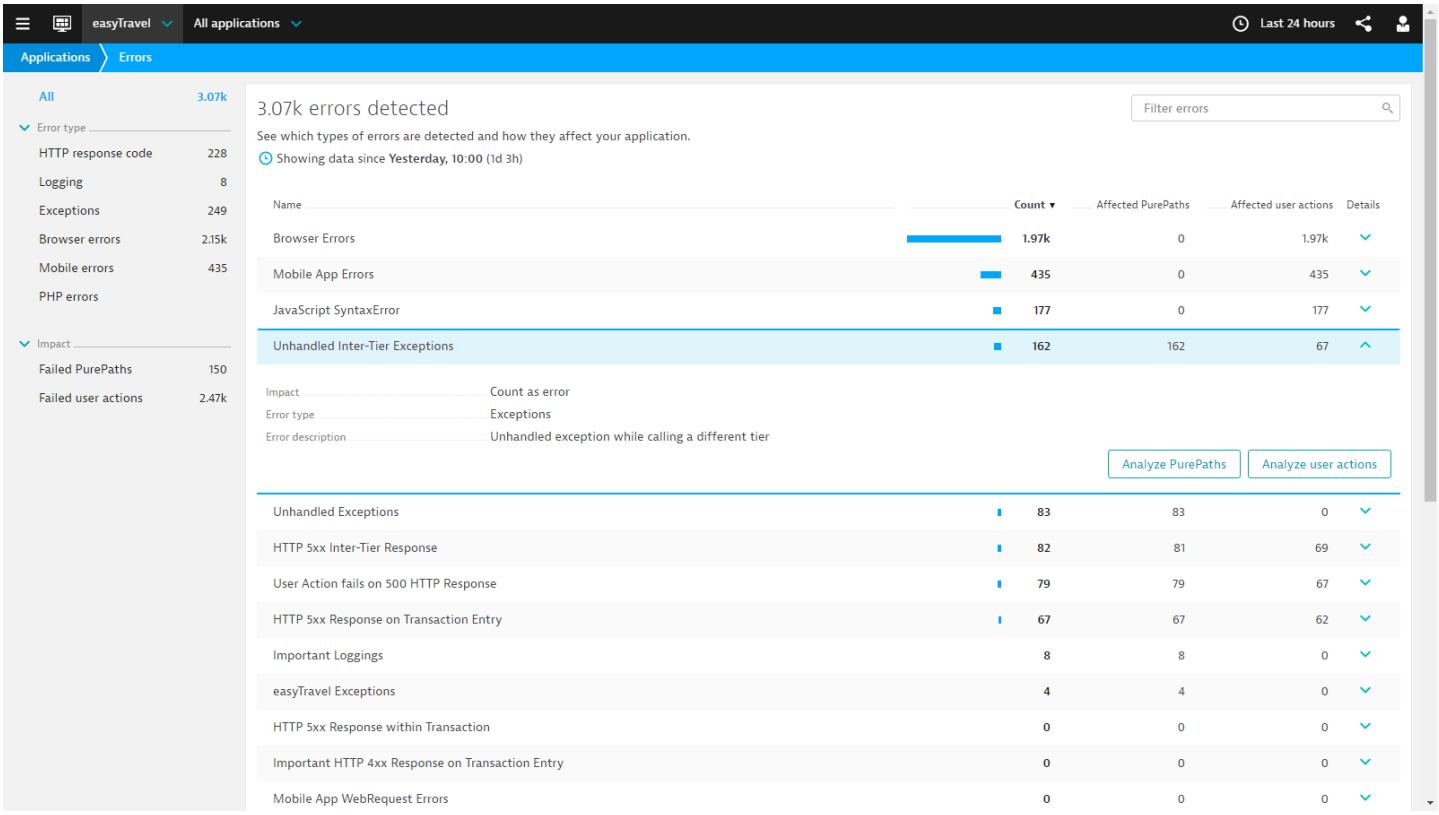
This release transforms your continuous delivery pipeline. Fully automated analysis of millions of PurePaths across multiple application deployments transforms your teams from problem hunters to

