BELGIUM TESTING DAYS 2013

# Agile Aspects of Performance Testing

**Alexander Podelko**

alex.podelko@oracle.com
www.alexanderpodelko.com
@apodelko

March 1, 2013

1

## About Me

- Specialize in performance last 16 years
- Currently performance testing and optimization of Hyperion products at Oracle
- Blog at http://alexanderpodelko.com/blog, have a collection of performance-related links at http://alexanderpodelko.com, on Twitter as @apodelko
- Involved in CMG http://cmg.org, a volunteer organization of performance and capacity planning professionals
  - Conference November 4-8, 2013 in La Jolla, CA



2

# Agenda

- Status Quo
- Performance Testing in Agile Project
- Agile Performance Testing

3

# Performance Testing

- Here used as an umbrella term including load, stress, concurrency, etc. testing
- An important project step in many corporations
- A project itself
  - We can apply software development methodologies to this project
- Waterfall approach in most cases
  - Often pre-production performance validation

4

# Assumptions

*Two important assumptions for this session:*

1. Stakeholders understand the need
2. The team knows the basics

*How to get there are very interesting topics, but outside this session*

5

# Waterfall Approach to Performance Testing

- Flowing steadily downwards through the phases
  - Get the system ready
  - Develop scripts requested
  - Run scripts in the requested combinations
  - Compare with the provided requirements
  - If requirements missed, throw back to development

6

## Main Problems

- We learn about system by preparing scripts and running tests
  - With each test
- We don't have much information about the system before tests to do a reliable plan
- Much more than in development and functional testing
  - Often based on internal communication
- It is agile / explorative by nature

7

## Main Problems

- The system must be ready
  - Late changes are expensive
  - If we want earlier, should be more agile / explorative
- Test scripts are also software
  - Even with record/playback you still need to correlate, parameterize, debug, and verify
  - Many details get uncovered as you go

8

## Main Problems

- Running all scripts together makes it very difficult to tune and troubleshoot
  - Shot-in-the-dark anti-pattern
  - Tuning and troubleshooting are iterative
- Minimal information about the system behavior
  - You cannot build any kind of model

9

## Manifesto for Agile Software Development

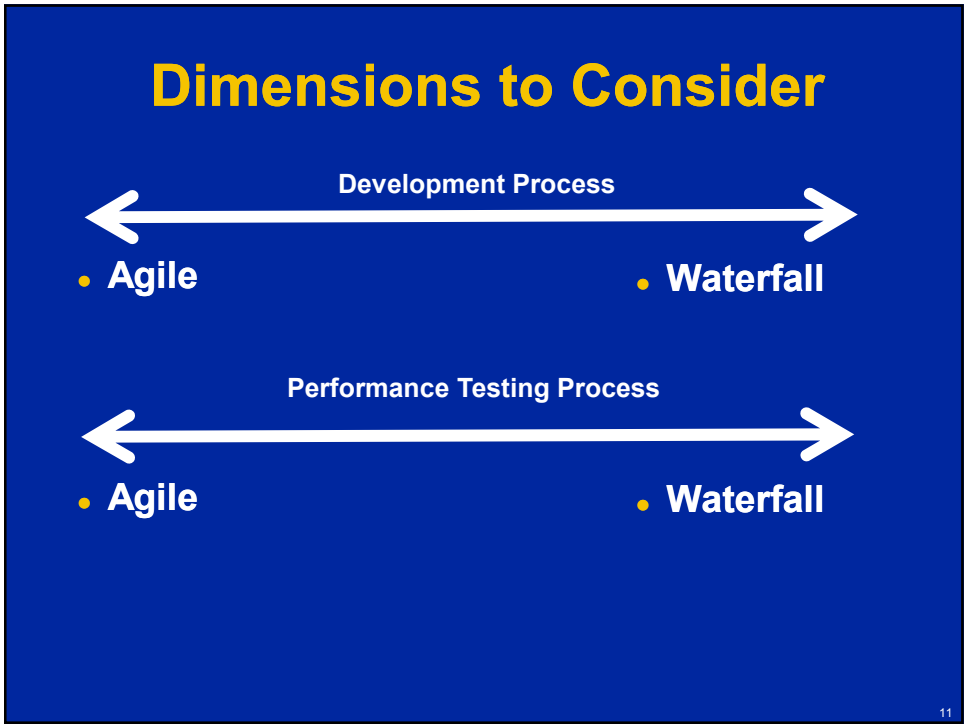
*We are uncovering better ways of developing software by doing it and helping others do it.*

