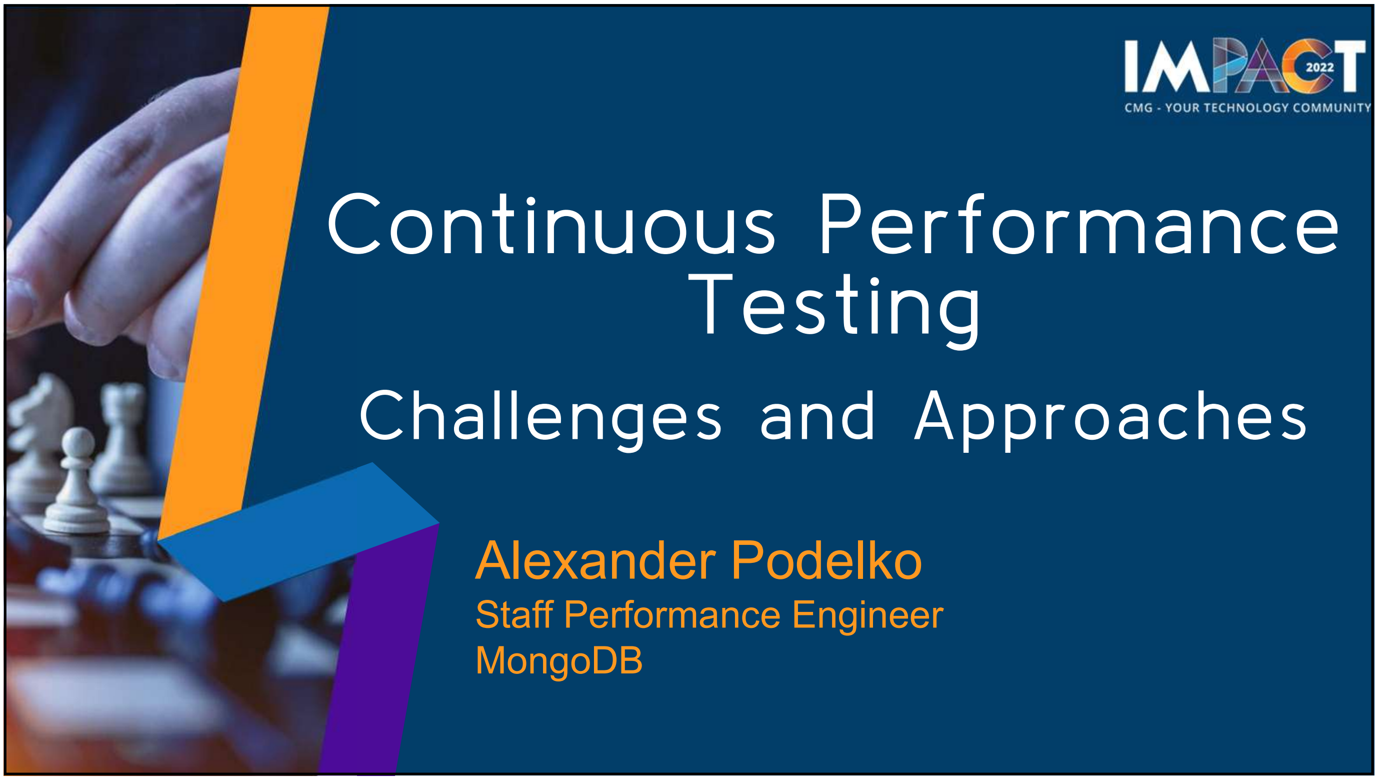


IMPACT 2022
CMG - YOUR TECHNOLOGY COMMUNITY

Continuous Performance Testing

Challenges and Approaches

Alexander Podelko
Staff Performance Engineer
MongoDB

1

ABOUT THE SPEAKER

Alex Podelko

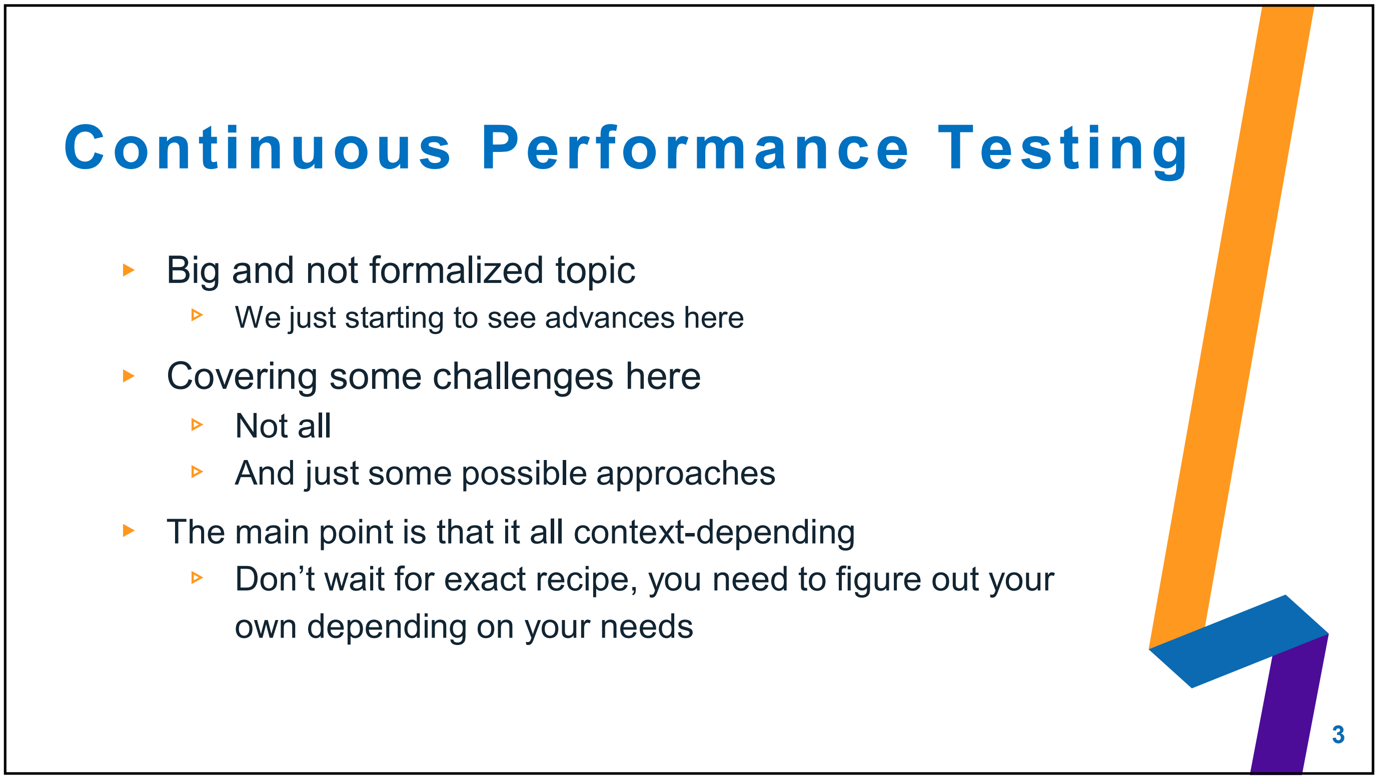
- Specializing in performance since 1997
- Staff Performance Engineer at MongoDB
  - Before worked for Oracle/Hyperion, Intel, and Aetna
- Board director at CMG

apodelko@yahoo.com
@apodelko
https://alexanderpodelko.com/blog/

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.

2

2

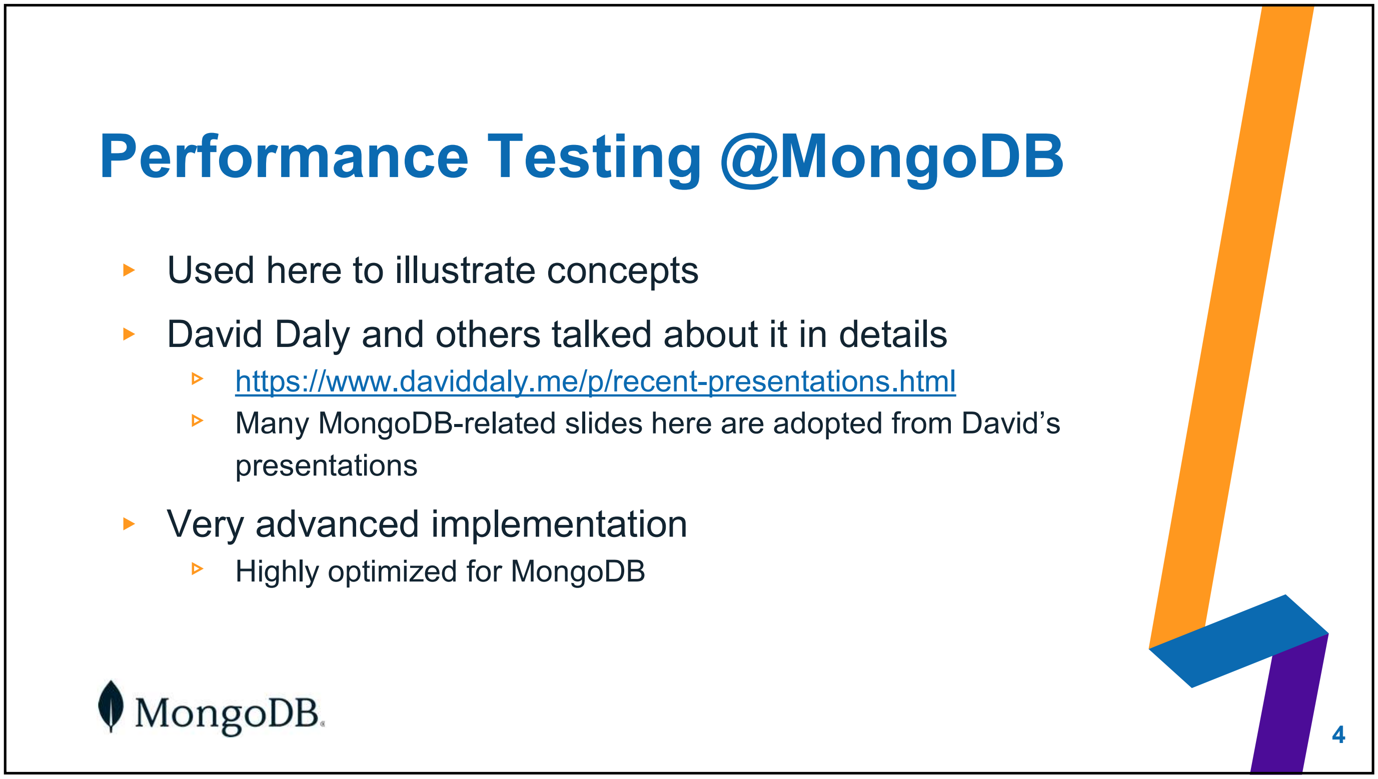


Continuous Performance Testing

- Big and not formalized topic
  - We just starting to see advances here
- Covering some challenges here
  - Not all
  - And just some possible approaches
- The main point is that it all context-depending
  - Don’t wait for exact recipe, you need to figure out your own depending on your needs

3

3



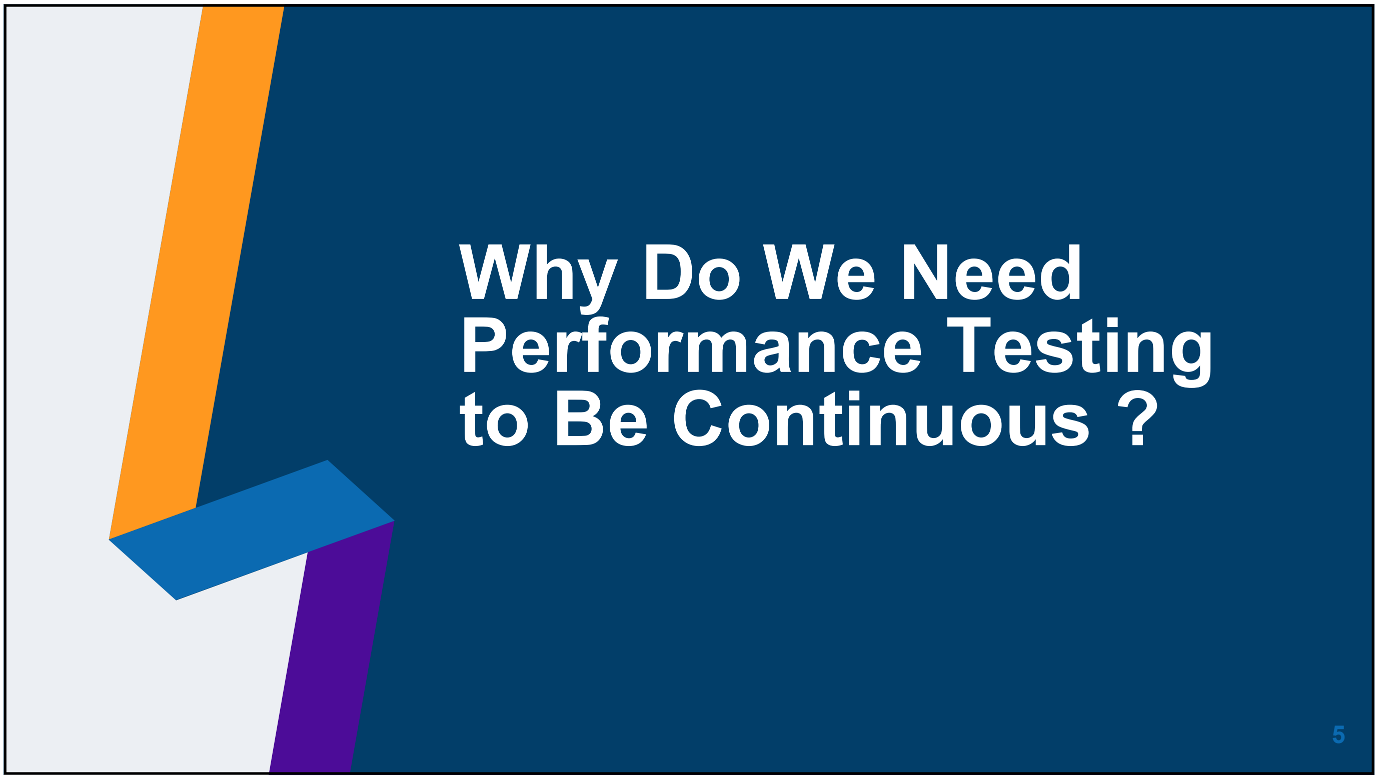
Performance Testing @MongoDB

- Used here to illustrate concepts
- David Daly and others talked about it in details
  - https://www.daviddaly.me/p/recent-presentations.html
  - Many MongoDB-related slides here are adopted from David’s presentations
- Very advanced implementation
  - Highly optimized for MongoDB

MongoDB.

4

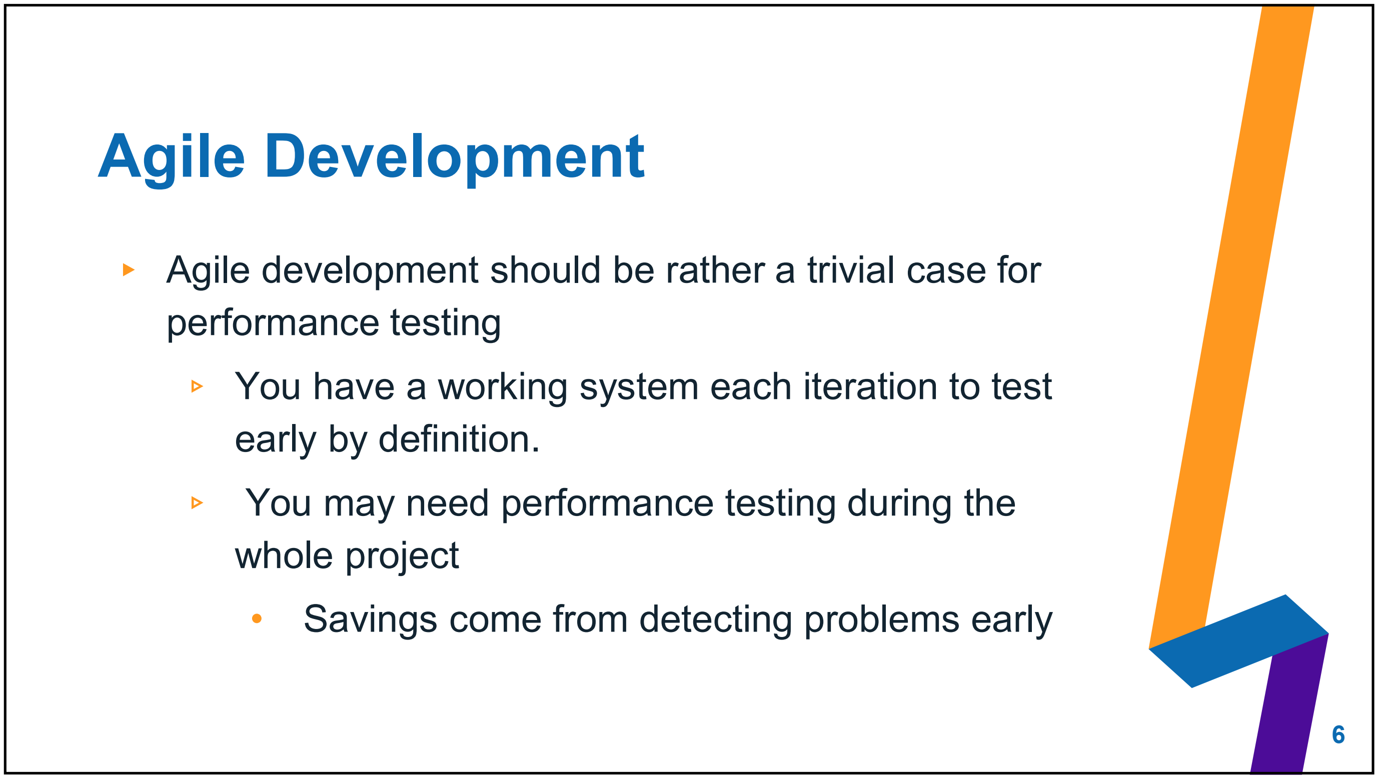
4



Why Do We Need Performance Testing to Be Continuous ?

