Enterprise monitoring

Continuous Performance

…as a self-service

…with fully-automated feedback loops


© 2002 HowStuffWorks

Continuous Integration

"…is the practice, in software engineering, of merging all developer working copies with a shared mainline several times a day."

Pragmatically:

- autonomous, pre-scheduled build and deploy for DEV and QA
- inclusive of an automated "build verification test"
- is rarely autonomous or unattended

Implementation basics:

- powered-by Visual Studio, Jenkins, Bamboo
- integrated with code version control and config mgmt
- originated with eXP and test driven development (TDD)

Continuous Delivery

“the practice of continuous delivery further extends CI by making sure the software checked in on the mainline is always in a state that can be deployed to users and makes the actual deployment process very rapid.“

Pragmatically:

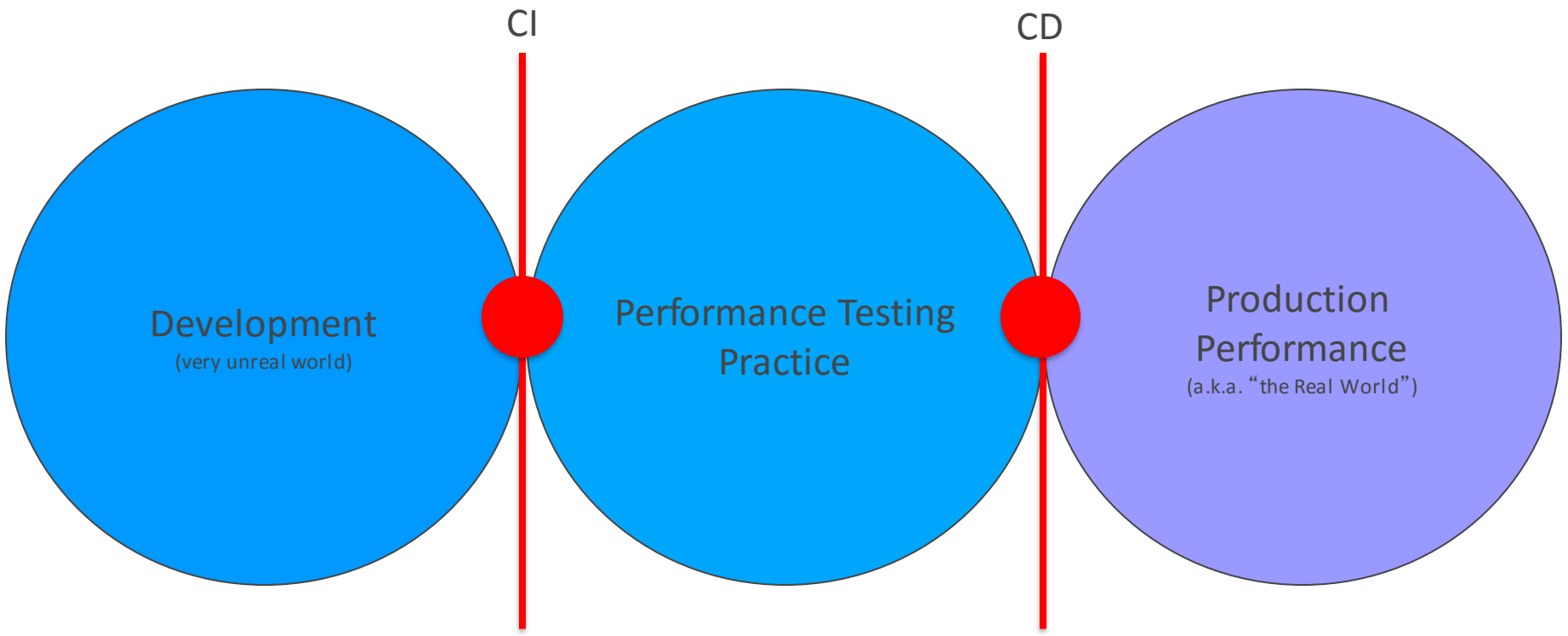
- automates release management between QA and PROD
- highly automated and orchestrated deploy/roll-back
- automated tests validate the code is “release ready”
- requires more rigorous, elaborate checking

Implementation basics:

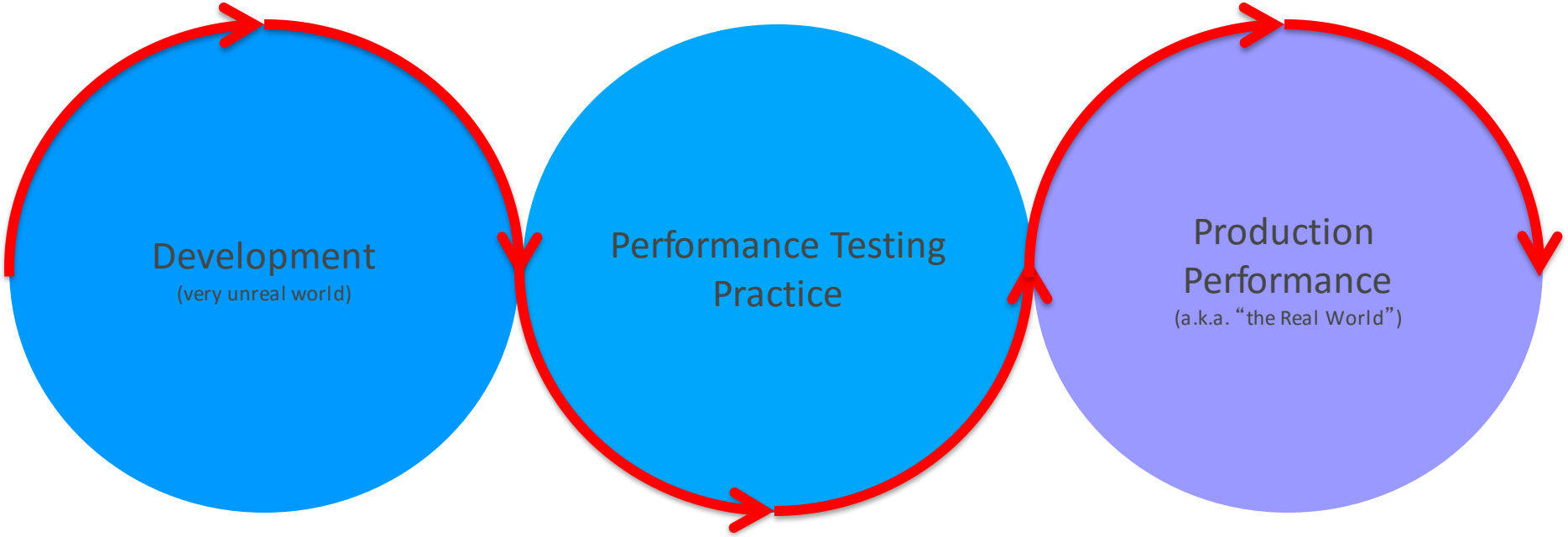
- powered-by Puppet, Chef, VMWare, Home-grown
- defined as a part of eXP – refined by Jez Humble/Dave Farley

