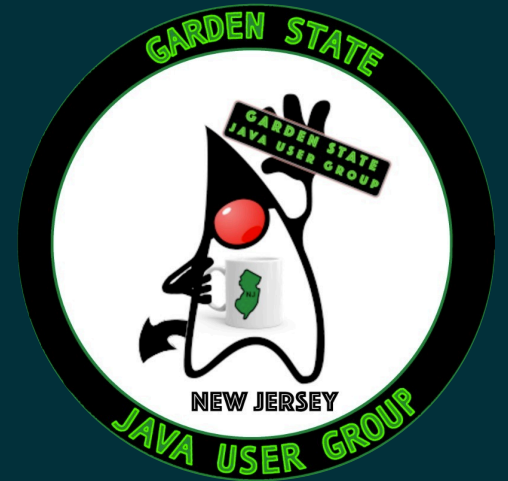


Developer Productivity Engineering

What's in it for Java Developers?



Gradle





Who is this guy?



Java™
Champions



Warner River
Bees



codemonkey.fm



Apache Directory



openstack



THE
APACHE®
SOFTWARE FOUNDATION

Maven™



Sonatype
Nexus



TABS



SPACES



VS



Gradle

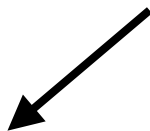
BUILD  TOOL

VS

Maven  TM

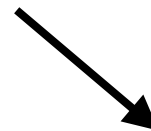


Gradle

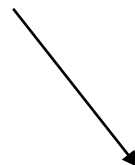


Gradle



BUILD  TOOL



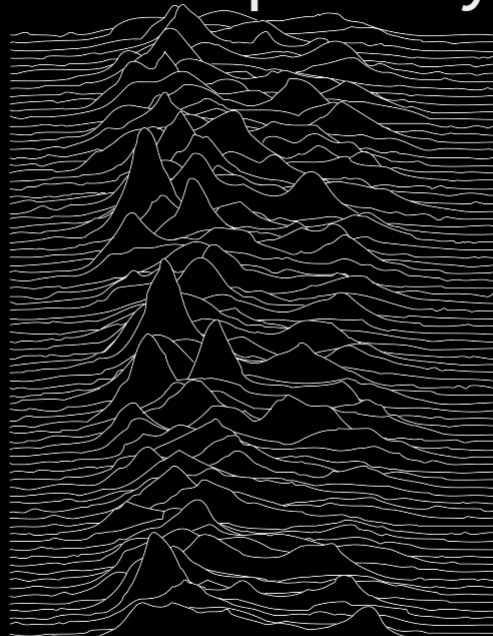
 DEVELOCITY



Developer
Productivity
Engineering

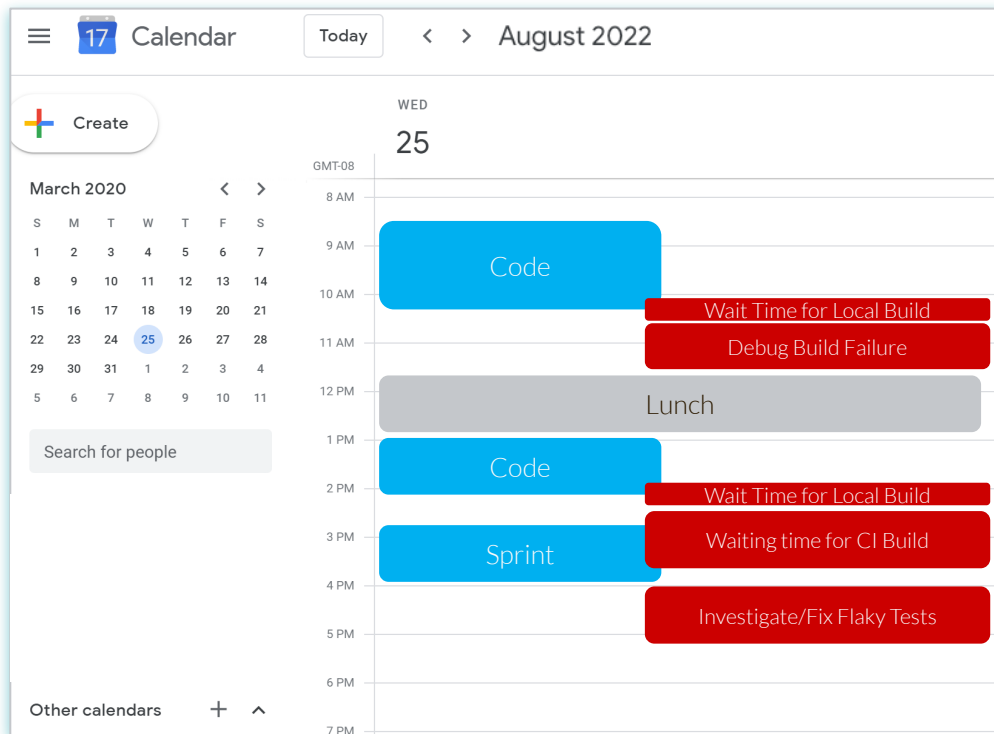
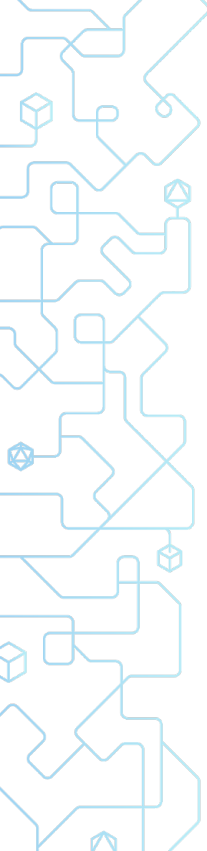
Gradle
BUILD  TOOL
Maven[™]
 Bazel
sbt

Developer Joy



Developer Productivity Engineering





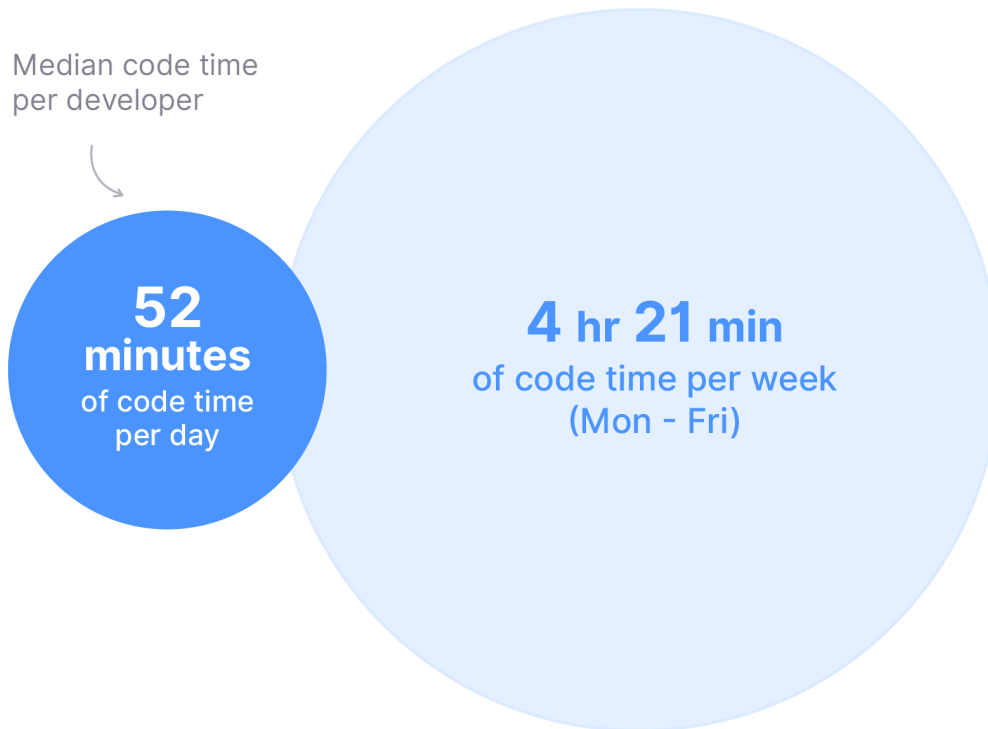
Bottlenecks to Productivity are Everywhere

