



imPACt 2016 by CMG

Reinventing Performance Testing

Alexander Podelko

alex.podelko@oracle.com

alexanderpodelko.com/blog

@apodelko

November 9, 2016

The Stereotype

- Load / Performance Testing is:
 - Last moment before deployment
 - Last step in the waterfall process
 - Large corporations
 - Expensive tools requiring special skills
 - Protocol level record-and-playback
 - Lab environment
 - Scale-down environment
 - Checking against given requirements / SLAs
 - Throwing it back over the wall if reqs are not met
 - ...

Agenda

- Industry Trends and their Impact
 - Cloud
 - Agile Development
 - Continuous Integration
 - New Architectures
 - New Technologies
- Performance Engineering Puzzle: Changing Dynamics

Disclaimer: The views expressed here are my personal views only and do not necessarily represent those of my current or previous employers. All brands and trademarks mentioned are the property of their owners.

3

Cloud

- No more excuse of not having hardware
- Lab vs. Service (SaaS) vs. Cloud (IaaS)
 - For both the system and load generators
- Test vs. Production

4

Scenarios

- System validation for high load
 - Outside load (service or cloud), production system
 - Wider scope, lower repeatability
- Performance optimization / troubleshooting
 - Isolated lab environment
 - Limited scope, high repeatability
- Testing in Cloud
 - Lowering costs (in case of periodic tests)
 - Limited scope, low repeatability

5

Find Your Way

- If performance risk is high it may be a combination of environments, e.g.
 - Outside tests against the production environment to test for max load
 - Lab for performance optimization / troubleshooting
 - Limited performance environments to be used as part of continuous integration

6

Scaling

- Becomes *critical* as you get to a large number of virtual users
- The number of supported users per unit of computing power may differ drastically
 - Depending on tool, protocol, scenario, system...
- If you need deploy it on a large number of machines automation would be helpful

7

Agenda

- *Industry Trends and their Impact*
 - Cloud
 - Agile Development*
 - Continuous Integration
 - New Architectures
 - New Technologies
- Performance Engineering Puzzle: Changing Dynamics

8

Agile Development

- Agile development should be rather a trivial case for performance testing
 - You have a working system each iteration to test early by definition.
 - You need performance engineer for the whole project
 - Savings come from detecting problems early
- You need to adjust requirements for implemented functionality
 - Additional functionality will impact performance

9

The Main Issue on the Agile Side

- It doesn't [always] work this way in practice
- That is why you have "Hardening Iterations", "Technical Debt" and similar notions
- Same old problem: functionality gets priority over performance

10

The Main Issue on the Testing Side

- Performance Engineering teams don't scale well
 - Even assuming that they are competent and effective
- Increased volume exposes the problem
 - Early testing
 - Each iteration
- Remedies: automation, making performance everyone's job

11

Mentality Change

- Making performance everyone's job
- Late record/playback performance testing -> Early Performance Engineering
- System-level requirements -> Component-level requirements
- Record/playback approach -> Programming to generate load/create stubs
- "Black Box" -> "Grey Box"

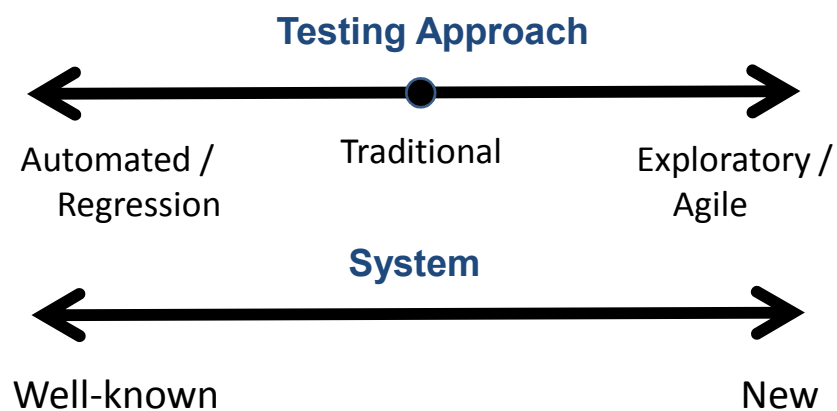
12

Exploratory Testing

- Rather alien for performance testing, but probably more relevant than for functional testing
- We learn about system's performance as we start to run test
 - Only guesses for new systems
- Rather a performance engineering process bringing the system to the proper state than just testing

13

Test Approach Dimension



14

Agenda

- *Industry Trends and their Impact*
 - Cloud
 - Agile Development
 - Continuous Integration*
 - New Architectures
 - New Technologies
- Performance Engineering Puzzle: Changing Dynamics

15

Continuous Integration

- CI support becoming the main theme
- Integration with Continuous Integration Servers
 - Jenkins, Hudson, etc.
 - Several tools announced integration recently
 - Making a part of automatic build process
- Automation support
- Cloud support
- Support of newest technologies

