Developer Productivity Engineering

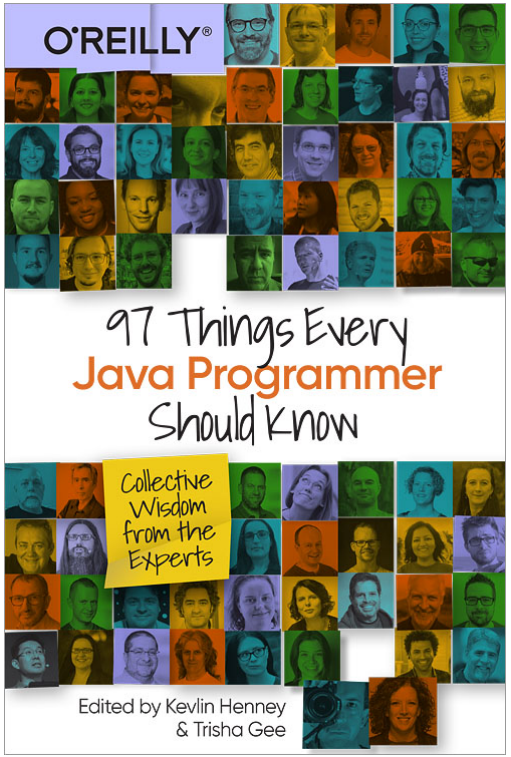
What's in it for me?

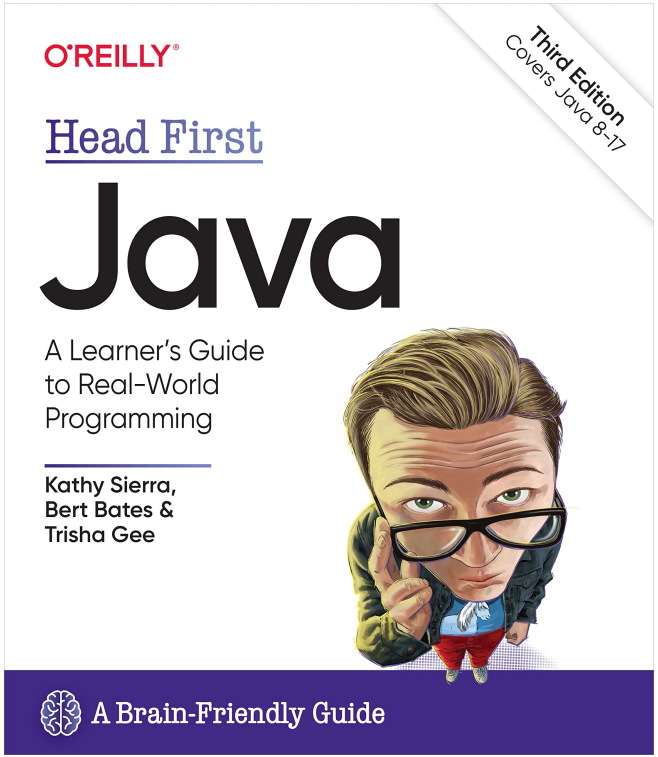
Gradle Enterprise

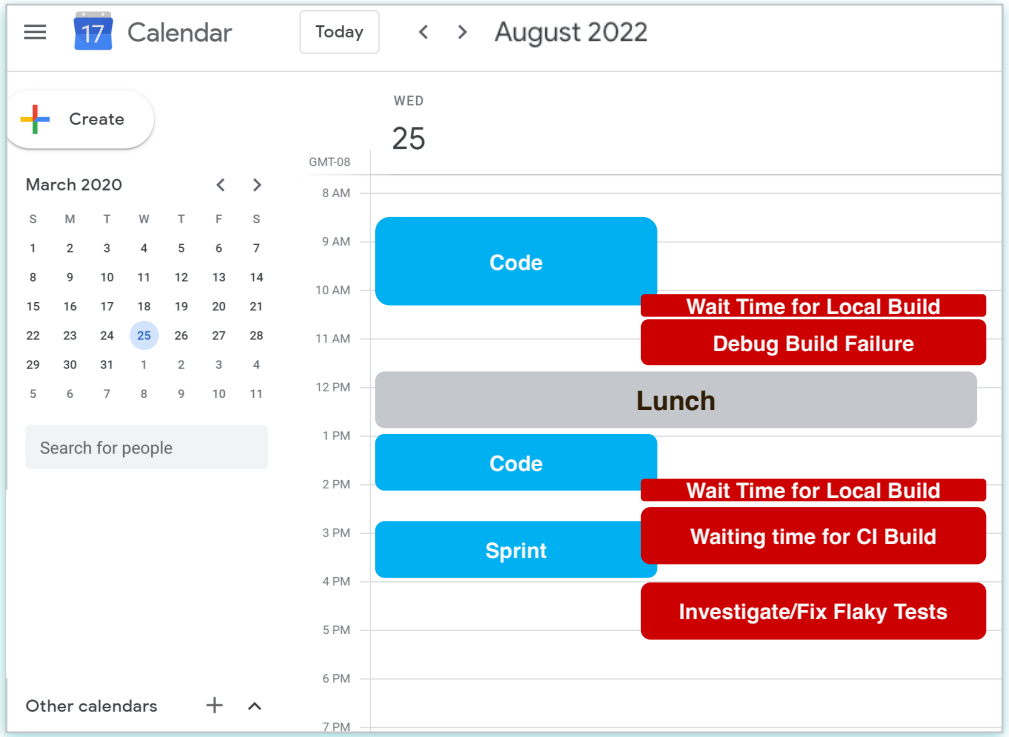
Trisha Gee

- Lead Developer Advocate
- Java Champion
- 20+ years Java experience
- …and author