*Through this work we have come to value:*

- *Individuals and interactions over processes and tools*
- *Working software over comprehensive documentation*
- *Customer collaboration over contract negotiation*
- *Responding to change over following a plan*

*That is, while there is value in the items on the right, we value the items on the left more.*

10


Dimensions to Consider
Development Process
Agile
Waterfall
Performance Testing Process
Agile
Waterfall
11


Agenda
Status Quo
Performance Testing in Agile Project
Agile Performance Testing
12

## Agile Development

- Agile development is 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

13

## 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

14

## 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
  - Each iteration
- Remedies: automation, making performance everyone's job

15

## Automation: Difficulties

- Complicated setups
- Long list of possible issues
- Complex  results (no pass/fail)
- Not easy to compare two result sets

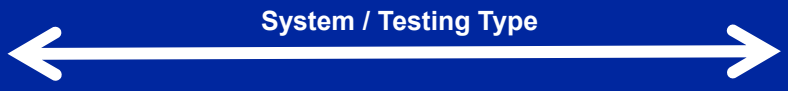
16

## 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 environment [with testing in the end]
- If the same system is tested again and again
  - It makes sense to invest in setting up automation

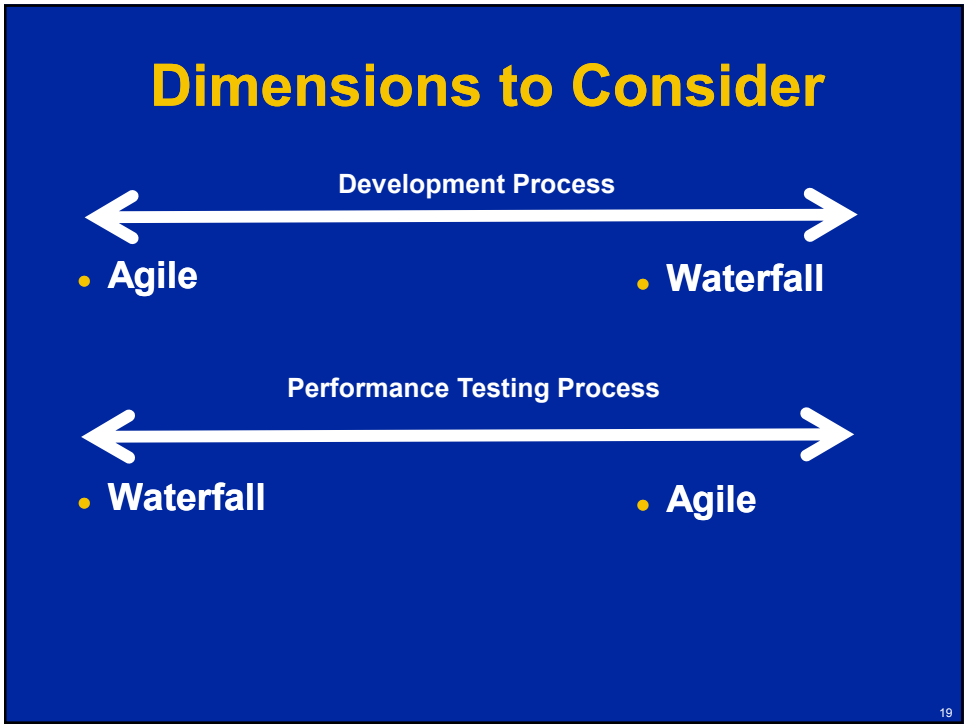
17

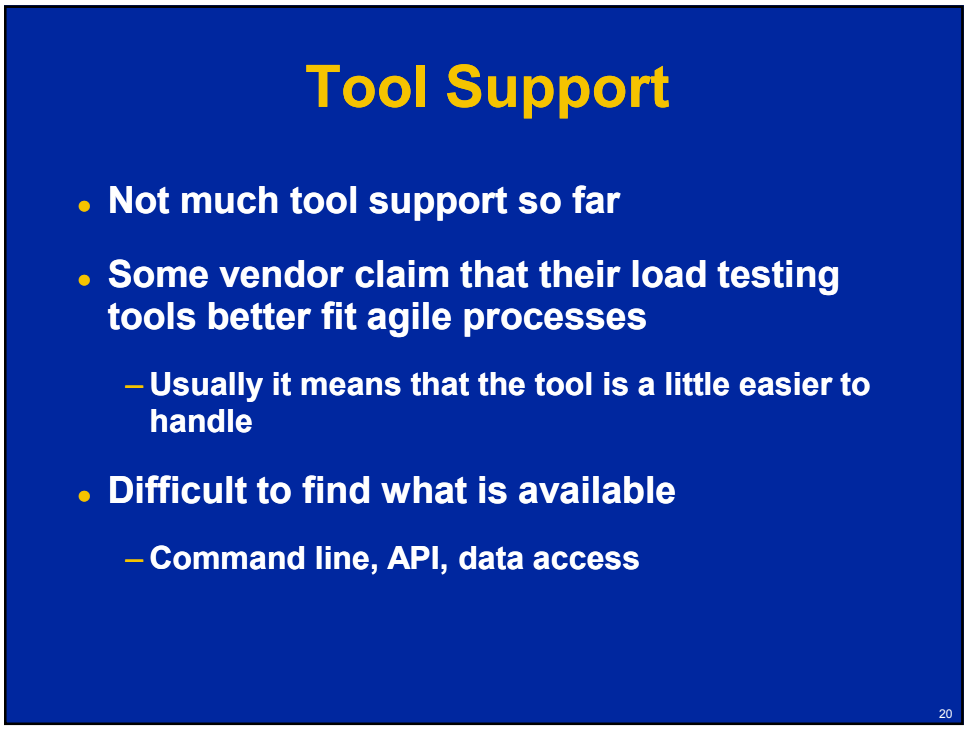
## One More Dimensions to Consider




- New (Exploratory)
- Well-known (Regression)

18


Dimensions to Consider
Development Process
Agile
Waterfall
Performance Testing Process
Waterfall
Agile
19


Tool Support
Not much tool support so far
Some vendor claim that their load testing tools better fit agile processes
Usually it means that the tool is a little easier to handle
Difficult to find what is available
Command line, API, data access
20

# Performance Challenges in Agile Projects

- Performance-related activities usually span the whole project
- Specifying performance requirements
  - Constraints

    As user stories should represent finite manageable tasks
  - User stories

    Separating of initial and ongoing compliance