5

5



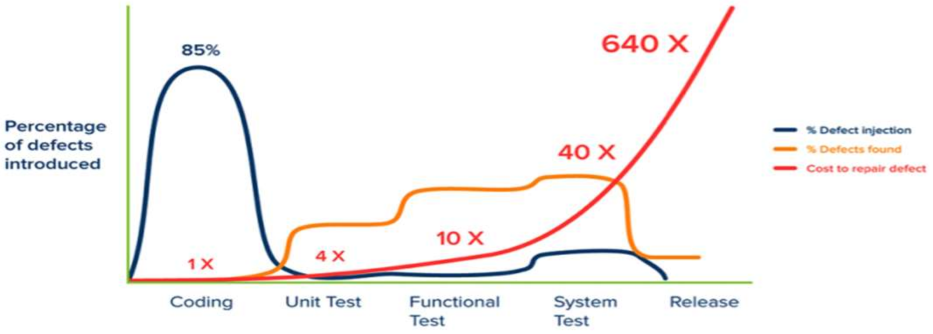
Agile Development

- Agile development should be rather a trivial case for performance testing
  - You have a working system each iteration to test early by definition.
  - You may need performance testing during the whole project
    - Savings come from detecting problems early

6

6

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



Jones, Capers. *Applied Software Measurement: Global Analysis of Productivity and Quality.*

7

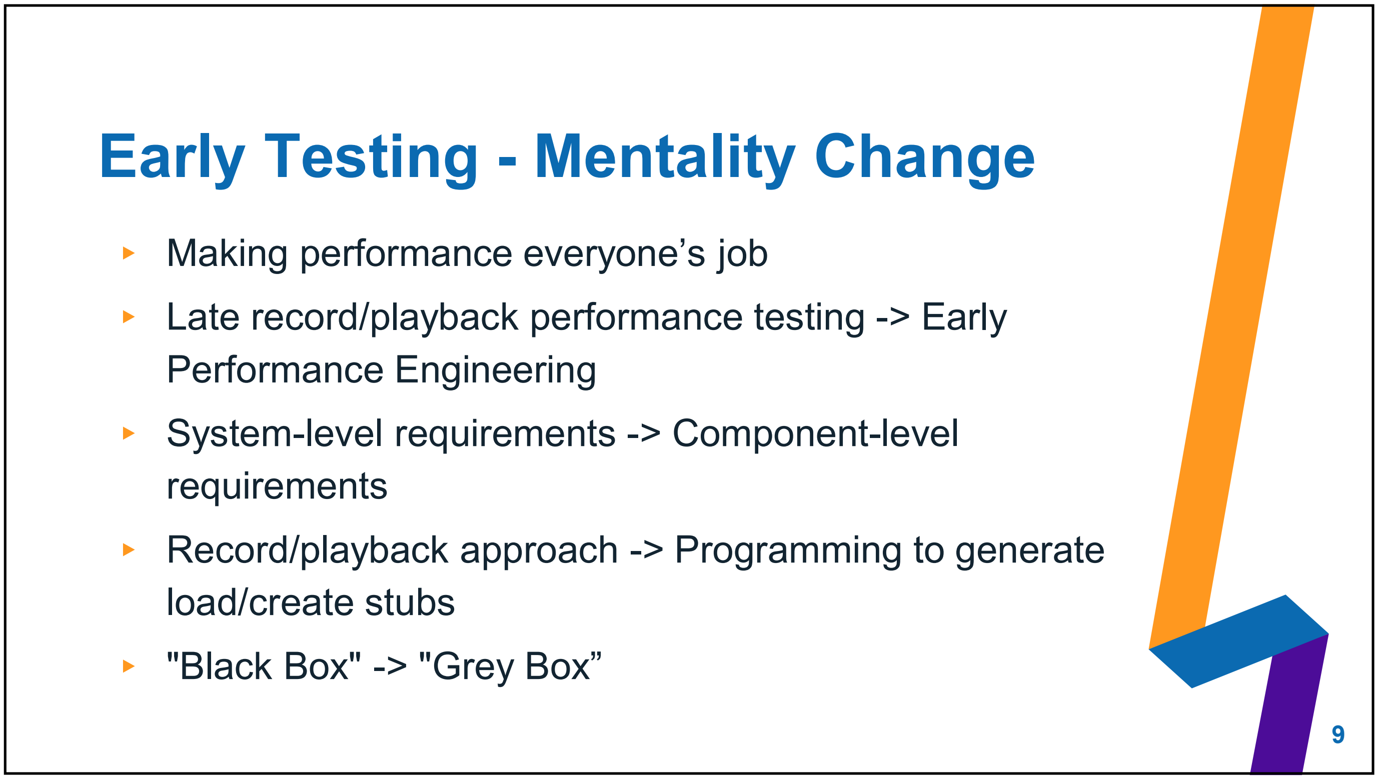
7

Paradigm Change

- Traditional performance approach don’t scale well
  - Even having competent and effective engineers
- Increased volume exposes the problem
  - Early testing
  - Each iteration
- Remedies: **automation, making performance everyone’s job**

8

8

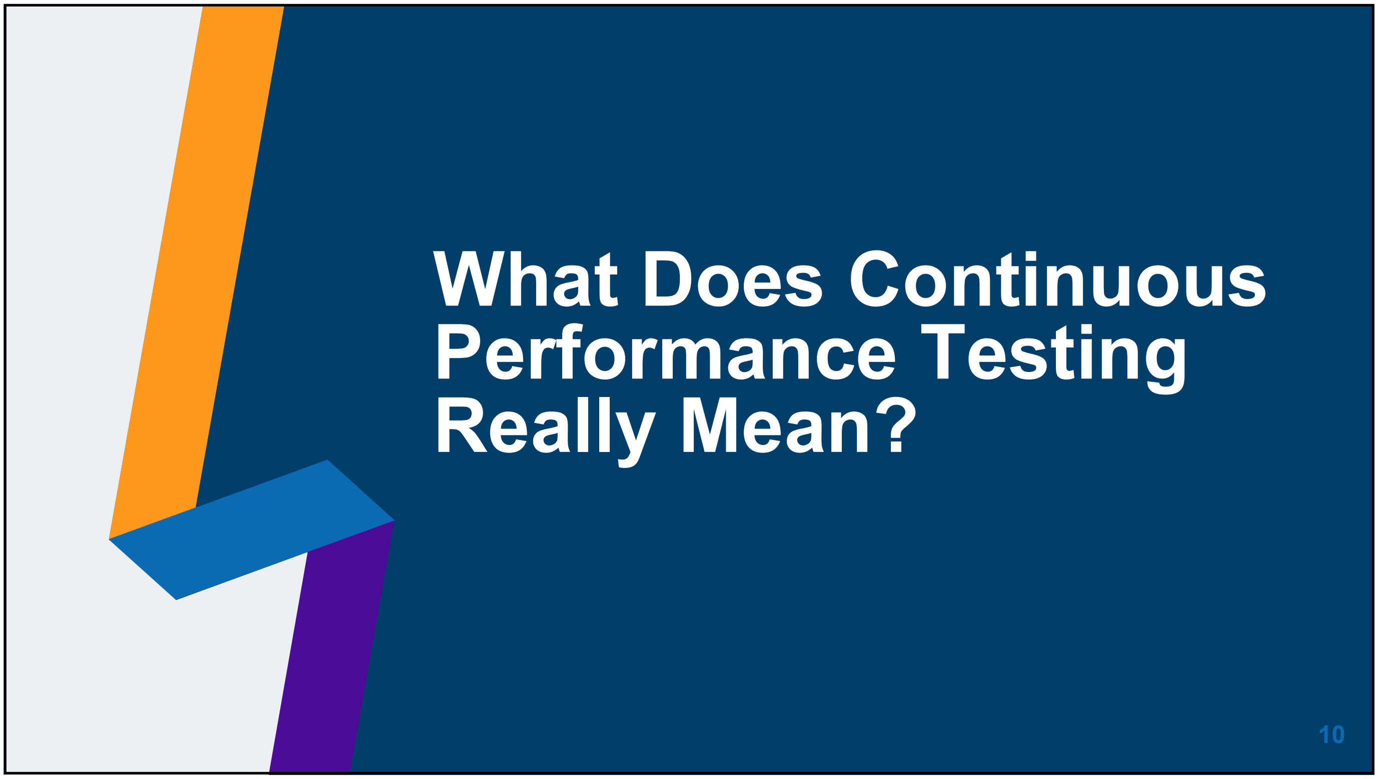


Early Testing - Mentality Change

- 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”

9

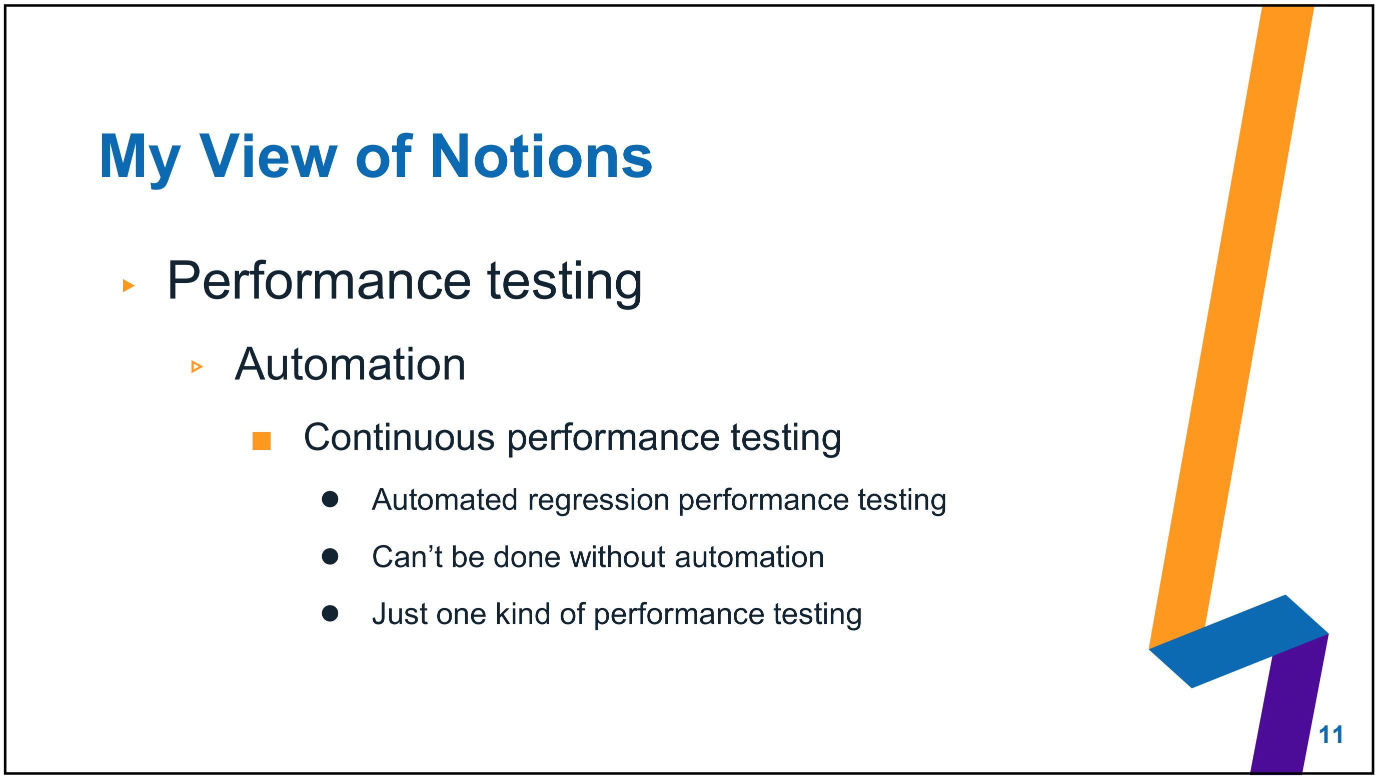
9



What Does Continuous Performance Testing Really Mean?

10

10

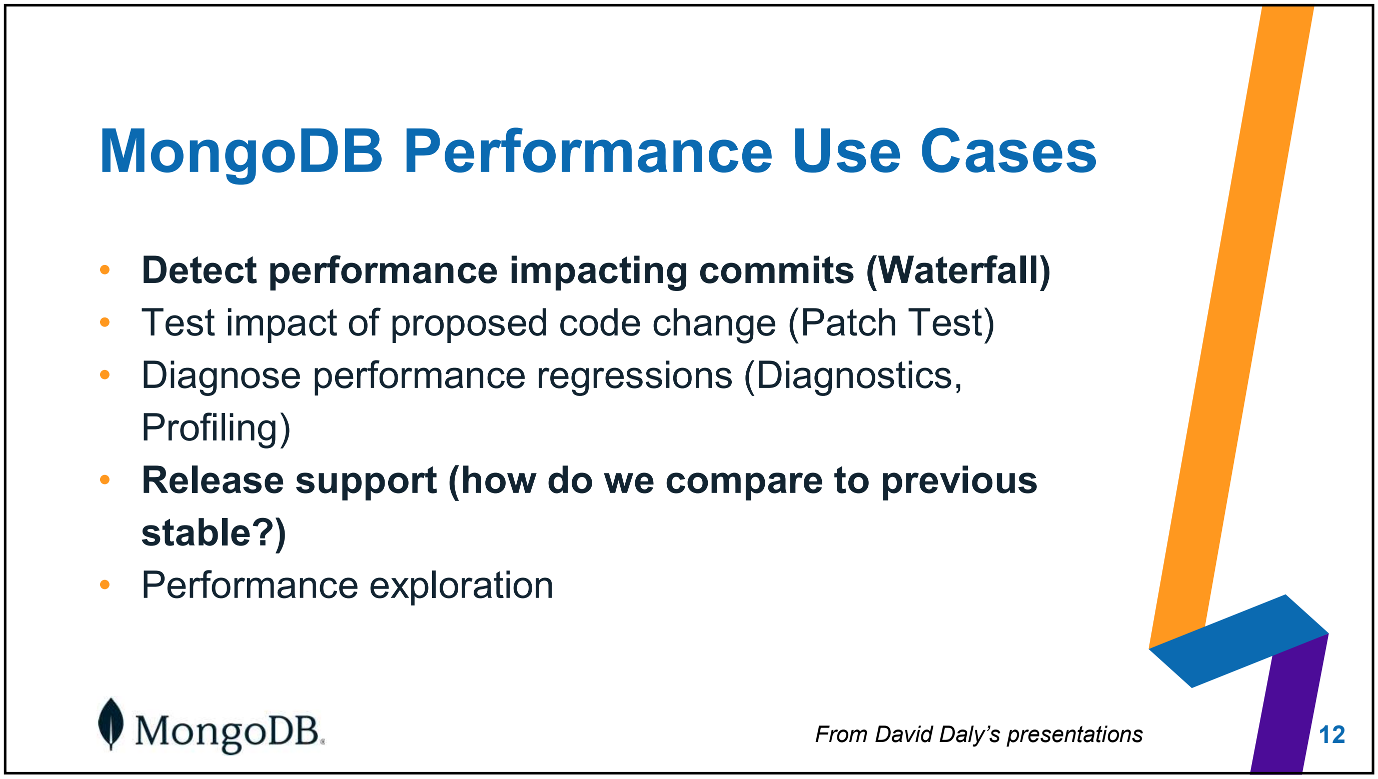


My View of Notions

- Performance testing
  - Automation
    - Continuous performance testing
      - Automated regression performance testing
      - Can’t be done without automation
      - Just one kind of performance testing