**“Bottlenecks in the toolchain are holding
back the rockstar 10x developers”**

Pete Smoot, Software Architect, Dell Technologies

It's True at Dell, and Everywhere Else

Developers code 52 minutes per day



<https://www.software.com/reports/code-time-report>



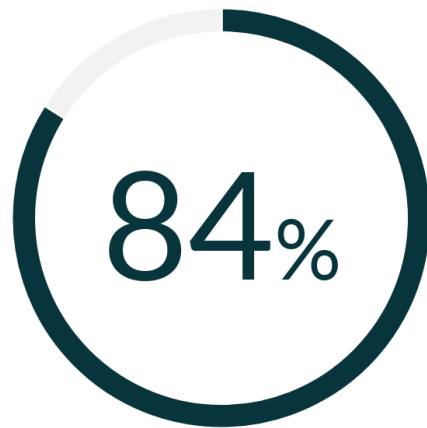
DPE is a new software development practice used by leading software development organizations to maximize developer productivity and happiness.

Gradle is Pioneering DPE

GRADLE ENTERPRISE CUSTOMER STATISTIC

DPE Fosters Developer Joy

84% of surveyed users agree that DPE's impact on their toolchain makes their job more enjoyable.



Source: TechValidate survey of 51 users of Gradle Enterprise

✓ Validated

Published: Jul. 2, 2023 TVID: 930-05A-A5F

 Gradle Enterprise

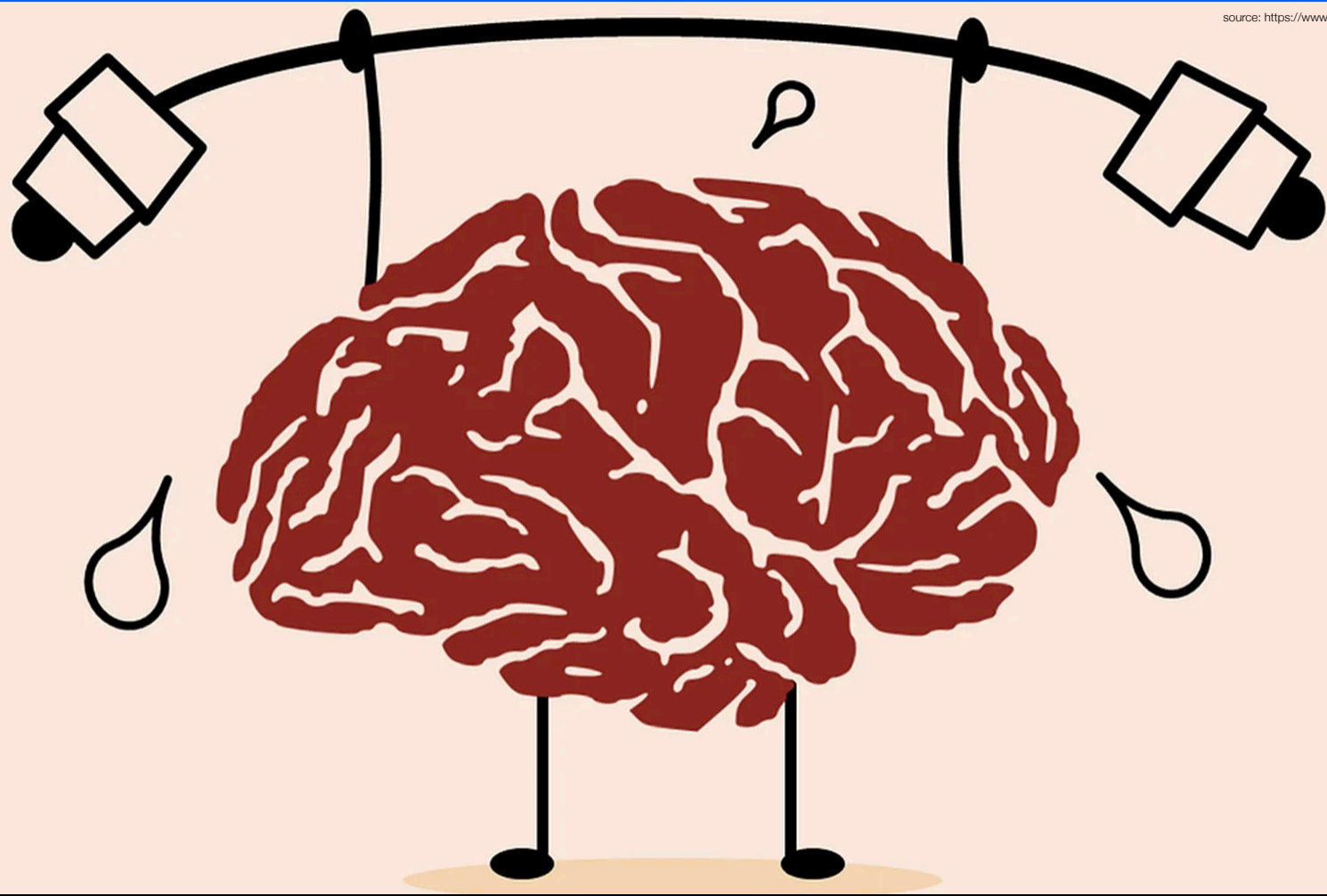
TechValidate
by SurveyMonkey

THE #1 PROGRAMMER EXCUSE
FOR LEGITIMATELY SLACKING OFF:

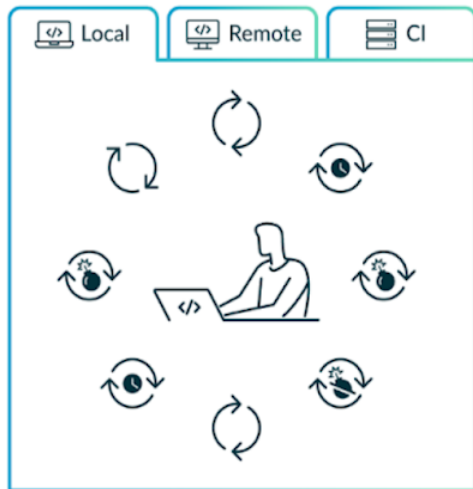
"MY CODE'S COMPILING."



This Doesn't Have To Be Our Reality



What Problems Does DPE Solve?



240 days per year



100s of developers



Productivity



Cost
10s of millions



This takes too long!



This takes too long to fix



This should have been observable

1970s+

JIT
Manufacturing

1980s+

Business
Process
Reengineering

1990s+

Change
management

2000s+

Agile, Lean Six
Sigma

2010s+

DevOps

2020+

DPE

GRADLE ENTERPRISE CUSTOMER STATISTIC

Builds and Tests are Slow!