21

# Heretic Thought

*"we have come to value:*
- *Individuals and interactions over processes and tools"*

- Is agile process oxymoron?
- Maybe different dimensions:
  - Waterfall – Iterative
  - Process Oriented – Agile
  - Build to Grow – Lean

22

# Agenda

- Status Quo
- Performance Testing in Agile Project
- Agile Performance Testing

23

# Agile

- 'Agile software development refers to a group of software development methodologies that promotes development iterations, open collaboration, and process adaptability through the life-cycle of the project'
- Fully applicable to performance testing
- Performance testing of new systems is agile by its nature
  - As far as we learn about the system with each test

24

## Agile Performance Testing

- Finding new opportunities inside existing processes using collaboration, iterative and adaptive processes
- Separate from performance testing in agile development, which was discussed earlier
- Most good performance engineers already use similar approaches
  - Although waterfall-like plan is usually presented to management

25

## Testing Early

- The main problem in waterfall development
- Nobody argues against, but rarely happens in practice
  - In agile projects it supposed to happened automatically
- Require different, more agile approach
- Making performance everyone's job

26

## Mentality Change

- 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"

27

## Unit Performance Testing

- Any part of the system
- Not a standard practice
- Do not wait the system is assembled
- Test cases are simpler, fewer variables
- Many systems are monolithic
  - Test-Driven Development may be an answer
- Third-party components

28

## Single-User Performance

- If performance for one user isn't good, it won't be any better for multiple users
- Single-user testing is conducted throughout the development life cycle
  - Gathering performance data can be extremely helpful
  - Can provide a good indication of what business functions need to be investigated further