11

11



MongoDB Performance Use Cases

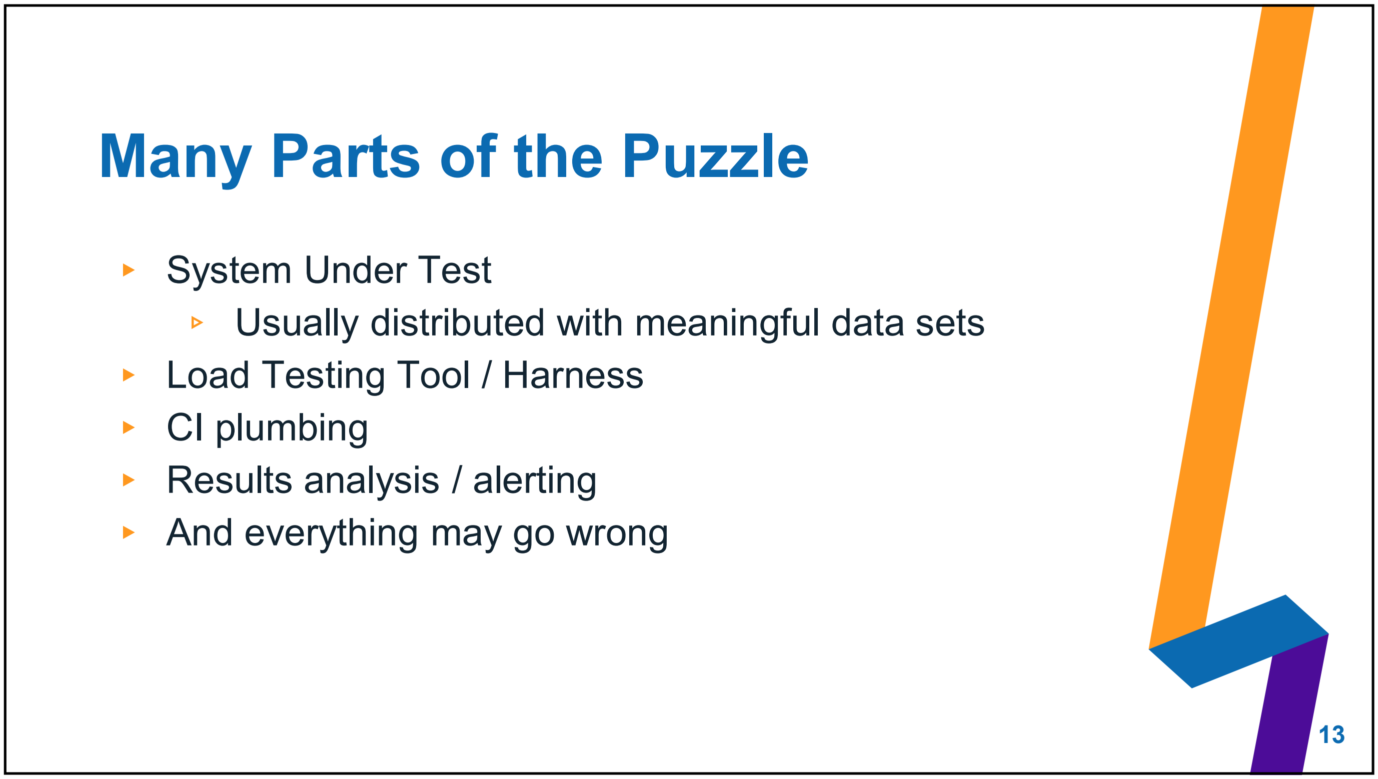
- **Detect performance impacting commits (Waterfall)**
- Test impact of proposed code change (Patch Test)
- Diagnose performance regressions (Diagnostics, Profiling)
- **Release support (how do we compare to previous stable?)**
- Performance exploration

MongoDB.

From David Daly’s presentations

12

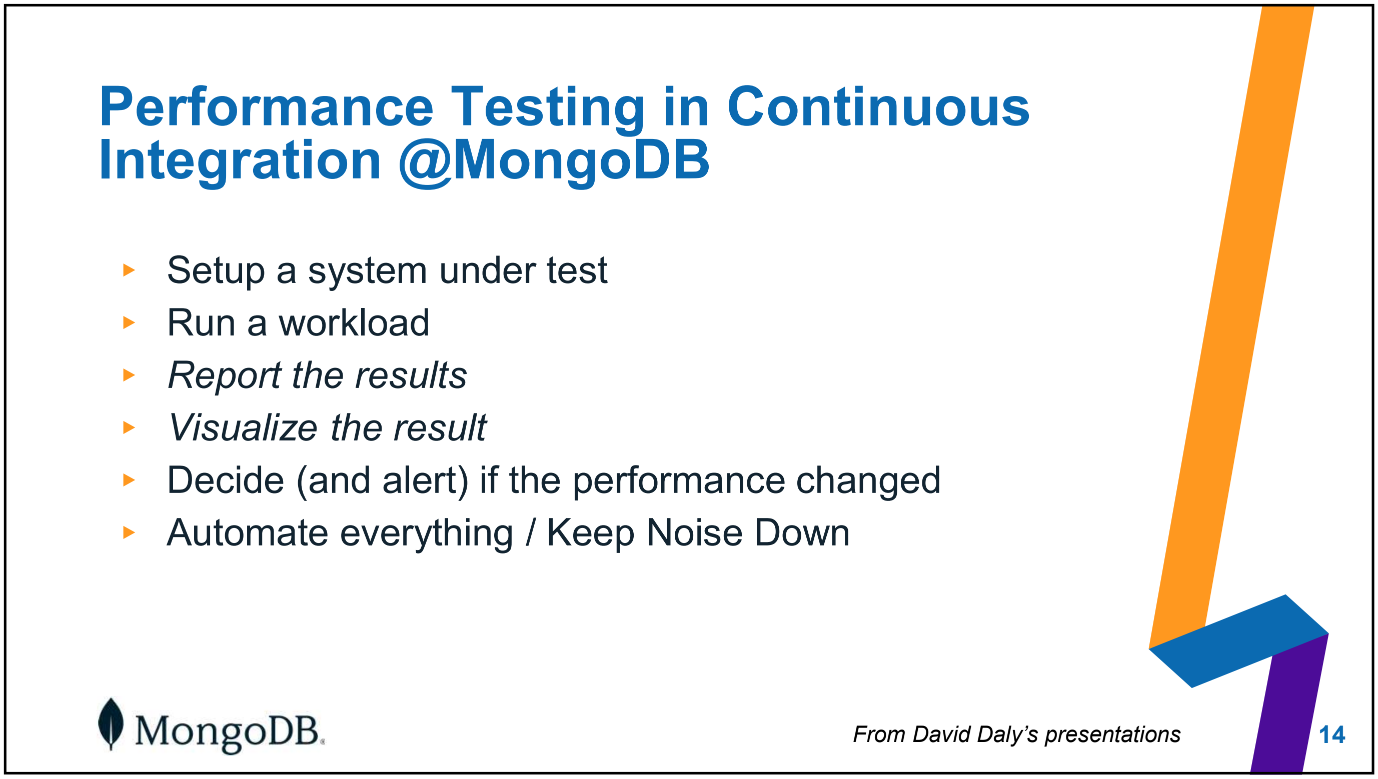
12



Many Parts of the Puzzle

- System Under Test
  - Usually distributed with meaningful data sets
- Load Testing Tool / Harness
- CI plumbing
- Results analysis / alerting
- And everything may go wrong

13



Performance Testing in Continuous Integration @MongoDB

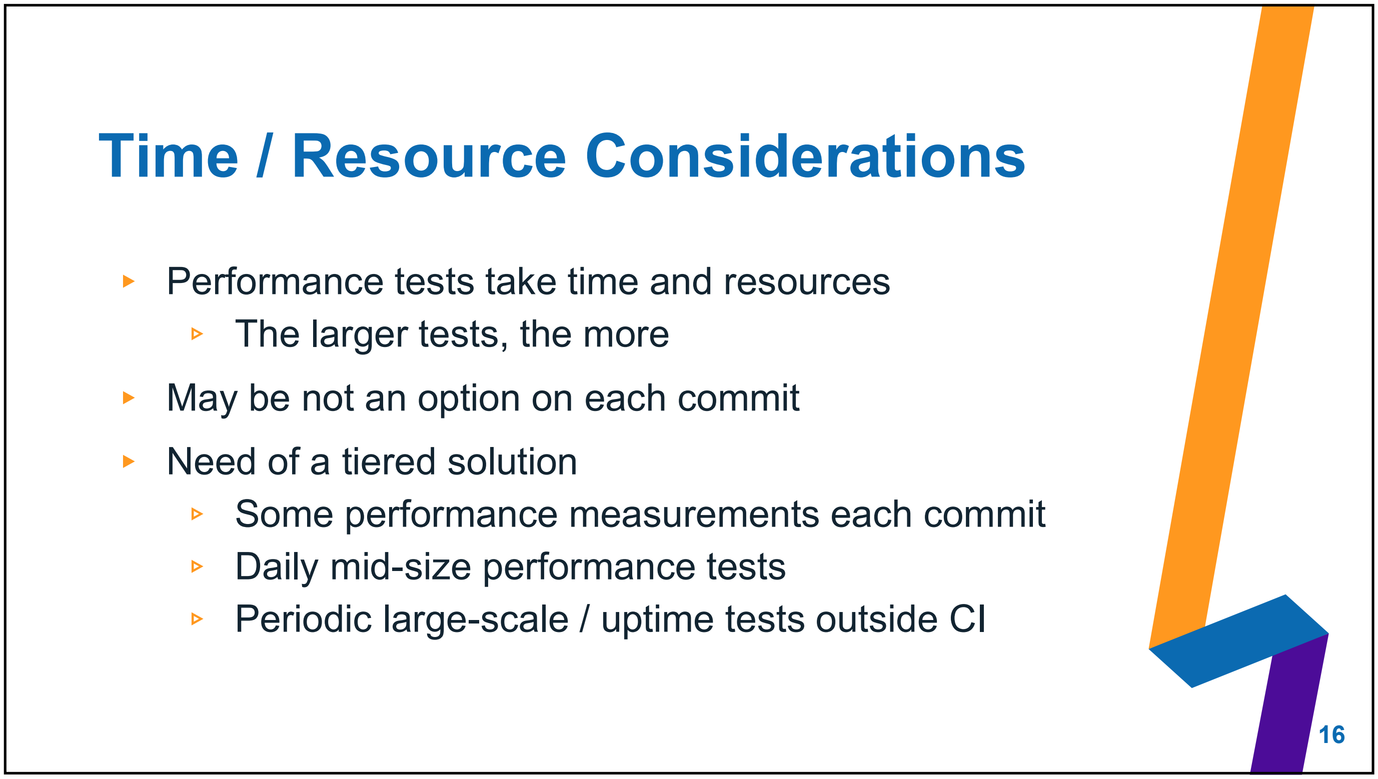
- Setup a system under test
- Run a workload
- *Report the results*
- *Visualize the result*
- Decide (and alert) if the performance changed
- Automate everything / Keep Noise Down

MongoDB.

From David Daly's presentations

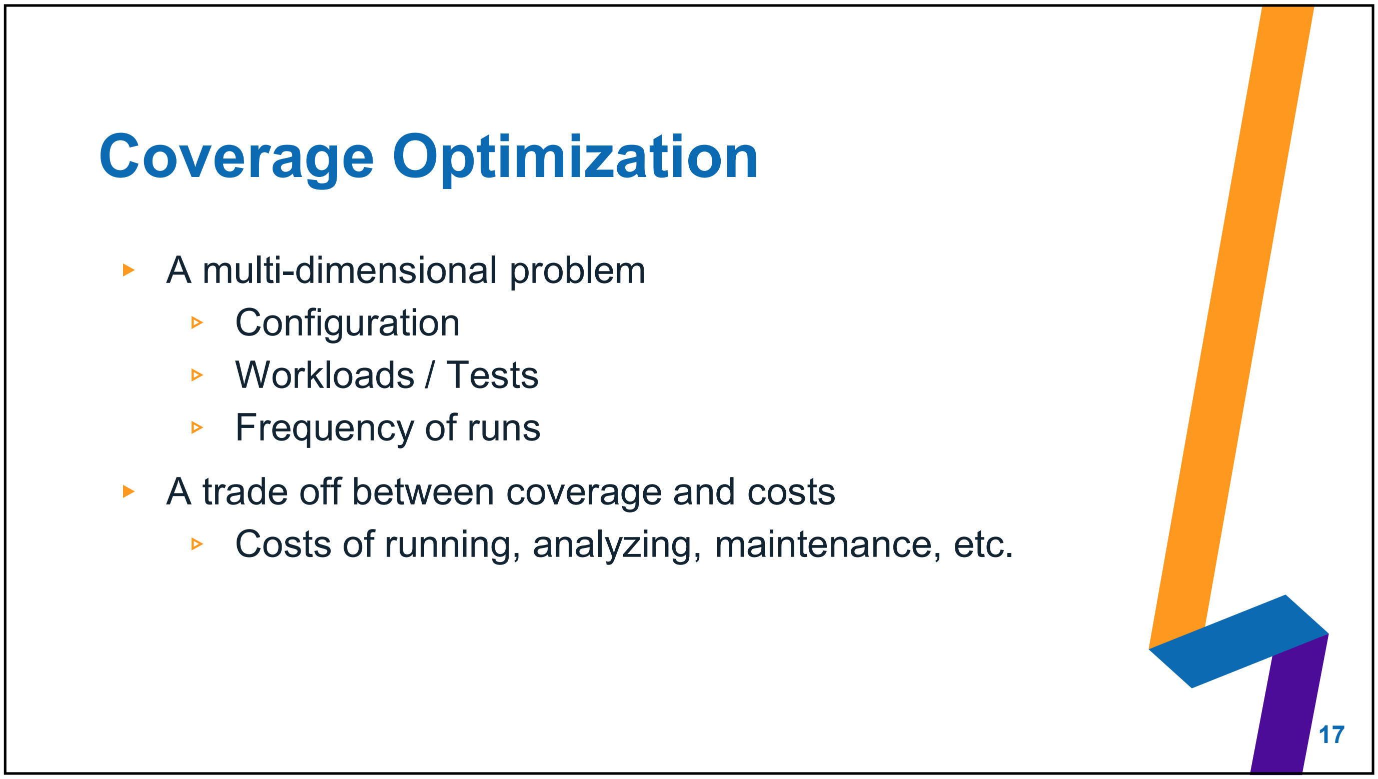
14

The Challenge of Coverage Optimization



Time / Resource Considerations

- Performance tests take time and resources
  - The larger tests, the more
- May be not an option on each commit
- Need of a tiered solution
  - Some performance measurements each commit
  - Daily mid-size performance tests
  - Periodic large-scale / uptime tests outside CI

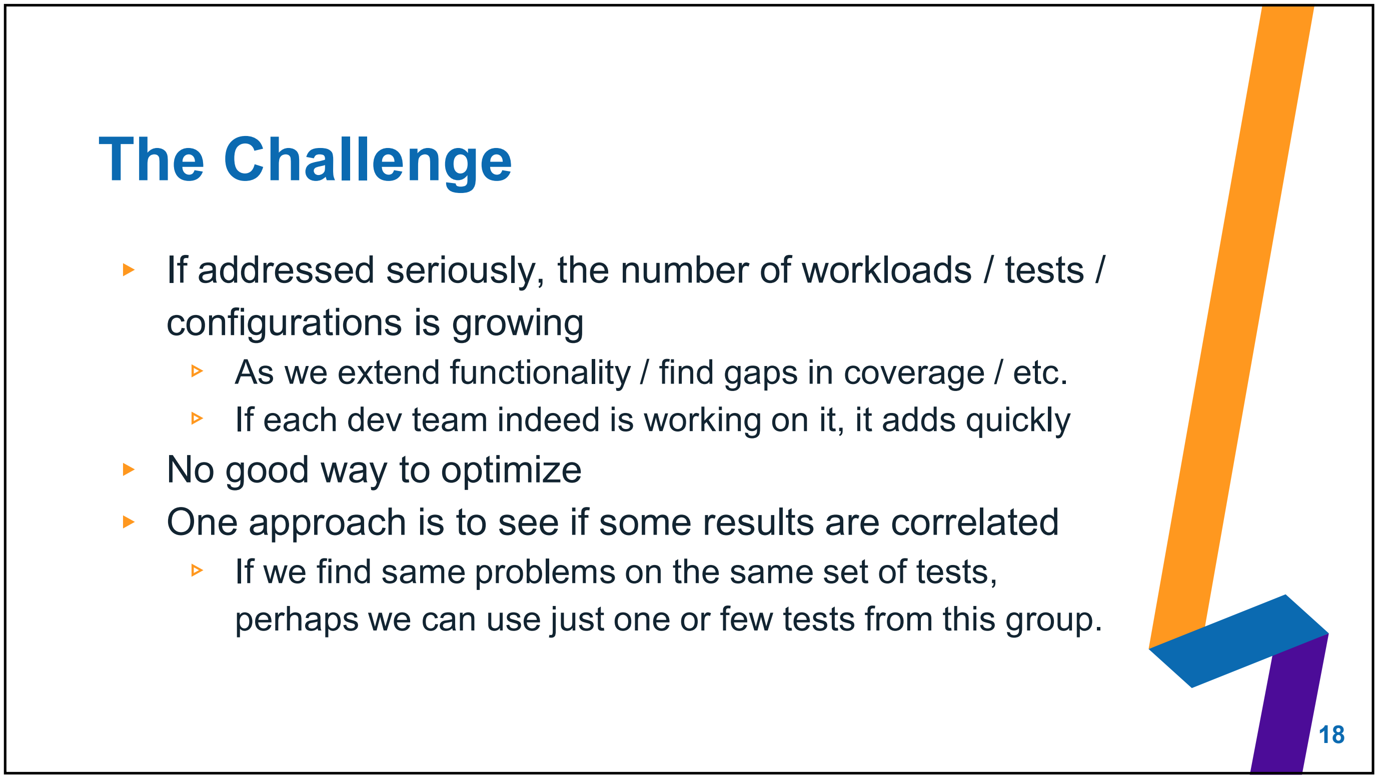


Coverage Optimization

- A multi-dimensional problem
  - Configuration
  - Workloads / Tests
  - Frequency of runs
- A trade off between coverage and costs
  - Costs of running, analyzing, maintenance, etc.