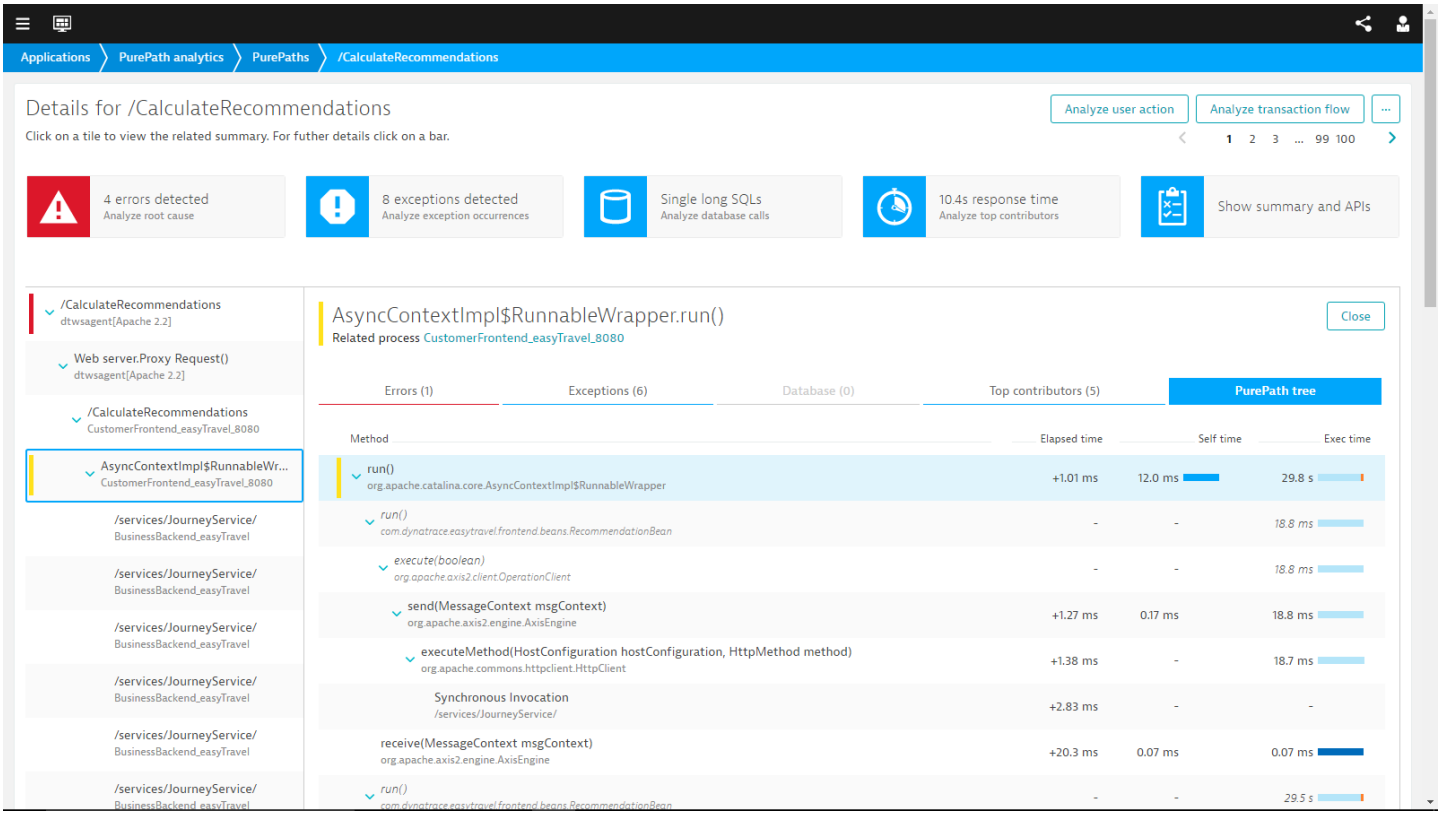
quality fanatics.