85% of surveyed IT organizations experienced challenges with too much time spent waiting on build and test feedback.



Source: TechValidate survey of 65 users of Gradle Enterprise



Validated

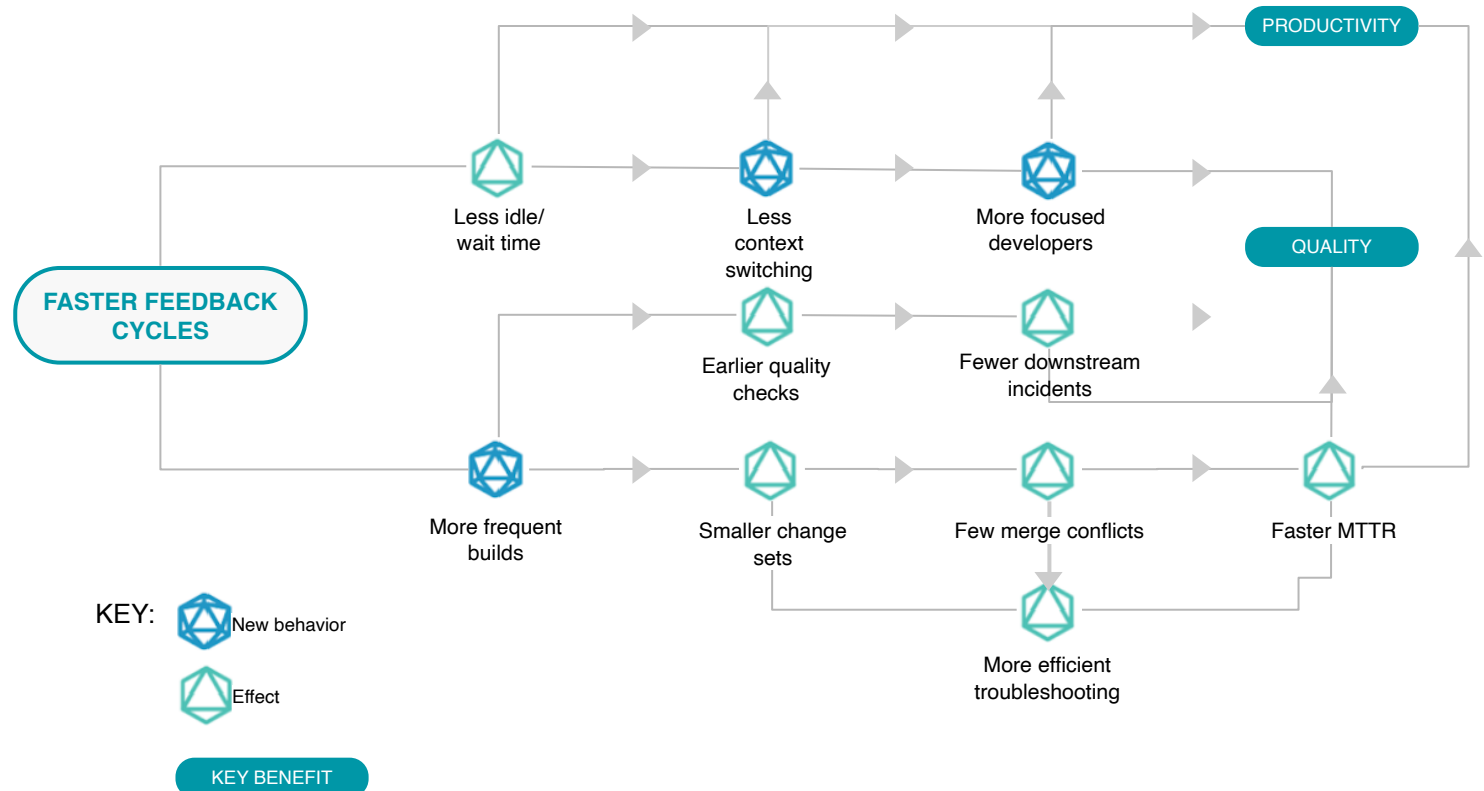
Published: Jul. 20, 2023 TVID: 7F3-942-85D



TechValidate
by SurveyMonkey



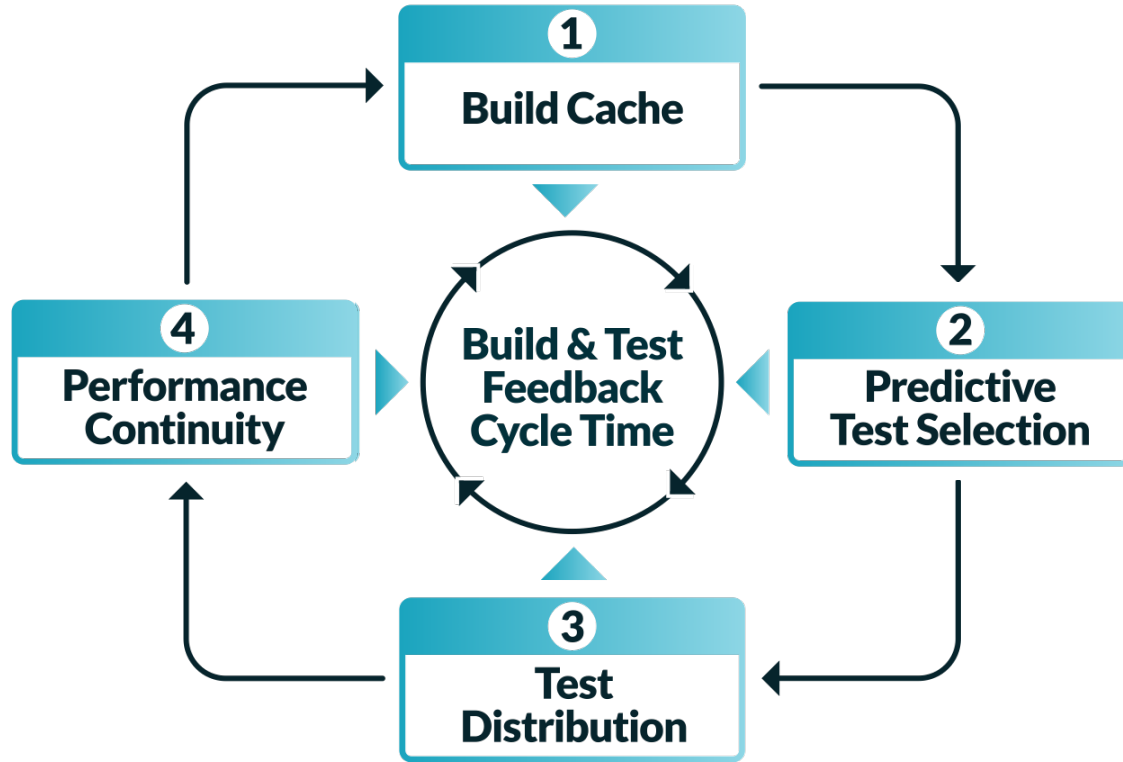
The anatomy and importance of fast feedback cycles



Faster Builds Improve Creative Flow



	Team 1	Team 2
No. of Devs	11	6
Build Time	4 mins	1 mins
No. of local builds	850	1010



Multiple Acceleration Technologies are Best



***Build caching delivers fast build and test
feedback cycles***



Build Caching

- Introduced to the Java world by Gradle in 2017
- Maven has an open source build cache too
- **Used by leading technology companies** like Google and Facebook
- Can support both **user local and remote caching** for distributed teams

- Build caches are **complementary to dependency caches**, not mutually exclusive:
 - A dependency cache caches **fully compiled dependencies**
 - A build cache accelerates **building a single source repository**
 - A build cache caches build actions (e.g. Gradle tasks or Maven goals)



What is a Build Cache?