29

## Early Involvement

- Any early involvement would be beneficial
  - Even if only asking a few key questions
  - Don't wait until everything gets in place
  - A few guerrilla-style actions can save a lot of time and resources later
- Unfortunately, you often get involved in the project at a later stage
  - Next sections are still fully applicable

30

## Don't Underestimate Workload Generation

### Only the Paranoid Survive

- The title of Andy Grove's book should be performance engineering's motto

31

## Be a Performance Test Architect

- Gather all requirements and project them onto the system architecture
  - Business requirements are not the "Holy Scripture"
  - Iterative process, evaluate and validate
  - Scrutinize workload
  - Project requirements onto the system architecture
    - What components are involved

32

## Be a Performance Test Architect

- Make sure that the system is properly configured and results are meaningful
  - Difference between environments
  - Data used
  - User access
- Make the test environment as close to the production as possible
- Performance testing isn't an exact science

33

## Scripting Process

- Record/playback is easy for static content and plain HTML
  - Not many such systems anymore
  - Learn as you script
- Software Development Project
  - Correlate and parameterize
  - Validate
    - Lack of error messages is not the proof
    - Test different input data

34

# Scripting Example

- Back-end calculation (Financial consolidation)
  - Long time, shows progress bar
  - Polling back-end
  - Explicit loop is needed to work properly

35

# Recorded Script

```
web_custom_request("XMLDataGrid.asp_7","URL={URL}/
  Data/XMLDataGrid.asp?Action=EXECUTE&TaskID=1024
  &RowStart=1&ColStart=2&RowEnd=1&ColEnd=2&SelTy
  pe=0&Format=JavaScript", LAST);
web_custom_request("XMLDataGrid.asp_8","URL={URL}/
  Data/XMLDataGrid.asp?Action=GETCONSOLSTATUS",
  LAST);
web_custom_request("XMLDataGrid.asp_9","URL={URL}/
  Data/XMLDataGrid.asp?Action=GETCONSOLSTATUS",
  LAST);
web_custom_request("XMLDataGrid.asp_9","URL={URL}/
  Data/XMLDataGrid.asp?Action=GETCONSOLSTATUS",
  LAST);
```

36

# Working Script

```
web_custom_request("XMLDataGrid.asp_7","URL={URL}/
  Data/XMLDataGrid.asp?Action=EXECUTE&TaskID=1024
  &RowStart=1&ColStart=2&RowEnd=1&ColEnd=2&SelTy
  pe=0&Format=JavaScript", LAST);
do {
  sleep(3000);
  web_reg_find("Text=1","SaveCount=abc_count",LAST);
  web_custom_request("XMLDataGrid.asp_8","URL={UR
  L}/Data/XMLDataGrid.asp?Action=GETCONSOLSTATU
  S",  LAST);
 } while (strcmp(lr_eval_string("{abc_count}"),"1")==0);
```

37

# Performance Testing

- Often is not separated from:
  - Tuning
    - System should be properly tuned
  - Troubleshooting / Diagnostics
    - Diagnose to the point when it is clear how to handle
  - Capacity Planning / Sizing
- "Pure" performance testing is rare
  - Regression testing ?

38

## Tuning and Troubleshooting

- Both are iterative processes
  - Make a change
  - Run the test
  - Analyze results
  - Repeat if necessary
- You apply the same synthetic workload
  - Easy to quantify the impact of the change

39

## Issues

- Requires close collaboration of all stakeholders
- Impossible to predict how many test iterations would be necessary
  - Shorter / simpler tests may be needed
  - Complex tests may make problems less evident
- Usually no indications at the beginning how much time and resources would be needed

40

## Process

- Asynchronous process doesn't work well
  - Problem often blocks further testing
  - Full setup is usually needed to reproduce
  - Debugging is a collaborative process
- Open collaboration needed
  - Synchronized work of all stakeholders to fix problems and complete performance testing

41

## Building a Model

- Significantly increases the value of performance testing
  - One more way to validate tests and help to identify problems
  - Answers questions about sizing the system
- Doesn't need to be a formal model
  - May be just simple observations as to how much resources are needed by each component

42

# Modeling

- Often is associated with queuing theory
  - Great mechanism, but not required in simple cases
- Most good performance engineers have a model in their mind
  - May be not documented / formalized
  - But they see when the system behaves in an unusual way

