

THE FUTURE OF PERFORMANCE ENGINEERING

EMPIRICAL ANALYSIS OF CURRENT TRENDS

Alexander Podelko
Senior Performance Engineer, AWS

2023 CONFERENCE | FEBRUARY 21-23, 2023 | VIRTUAL :: ATLANTA | [CMGIMPACT.COM](https://cmgimpact.com)

1

Alexander Podelko

- Specializes in performance since 1997
- Senior Performance Engineer at AWS – Amazon Aurora
 - Before worked for MongoDB, Oracle/Hyperion, Intel, and Aetna
- CMG Board Director
- SPEC RG Steering Committee Member



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.

2023 CONFERENCE | FEBRUARY 21-23, 2023 | VIRTUAL :: ATLANTA | [CMGIMPACT.COM](https://cmgimpact.com)

2

Industry Changes

3

History

- Mainframes – late 50s
 - Instrumentation, Monitoring, Capacity Planning – late 60s
- Distributed Systems - late 70s
 - Performance testing tools, performance engineering – late 80s
 - Web - mid 90s
 - Mobile - late 00s
- Cloud - late 00s

<https://alexanderpodelko.com/docs/The%20Past%20Present%20and%20Future%20of%20Performance%20Engineering%20CMG%202021.pdf>

4

4

What Changed?

- Web
 - Centralization [back to mainframes ?]
 - Open / unlimited workload
- Cloud
 - Further centralization [chargeback - back to mainframes ?]
 - Dynamic Configurations
- Agile / iterative development
 - Continuous Integration / Delivery / Deployment
- Mobile / WPO / Rich Internet Client
 - Shift to single-user client performance optimization

5

5

All Interconnected

Centralization

- => Control over deployments
- => Ability to deploy small changes
- => Agile development
 - => Fuzzier line between Dev and Ops (DevOps, SRE)
 - => Need for continuous performance engineering

6

Performance Risk Mitigation - Experimental

- Single-user performance engineering
 - Profiling, WPO, single-user performance
- Multi-user performance testing
 - Load, stress, etc. testing
- Resilience engineering
 - Reliability testing, chaos engineering

7

7

Performance Risk Mitigation - Observational

- Instrumentation / APM / Monitoring / Observability
 - Production system insights
- Testing in production
 - Canary, A/B, etc testing
- Continuous Integration / Deployment
 - Ability to deploy and remove changes quickly
- Self-managed systems
 - Auto-scaling, etc.

8

8

Performance Risk Mitigation - Analytical

- Software Performance Engineering
 - Modeling, Performance Patterns / Anti-patterns
- Capacity Planning/Management
 - Resources Allocation
 - FinOps, Cost Management

9

9

Changing Dynamic / Historical View

- Mainframes
 - Methods: Instrumentation, Scheduling, Capacity Planning
 - Titles: Performance Analyst, Capacity Planners
- Distributed Systems
 - Load Testing, System Monitoring
 - Titles: Performance Tester [Engineer, Architect]
- Web / Cloud
 - Methods: App Monitoring, Observability, CI/CD
 - Titles: Performance Engineer ???

10

10

Existing Surveys

- [How is Performance Addressed in DevOps? A survey on Industrial Practices](#)
 - “most surveyed companies do not regularly conduct performance evaluations”
- The flaw: Trends are defined by frontrunners, not by the majority
 - It is still in the beginning, but the trend is clear - integration of performance engineering into agile development, DevOps, etc. -

11

11

Profession Changes

12

Different Roles

- Traditional: “Consultant”
 - Test the system in its current state
 - External or internal
 - Why bother about CI / automation / etc?
- New: “Performance Engineer”
 - On an agile team
 - Need to test it each build/iteration/sprint/etc.
- Automation Engineer / SDET / etc.

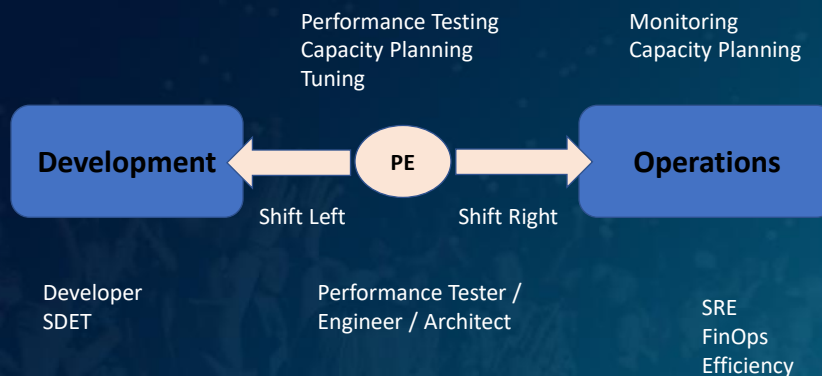
Still majority in the industry

What frontrunners looking for

13

13

Integrating Performance Engineering into DevOps



Expand or be Squeezed Out ?