17

17



The Challenge

- If addressed seriously, the number of workloads / tests / configurations is growing
  - As we extend functionality / find gaps in coverage / etc.
  - If each dev team indeed is working on it, it adds quickly
- No good way to optimize
- One approach is to see if some results are correlated
  - If we find same problems on the same set of tests, perhaps we can use just one or few tests from this group.

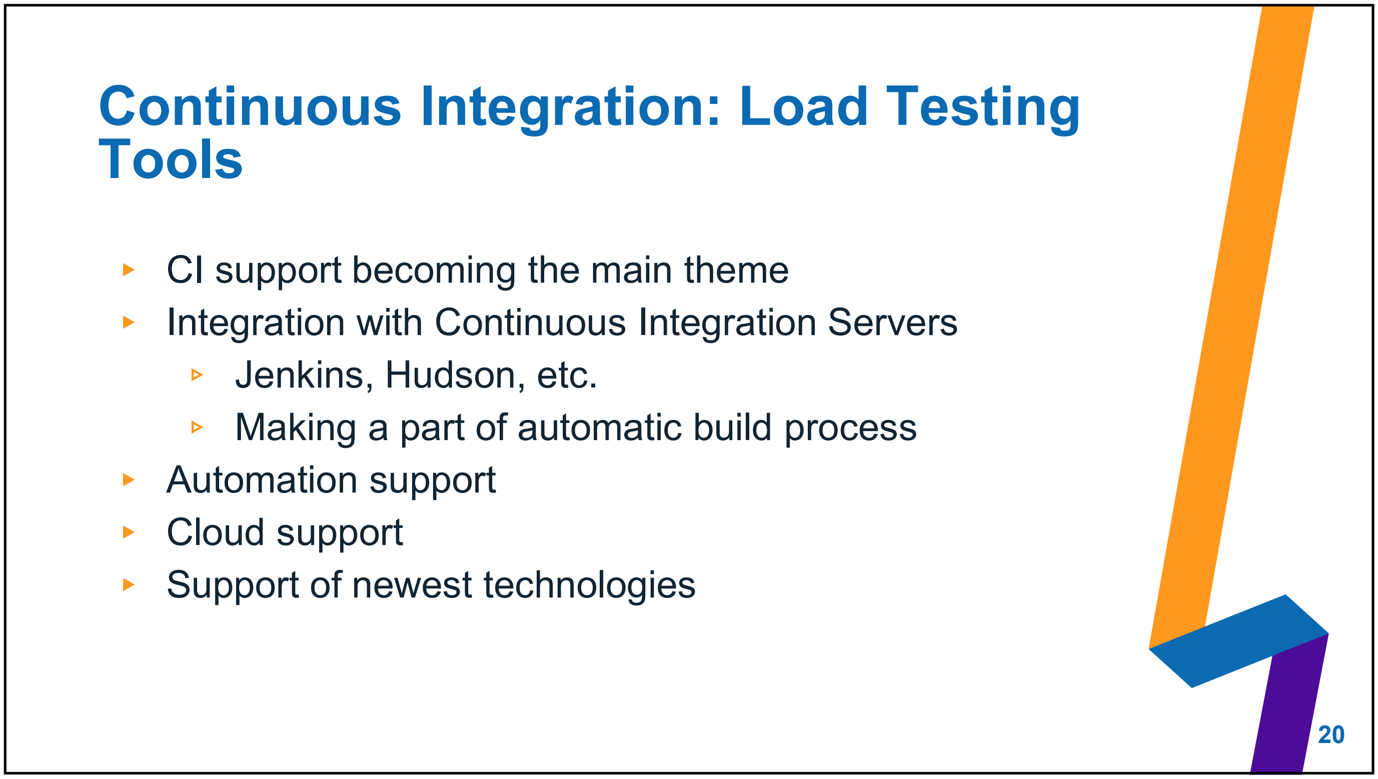
18

18

The Challenge of Integration

19

19



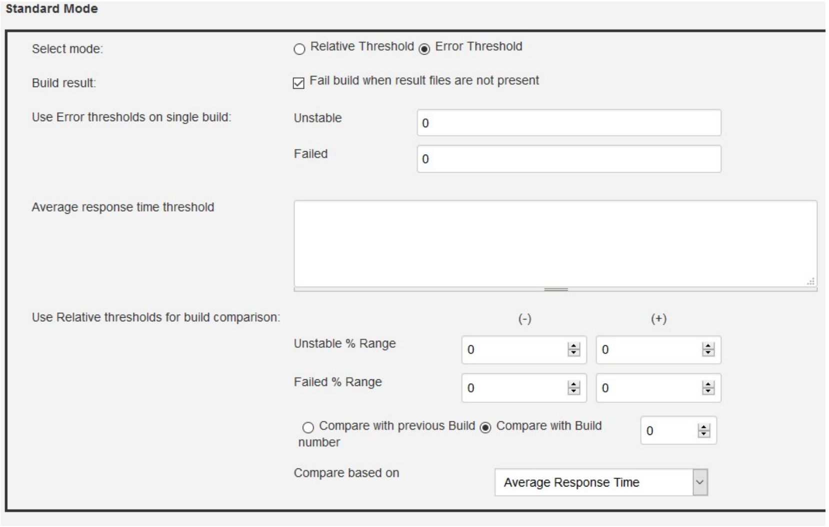
Continuous Integration: Load Testing Tools

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

20

20

Jenkins Performance Plugin



Standard Mode

Select mode: ○ Relative Threshold ● Error Threshold

Build result: ☑ Fail build when result files are not present

Use Error thresholds on single build:

Unstable	0
Failed	0

Average response time threshold

Use Relative thresholds for build comparison:

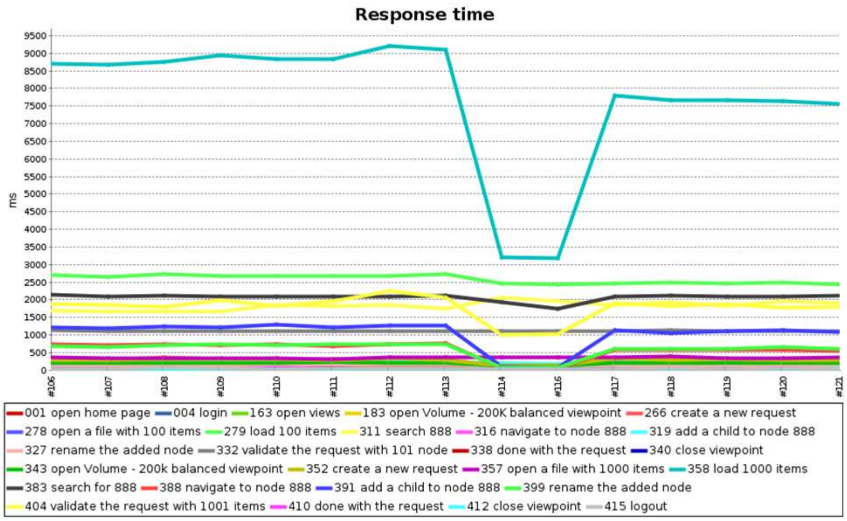
	(-)	(+)
Unstable % Range	0	0
Failed % Range	0	0

○ Compare with previous Build ● Compare with Build number 0

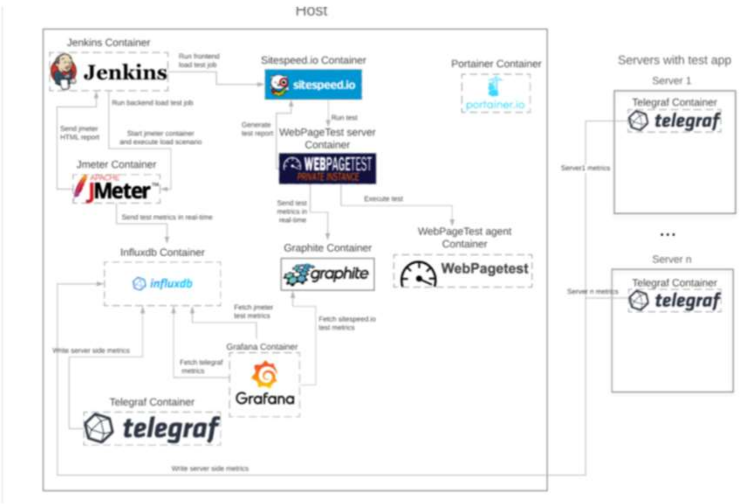
Compare based on: Average Response Time

https://plugins.jenkins.io/performance/

Performance Plugin: Visualization

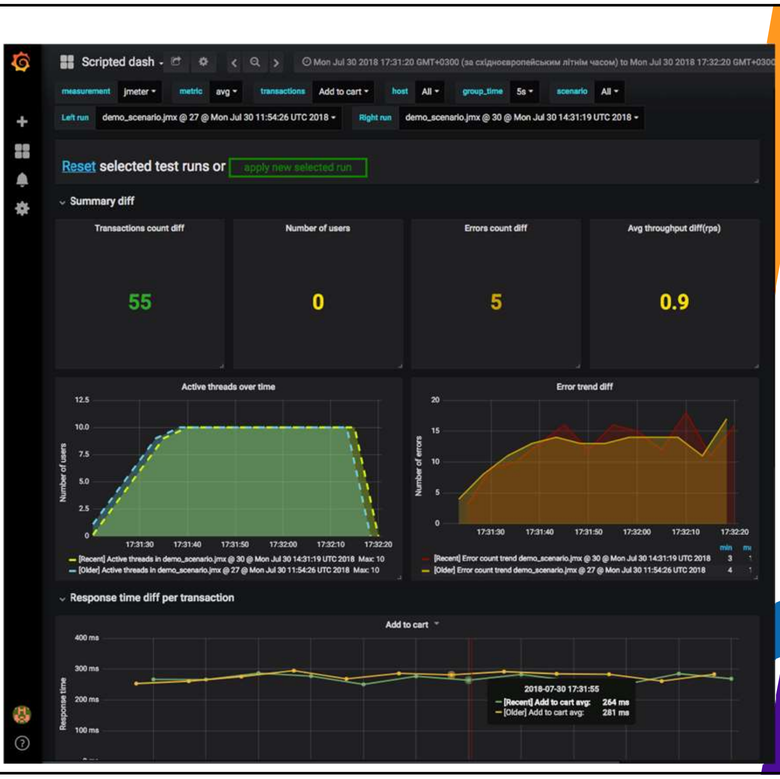


A Performance Testing Framework



https://github.com/serputko/performance-testing-framework

Performance Testing Framework: Visualization


Scripted dash

Mon Jul 30 2018 17:31:20 GMT+0300 (за східноєвропейським літнім часом) to Mon Jul 30 2018 17:32:20 GMT+0300

measurement jmeter · metric avg · transactions Add to cart · host All · group_time 5s · scenario All

Left run demo_scenario.jmx @ 27 @ Mon Jul 30 11:54:26 UTC 2018

Right run demo_scenario.jmx @ 30 @ Mon Jul 30 14:31:19 UTC 2018

Reset selected test runs or apply new selected run

Summary diff

Transactions count diff	Number of users	Errors count diff	Avg throughput diff(rps)
55	0	5	0.9

Active threads over time

- [Recent] Active threads in demo_scenario.jmx @ 30 @ Mon Jul 30 14:31:19 UTC 2018 Max: 10
- [Older] Active threads in demo_scenario.jmx @ 27 @ Mon Jul 30 11:54:26 UTC 2018 Max: 10

Error trend diff

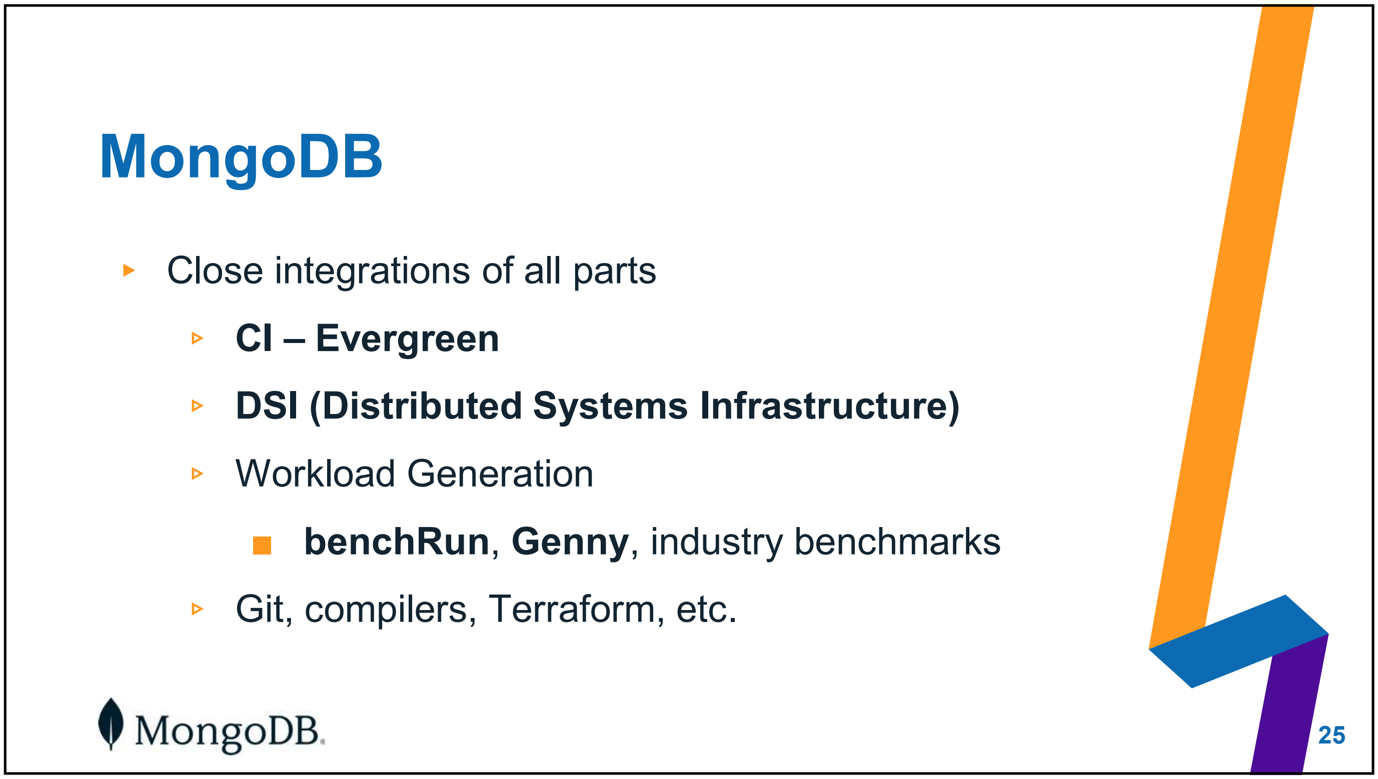
	min
[Recent] Error count trend demo_scenario.jmx @ 30 @ Mon Jul 30 14:31:19 UTC 2018	3
[Older] Error count trend demo_scenario.jmx @ 27 @ Mon Jul 30 11:54:26 UTC 2018	4