Inputs



- Gradle Tasks
- Maven Goal Executions

Outputs



When the inputs have not changed, the **output can be reused** from a previous run.



Cache Key/Value Calculation

The **cacheKey** for Gradle Tasks/Maven Goals is based on the Inputs:

```
cacheKey(javaCompile) = hash(sourceFiles,  
                             jdk version,  
                             classpath,  
                             compiler args)
```

The **cacheEntry** contains the output:

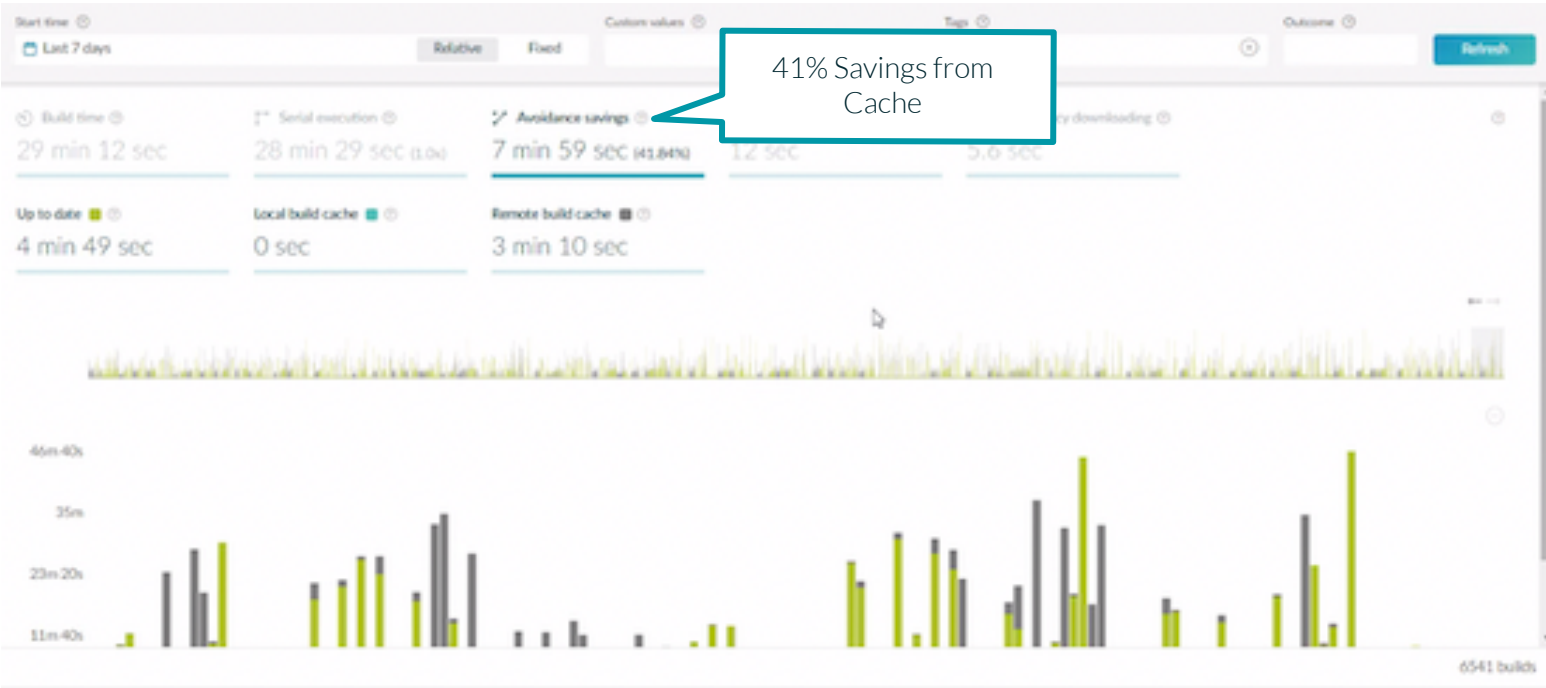
```
cacheEntry[cacheKey(javaCompile)] = fileTree(classFiles)
```

For more information, see:

https://docs.gradle.org/current/userguide/build_cache.html



Remote Build Cache Savings at Dell





Reproducible Builds

<https://reproducible-builds.org/>





Predictive Test Selection leads to greater efficiencies



Predictive Test Selection

International Conference on Software Engineering (ICSE)

Abstract

Change-based testing is a key component of continuous integration at Facebook. However, a large number of tests coupled with a high rate of changes committed to our monolithic repository make it infeasible to run all potentially impacted tests on each change. We propose a new *predictive test selection strategy* which selects a subset of tests to exercise for each change submitted to the continuous integration system. The strategy is *learned* from a large dataset of historical test outcomes using basic machine learning techniques. Deployed in production, the strategy reduces the total infrastructure cost of testing code changes by a factor of two, while guaranteeing that over 95% of individual test failures and over 99.9% of faulty changes are still reported back to developers. The method we present here also accounts for the non-determinism of test outcomes, also known as test flakiness.