14

14

Five Trends of Performance Engineering

- Adjusting to Agile and CI/CD
- Integrating performance engineering into DevOps
- Context-driven performance engineering
- Integrating everything: tools, processes, roles
- Chaos engineering: renaissance of reliability

Originally published in State of Performance Engineering Report 2020

15

15

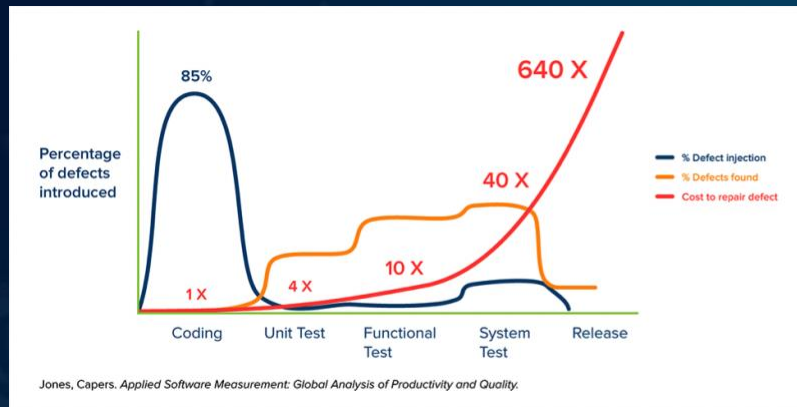
Adjusting to Agile and CI/CD

- Agile development should be rather a trivial case for performance testing
 - Working system on each iteration by definition
 - You need performance engineer for the whole project
 - Savings come from detecting problems early
- Performance Engineering teams don't scale well
 - Increased volume exposes the problem
 - Early testing
 - Each iteration
- Remedies: automation, making performance everyone's job

16

16

Cost of fixing defects during earlier phases of application life cycle is significantly lower



17

17

Early Testing - 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"

18

18

Context-Driven Performance Engineering

- Select a combination of performance mitigation methods optimal for *your* context
 - Software Performance Engineering (SPE)
 - Single-user performance engineering
 - Performance Testing
 - Instrumentation / APM / Monitoring / Observability
 - Capacity Planning/Management
 - Continuous Integration / Deployment
- A holistic approach to performance
 - Get all these methods complement each other and work together

19

19

Integrating Everything: Tools, Processes, Roles

- Tools get integrated. No name / concept of the integrated tool yet.
 - ITSM (IT Service Management)
 - ITOA (IT Operations Analytics)
 - AIOps (Artificial Intelligence for / Algorithmic IT Operations)
 - MLOps (Machine Learning for IT Operations)
 - Digital Operations Management
 - Digital Experience Management
- Processes
 - Integration into CI/CD and DevOps
- Roles
 - Less need for silo performance roles