43

# Sometimes Linear Model Works

- Linear model may often work for multi-processor machines
  - Considering the working range of CPU utilization

    Most IT shops don't want more than 70-80%
  - Throughput and CPU utilization increase proportionally
  - Response times increase insignificantly

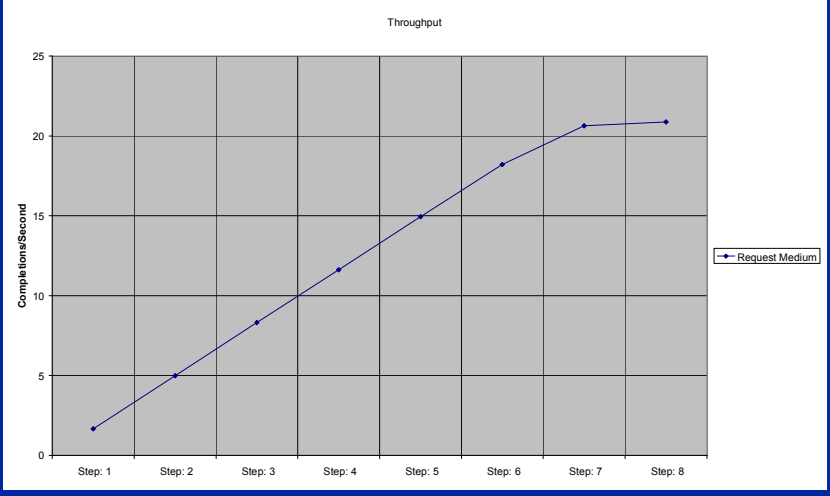
44

# Example

- Simple queuing model was built using TeamQuest
- Specific workload executed on a four-way server
- Eight different load levels
  - Step 1 represents 100-user workload, each next step adds 200 users, step 8 represents 1,500 users
- Linear model would be good through step 6
  - With CPU utilization 87%