Response time diff per transaction

Add to cart

2018-07-30 17:31:55
- [Recent] Add to cart avg: 264 ms
- [Older] Add to cart avg: 281 ms

https://github.com/serputko/performance-testing-framework



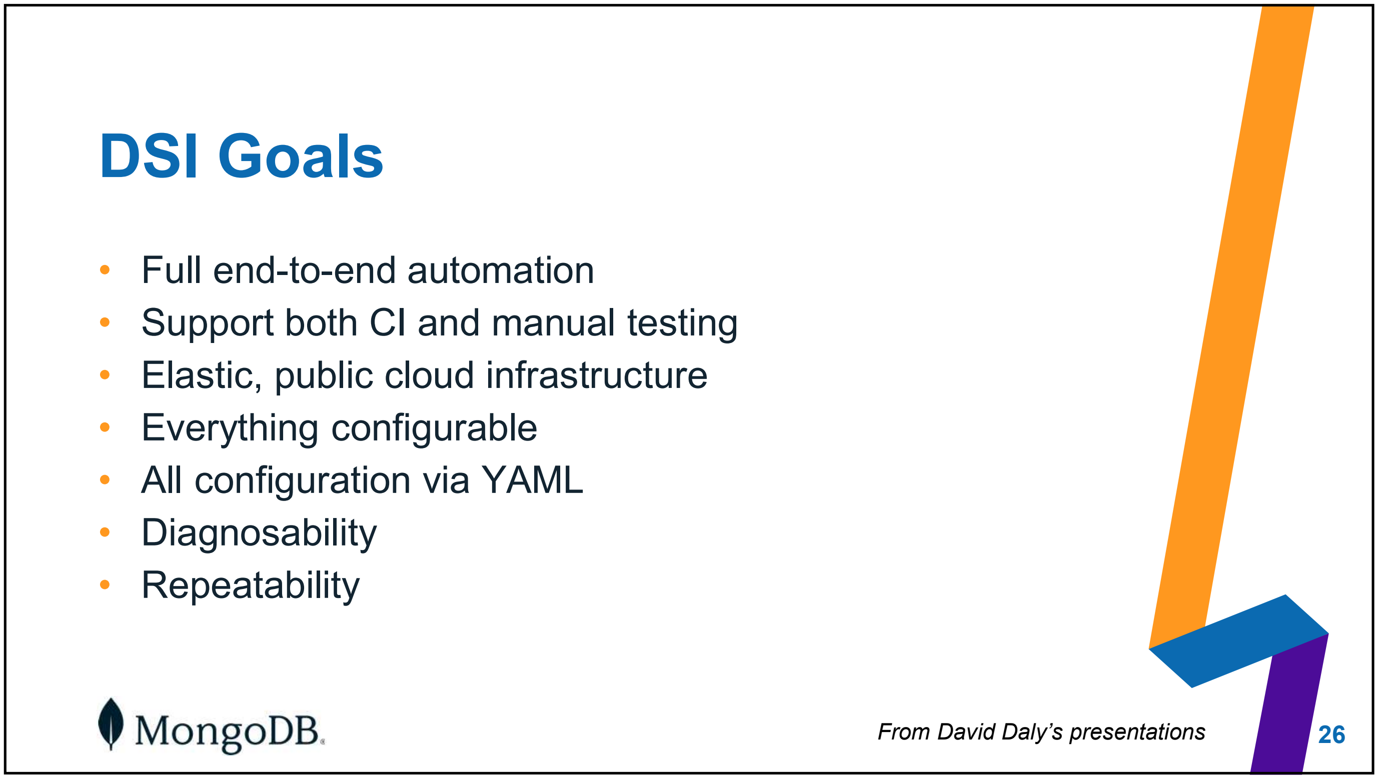
MongoDB

- Close integrations of all parts
  - **CI – Evergreen**
  - **DSI (Distributed Systems Infrastructure)**
  - Workload Generation
    - **benchRun**, **Genny**, industry benchmarks
  - Git, compilers, Terraform, etc.

MongoDB

25

25



DSI Goals

- Full end-to-end automation
- Support both CI and manual testing
- Elastic, public cloud infrastructure
- Everything configurable
- All configuration via YAML
- Diagnosability
- Repeatability

MongoDB

From David Daly's presentations

26

26

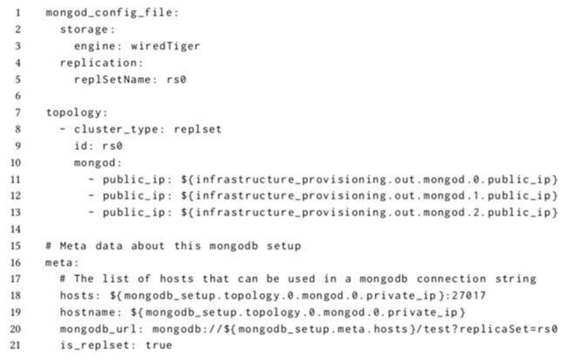
DSI Modules

- Bootstrap
- Infrastructure provisioning
- System setup
- Workload setup
- MongoDB setup
- Test Control
- Analysis
- Infrastructure teardown

MongoDB

From David Daly's presentations

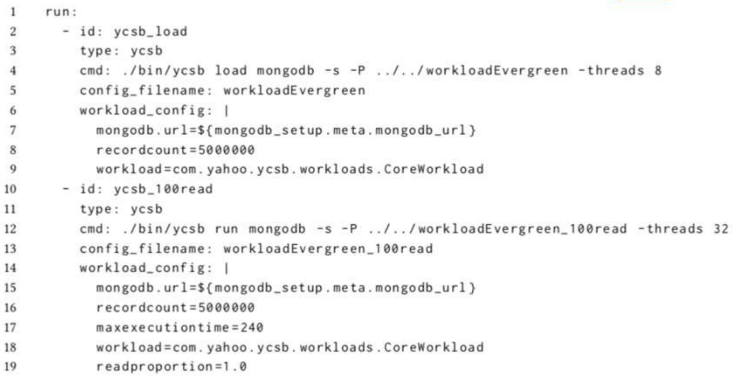
Configuration Files



```
mongod_config_file:
  storage:
    engine: wiredTiger
  replication:
    replSetName: rs0

topology:
  - cluster_type: replset
    id: rs0
    mongod:
      - public_ip: ${infrastructure_provisioning.out.mongod.0.public_ip}
      - public_ip: ${infrastructure_provisioning.out.mongod.1.public_ip}
      - public_ip: ${infrastructure_provisioning.out.mongod.2.public_ip}

# Meta data about this mongodb setup
meta:
  # The list of hosts that can be used in a mongodb connection string
  hosts: ${mongodb_setup.topology.0.mongod.0.private_ip}:27017
  hostname: ${mongodb_setup.topology.0.mongod.0.private_ip}
  mongodb_url: mongodb://${mongodb_setup.meta.hosts}/test?replicaSet=rs0
  is_replset: true
```

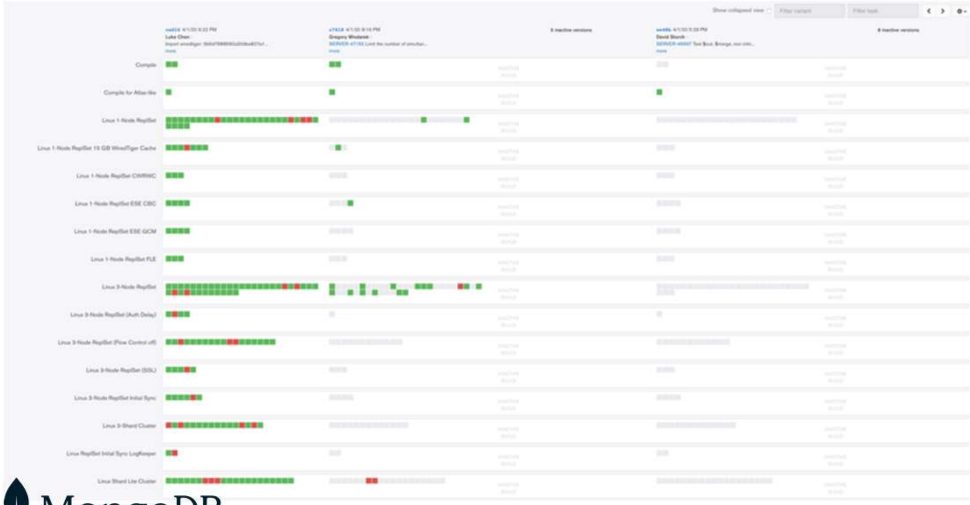


```
run:
  - id: ycsb_load
    type: ycsb
    cmd: ./bin/ycsb load mongodb -s -P ../../workloadEvergreen -threads 8
    config_filename: workloadEvergreen
    workload_config: |
      mongodb.url=${mongodb_setup.meta.mongodb_url}
      recordcount=5000000
      workload=com.yahoo.ycsb.workloads.CoreWorkload
  - id: ycsb_100read
    type: ycsb
    cmd: ./bin/ycsb run mongodb -s -P ../../workloadEvergreen_100read -threads 32
    config_filename: workloadEvergreen_100read
    workload_config: |
      mongodb.url=${mongodb_setup.meta.mongodb_url}
      recordcount=5000000
      maxexecutiontime=240
      workload=com.yahoo.ycsb.workloads.CoreWorkload
      readproportion=1.0
```

MongoDB

From David Daly's presentations

Evergreen Integration


Show collapsed view
Filter variant
Filter task
Luke Chen
Gregory Wlodarek
3 inactive versions
David Storch
8 inactive versions
Compile
Compile for Atlas-like
Linux 1-Node ReplSet
Linux 1-Node ReplSet 15 GB WiredTiger Cache
Linux 1-Node ReplSet CWRWC
Linux 1-Node ReplSet ESE CBC
Linux 1-Node ReplSet ESE GCM
Linux 1-Node ReplSet FLE
Linux 3-Node ReplSet
Linux 3-Node ReplSet (Auth Delay)
Linux 3-Node ReplSet (Flow Control off)
Linux 3-Node ReplSet (SSL)
Linux 3-Node ReplSet Initial Sync
Linux 3-Shard Cluster
Linux ReplSet Initial Sync LogKeeper
Linux Shard Lite Cluster

MongoDB.

29

29

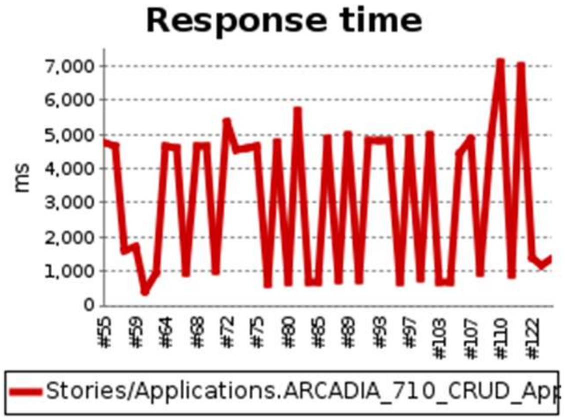
The Challenge of Variability

30

30

Variability - Environment

- Due to difference in environments

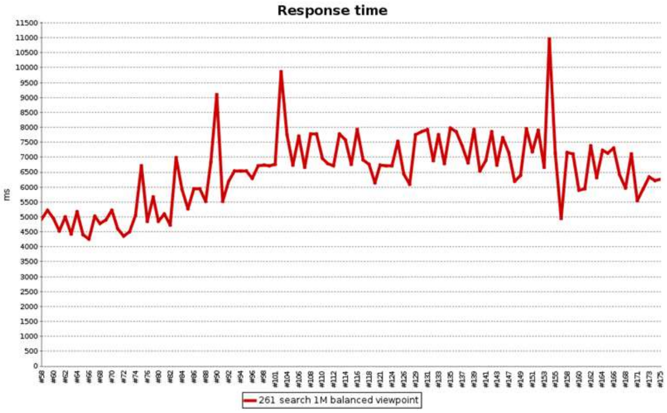


31

31

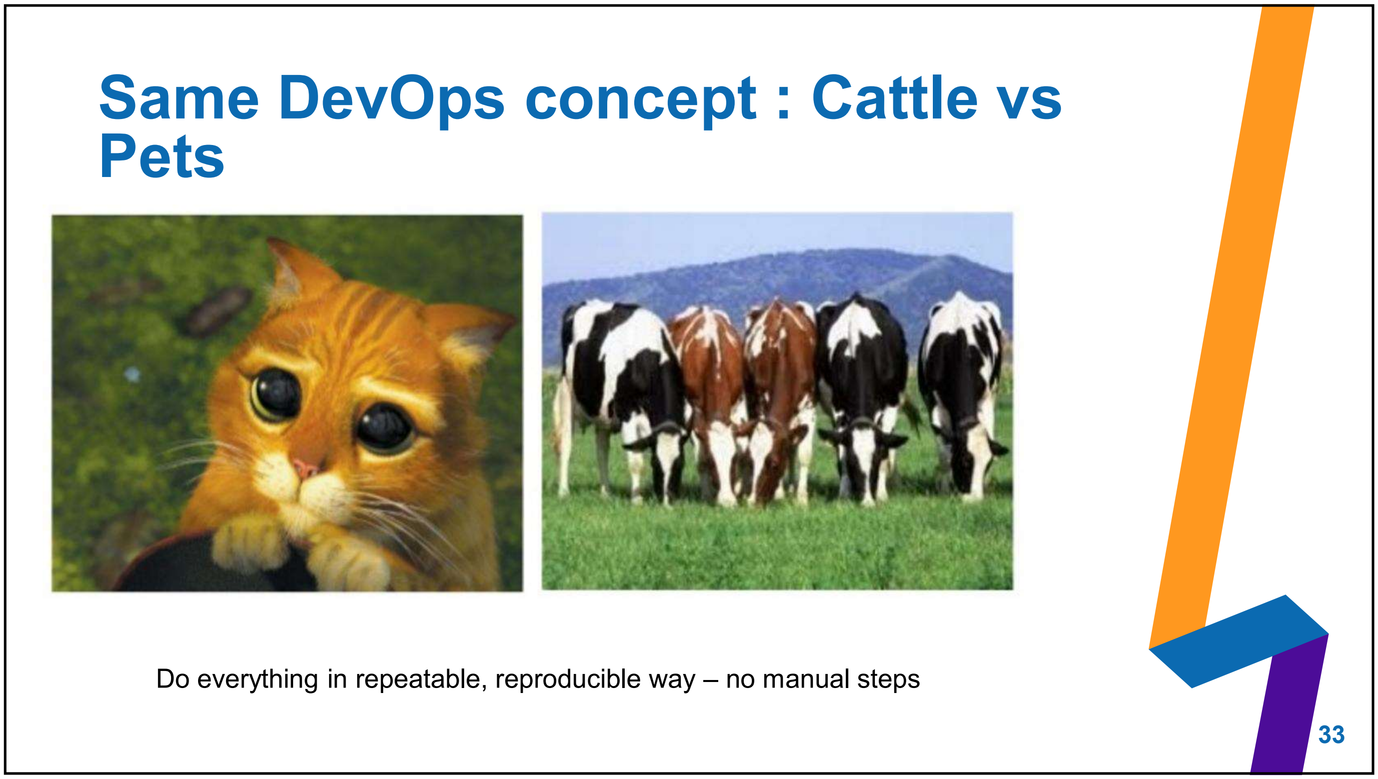
Variability - System

- Inherent to the test setup



32

32

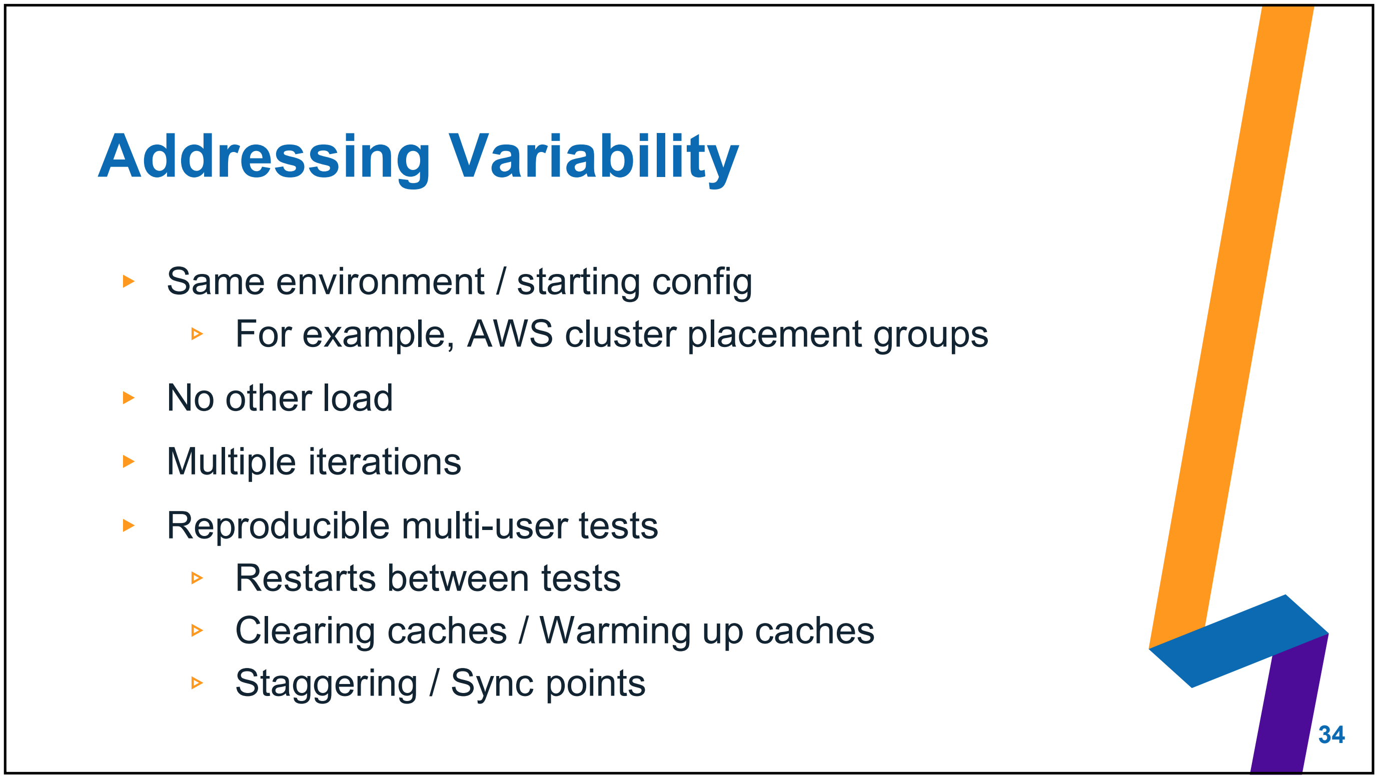


Same DevOps concept : Cattle vs Pets

Do everything in repeatable, reproducible way – no manual steps

33

33



Addressing Variability

- Same environment / starting config
  - For example, AWS cluster placement groups
- No other load
- Multiple iterations
- Reproducible multi-user tests
  - Restarts between tests
  - Clearing caches / Warming up caches
  - Staggering / Sync points