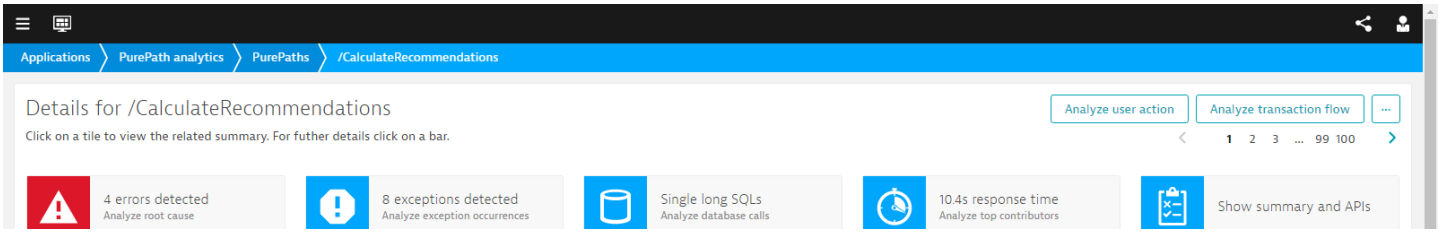


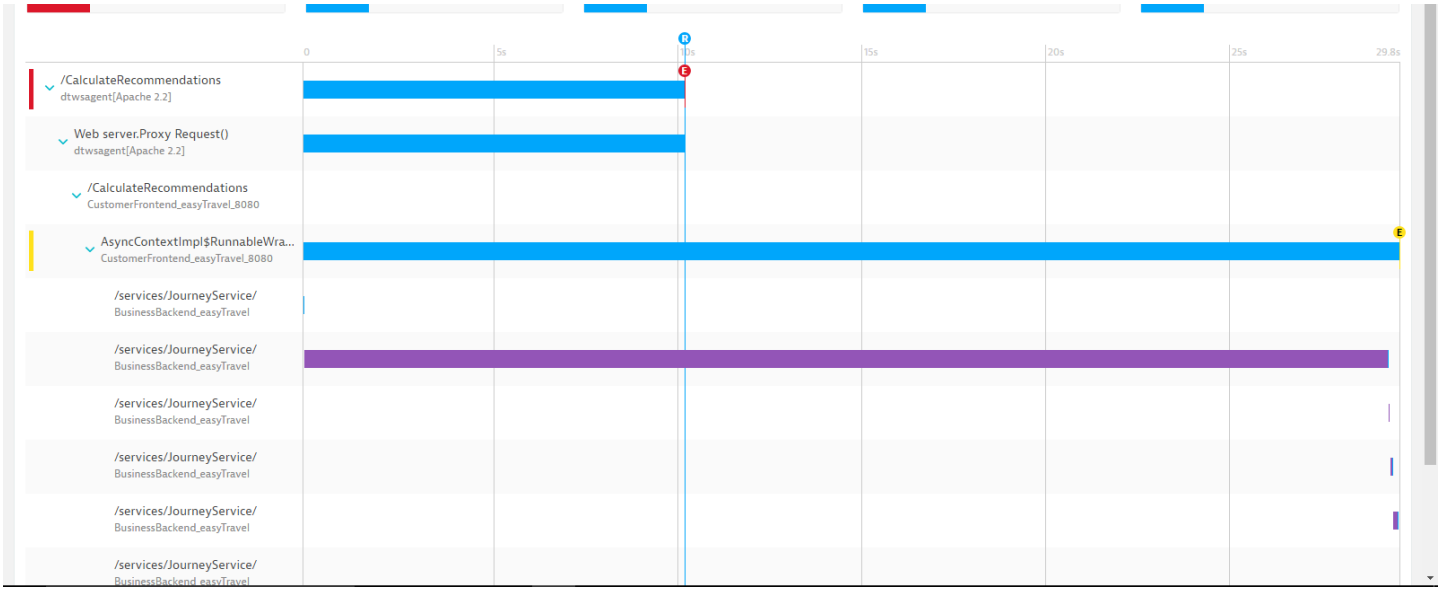
Better apps with every deployment




easyTravel
All applications
Last 24 hours
Applications
Errors
All 3.07k
Error type
HTTP response code 228
Logging 8
Exceptions 249
Browser errors 2.15k
Mobile errors 435
PHP errors
Impact
Failed PurePaths 150
Failed user actions 2.47k
3.07k errors detected
See which types of errors are detected and how they affect your application.
Showing data since Yesterday, 10:00 (1d 3h)
Filter errors
Name
Count
Affected PurePaths
Affected user actions
Details
Browser Errors 1.97k 0 1.97k
Mobile App Errors 435 0 435
JavaScript SyntaxError 177 0 177
Unhandled Inter-Tier Exceptions 162 162 67
Impact Count as error
Error type Exceptions
Error description Unhandled exception while calling a different tier
Analyze PurePaths
Analyze user actions
Unhandled Exceptions 83 83 0
HTTP 5xx Inter-Tier Response 82 81 69
User Action fails on 500 HTTP Response 79 79 67
HTTP 5xx Response on Transaction Entry 67 67 62
Important Loggings 8 8 0
easyTravel Exceptions 4 4 0
HTTP 5xx Response within Transaction 0 0 0
Important HTTP 4xx Response on Transaction Entry 0 0 0
Mobile App WebRequest Errors 0 0 0


Applications
PurePath analytics
PurePaths
/CalculateRecommendations
Details for /CalculateRecommendations
Click on a tile to view the related summary. For futher details click on a bar.
Analyze user action
Analyze transaction flow
1 2 3 ... 99 100
4 errors detected
Analyze root cause
8 exceptions detected
Analyze exception occurrences
Single long SQLs
Analyze database calls
10.4s response time
Analyze top contributors
Show summary and APIs
/CalculateRecommendations
dtwsagent[Apache 2.2]
Web server.Proxy Request()
dtwsagent[Apache 2.2]
/CalculateRecommendations
CustomerFrontend_easyTravel_8080
AsyncContextImpl$RunnableWr...
CustomerFrontend_easyTravel_8080
/services/JourneyService/
BusinessBackend_easyTravel
AsyncContextImpl$RunnableWrapper.run()
Related process CustomerFrontend_easyTravel_8080
Close
Errors (1)
Exceptions (6)
Database (0)
Top contributors (5)
PurePath tree
Method
Elapsed time
Self time
Exec time
run()
org.apache.catalina.core.AsyncContextImpl$RunnableWrapper
+1.01 ms 12.0 ms 29.8 s
run()
com.dynatrace.easytravel.frontend.beans.RecommendationBean
18.8 ms
execute(boolean)
org.apache.axis2.client.OperationClient
18.8 ms
send(MessageContext msgContext)
org.apache.axis2.engine.AxisEngine
+1.27 ms 0.17 ms 18.8 ms
executeMethod(HostConfiguration hostConfiguration, HttpMethod method)
org.apache.commons.httpclient.HttpClient
+1.38 ms 18.7 ms
Synchronous Invocation
/services/JourneyService/
+2.83 ms
receive(MessageContext msgContext)
org.apache.axis2.engine.AxisEngine
+20.3 ms 0.07 ms 0.07 ms
run()
29.5 s