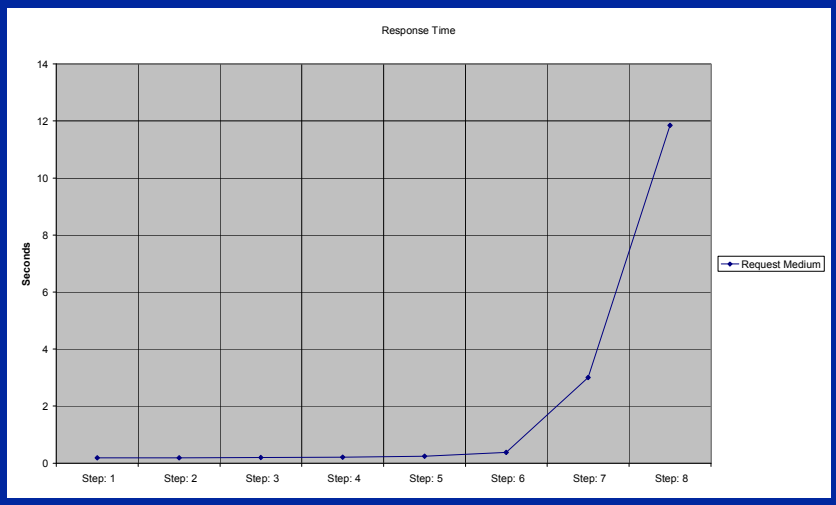
45

# Throughput



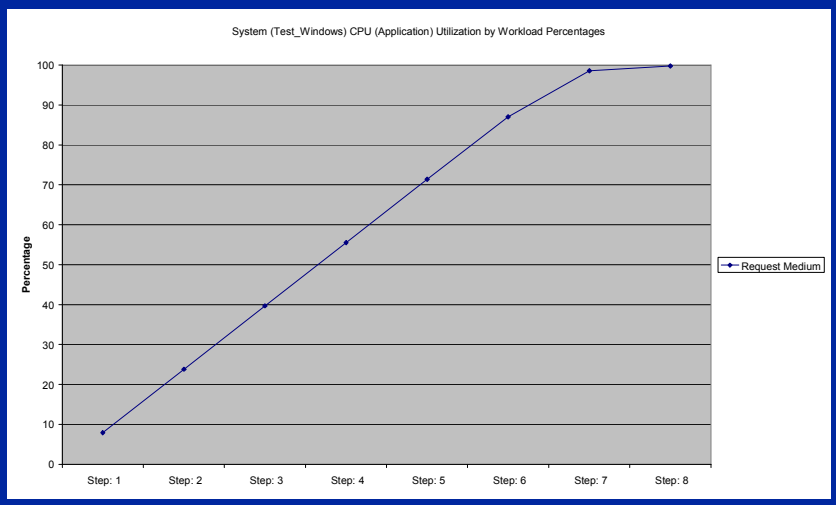

46

# Response Time




47

# CPU Utilization




48

## How to Approach Modeling?

- Run independent tests for each business function
  - To see how much resources each function requires
  - Load shouldn't be too light
    - To get steady, not distorted by noise resource utilization
  - Load shouldn't be too heavy
    - To avoid non-linear effects

49

## Agile Performance Testing

- Get involved early if possible
- Have a plan, but it needs to be very adaptable
- Expect iterative process involving tuning and troubleshooting
  - Each test provides a lot of information
  - If you find a bottleneck, you need to fix it before running further tests

50

## Agile Performance Testing

- As soon as you get a working script
  - Run it for one, a few, and many users
  - Sort out errors
  - Note resource utilization (build a "model")
- Even with a single script you can find many problems and, at least partially, tune the system

51

## Agile Performance Testing

- Make sure that every script works before running a real-life mixed scenario
  - Verify that the system behaves as expected
  - Pay attention to every issue / deviation
- Work in close cooperation with developers, administrators, and other experts

52

## Summary – Agile Projects

- If agile implemented properly, it is much easier and less risky to do performance testing
  - System to test at the end of each iteration
- Automation and making performance everyone's job to handle work increase
- Probably would require more resources
  - Payoff is in decreasing risks and finding problems early

53

## Summary – Agile Testing

- Small extra efforts, making the process more agile, increase efficiency significantly – and usually pay off multi-fold
  - Get involved early
  - Be paranoid about workload generation / system setup
  - Expect multiple tuning and troubleshooting iterations
  - Build a "model"

54

# References


Scott Barber. An Explanation of Performance Testing on an Agile Team.
http://www.logigear.com/newsletter-2007/320-an-explanation-of-performance-testing-on-an-agile-team-part-1-of-2.html
http://www.logigear.com/newsletter-2007/321-an-explanation-of-performance-testing-on-an-agile-team-part-2-of-2.html

Lisa Crispin, Janet Gregory. Agile Testing: A Practical Guide for Testers and Agile Teams. Pearson Education, 2009.

Jamie Dobson. Agile Performance Testing.
http://jamiedobson.co.uk/blog/agile-performance-testing

Chapter 6 of Performance Testing Guidance for Web Applications. Managing an Agile Performance Test Cycle.
http://perftestingguide.codeplex.com/Wiki/View.aspx?title=Chapter%206%20%u2013%20Managing%20an%20Agile%20Performance%20Test%20Cycle

Alexander Podelko. Agile Performance Testing .
http://www.alexanderpodelko.com/docs/Agile_Performance_Testing_CMG08.pdf


55

# References


Agile Performance Testing process. AgileLoad whitepaper.
http://www.agileload.com/agileload//blog/2012/10/22/agile-performance-testing-process---whitepaper

Scott Barber. Performance Testing in the Agile Enterprise.
http://www.slideshare.net/rsbarber/agile-enterprise

Jason Buksh. Performance By Design – an Agile Approach.
http://www.perftesting.co.uk/performance-by-design-an-agile-approach/2011/11/18/

Vikas Hazrati. Nailing Down Non-Functional Requirements.
http://www.infoq.com/news/2011/06/nailing-quality-requirements

Patrick Kua. Top Ten Secret Weapons For Agile Performance Testing
http://www.slideshare.net/melnykenator/top-ten-secret-weapons-for-agile-performance-testing


56

# Questions?

**Alexander Podelko**

alex.podelko@oracle.com
@apodelko

*Additional links and references may be found at www.alexanderpodelko.com*

57