16

Automation: Difficulties

- Complicated setups
- Long list of possible issues
- Complex results (no pass/fail)
- Not easy to compare two result sets
- Changing Interfaces
- Tests may be long

17

Automation: Considerations

- You need know system well enough to make meaningful automation
- If system is new, overheads are too high
 - So almost no automation in traditional environments
- If the same system is tested again and again
 - It makes sense to invest in setting up automation
- Automated interfaces should be stable enough
 - APIs are usually more stable on early stages

18

Automation: Limitations

- Works great to find regressions and check against requirements
- Doesn't cover:
 - Exploratory tests
 - Large scale / scope / duration / volume
- “Full Automation” is not a real option, should be a combination

19

Agenda

- *Industry Trends and their Impact*
 - Cloud
 - Agile Development
 - Continuous Integration
 - New Architectures*
 - New Technologies
- Performance Engineering Puzzle: Changing Dynamics

20

New Architectures

- Software as a Service (SaaS)
 - Centrally managed
 - Multi-tenant
- Dynamic Architectures
 - Auto-scaling
- Third-party components and services

21

Consequences

- Major challenges for monitoring and analysis
- Service virtualization
- New application of performance testing

22

The Main Change in Monitoring

- Configuration becomes dynamic, changing on the fly
 - Auto scaling, auto provisioning, etc.
 - Challenge to monitor all moving parts
 - Challenge to compare results of dynamic configurations
 - Shift to application monitoring

23

The Main Change in Analysis

- Not only comparison with the requirements
- Many different forms of analysis depending on the tests
 - Adjusting to configuration / type of the test
 - Component testing
 - Automatic analysis / alerting
 - Continuous Integration / Delivery / Deployment
 - Input for tuning / optimization / sizing

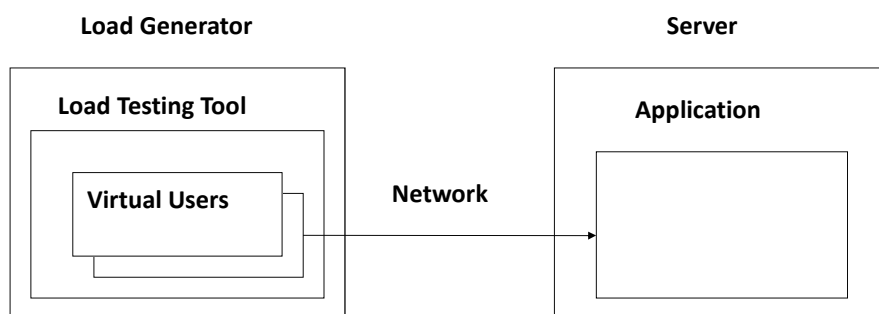
24

Agenda

- *Industry Trends and their Impact*
 - Cloud
 - Agile Development
 - Continuous Integration
 - New Architectures
 - New Technologies*
- Performance Engineering Puzzle: Changing Dynamics

25

Record and Playback: Protocol Level



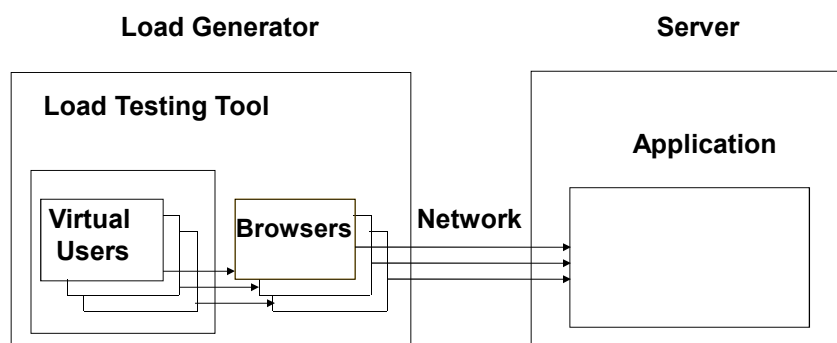
26

Considerations

- Usually doesn't work for testing components
- Each tool support a limited number of technologies (protocols)
- Some technologies are very time-consuming
- Workload validity in case of sophisticated logic on the client side is not guaranteed

27

Record and Playback: UI Level



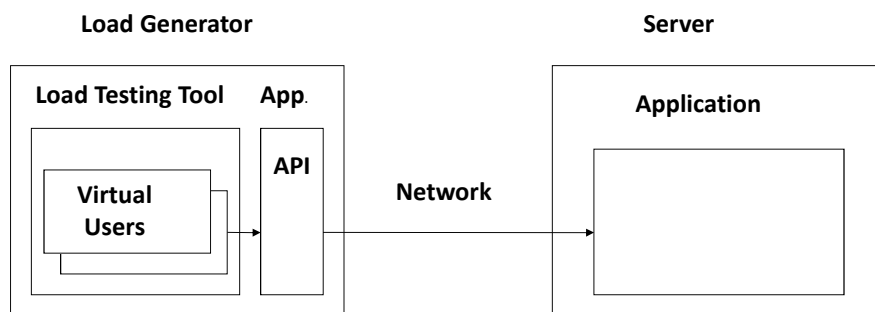
28

Considerations

- Scalability
 - Still require more resources
- Supported technologies
- Timing accuracy
- Playback accuracy
 - For example, for HtmlUnit