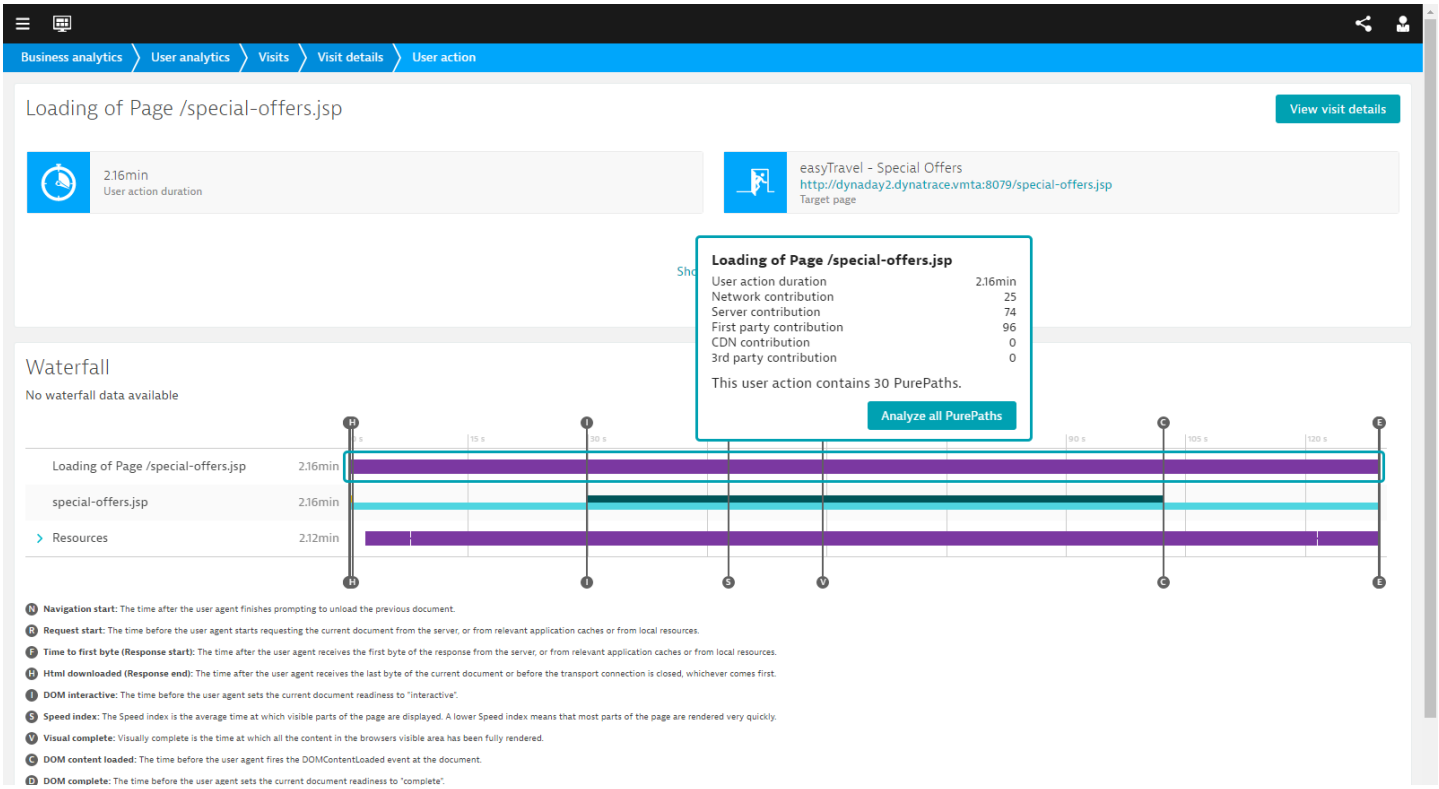

Applications
PurePath analytics
PurePaths
/CalculateRecommendations
Details for /CalculateRecommendations
Click on a tile to view the related summary. For futher details click on a bar.
Analyze user action
Analyze transaction flow
1 2 3 ... 99 100
4 errors detected
Analyze root cause
8 exceptions detected
Analyze exception occurrences
Single long SQLs
Analyze database calls
10.4s response time
Analyze top contributors
Show summary and APIs



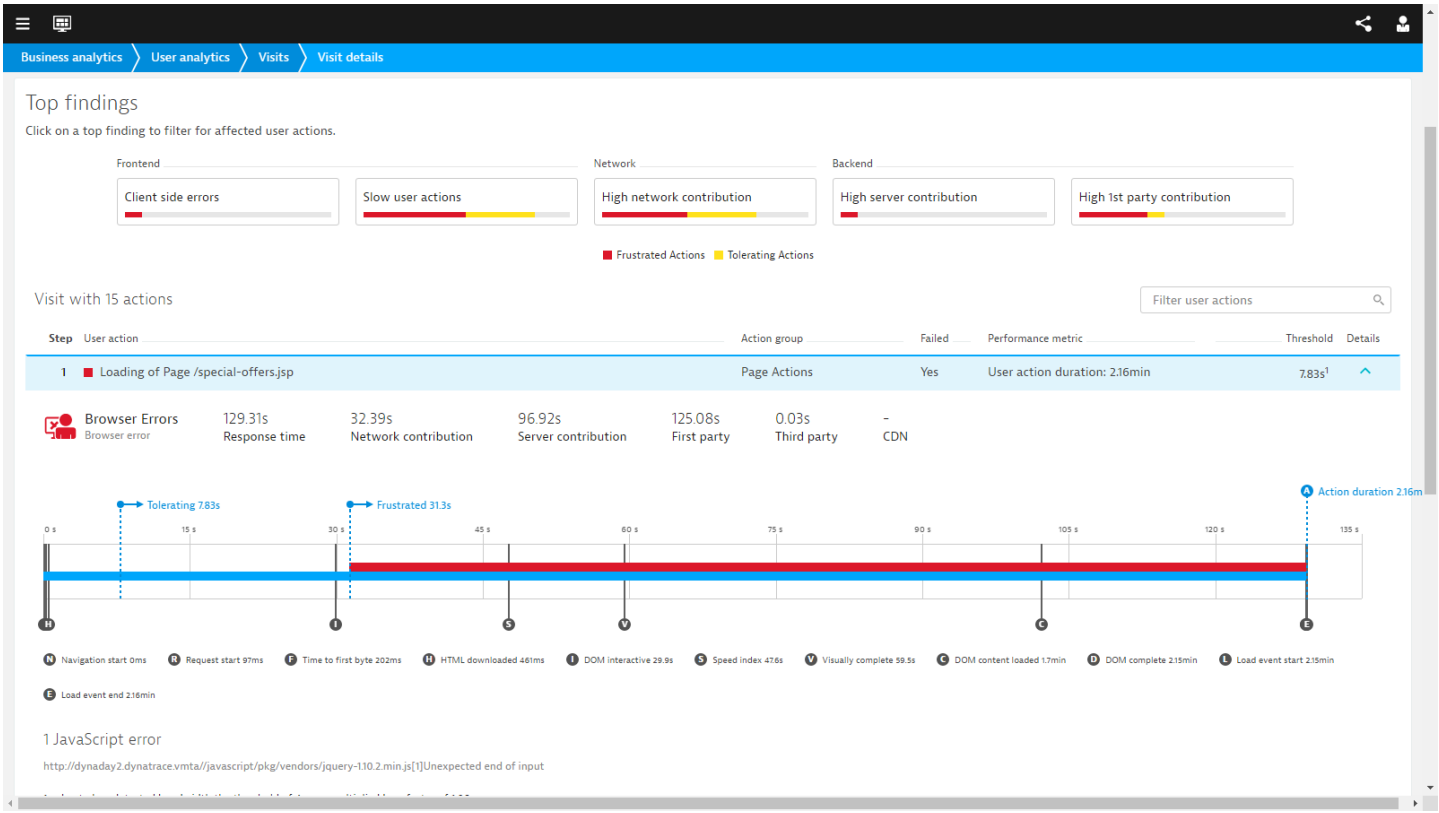
- **PurePath for all teams**. Leverage top findings, threading issues, slow statements, and common issues much faster than ever before.
- Everything is built putting **collaboration first**. Your teams can always share what they are looking at, including sessions.