Continuous Performance Lifecycle

Continuous Performance Lifecycle



Promotional Flow



Promotional Flows

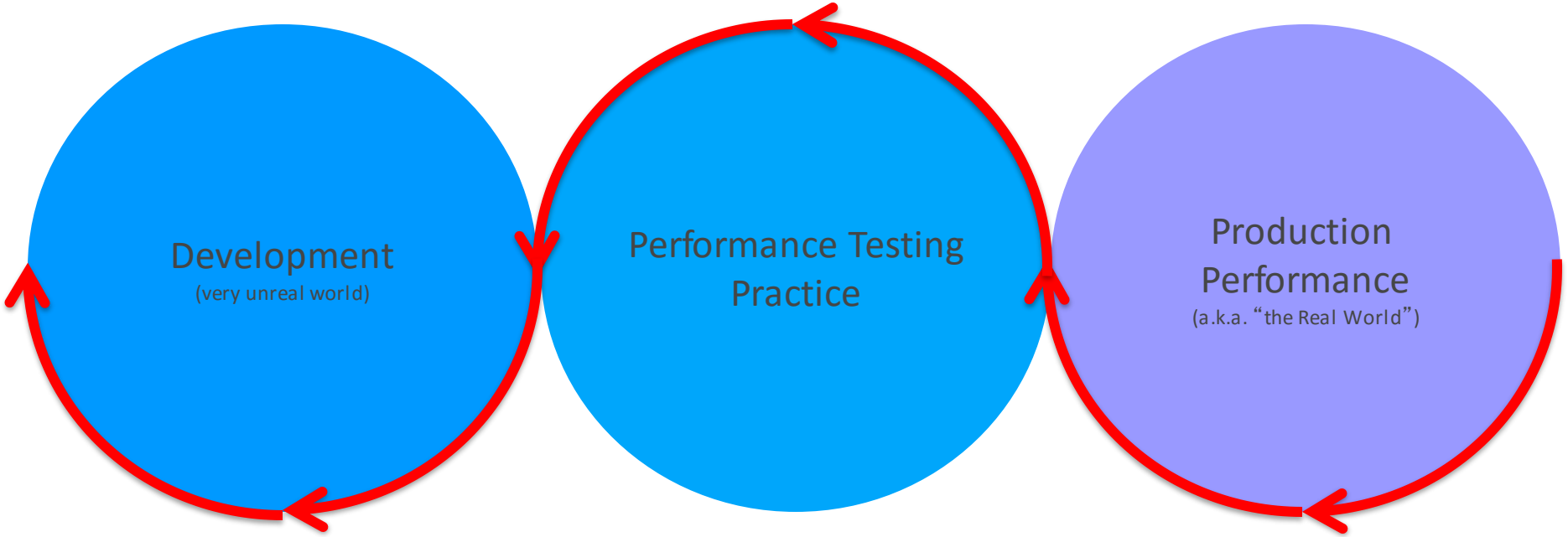
From DEV to PERF (via Continuous Integration)

- Performance regression testing – repeated testing and trending
- New code and app config changes
- New applications and integrations to be tested
- Unit-level performance information data

From PERF to OPS (via Continuous Delivery)

- Validation that a release will meet performance demand
- Performance guidance – top ten “worst offenders”
- Performance threshold update – re-configure APM monitors
- Performance test report generated and published to RM

Feedback Flow



Feedback Flow

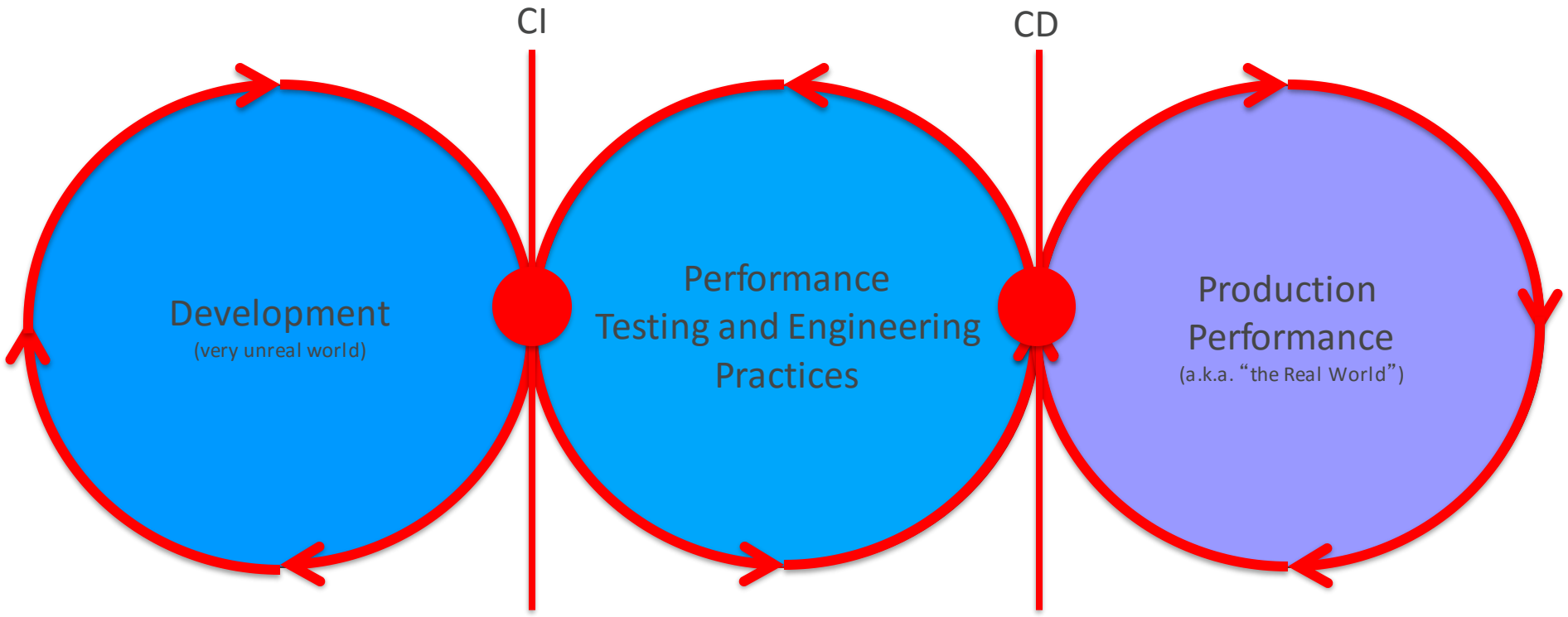
From OPS -> PERF

- Monitoring performance in PROD, setting new trends/thresholds
- Synchronizing test simulation with PROD (volume, mix, throughput)
- Synchronizing thresholds with PROD (resources, app metrics)
- Production issue repro/remediation – performance escalation