29

Programming



30

Considerations

- Requires programming / access to APIs
- Tool support
 - Extensibility
 - Language support
- May require more resources
- Environment may need to be set

31

Agenda

- Industry Trends and their Impact
 - Cloud
 - Agile Development
 - Continuous Integration
 - New Architectures
 - New Technologies
- *Performance Engineering Puzzle: Changing Dynamics*

32

Load Testing

- Traditional load testing is not enough anymore
- New industry trends change a lot
 - Cloud
 - Continuous Integration / Delivery / Deployment
 - DevOps
 - Agile
- Some even say that load testing is not needed anymore
 - Due to other ways to mitigate performance risk

33

Performance Risk Mitigation

- Single-user performance engineering
 - Profiling, WPO, single-user performance
- Software Performance Engineering
 - Modeling, Performance Patterns
- Instrumentation / APM / Monitoring
 - Production system insights
- Capacity Planning/Management
 - Resources Allocation
- Continuous Integration / Deployment
 - Ability to deploy and remove changes quickly

34

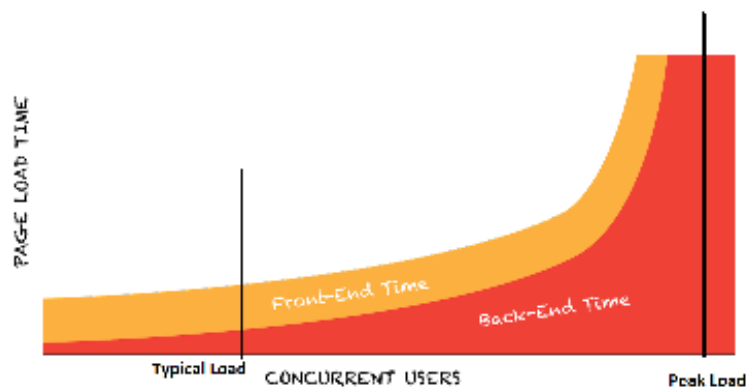
*But all of them
don't replace load
testing:*

*Load testing
complements them in
several important
ways !*

35

Can System Handle Peak Load?

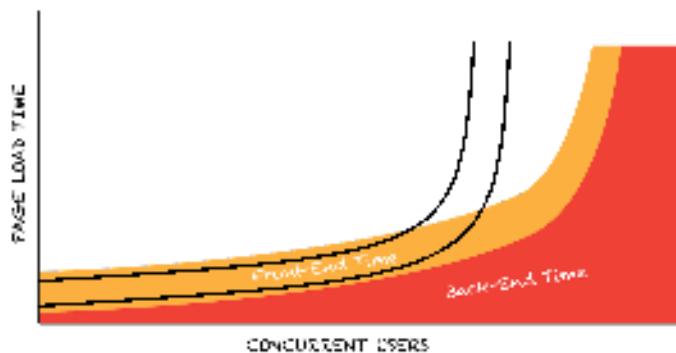
- You can't know without testing:



36

Verify Multi-User Performance

- Single-user improvement may lead to multi-user performance degradation



37

What Else Load Testing Adds

- Performance optimization
 - Apply exactly the same load
 - See if the change makes a difference
- Debugging/verification of multi-user issues
- Testing self-regulation functionality
 - Such as auto-scaling or changing the level of service depending on load

38

Changing Dynamic / Historical View

- Mainframes
 - Instrumentation, Scheduling, Capacity Planning
- Distributed Systems
 - Load Testing, System Monitoring
- Web / Cloud
 - App Monitoring, Perf Engineering

39

So What Is Going On?

- I believe that load testing is here to stay, but should fully embrace the change
 - Not one-time, to become dynamic
- Dynamic of using different PE approaches is changing
 - As it was during the whole history of PE
- Probably there would be less need for "performance testers" limited only to running tests, but more need for performance experts who would be able to see the whole picture using all available tools and techniques.

40

Summary

- The industry is rapidly changing – performance testing should change too
 - Much more context- and business-driven
 - Fully embracing agile (early testing), cloud, continuous integration, automation, new architectures and technologies
- Good tools help, but there is no “best tool” – it depends on your needs
 - Any tool is only a tool

41

Questions?

Alexander Podelko

alex.podelko@oracle.com

alexanderpodelko.com/blog

[@apodelko](#)