 Download Paper

 Copy PDF URL

By: Mateusz Machalica, Alex Samykin, Meredith Porth, Satish Chandra

November 23, 2020

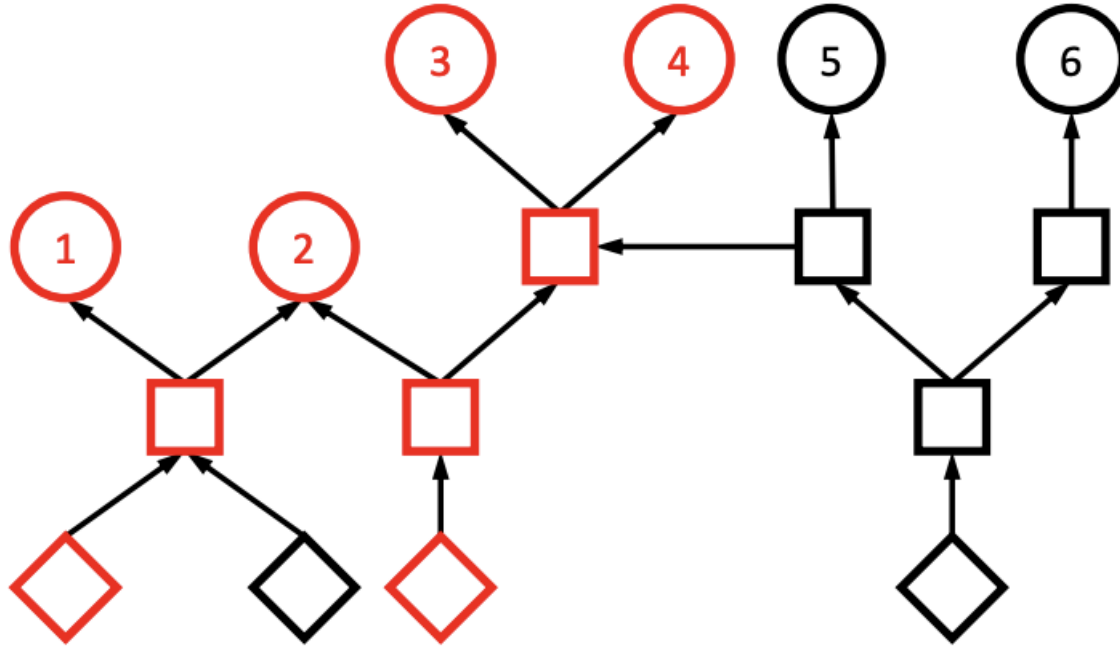
Areas **AR/VR**

Tags **PROBABILITY**

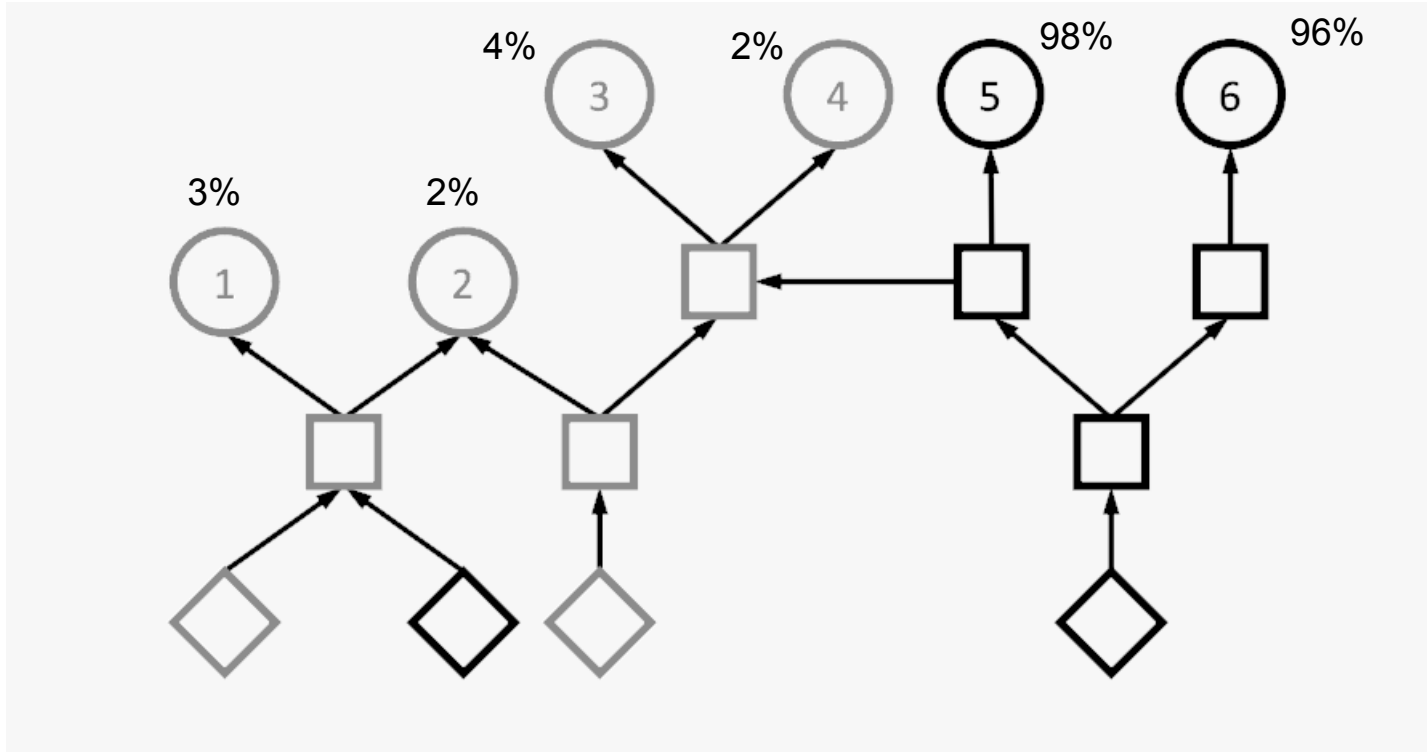
Share   



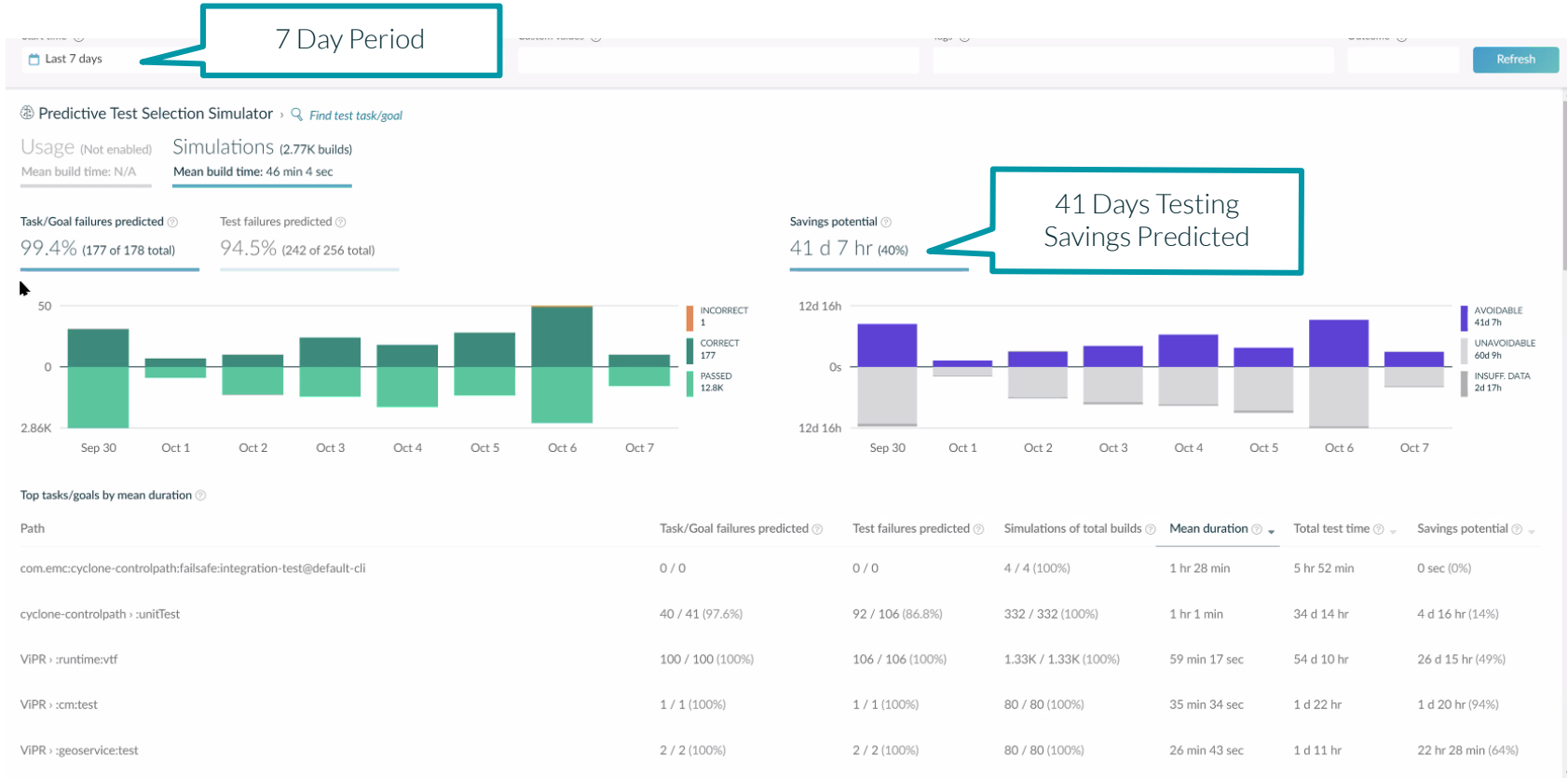
Conventional Test Selection Approach



Predictive Test Selection Approach



Predictive Test Selection Savings at Dell

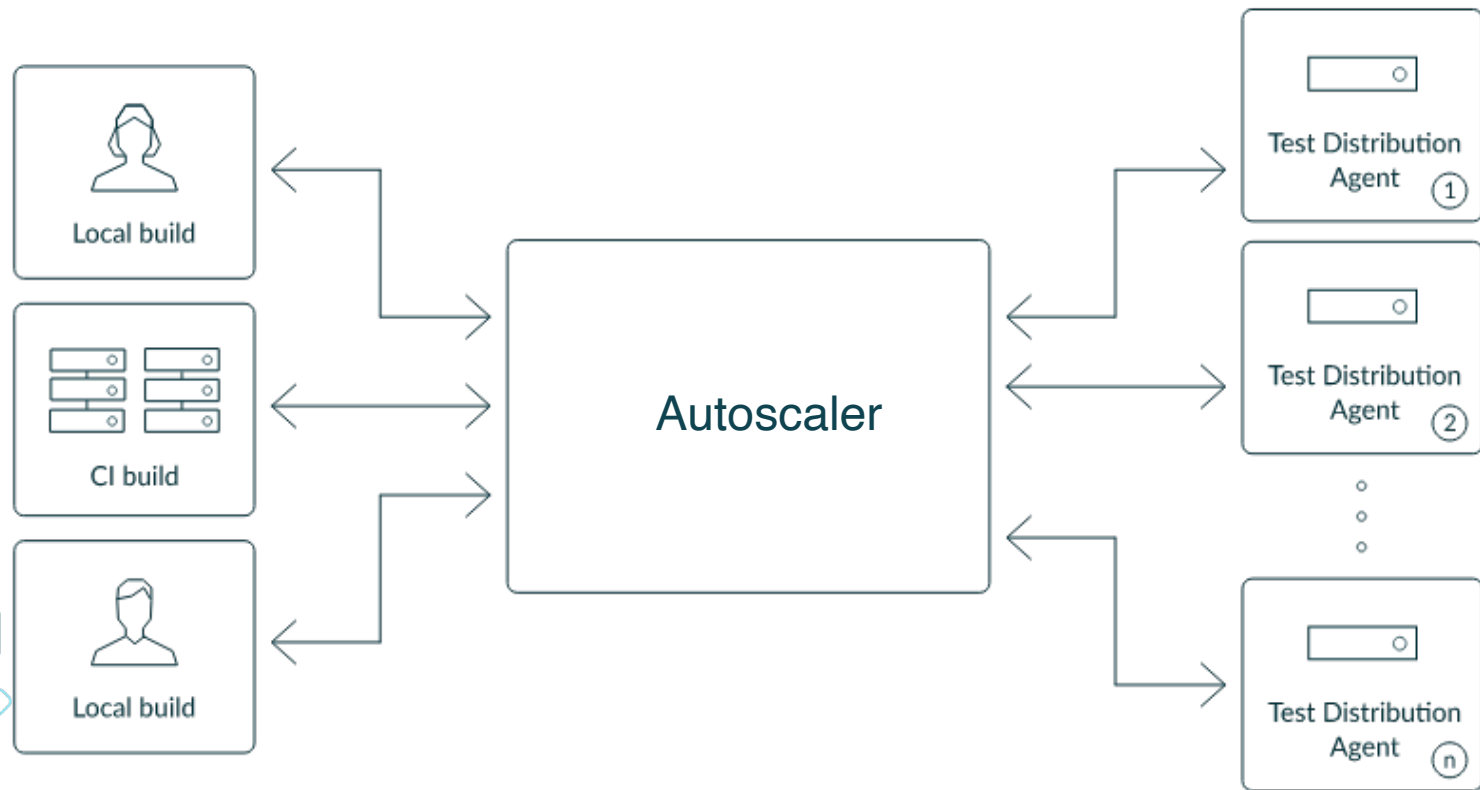




Test distribution can make tests even faster



How it works



Existing solutions - CI fanout

Test execution is distributed by manually partitioning the test set and then running partitions in parallel on several CI nodes.

```
pipeline {
  stage('compile') { ... }
  parallelStage('test') {
    step {
      sh './gradlew :testGroup1'
    }
    step {
      sh './gradlew :testGroup2'
    }
    step {
      sh './gradlew :testGroup3'
    }
  }
}
```





Assessment of existing solutions

- ◆ **Build Caching** is great in many cases but doesn't help when test inputs have changed.
- ◆ **Single machine parallelism** is limited by that machine's resources.