20

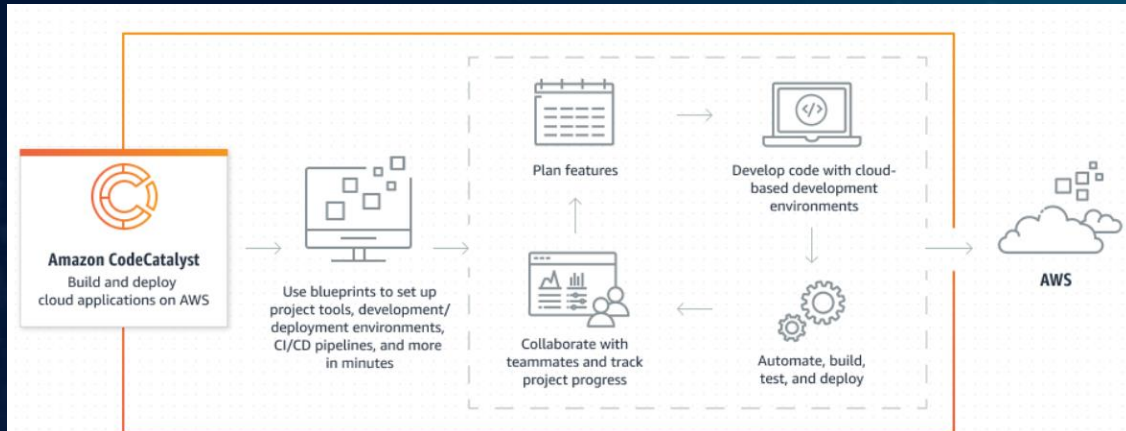
20

Unified Development Services



- Such as AWS CodeCatalyst (Preview)

<https://aws.amazon.com/codecatalyst/>



21

21

Chaos Engineering: Renaissance of Reliability

- Reincarnation of reliability testing
 - Which was practically ignored for a long time
 - Started in production
 - Slowly moving back to testing
 - Basically reliability testing tools
 - Not touched by load testing tool vendors as far as I know (except Network simulation)
- Developing into a separate discipline – **Resilience Engineering**

22

22

Why Do We Need Performance Testing to Be Continuous ?

23

23

My View / Use of Terminology

Performance Testing

Exploratory Testing, Profiling, etc.

Automation

Patch Testing, Optimization, etc.

**Continuous Performance
[Regression] Testing**

24

24

Continuous Performance Testing

- Integration into Agile and CI/CD
 - We just starting to see advances here
- All context-dependent
 - Don't wait for exact recipe, you need to figure out your own depending on your needs
- Change from realistic testing / SLO checking to coverage / difference between builds

25

25

Challenges of Continuous Performance Testing

- Integration
- Coverage Optimization
- Variability / Noise Reduction
- Change Detection
- Advanced Analysis
- Operations / Maintenance

<https://alexanderpodelko.com/docs/CMG%20IMPACT%202022%20-%20Continuous%20Performance%20Testing.pdf> and

https://alexanderpodelko.com/docs/LTB22_Review_Modern_Challenges_in_Performance_Testing.pdf

26

26

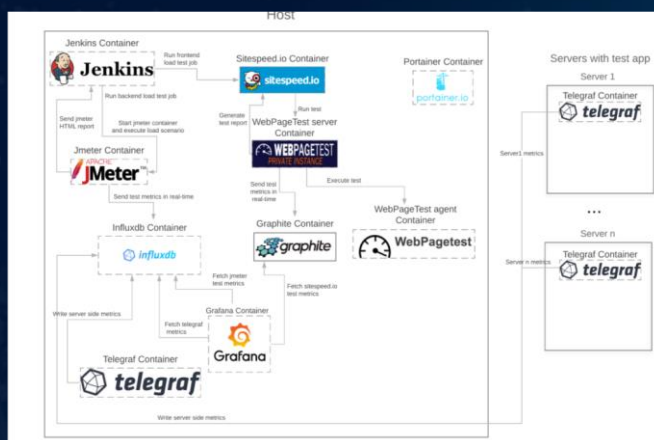
Continuous Integration: Load Testing Tools

- CI support in load testing tools
 - Integration with CI Servers (Jenkins, Hudson, etc.)
 - Automation support
- CI tools support for performance testing
 - [Jenkins Performance Plugin](#)
- Performance Testing Frameworks / Solutions
 - Combining multiple tools