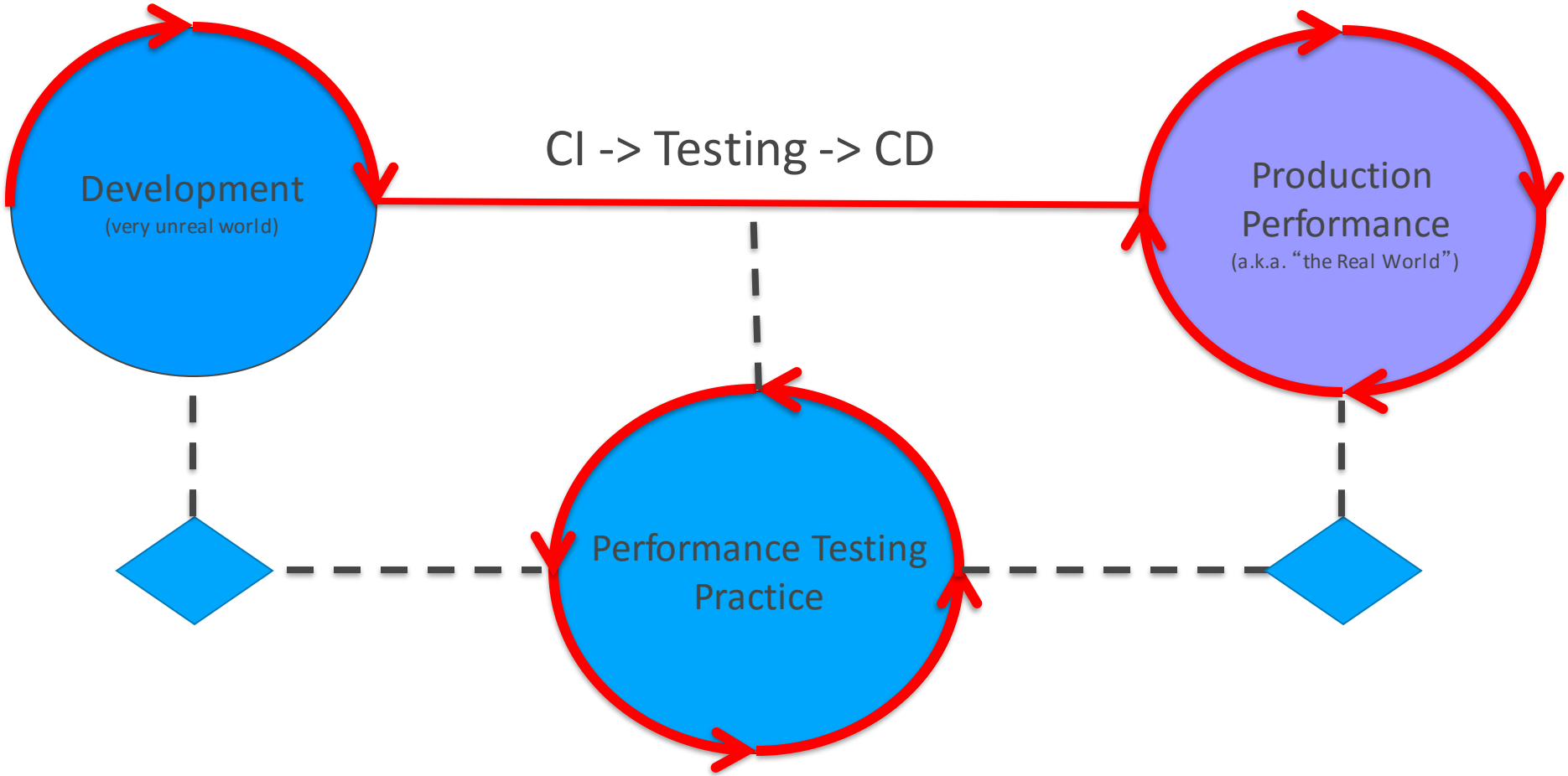
PERF->DEV

- Performance defects – bottlenecks and proposed remediation
- Early Performance Testing – unit-level performance results
- Strategic performance measurements – for architecture
- System future estimations – for PM's and Biz

Continuous Performance Flows



Out-of-Flow Continuous Performance



Performance Decisions

How do we make decisions about performance?

- We make decisions based on the information we have – not with the information we wish we had
- We make decision based on our understanding of the information we have – with our default perspective
- If the information is limited – the decision will be limited
- If the information isn't timely – our decision will be inaccurate
- If we don't know what came before – we can't estimate what will come next

With every sacrifice and limitation, our value decreases.

Performance Decisions

We can analyze performance at each point of the flow:

- Every time we move code along the promotional flow
- Every time we accept feedback about performance
- Every time we choose to reverse our decision to promote
- Every time we decide to repeat a test
- Every time we decide to log a bug
- Every time we change our plans according to new information

If the performance information is not timely:

- A wrong decision may delay competitive features for the biz
- A right decision may come too late to avoid disaster
- Any decision can and should be revisited in the face of change

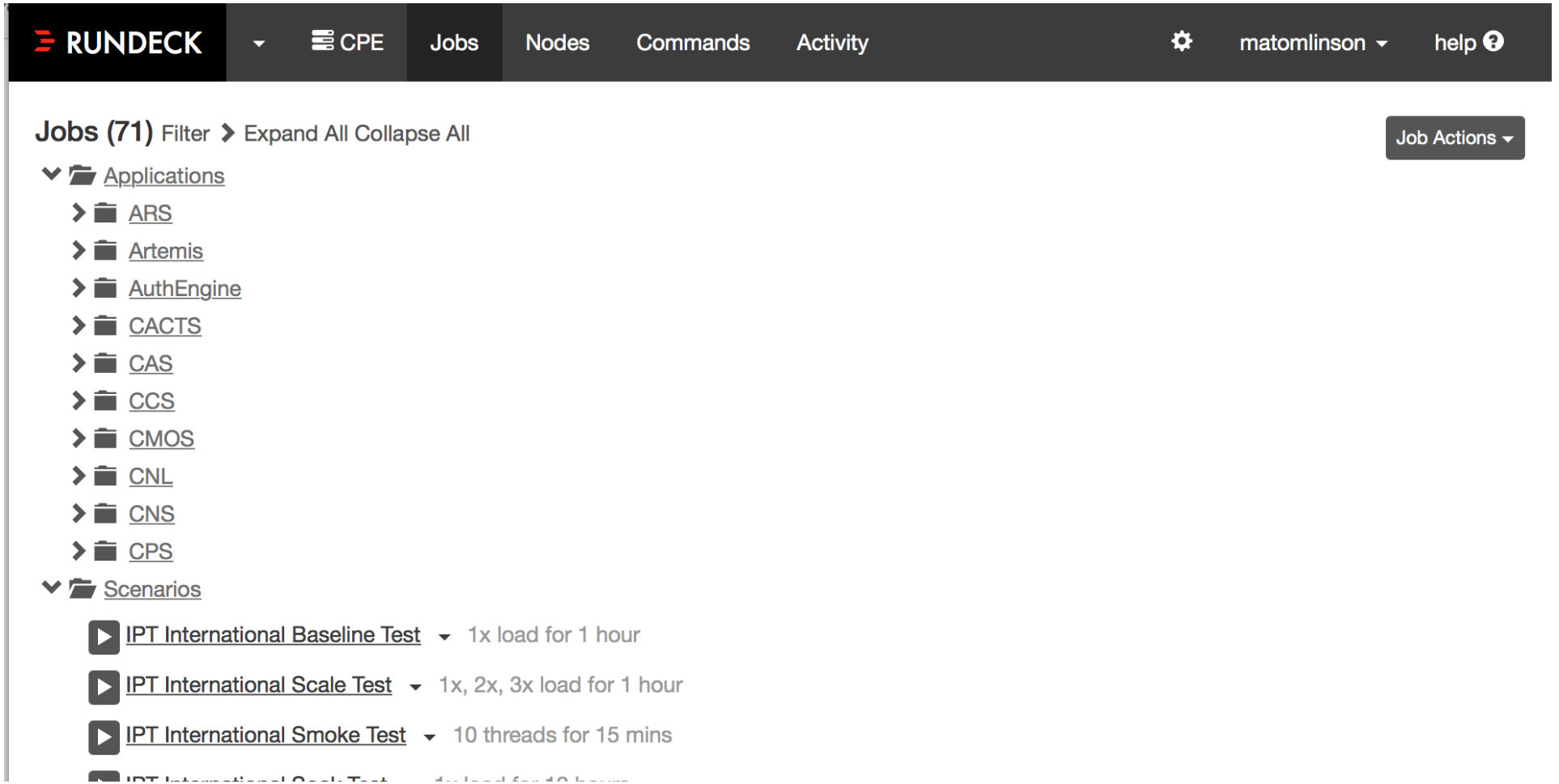

Continuous Performance Tooling

Continuous Performance Tooling

There are 3 primary “new tools” to add to your typical performance testing effort/team:

- tools for unattended automation
- tools for data trending/visualization
- tools for test notifications

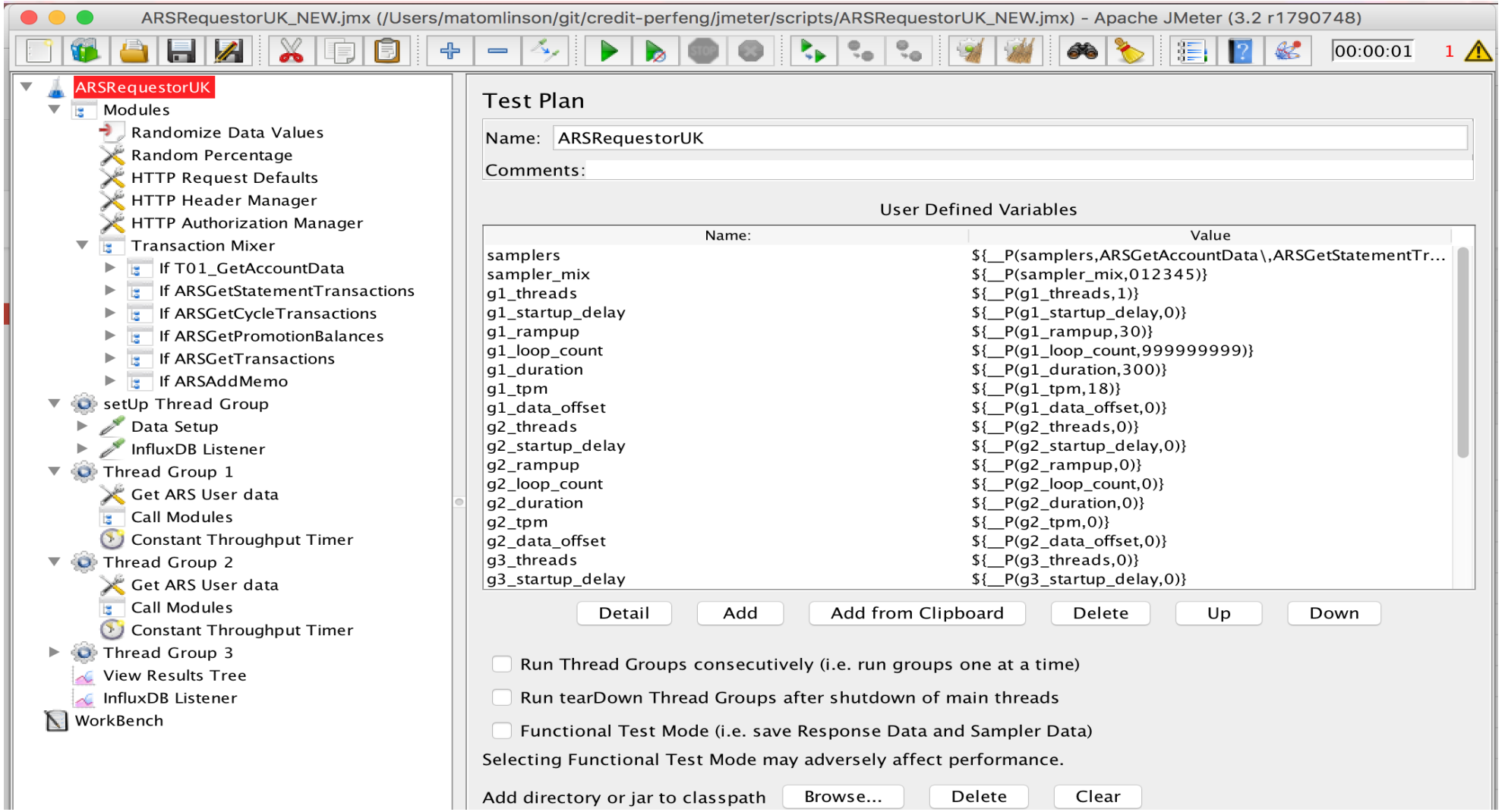
DEMONSTRATION: UNATTENDED AUTOMATION WITH RUNDECK



REVIEW: UNATTENDED AUTOMATION WITH RUNDECK

- Using Rundeck we achieve the following:
  - Separation of jobs for scripts, scenarios, utilities
  - Ability to schedule automatic start
  - Ability to “halt the automation” (e.g. a hold button)
  - Ability to run a script or scenario manually
  - Job status notification
  - Job activity history