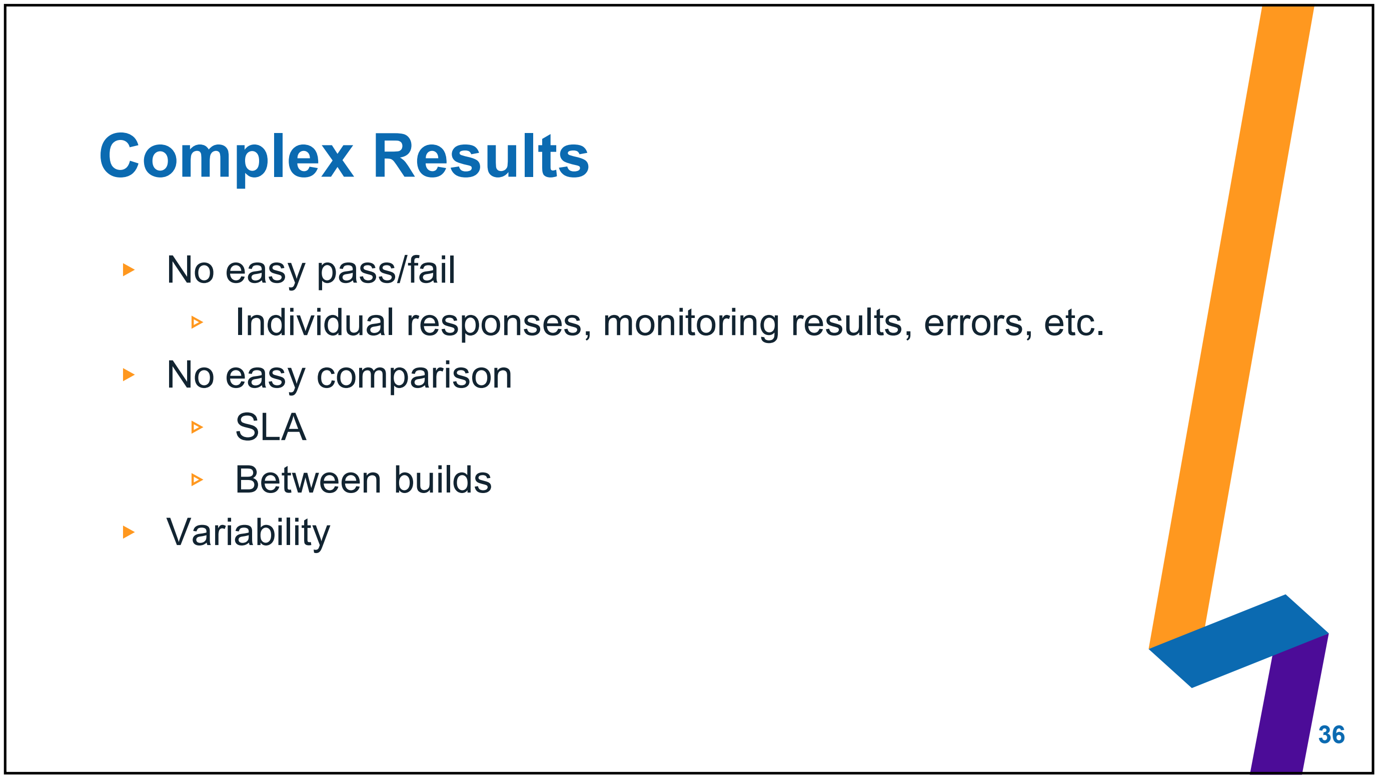
34

34

The Challenge of Change Detection

35

35



Complex Results

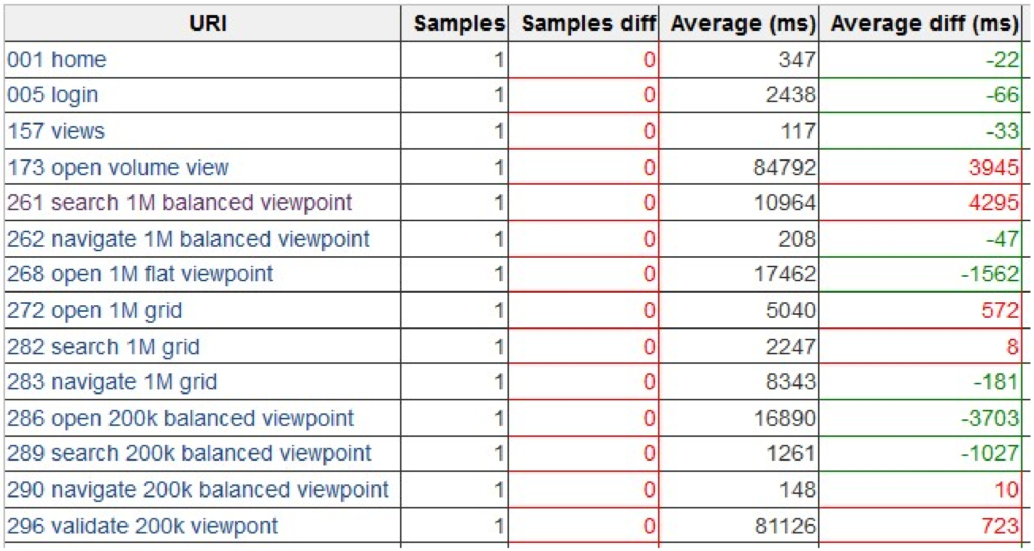
- No easy pass/fail
  - Individual responses, monitoring results, errors, etc.
- No easy comparison
  - SLA
  - Between builds
- Variability

36

36

Simple Comparison

Jenkins Performance Plugin



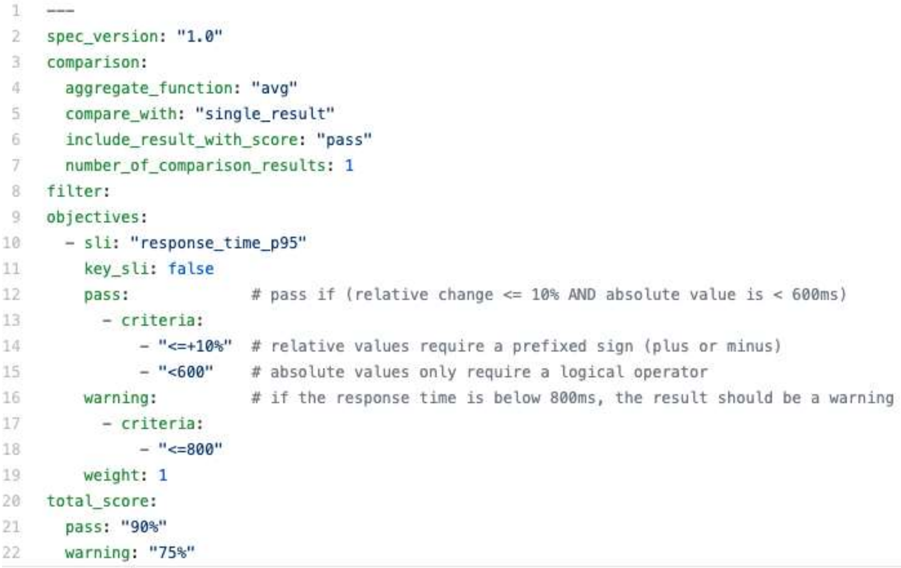
URI	Samples	Samples diff	Average (ms)	Average diff (ms)
001 home	1	0	347	-22
005 login	1	0	2438	-66
157 views	1	0	117	-33
173 open volume view	1	0	84792	3945
261 search 1M balanced viewpoint	1	0	10964	4295
262 navigate 1M balanced viewpoint	1	0	208	-47
268 open 1M flat viewpoint	1	0	17462	-1562
272 open 1M grid	1	0	5040	572
282 search 1M grid	1	0	2247	8
283 navigate 1M grid	1	0	8343	-181
286 open 200k balanced viewpoint	1	0	16890	-3703
289 search 200k balanced viewpoint	1	0	1261	-1027
290 navigate 200k balanced viewpoint	1	0	148	10
296 validate 200k viewpont	1	0	81126	723

37

37

keptn.sh

Quality Gates
SLIs / SLOs as code



```
---
spec_version: "1.0"
comparison:
  aggregate_function: "avg"
  compare_with: "single_result"
  include_result_with_score: "pass"
  number_of_comparison_results: 1
filter:
objectives:
  - sli: "response_time_p95"
    key_sli: false
    pass:             # pass if (relative change <= 10% AND absolute value is < 600ms)
      - criteria:
          - "<=+10%"  # relative values require a prefixed sign (plus or minus)
          - "<600"    # absolute values only require a logical operator
    warning:          # if the response time is below 800ms, the result should be a warning
      - criteria:
          - "<=800"
    weight: 1
total_score:
  pass: "90%"
  warning: "75%"
```

38

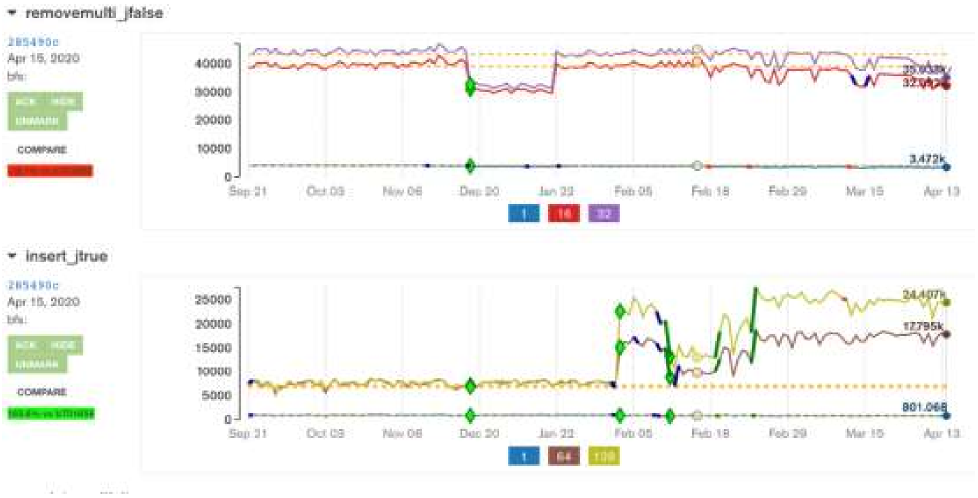
38

Change Point Detection

- Statistical methods taking noise in consideration
  - E-Divisive means algorithm
    - See ICPE Paper: Change Point Detection in Software Performance Testing
    - https://github.com/mongodb/signal-processing-algorithms
      - Open sourced, generic
    - Need several data points. May miss a gradual degradation.

MongoDB.

Change Point Detection - Visualization

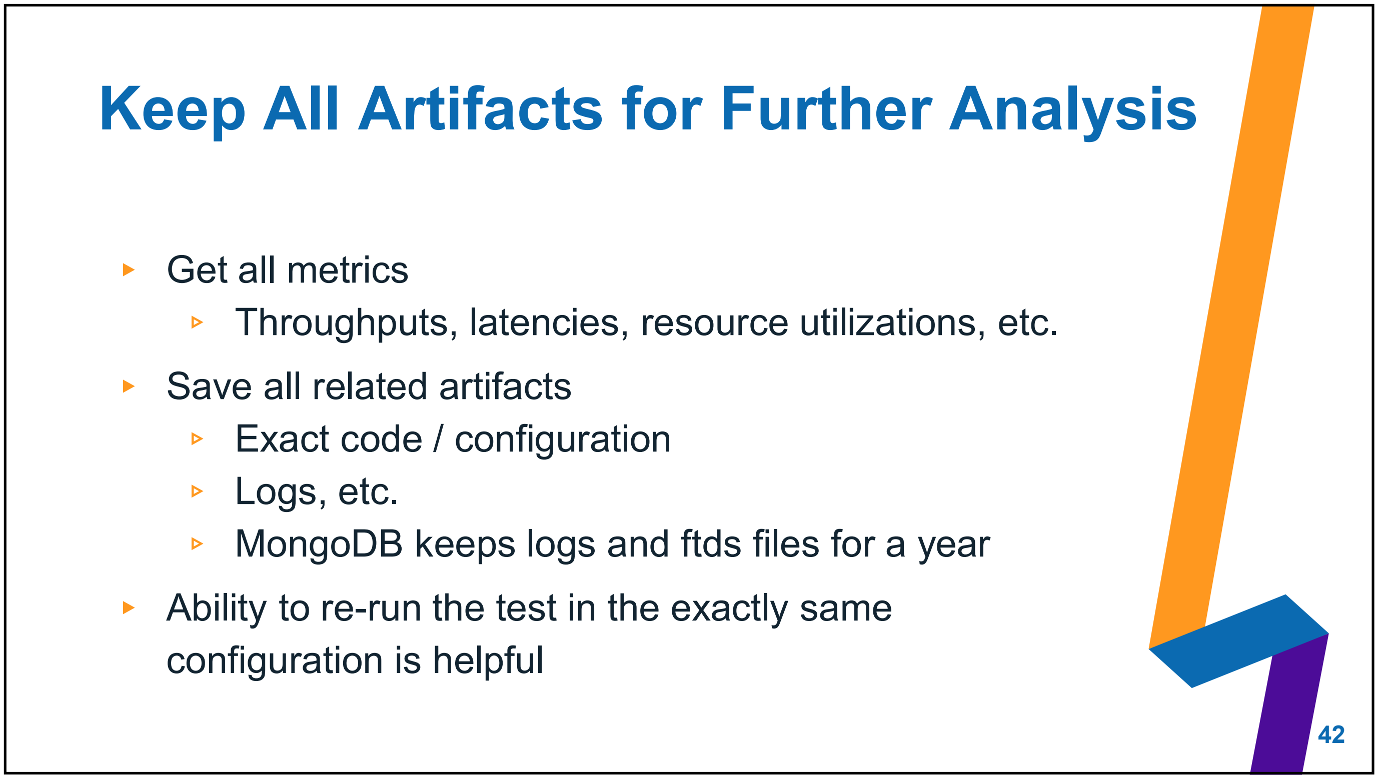


MongoDB.