27

27

Continuous Integration: Testing Frameworks

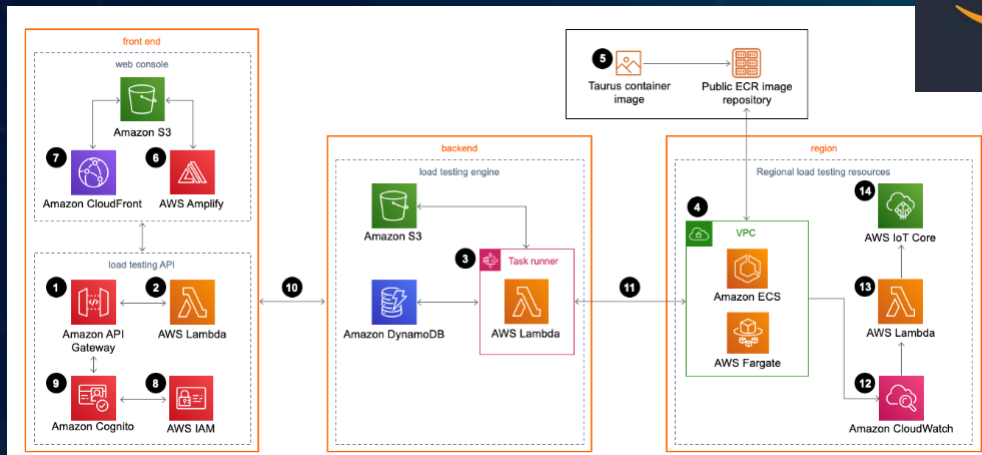


From <https://github.com/serputko/performance-testing-framework>

28

28

Distributed Load Testing on AWS



From AWS Solutions Library

<https://aws.amazon.com/solutions/implementations/distributed-load-testing-on-aws/>

29

29

Closely Integrated Systems

- [Creating a Virtuous Cycle in Performance Testing at MongoDB](#)
- [Fallout: Distributed Systems Testing as a Service \(DataStax\)](#)
- [Tracking Performance of the Graal Compiler on Public Benchmarks](#) (Charles University / Oracle Labs)
- [Introducing Ballast: An Adaptive Load Test Framework](#) (Uber)



30

30

Where Is It Going?

31

What is Ahead for Performance Testing ?

- 
- 
- Replaced by other methods of performance risk mitigation
 - Due to centralization
 - Moving responsibilities
 - Left – to developers
 - Right – to SRE (admins, operations)
 - Self-managed systems
 - Auto-scaling, etc.
 - Systems scale skyrocket
 - Systems sophistication skyrocket
 - Cost of resources becomes significant for large-scale systems
 - Cost of failures increases
 - Decentralization trends
 - Mobile, IoT, Fog, Edge

32

32

Skills

- All old good performance knowledge / skills
 - Not as much around load testing tools anymore
- Development / Scripting / Automation
- Performance understanding becoming a must in the industry
 - Need to go one level deeper

33

Algorithmic Complexity

- Time Complexity
- Space Complexity
- Big-O notation

*Almost in every
interview around the
globe !*

34

34

The 6 Pillars of the AWS Well-Architected Framework



<https://aws.amazon.com/blogs/apn/the-6-pillars-of-the-aws-well-architected-framework/>

- Operational Excellence
- Security
- **Reliability**
- **Performance Efficiency**
- **Cost Optimization**
- Sustainability

35

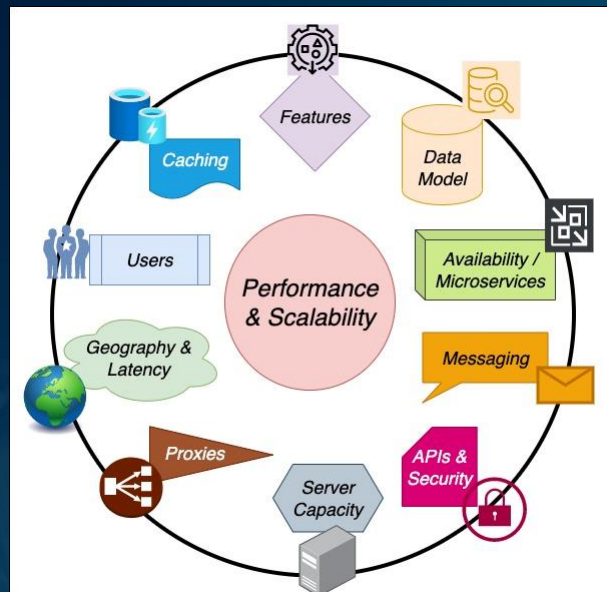
35

System Design Interview Cheat Sheet by Vahid Dejwakh

From:

<https://vahid.blog/post/2022-05-05-system-design-interview-cheat-sheet/>

Just *an example* of changing attitude



36

36

Summary

- Systems scale and sophistication skyrocket – performance gets more attention
 - Performance engineering gets more integrated
- There are many way to mitigate performance risks
 - Define a strategy based on your context
- Performance engineering must adjust to industry trends
 - Some trends are clear: continuous testing, automation, integration, performance as code, etc.
 - We are rather in the beginning; the future of the trade is not set yet...

37

37

Questions ?

apodelko@yahoo.com

[@apodelko](#)

<https://alexanderpodelko.com/blog/>

<https://www.linkedin.com/in/alexanderpodelko/>

38

38