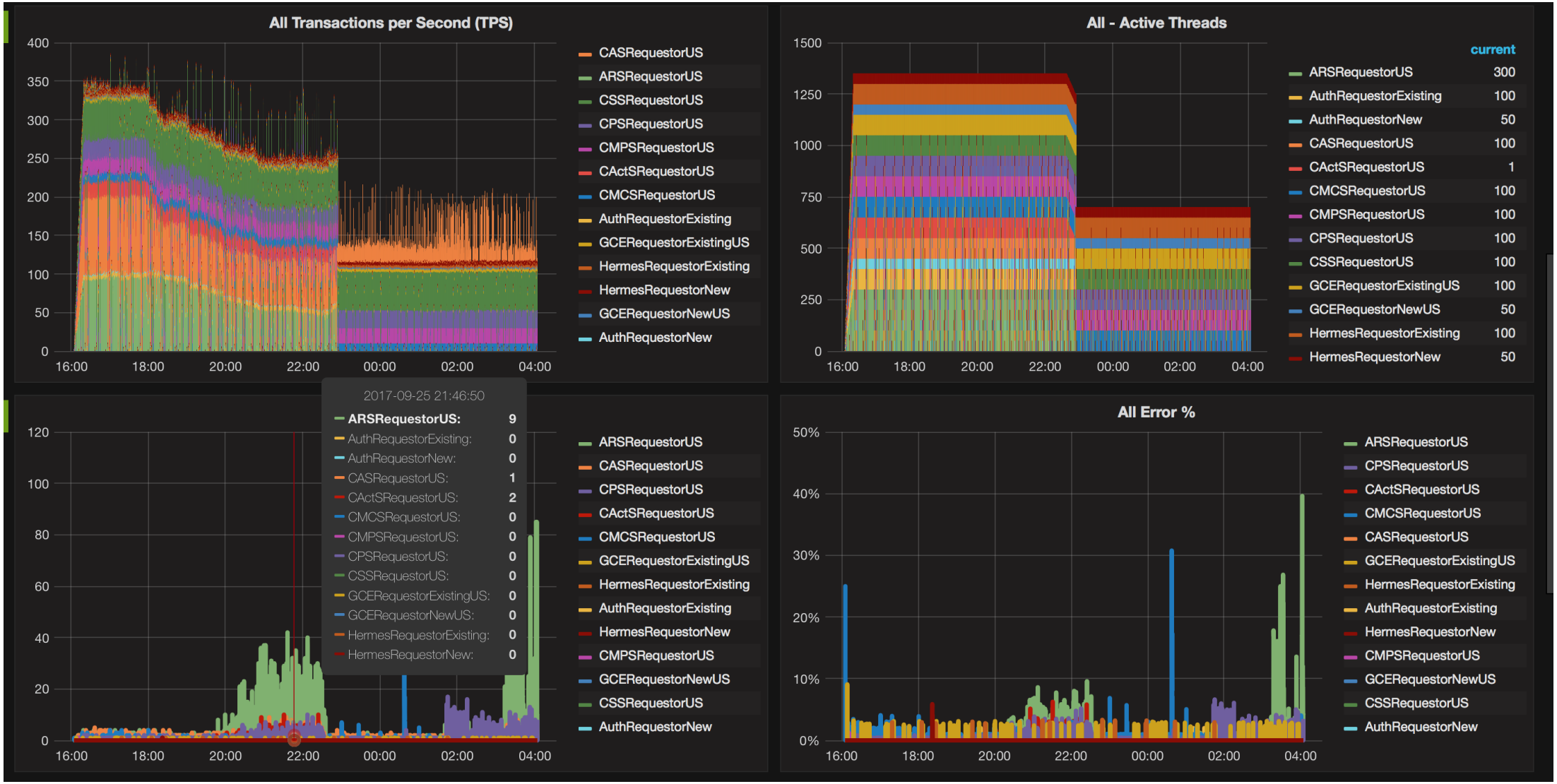
DEMONSTRATION: JMETER CUSTOMIZATIONS



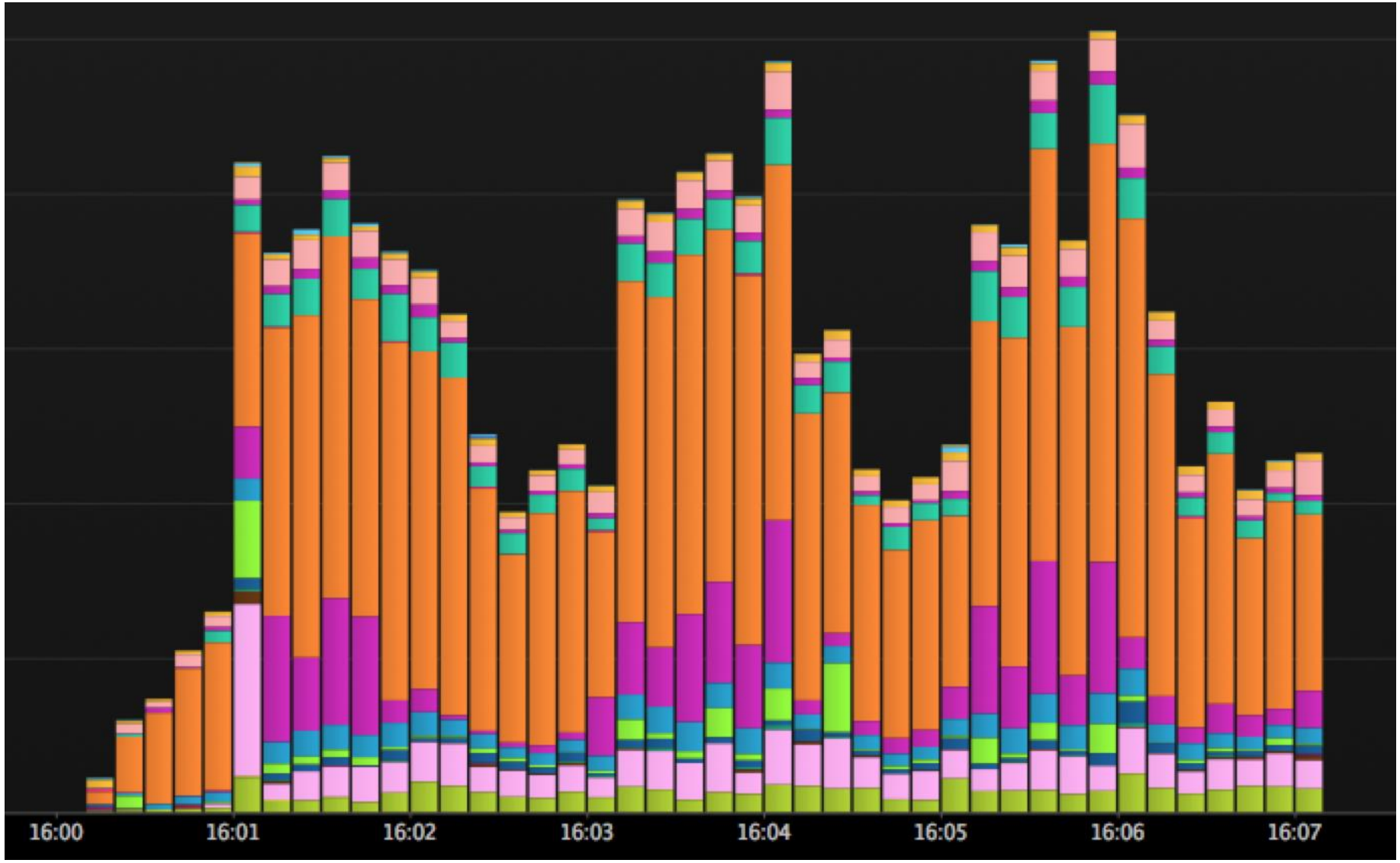
REVIEW: JMETER CUSTOMIZATIONS

- By Extending the testing tool we achieve the following:
  - Portability to different environments
  - Adaptability to serve multiple, different scenarios
  - Improved code-reuse and collaboration
  - Results extensibility to external data stores