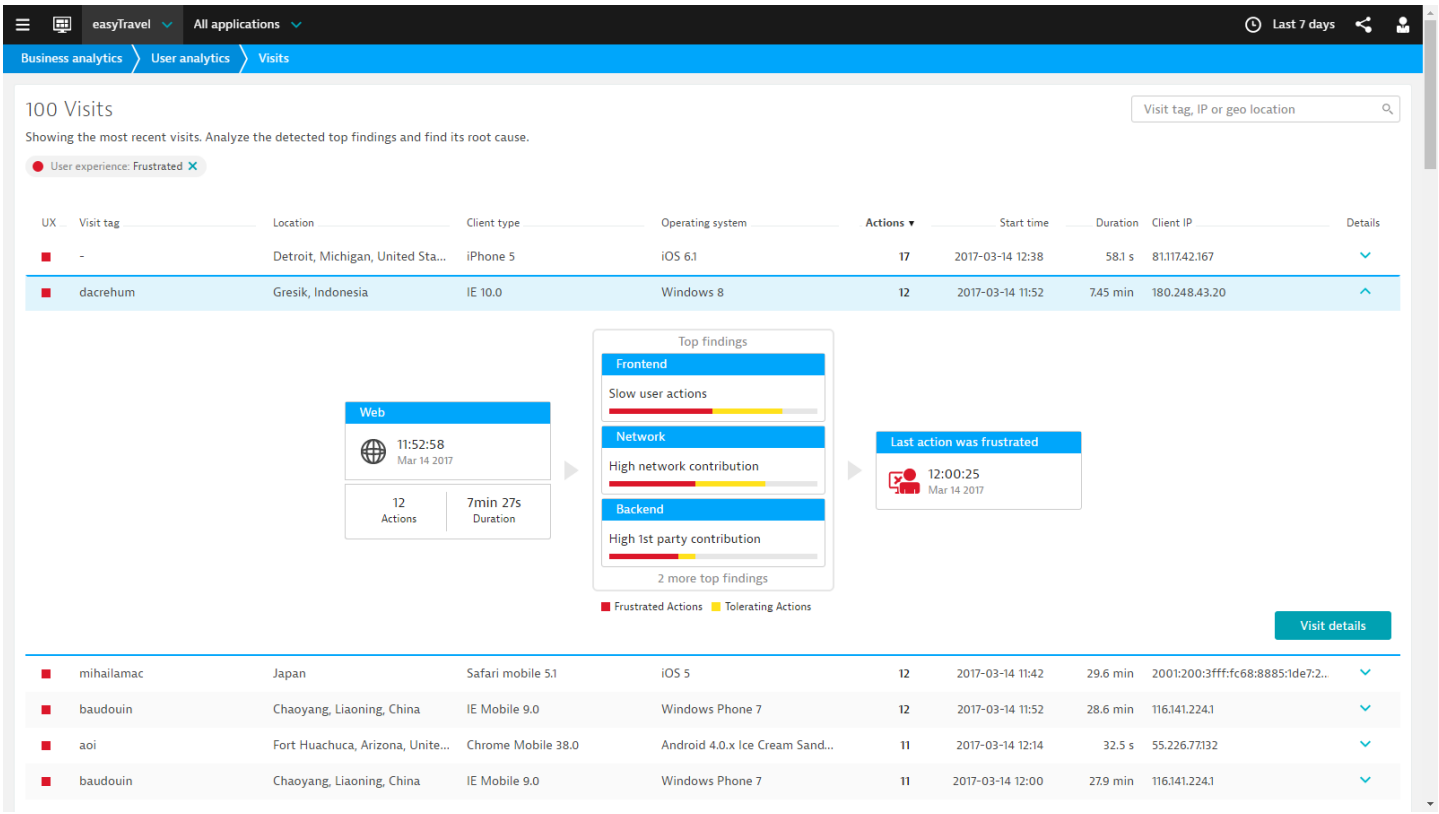
Source-code level analysis is ten times faster than ever before. Speed-up MTTR by guiding your analysis from baseline violation down to the root cause.