- ◆ **CI fanout** does not help during local development, is inefficient (in particular on ephemeral CI agents or without build cache), requires manual setup and test partitioning, and result collection/aggregation





Build Scans speeds up troubleshooting



Improved Troubleshooting



☰ Summary

☞ Console log

☒ Failure

⌘ Timeline

📊 Performance

☑ Tests

📦 Projects

🔗 Dependencies

🔧 Extensions

📁 Plugins

☰ Custom values

🔑 Switches

🏗 Infrastructure

🕒 See before and after

Started today at 10:25:26 AM EDT, finished today at 10:26:16 AM EDT 🌐

Maven 3.8.5, Gradle Enterprise Maven Extension 1.15

[Explore console log](#)

1 failure

Failed to execute goal `moderne:ast (default-cli) @ shopping`: Execution default-cli of goal `io.moderne:moderne-maven-plugin:0.27.0:ast` failed

255 other builds with similar failures in last 7 days [View failure history](#)

No version provided for dependency `commons-beanutils:commons-beanutils`

[Explore failure](#)

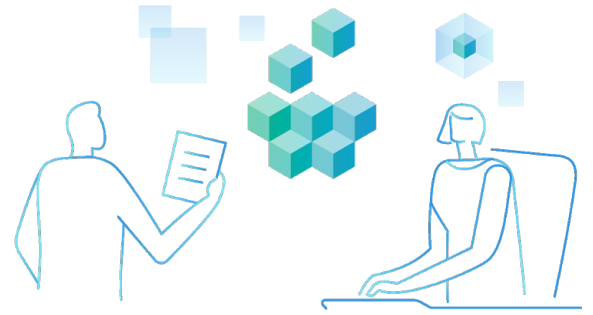
9 goals executed in 1 project, 1 failed goal in 50s

<code>moderne:ast @ shopping</code>	FAILED	12.948s
<code>compiler:compile @ shopping</code>		14.741s
<code>spring-boot:repackage @ shopping</code>		4.891s
<code>war:war @ shopping</code>		4.586s

Without focus, problems can sneak back in...