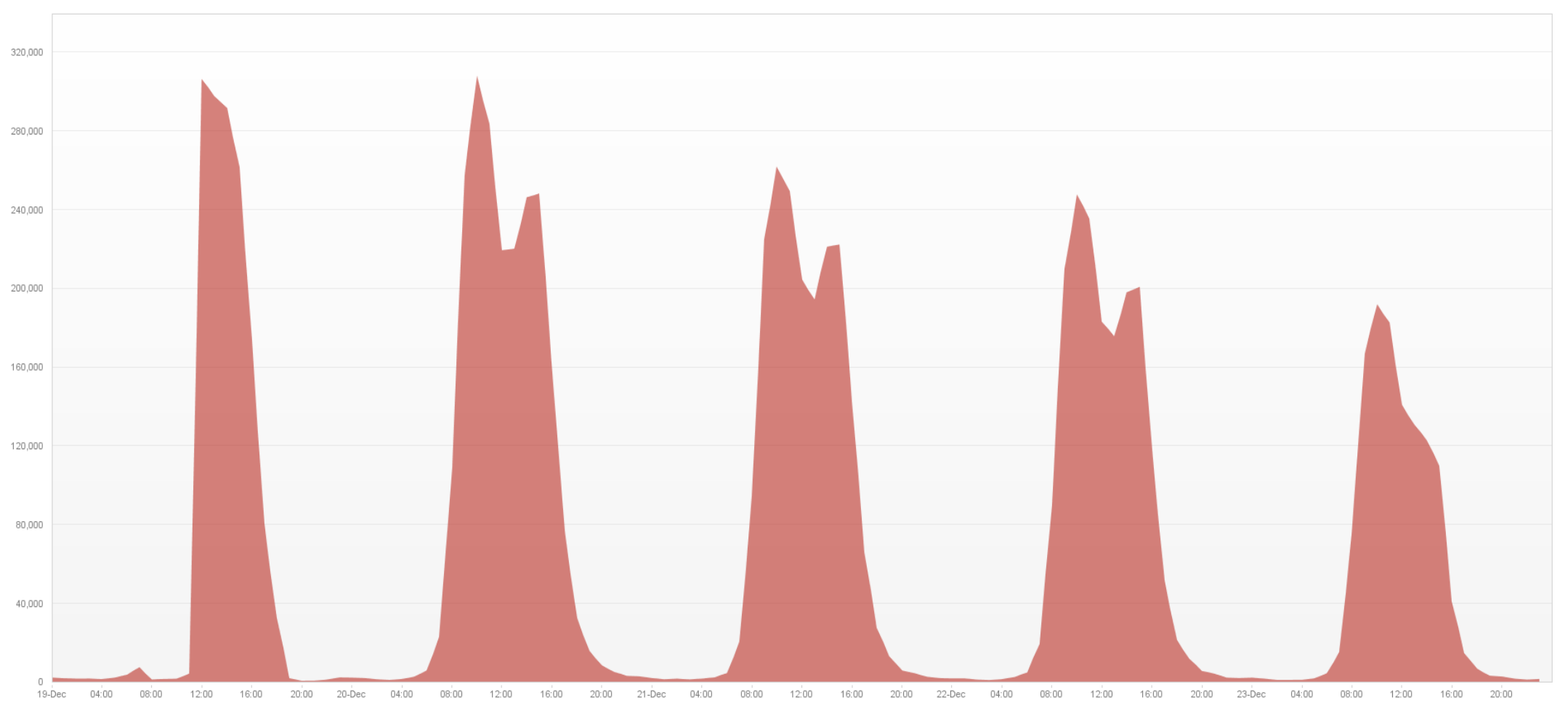
DEMONSTRATION: PERFORMANCE VISUALIZATION



DEMONSTRATION: PERFORMANCE VISUALIZATION



DEMONSTRATION: PERFORMANCE VISUALIZATION



REVIEW: PERFORMANCE VISUALIZATIONS

We reviewed the following in Grafana and Dynatrace:

- Connection to a time-series database (e.g. InfluxDB)
- Separation of dashboards by app
- Combined dashboard for all traffic
- Drill-down capability: digging deeper into metrics
- Consistent coloring, helps to remember metrics
- Advanced features for viewing timeframe and refresh
- Advanced metrics querying, granularity and aggregation

DEMONSTRATION: NOTIFICATION AND COMMUNICATION

PayPal
Mark Tomlinson

All Unreads
All Threads

Starred
help-credit-lnp
lt-1054
lt-1056
lt-1069
lt-db-sync 1
lt-ipt-domestic
lt-ipt-international
lt-pre-push
timonium-lt-te... 3
John Pfeifer IV
saiprasad setty
William Safee 1

Channels

#lt-ipt-domestic
27 | 0 | Add a topic

Search

Yesterday

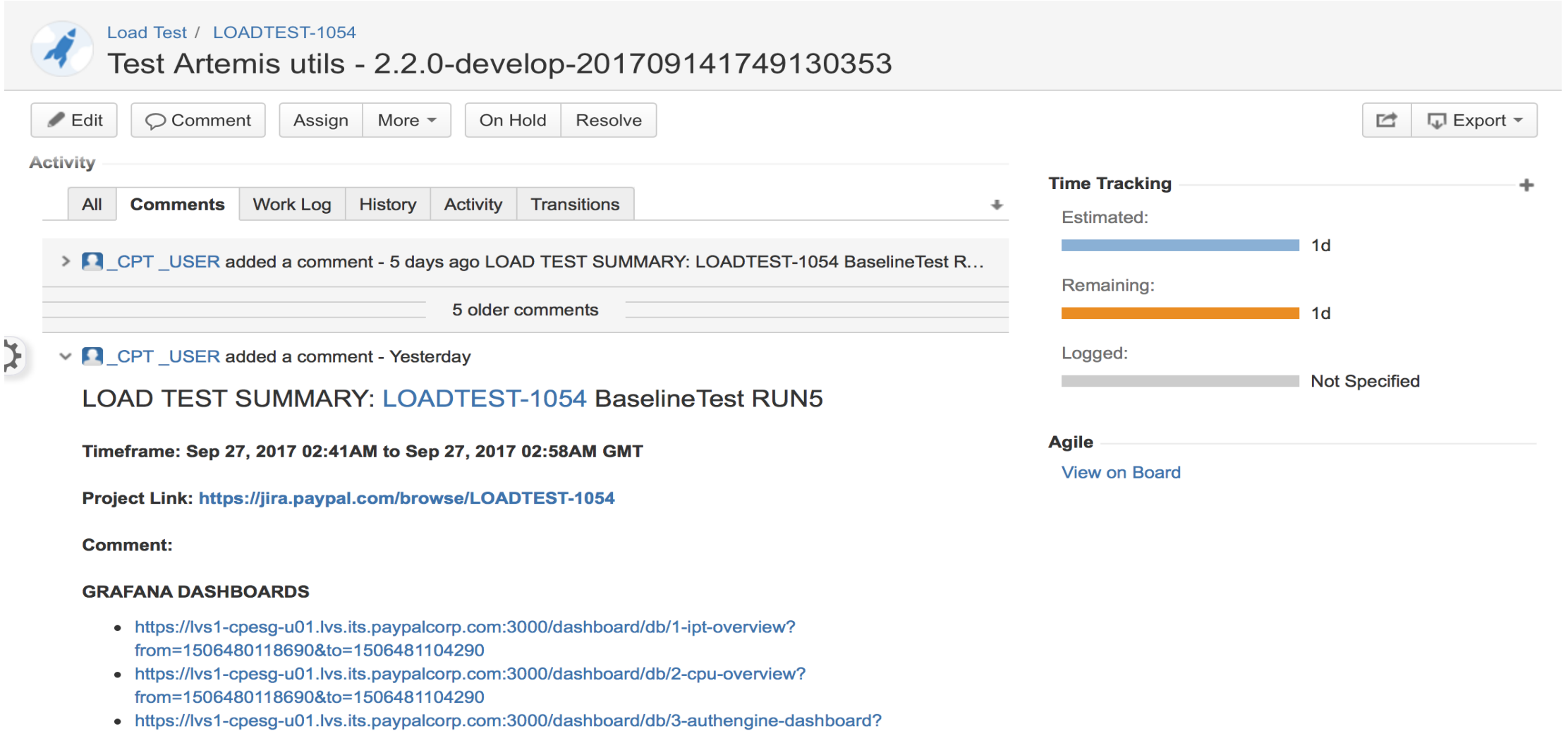
saiprasad setty 9:27 AM
— CNR + flashback
in LT01

saiprasad setty 9:33 AM
uploaded this image: Not a perfect Soak Run

saiprasad setty 9:33 AM
ARS requestor failed and error % was touching 40% for quite good amount of time
investigating the last night run

Terrance APP 9:38 AM

DEMONSTRATION: NOTIFICATION AND COMMUNICATION



REVIEW: NOTIFICATION AND COMMUNICATION

We reviewed the following in Slack:

- Individual channels for apps/teams
- Combined channels for all traffic
- Real-time notification of test events
- Bot-identity ..making the automation more personable

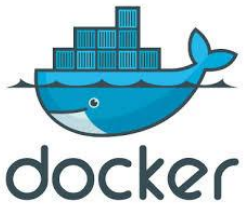
We reviewed the following in JIRA Comments:

- Automatic summary comment from the test results
- Links to drill-down into the results tools/dashboards

OTHER TOOLS

Consider a few other things you might need:

- Managing test data state continuously (flashbacks/restore)
- Service Virtualization - simulating external systems
- Support for different load generation tools
- Application log analysis – grepping for exceptions/errors
- Storage for all the results data…can get HUGE over-time

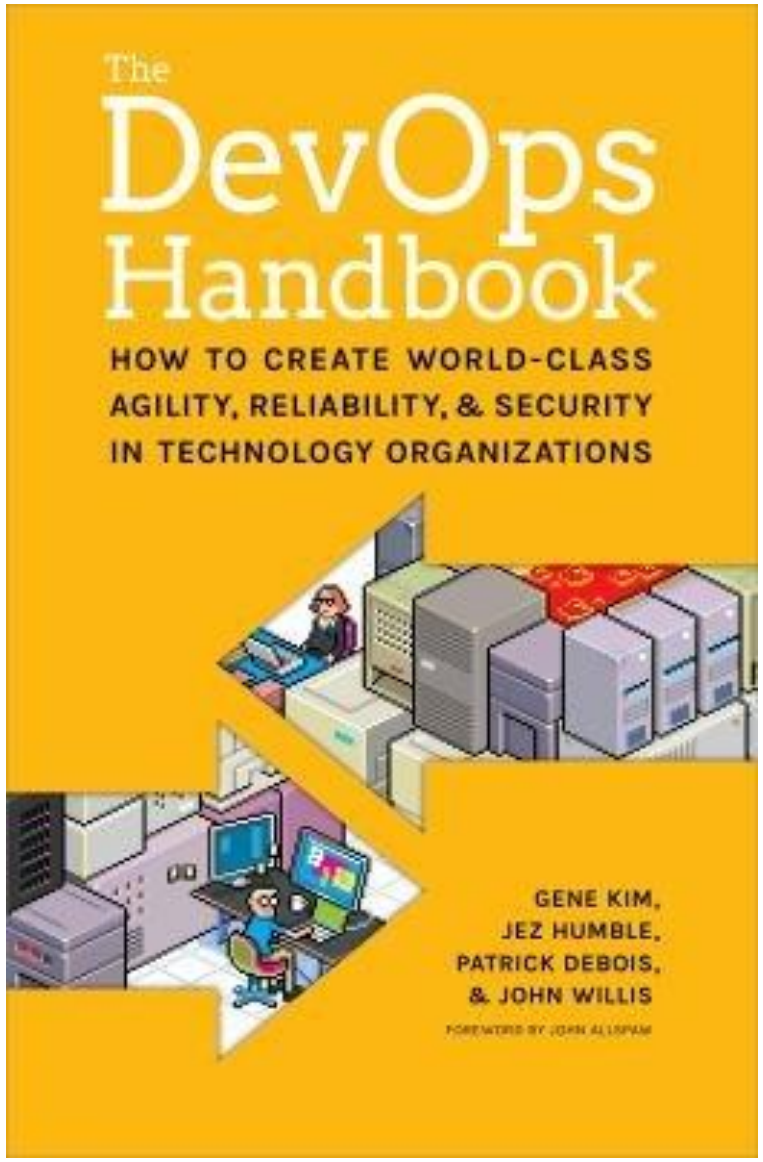



WHERE TO START...

WHERE TO START…

DevOps Handbook Part 2 “Where to Start”

- Greenfield vs. “Brownfield”
- Systems of Record vs. Engagement
- Sympathetic and Innovative Teams
- Expanding DevOps/Continuous-ness



Consider the following questions at the start:

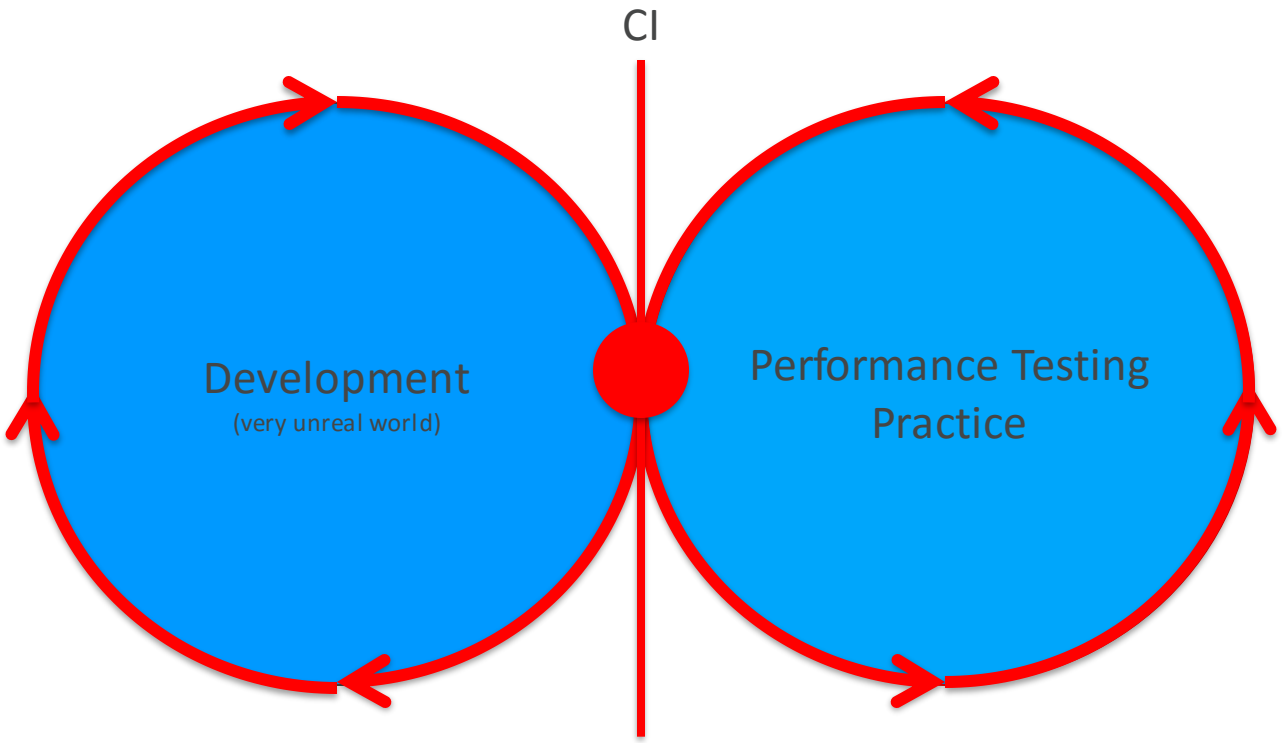
- are your stakeholders/customers ready for continuous feedback? How will they react? Will they value faster feedback?
- do you use information radiators/visual mediums to deliver the performance trends visually?
- do you have notification systems (email, chatops, Slack) for communication to results and trends?
- are your test environments configured to support unattended, non-stop test runs?
- do you have the tools and licenses ready for more executions?