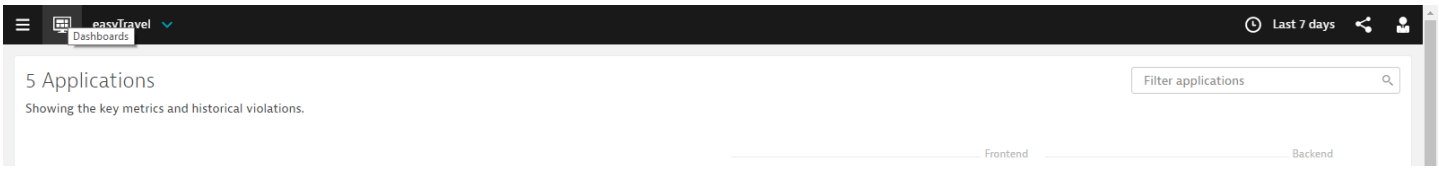
Sourcecode level insight for everyone

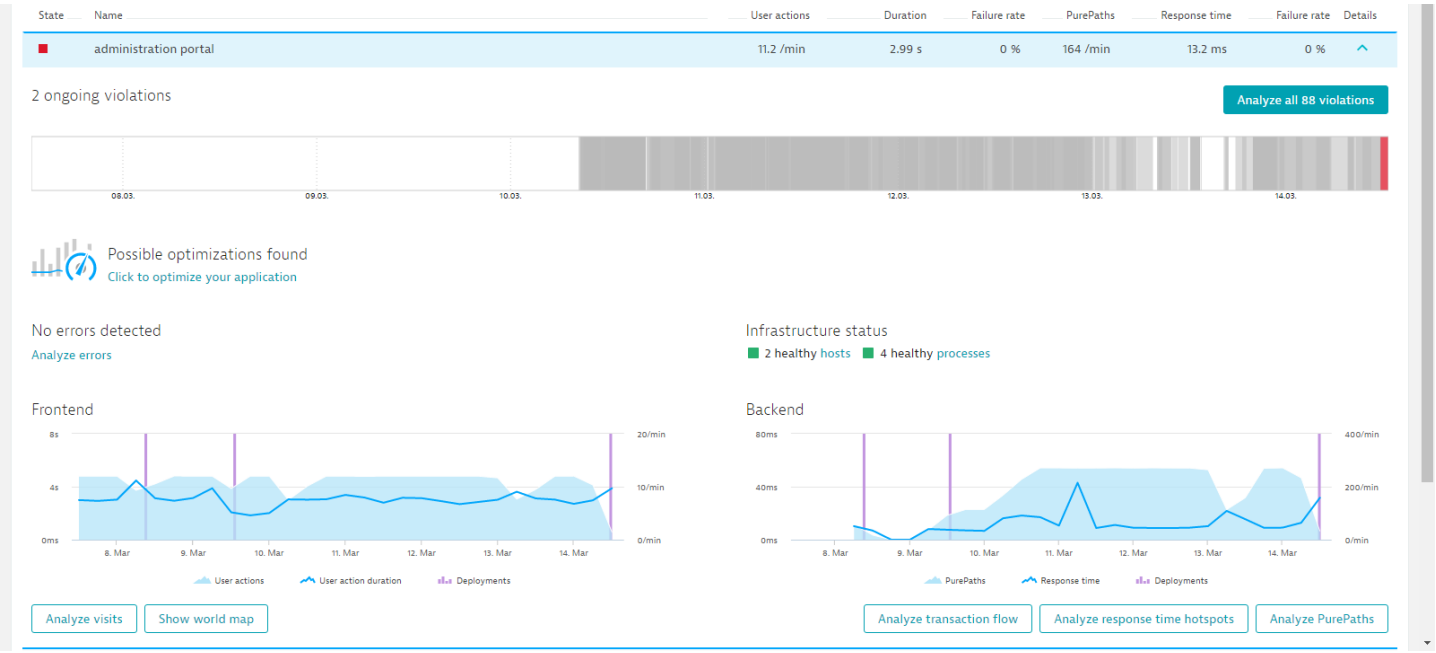


Load event start: The time before the load event of the current document is fired.









- **User action waterfall**: Understand what exactly affects your user experience and leverage the W3C metrics for precise insight into resource loading. Learn more.
- **Smarter than ever before**: **Visually complete** is your feedback to what the user is really experiencing, no matter which device or browsers. It is now an integral part of the user action timeline as well as the waterfall.
- **From user interaction down to the server call**: Everything is connected, and frontend teams can collaborate with backend teams in context.

This release delivers insight into every browser and app and provides the best metrics that reflect your customer's perspective.

Speed up your frontend teams

- Key features and enhancements
- Upgrading and getting started with this release
- News from the Dynatrace Community
- Licensing update
- Resolved and known issues
- System requirements & Supported technologies
- Dynatrace Support levels and End-of-life policy

- User documentation
- Feedback for this release

What's new in Dynatrace AppMon & UEM May 2017?