- ◆ Infrastructure changes
 - Binary management
 - Caching
 - CI agents
- ◆ New annotation processors or versions of annotation processors
- ◆ Build logic configurations settings
 - Build tool version and plugins
 - Compiler and/or Memory settings
- ◆ Code refactoring
- ◆ New office locations
- ◆ Without observability, it is impossible to have a great and fast developer experience.





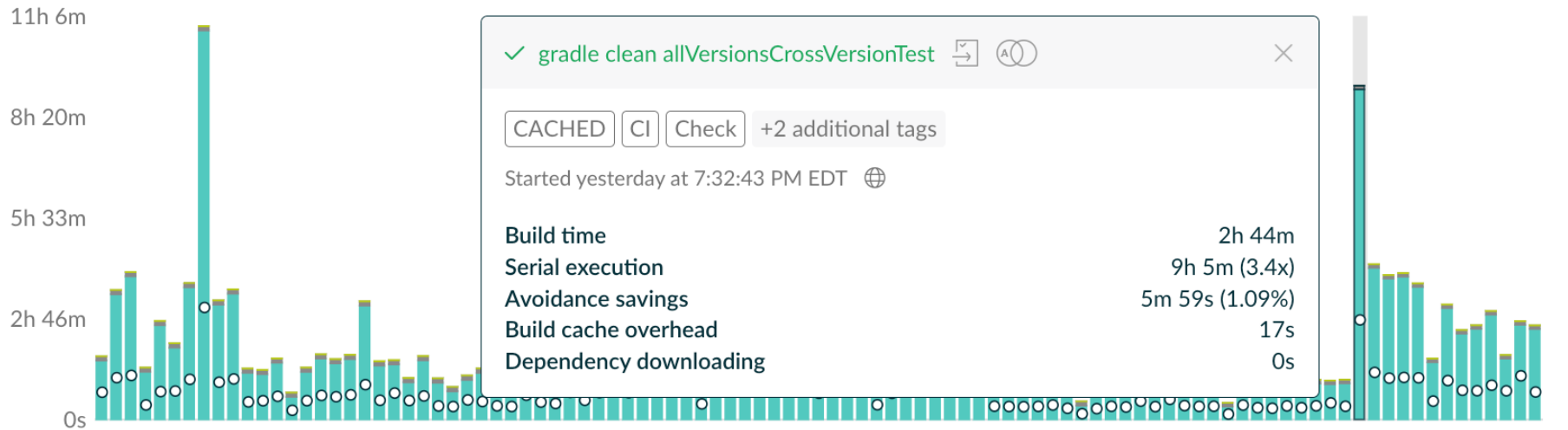
“You can observe a lot by just watching.”

- Yogi Berra, Catcher and Philosopher



Performance Insights

Are you tracking local
build and test times?



463 builds

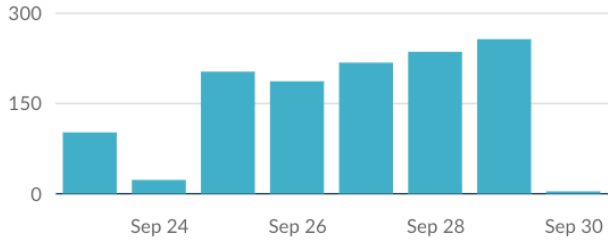
DPE Organizations Track Build and Test Times

> There were failing tests. See the report at: file:///.../build/reports/tests/*/index.html

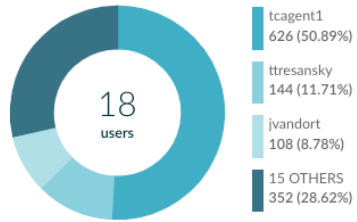


Builds with matching failures

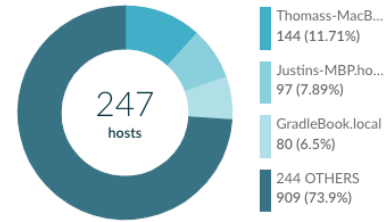
1.23K (2% of 59K total builds)



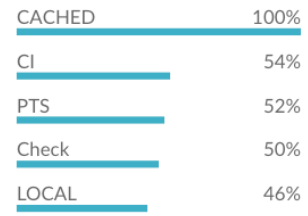
Affected users



Affected hosts



Top Tags

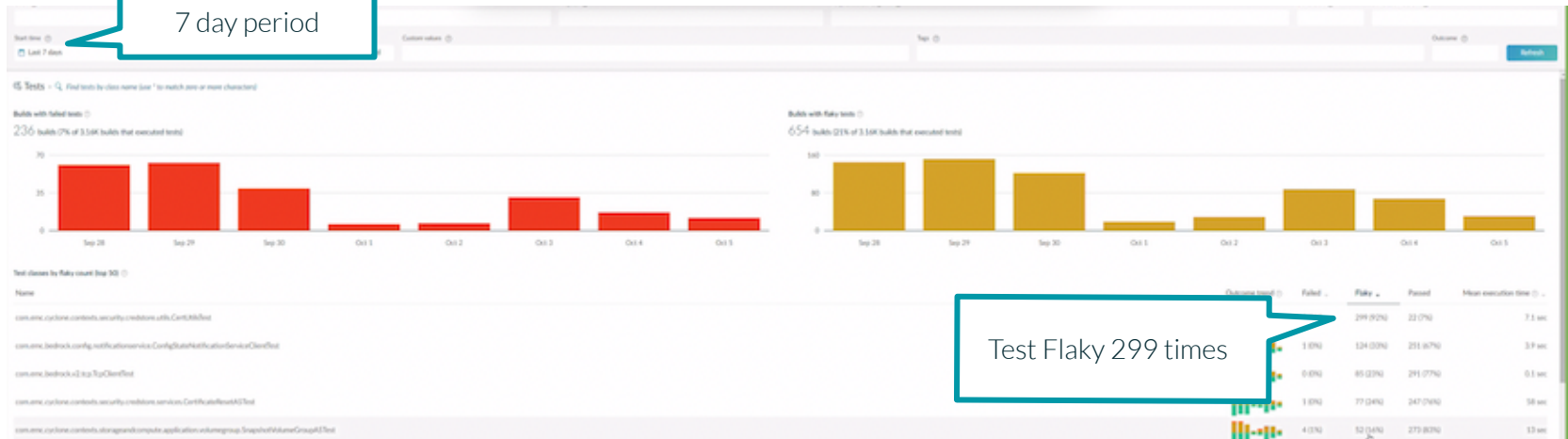


Failed builds (50 most recent)

Start time	Project	Requested tasks/goals	User	Hostname
today at 1:00:47 AM	gradle	:core:embeddedIntegTest --tests org.grad	javandort	Justins-MBP.home
	<div style="display: flex; gap: 5px;"> CACHED LOCAL IDEA dirty </div> <p>Execution failed for task ':core:embeddedIntegTest'.</p> <p>> There were failing tests. See the report at: file:///Users/javandort/work/gradle/subprojects/core/build/reports/tests/embeddedIntegTest/index.html</p>			

DPE Organizations Track Failure Rates

Flaky Tests Are Everywhere



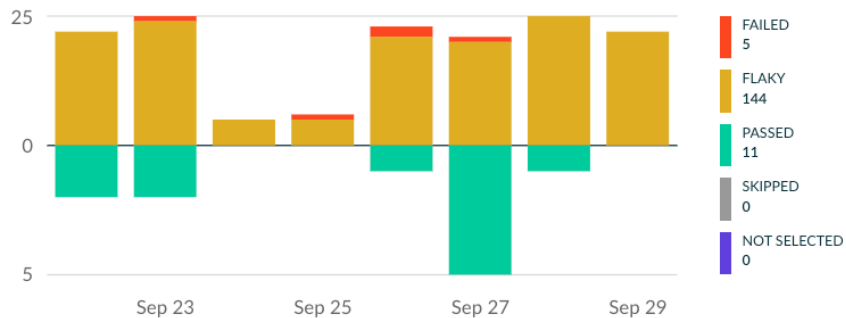
Dealing with Flaky Tests

The test is flaky. What do you do now?