GRADUAL ADOPTION:

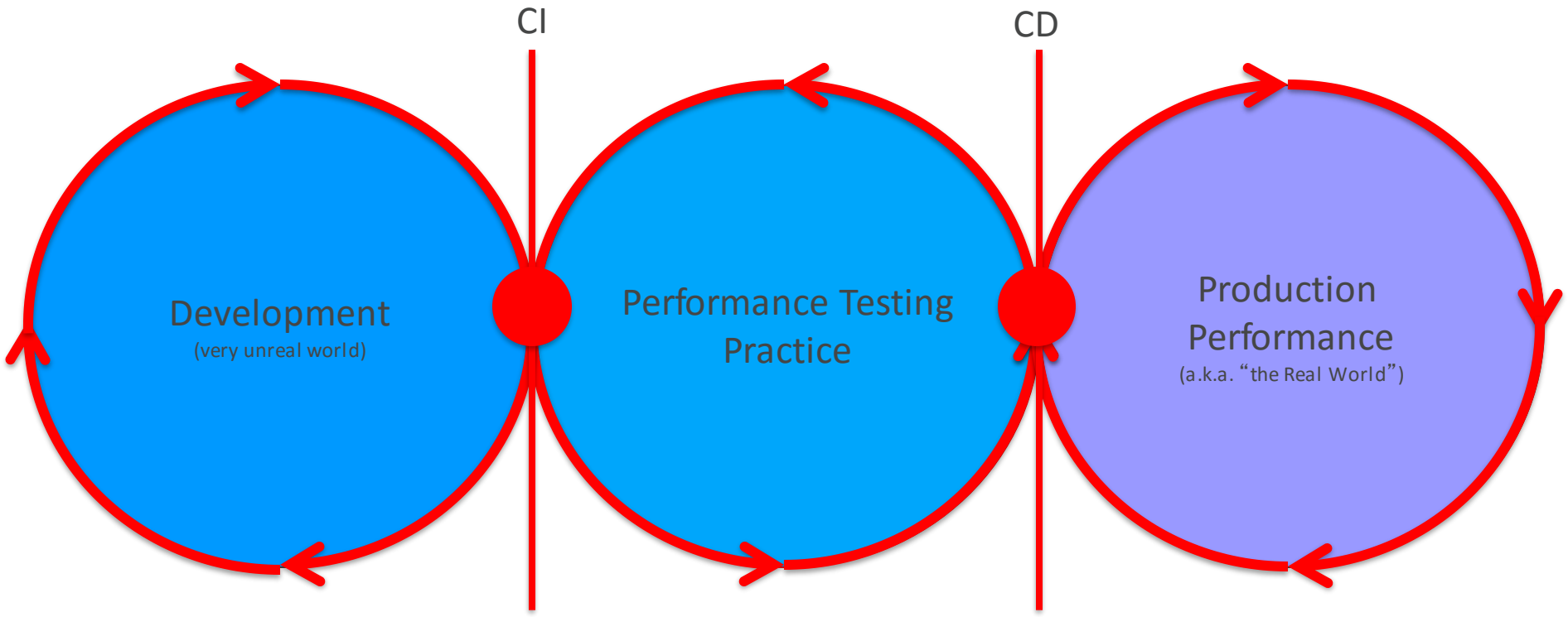
Slowly introduce continuous performance:

- Consider just 1 day a week to run non-stop testing
- Consider a team that already dominates your schedule
- It takes time for people to adjust to frequent feedback, how to react and behave
- You will also receive feedback on your tooling/visuals – take time to improve

Start with Dev…



…then add Production

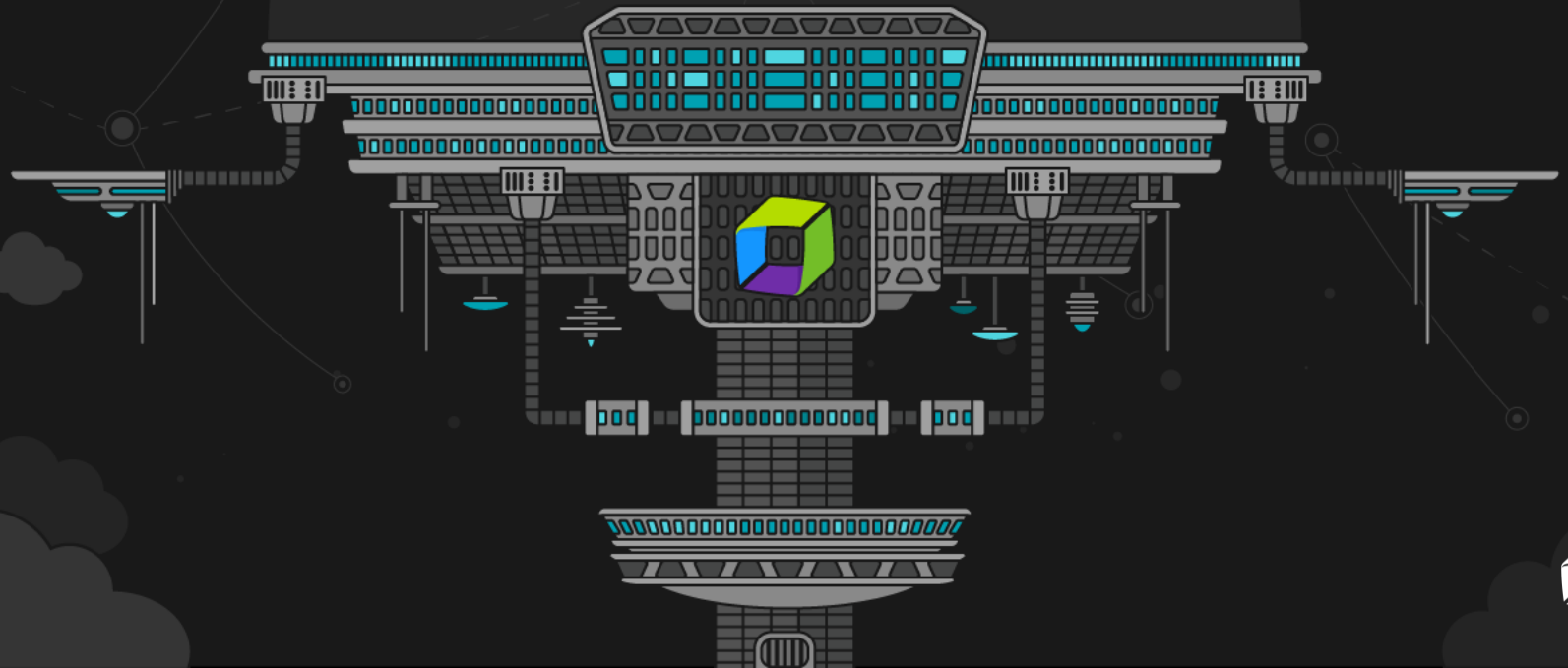


SUMMARY

- Performance can be measured continuously
- Know the difference: promotion and feedback
- One-time performance tests vs. repeated testing
- Absolute vs. relative measurements
- Performance decisions are made continuously
- Decisions and analysis are based on data
- Find ways to capture metrics across the lifecycle

Mark Tomlinson, Performacologist
mtomlins@perfbytes.com
mark-on-task.blogspot.com @mark_on_task
PerfBytes Podcast: @perfbytes

Thank you



dynatrace