Get AppMon 2017 May



Here's what our customers say: *"The newcustomization abilities of the web dashboards alone make this release an absolute must have within your IT organization."* Derek Abing, IT-Operations manager at Sentry Insurance.
May 31st, 2017. Supercharge all teams with this release of Dynatrace AppMon & UEM.

Dynatrace AppMon 2017 May Mainframe Release Notes

Date	PTF	What's Impacted	Description
2016-Oct	DTR010A	DTAX	**JLT-188327 – SOC4 in ZDTPLT when shutting down a system** • *Impact*: Low, should only happen if VTAM is recycled. • *Cause*: When VTAM comes down, ZDTPLT is trying to refer to storage that is not available. • *Solution*: Added a condition to check if the pointer is null before using it.

Resolved and known issues

Monthly PTF's have been made available for Dynatrace AppMon 7.0 mainframe components thru the cumulative monthly maintenance. Recommended for all users!

Update 2

- **ONE-8408** - Non-ASCII characters in the request URI path are shown as '?' in the PurePath.

Nginx

- **JLT-190358** - zRemoteAgent crashes when processing capture-type BCD

Mainframe

- **JLT-190248** - Fix NPE on session analysis.
- **JLT-190800** - Visits potentially not stored after maximum storage queue size has been reached although queue size already below maximum again.

Session Storage

- **JLT-190108** - Incident chart does not allow chart splitting of more than ten incident rules.

Charting

- **JLT-190028** - The "Start Time" field was missing when exporting User Action PurePaths to XML

Dashboard

- **JLT-189425** - Added cassandra sensors early return in case of disabled bind value capturing
- **JLT-189699** - Groovy methods that previously would show in PurePaths don't show anymore after upgrade to 6.5 - add debugEnableGroovyNative, so groovy.lang.GroovyObjects, groovy.lang.Scripts and groovy.lang.Closures may be instrumented again (although not supported)

Java Agent

- **JLT-189398** - Memory charts for HP-JVMs are not shown.
- **JLT-189410** - Keypress check in WebUI blocks entering into number fields on french keyboards
- **JLT-189698** - Bar chart bars aren't rendered correctly with high tile resolution on a small screen resolution.
- **JLT-191273** - Open-end drilldown timeframe is not properly displayed in the filterbar

- **JLT-191982** - [WebUI] Properly handle owner information when deleting a dashboard

WebUI

- **JLT-189009** - Fix synchronization bottleneck in instrumentation of .net agents.

Instrumentation

- **JLT-188749** - Simultaneous linking during UEM correlation and modification of sub-path transaction from sub-path correlation engine may lead to CME and consequently dropped user actions.
- **JLT-188758** - Paths from OneAgents that run into a timeout are marked as corrupted.
- **JLT-189566** - Fix waiting for string resolution of mobile tags.
- **JLT-191125** - Fix propagating CICS Transaction ID to a captured argument.
- **JLT-191984** - Check argument for null to avoid a late NPE.
- **JLT-192112** - Fix race condition with UEM for paths from OneAgents.
- **JLT-192304** - Fix NPE happening due to a race condition which was introduced in 7.0.1
- **JLT-192615** - Elapsed time column doesn't show values for paths from OneAgents.

Correlation and Analysis

- **JLT-188426** - Handle Leak when invoking .NET Thread Dump. Could lead to timeouts of Thread Dumps unter certain curcumstances.

.NET Agent

- **JLT-188091** - DB Agent Live View slow queries issue with large databases - improved logging
- **JLT-189160** - DB2 explain plan retrieval leaves rows in temporary tables
- **JLT-190593** - DB2 dataaccess permission made optional: may be replaced by access to all required views and functions (see Database Agent configuration section in AppMon documentation)
- **JLT-190879** - DB2 connection validation — missing permissions to explain plan tables prevented db2 agent creation
- **JLT-191944** - Expanded error message on failing to retrieve explain plans for truncated SQL on MySQL

DB Agent

- **JLT-187076** - No more agent connections possible, "Unable to create connection for request …" in log.

- **JLT-189564** - Agent property “agent.basename” not set on agent recovery
- **JLT-191599** - .NET Agent - ADO.NET Connection Pool attachment reported workstation id, instead of database name when the keyword ‘database’ was used in the connection string.

Collector

- **JLT-186179** - Possibly server not starting due to Performance Warehouse connection issue.
- **JLT-189077** - Add possibility to skip validation of entered performance warehouse configuration data.
- **JLT-189338** - Improve caching in lookup tables.
- **JLT-190086** - Improved exception handling for performance warehouse.
- **JLT-190959** - Fixed default strategy for embedded performance warehouse.
- **JLT-192357** - Improved collision detection for performance warehouse migration.
- **JLT-192489** - Improved performance of repairing inconsistent data in performance warehouse during migration.
- **JLT-192616** - Improve the 7.0.0.8 performance warehouse migration step to resume after a failed execution.