- a. Try it again
 - b. Re-run it
 - c. Re-run it again
 - d. Ignore it and approve PR
 - e. All of the above
-

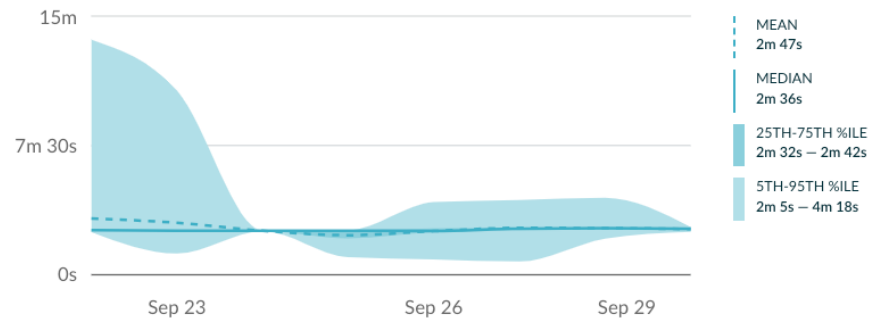
Builds that executed test class

160 builds



Mean execution time for test class

2 min 47 sec



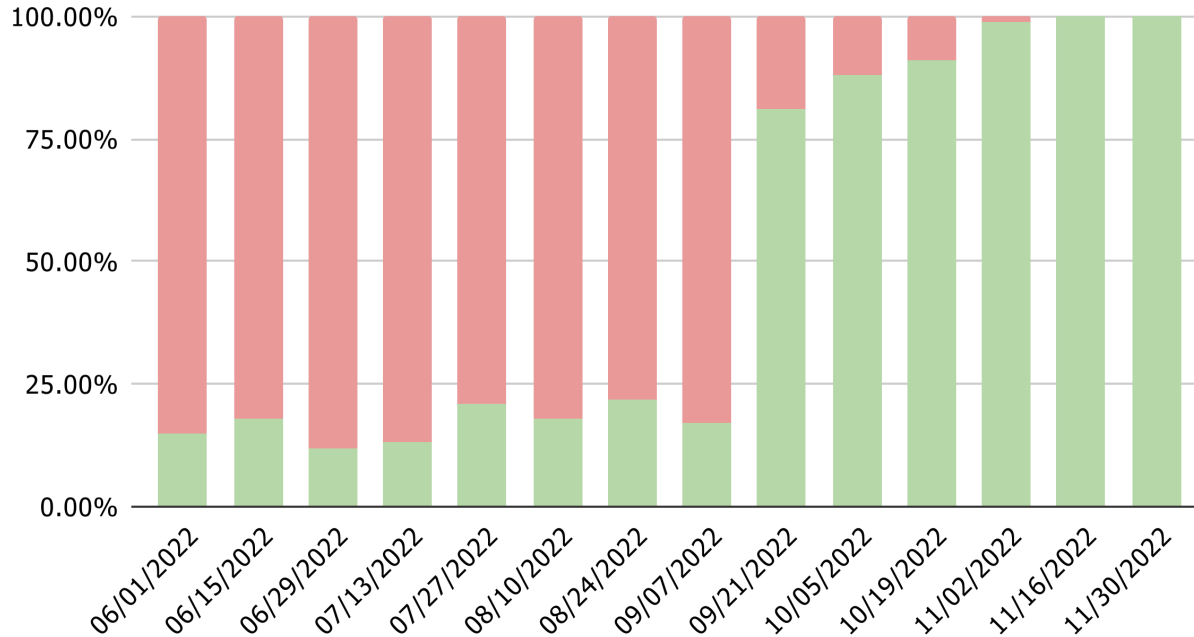
Tests by flaky count

Name	Outcome trend	Failed	Flaky	Passed	Mean execution time
performs static validation of plugins used by the Gradle build		5 (3%)	144 (90%)	11 (7%)	2 min 47 sec

DPE Organizations Analyze Flaky Tests

All Of This Will Improve CI

Distributed Agent Availability - Main Branch





Tips

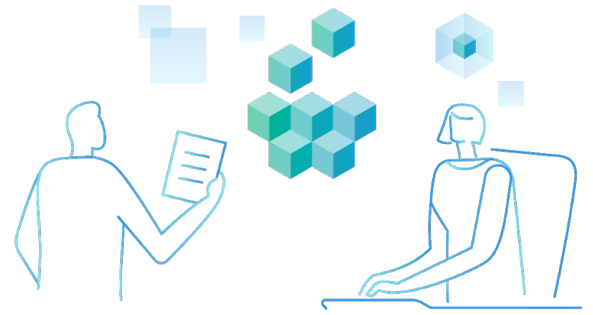




TOOL ALLOWANCE

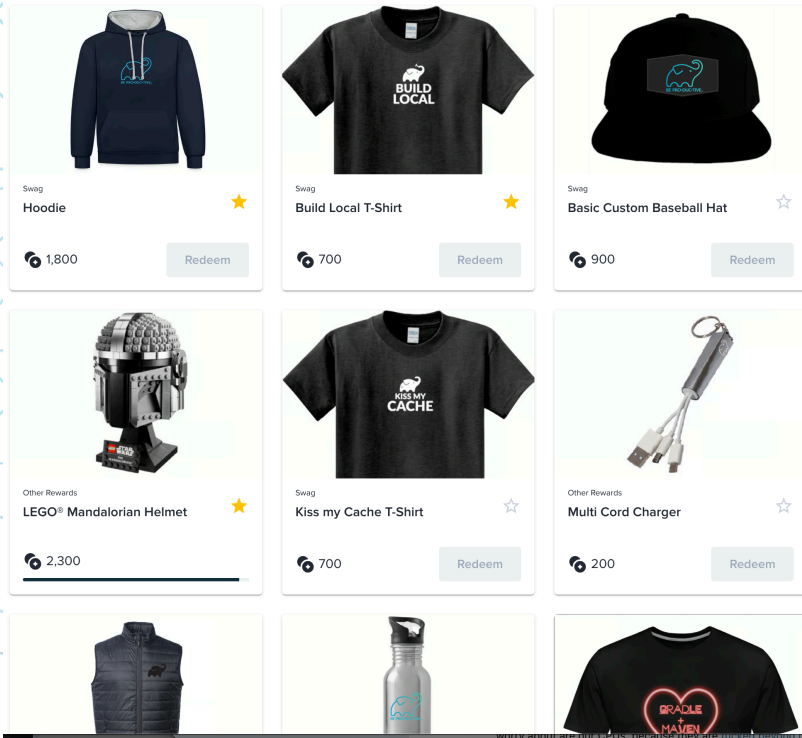
Block out time in your calendar!





***DPE will become standard practice
Because the world should foster developer joy***





 **BrianDemers**

 **bdemers**



Learn more & get free swag