The Challenge of Advanced Analysis

41

41



Keep All Artifacts for Further Analysis

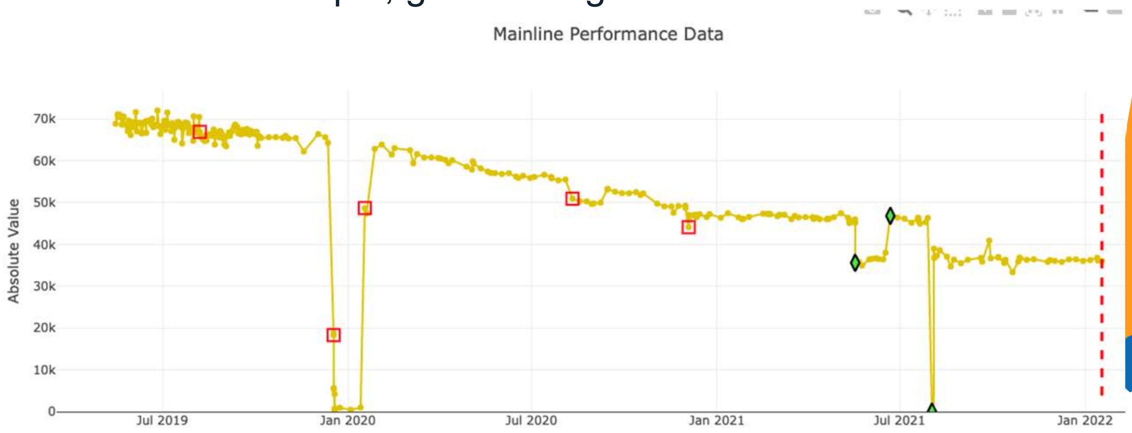
- Get all metrics
  - Throughputs, latencies, resource utilizations, etc.
- Save all related artifacts
  - Exact code / configuration
  - Logs, etc.
  - MongoDB keeps logs and ftds files for a year
- Ability to re-run the test in the exactly same configuration is helpful

42

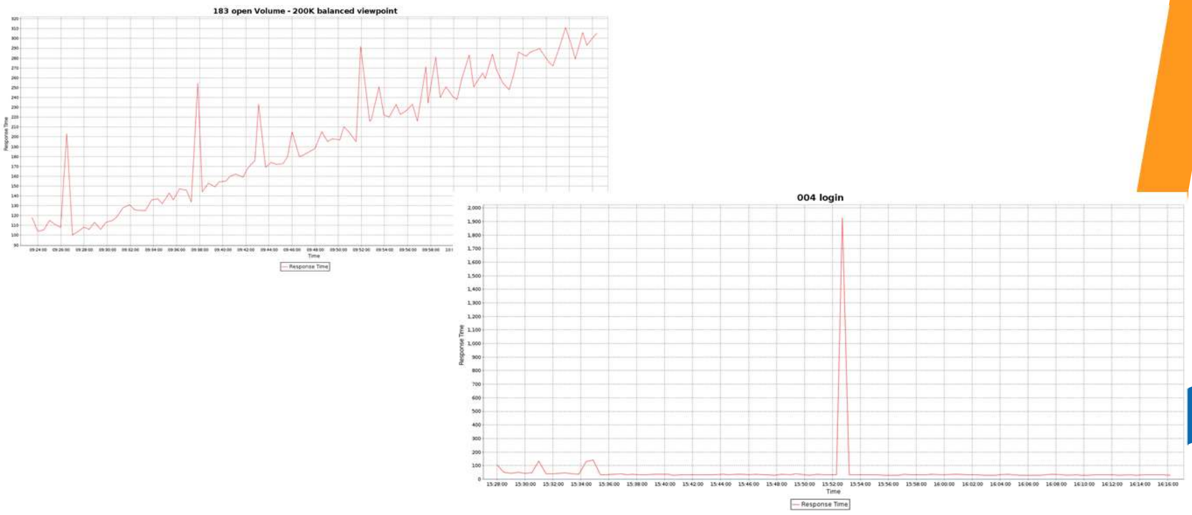
42

Visualization

- Sometimes helps to catch an issue
  - For example, gradual degradation:

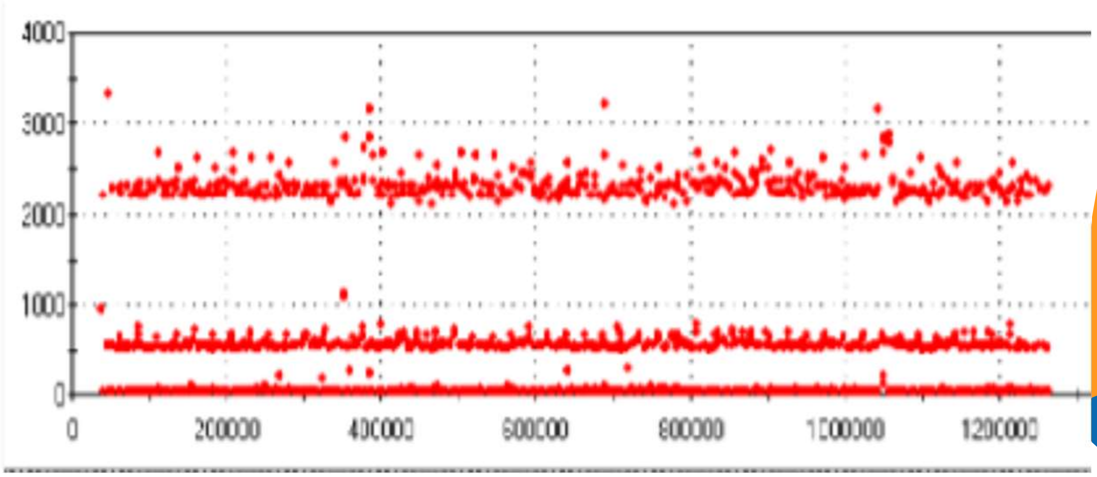


Looking Beyond Aggregate Info



Looking at Individual Results Patterns

Scatter charts – a “banding” pattern from http://www.perftestplus.com/resources/BPT6.pdf



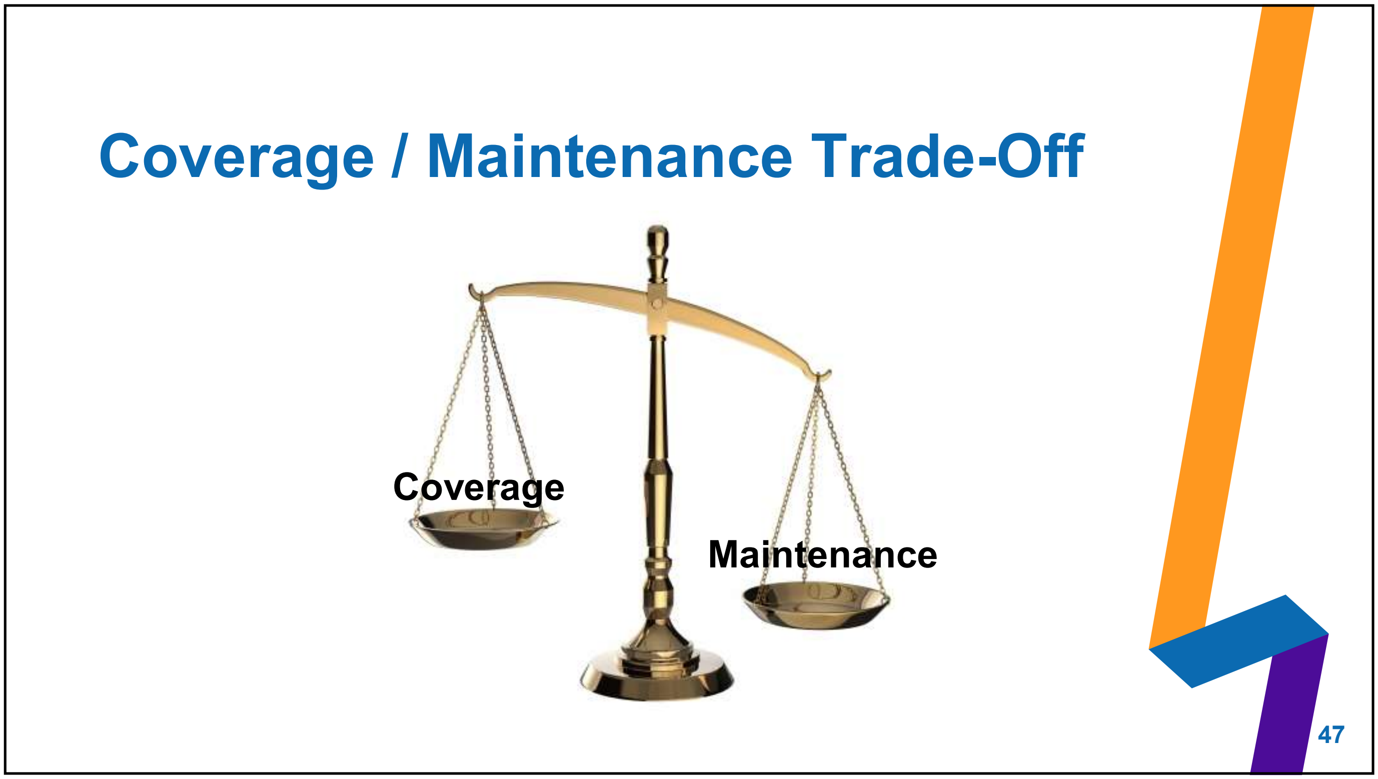
45

45

The Challenge of Maintenance

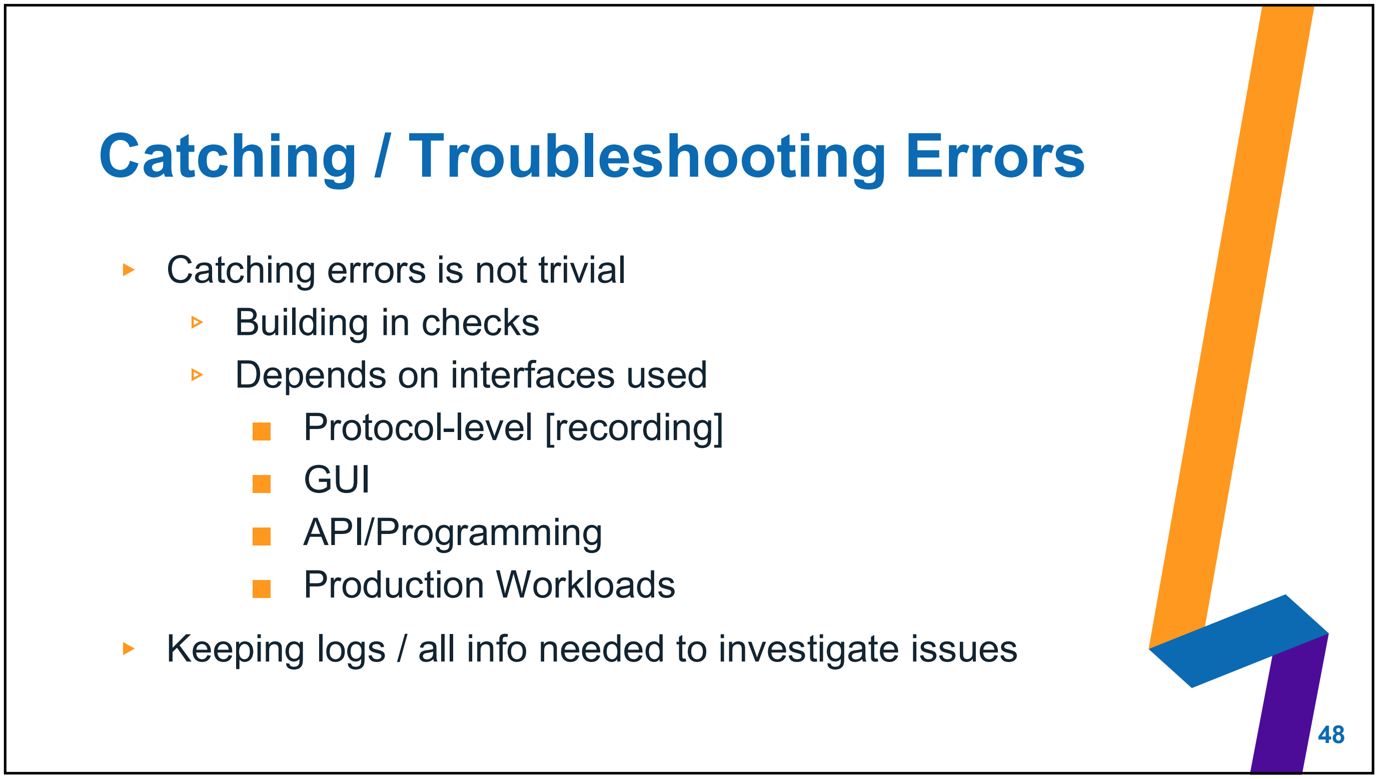
46

46



Coverage / Maintenance Trade-Off

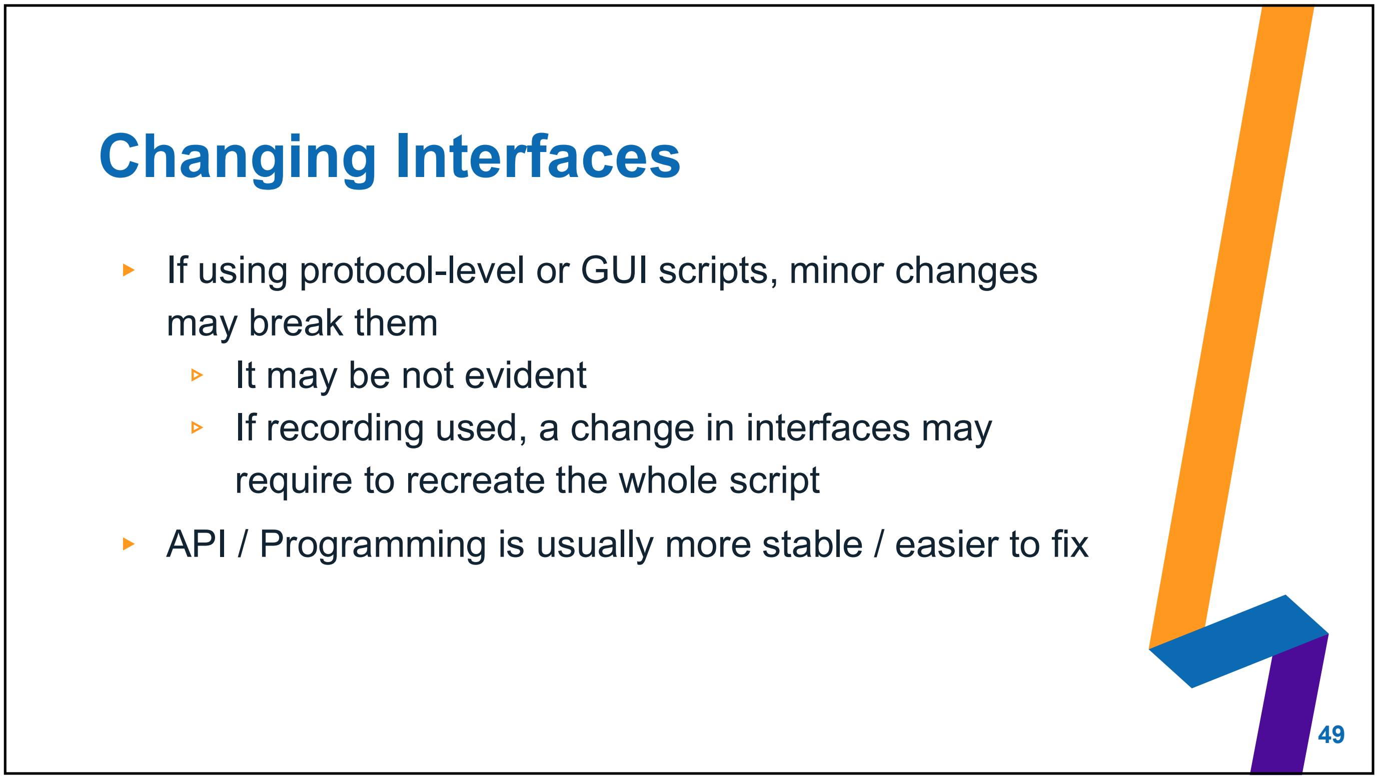
47



Catching / Troubleshooting Errors

- Catching errors is not trivial
  - Building in checks
  - Depends on interfaces used
    - Protocol-level [recording]
    - GUI
    - API/Programming
    - Production Workloads
- Keeping logs / all info needed to investigate issues

48



Changing Interfaces

- If using protocol-level or GUI scripts, minor changes may break them
  - It may be not evident
  - If recording used, a change in interfaces may require to recreate the whole script
- API / Programming is usually more stable / easier to fix