PWH

- **JLT-183715** - Client sometimes throws SWTException about a disposed widget in the deployed certificates tab
- **JLT-188190** - Wrong Agent Platform configuration may be shown in Agent Overview after server restart
- **JLT-188619** - Prevent NullPointerException when opening stored TestAutomation dashlet
- **JLT-189148** - Synthetic Monitoring: Disable ‘Data feed’ and remove configuration UI as desupported by Synthetic
- **JLT-189545** - Scrollbars in nightmode are misplaced in some instances.
- **JLT-189659** - Prevents potential null pointer exception when storing a system profile
- **JLT-189974** - Misleading percentage in Transaction Flow when there are First Party Calls.
- **JLT-190900** - Add tier wizard for Node.js was not supporting Windows
- **JLT-191848** - Error shown when creating Measures from PurePath nodes generated by web server OneAgents.

Client

- **JLT-179651** - Improved log for ArrayIndexOutOfBoundsException in CachingTimeSeries.

Server Platform

- **JLT-178235** - Potential keystore corruptions due to frequent unnecessary and non-atomic write access. Missing automatic restore mechanism when keystore is corrupted.
- **JLT-187038** - Server log spam: if password was wrong don't log a stack trace anymore
- **JLT-187280** - Thread Dump on server stays in-progress until server restart in case monitors and/or threads change during Thread Dump creation.
- **JLT-187531** - Fix a potential BufferUnderflow exception while reading incidents from sessions.
- **JLT-188082** - Controlled rollout based on Host + Agent Name does not work for Web Server OneAgents that were migrated from AppMon
- **JLT-188303** - No 'Continuous transaction recording was aborted' incident will be created if there is already one open.
- **JLT-188618** - Fix an Exception when reading path attachments from session-storage
- **JLT-188678** - Table order of resource-dump warnings table not correctly restored.
- **JLT-189128** - Added log output for enabled TLS ciphers and protocols
- **JLT-189134** - Prevented server restart by fixing session synchronization bottleneck with large amount of sessions.
- **JLT-189404** - Incorrect log supression in MultiMessageBroadcaster
- **JLT-189572** - Prevent manually editing of unsupported sizings in ini file.
- **JLT-189677** - Global incident downtimes do not activate correctly.
- **JLT-189719** - Ensure that the bytecode of java proxy classes are not written to the classcache
- **JLT-190078** - Support DCRum integration in new correlation out of the box.
- **JLT-190156** - Combined weekly/monthly schedule exclusions containing one "exclude" all schedule cause unexpected schedule behavior like scheduled executions unexpectedly ending after a week or month.
- **JLT-190564** - SSO: validation of Identity Provider Certificate fails
- **JLT-190607** - Add additional log output in case measure reference is null
- **JLT-190777** - Make recommended cpu cores of sizing global changeable via debug flag.
- **JLT-191306** - Null pointer check added to mitigate JVM crash.
- **JLT-191363** - Mimimal external measures' timestamp populated to test execution if not provided
- **JLT-191415** - Security vulnerability found in Apache POI version used by AppMon
- **JLT-191419** - Improved detection of server health to avoid early discarded PurePaths, especially in environments with many system profiles.
- **JLT-191700** - Reduce performance overhead for calcuation of the Failed Transaction and Failed User Action Count measures
- **JLT-191887** - Potential NPE in ActionCenterCheck
- **JLT-192133** - Markin Lost and found transactions as corrupt can be disabled now.

Server

- **JLT-172631** - NullPointerException in ActionDefinitions.isKnownAction(..)
- **JLT-188390** - Using incremental responses with XHRs leads to readyState 3 events not always firing, thus information is received too late.
- **JLT-191148** - Visit mapped to IE7 when IE11 compat mode was used (Trident/8.0 user-agent)
- **JLT-191305** - Onreadystatechange events of XHRs are fired multiple times in rare cases on IE, if the agent is injected asynchronously.

UEM

All resolved issues

- **JLT-192304 - NullPointerException happening due to a race condition which was introduced in 7.0.1**
  - **Impact:** Affected subpaths can never complete.
  - **Cause:** Usage of a reference after the memory barrier.
  - **Solution:** Fix order of statements.

Correlation and Analysis

- **JLT-190358 - zRemoteAgent crashes when processing capture-type BCD**
  - **Impact:** zRemoteAgent crashes when processing capture-type BCD
  - **Cause:** zRemoteAgent crashes when processing capture-type BCD
  - **Solution:** Fixed an issue with zRemoteAgent incorrectly parses the capture-type BCD causing it to crash.

Mainframe

Most critical issues since the last public updates

Component	Total	New
.NET Agent	2	1
Charting	1	1
Client	12	9

Collector	9	3
Core	1	0
Correlation and Analysis	13	8
DB Agent	5	5
Dashboard	1	1
Instrumentation	1	1
Integration and Services	1	0
Java Agent	2	2
JavaScript	1	0
Mainframe	3	1
MessageBroker Agent	1	0
Mobile	7	0
Nginx	1	1
PWH	14	8
Plugins	1	0
Server	33	26

Server Platform	1	1
Session Storage	2	2
UEM	9	4
WebUI	5	5
Webserver Agent	1	0

7 (+ 2) critical issues,127 (+ 79) total resolved issues

7.0.2.1011 Resolved Issues

- Public update 6.5.18 and earlier (not affected by JLT-192329)
- All updates released for versions 6.3 and earlier

This update contains all changes of:

Resolved Issues

A new update for Dynatrace AppMon 2017 May has been released! This update contains critical fixes and improvements. Using a controlled installation , you can select which agents when to update, without migration! The installation of updates containing critical fixes can prevent monitored application impact or monitoring data loss and therefore is strongly recommended for all users!

2017-07-04