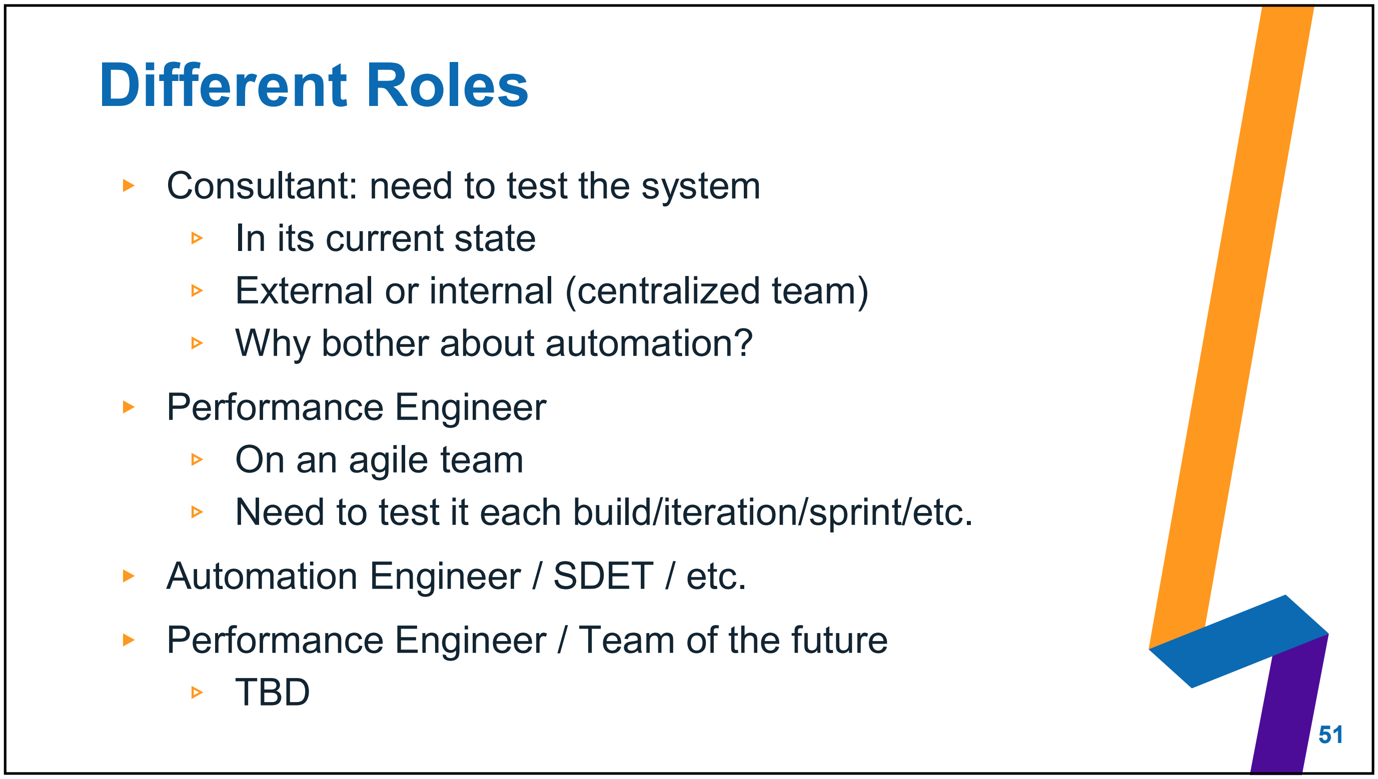
49

49

The Challenge of Organization

50

50

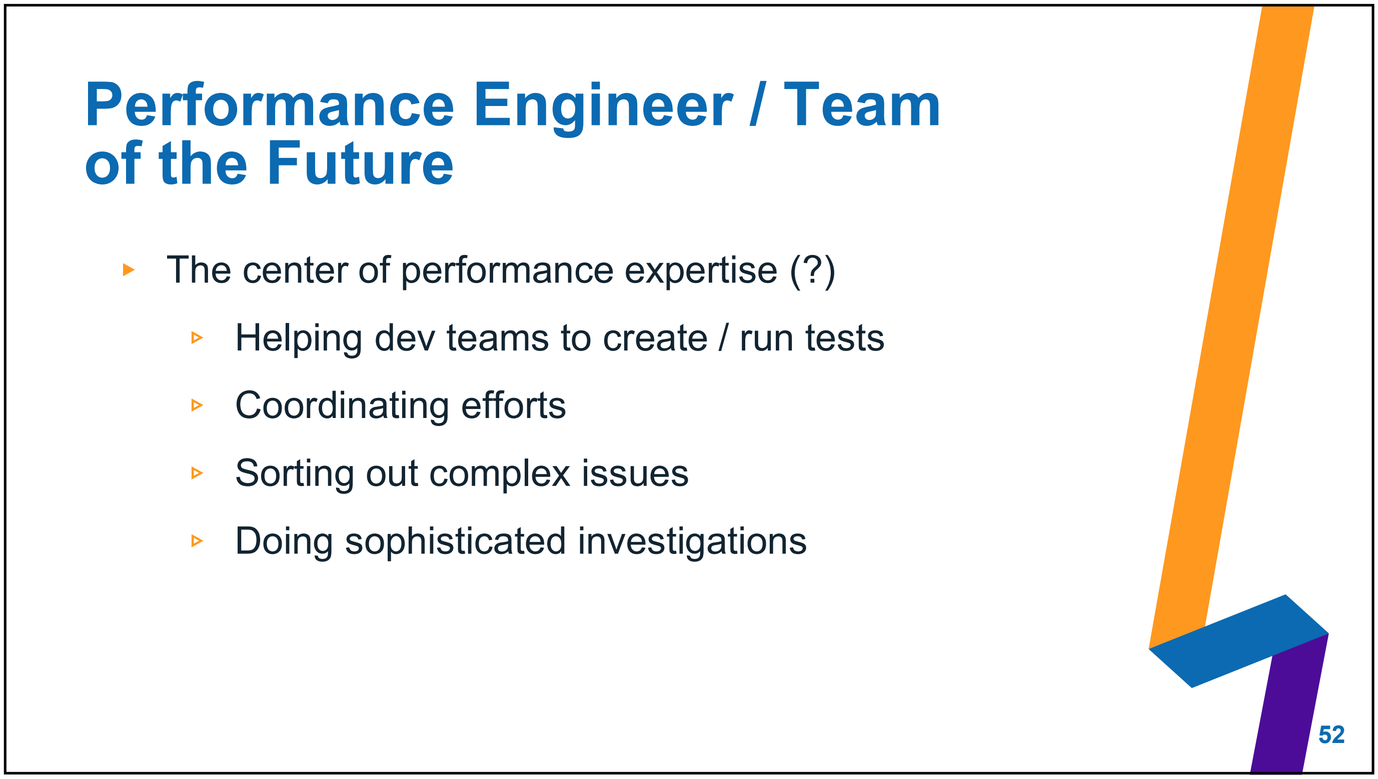


Different Roles

- Consultant: need to test the system
  - In its current state
  - External or internal (centralized team)
  - Why bother about automation?
- Performance Engineer
  - On an agile team
  - Need to test it each build/iteration/sprint/etc.
- Automation Engineer / SDET / etc.
- Performance Engineer / Team of the future
  - TBD

51

51

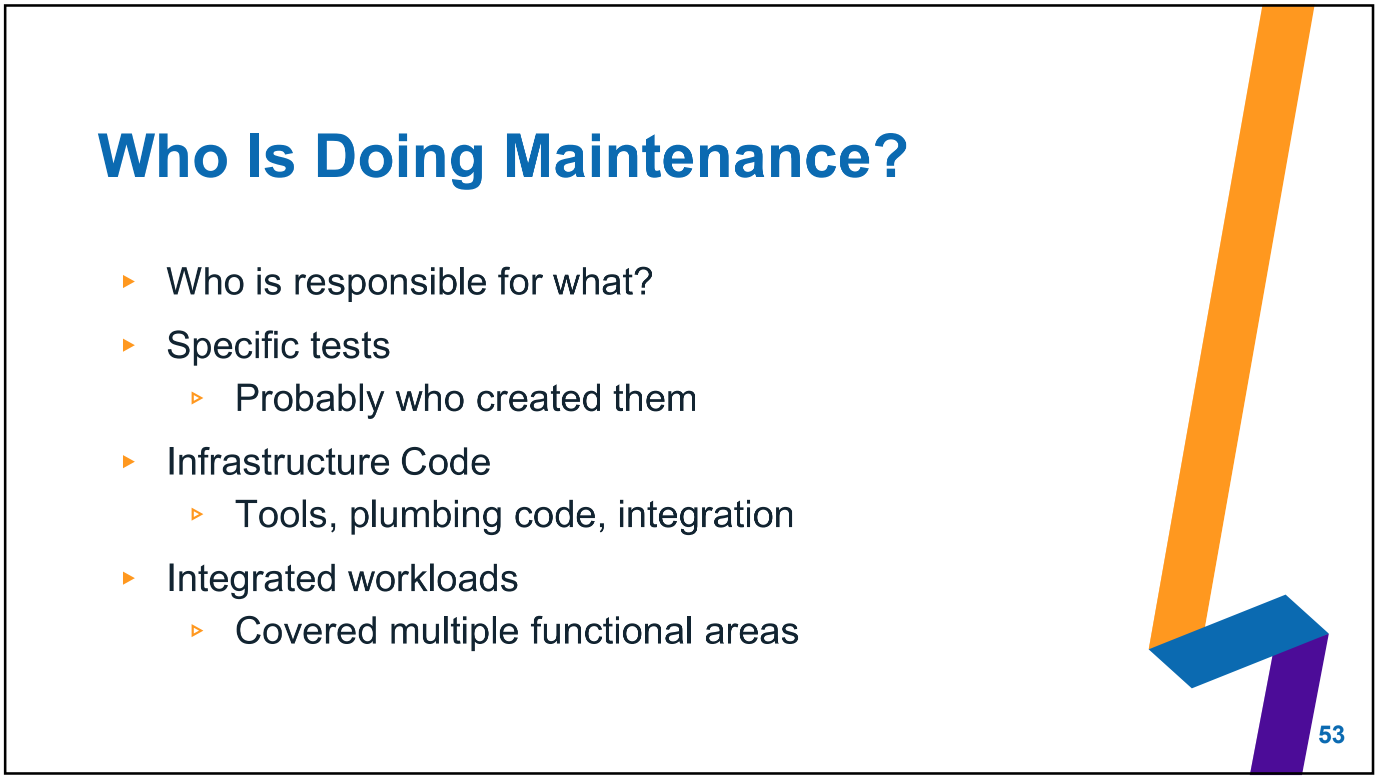


Performance Engineer / Team of the Future

- The center of performance expertise (?)
  - Helping dev teams to create / run tests
  - Coordinating efforts
  - Sorting out complex issues
  - Doing sophisticated investigations

52

52

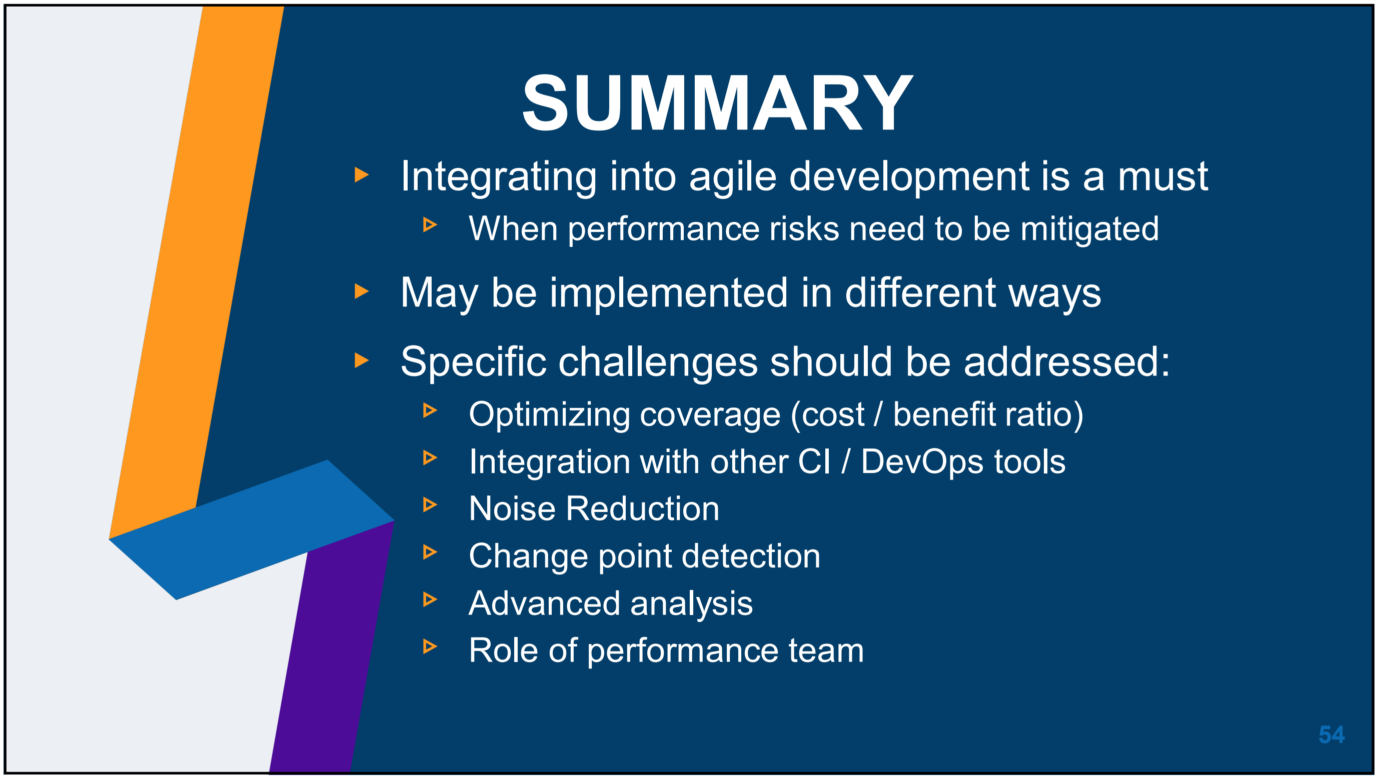


Who Is Doing Maintenance?

- Who is responsible for what?
- Specific tests
  - Probably who created them
- Infrastructure Code
  - Tools, plumbing code, integration
- Integrated workloads
  - Covered multiple functional areas

53

53



SUMMARY

- Integrating into agile development is a must
  - When performance risks need to be mitigated
- May be implemented in different ways
- Specific challenges should be addressed:
  - Optimizing coverage (cost / benefit ratio)
  - Integration with other CI / DevOps tools
  - Noise Reduction
  - Change point detection
  - Advanced analysis
  - Role of performance team

54

54

MongoDB is Open to the Community

David Daly and others discussed implementation in detail

- https://www.daviddaly.me/p/recent-presentations.html

Our code is open source: signal-processing-algorithms, infrastructure code

Our regression environment is open, and the platform is open source

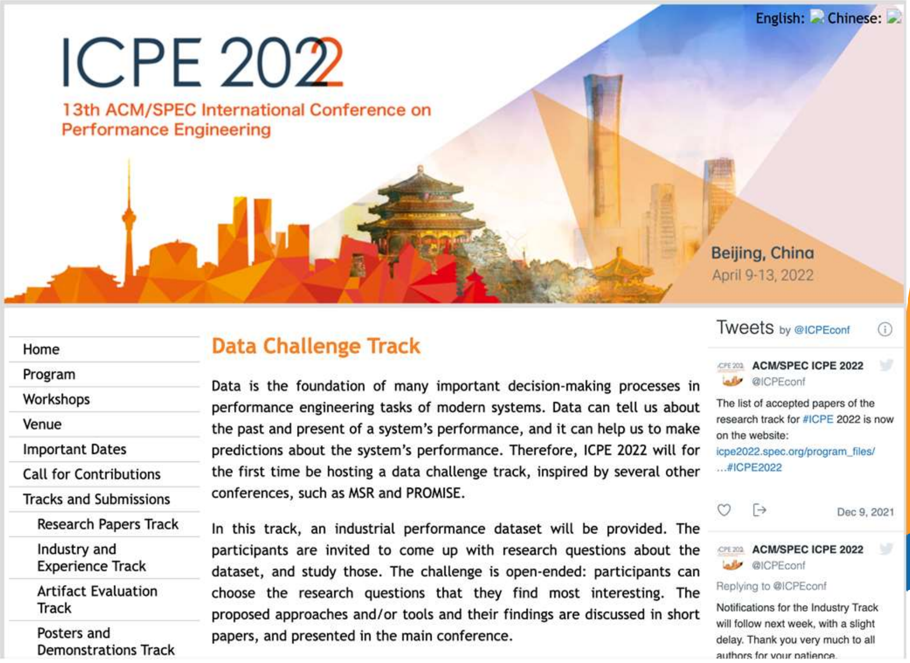
MongoDB.

55

55

Data Challenge

- CFP
- Notes
- Data



ICPE 2022

13th ACM/SPEC International Conference on Performance Engineering

Beijing, China
April 9-13, 2022

Data Challenge Track

Data is the foundation of many important decision-making processes in performance engineering tasks of modern systems. Data can tell us about the past and present of a system's performance, and it can help us to make predictions about the system's performance. Therefore, ICPE 2022 will for the first time be hosting a data challenge track, inspired by several other conferences, such as MSR and PROMISE.

In this track, an industrial performance dataset will be provided. The participants are invited to come up with research questions about the dataset, and study those. The challenge is open-ended: participants can choose the research questions that they find most interesting. The proposed approaches and/or tools and their findings are discussed in short papers, and presented in the main conference.

Tweets by @ICPEconf

ACM/SPEC ICPE 2022
@ICPEconf

The list of accepted papers of the research track for #ICPE 2022 is now on the website: icpe2022.spec.org/program_files/ …#ICPE2022

Dec 9, 2021

ACM/SPEC ICPE 2022
@ICPEconf

Replying to @ICPEconf

Notifications for the Industry Track will follow next week, with a slight delay. Thank you very much to all authors for your patience.

MongoDB.

56

56

Questions ?

apodelko@yahoo.com
@apodelko
https://alexanderpodelko.com/blog/
https://www.linkedin.com/in/alexanderpodelko/

57

57