https://trishagee.com/books/





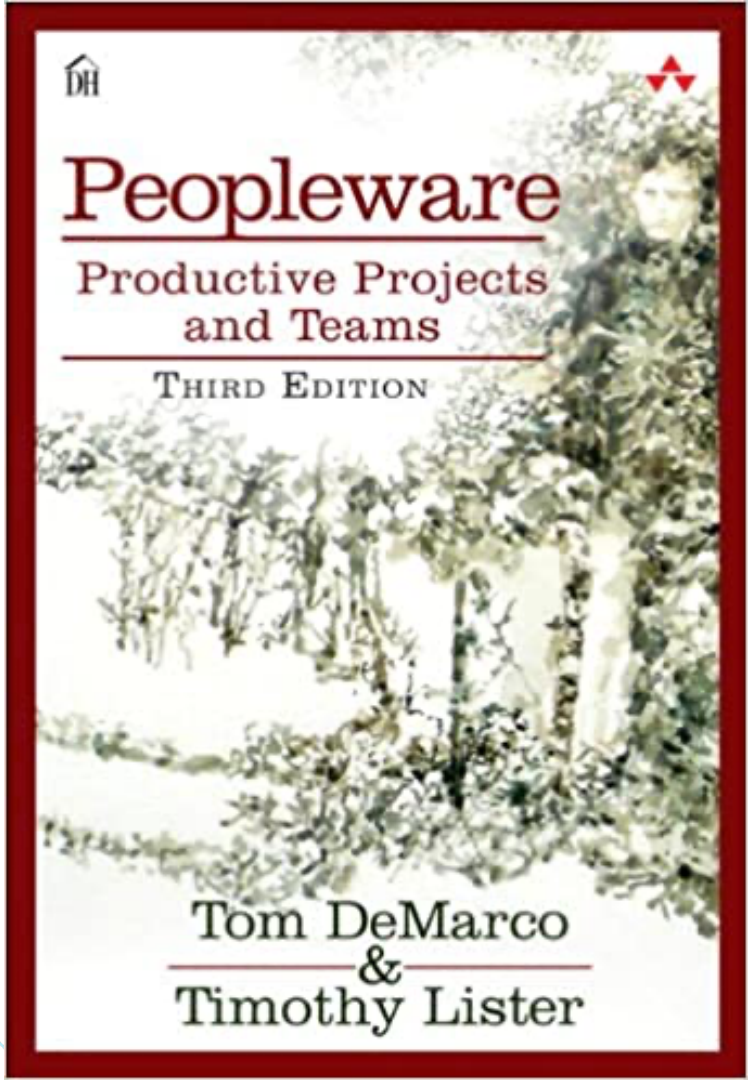


But Bottlenecks to Productivity are Everywhere

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

Pete Smoot, Software Architect, Dell Technologies

10X ENGINEER


DH
Peopleware
Productive Projects and Teams
Third Edition
Tom DeMarco
&
Timothy Lister

The “best” programmers outperformed the worst by **roughly a 10:1 ratio**

What Mattered?

What Mattered?

- Paired programmers **performed at roughly the same level**

What Mattered?

- Paired programmers **performed at roughly the same level**
- The average **difference was only 21%** between paired participants

What Mattered?

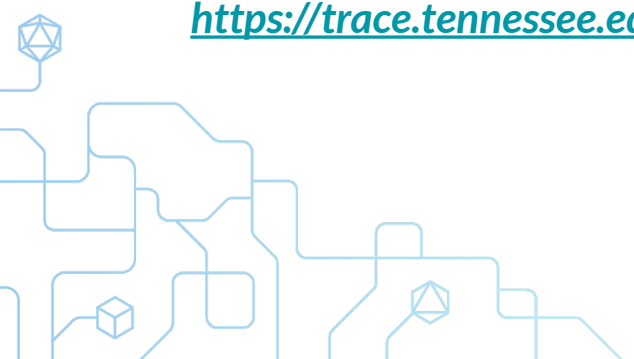
- Paired programmers **performed at roughly the same level**
- The average **difference was only 21%** between paired participants
- They didn’t work together on the task, but they **came from the same organization**

What Mattered?

- Paired programmers **performed at roughly the same level**
- The average **difference was only 21%** between paired participants
- They didn’t work together on the task, but they **came from the same organization**
- **The best organization performed 11.1x better than the worst**

“While this productivity differential among programmers is understandable, there is also a 10 to 1 difference in productivity among software organizations.”

Harlan (HD) Mills, Software Productivity in the Enterprise
https://trace.tennessee.edu/cgi/viewcontent.cgi?article=1010&context=utk_harlan



“The bald fact is that many companies provide developers with a workplace that is so crowded, noisy, and interruptive as to fill their days with frustration. That alone could explain reduced efficiency as well as a tendency for good people to migrate elsewhere.”

Peopleware: Productive Projects and Teams, Third Edition
Tom DeMarco, Tim Lister

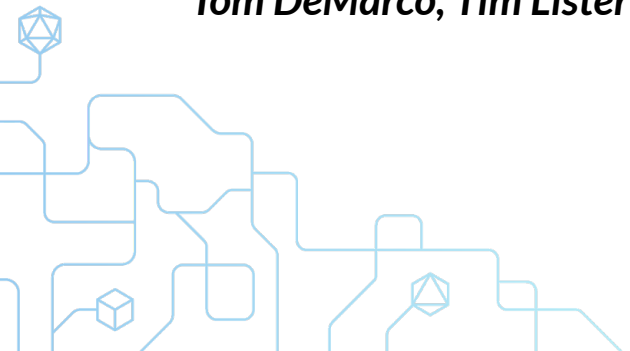
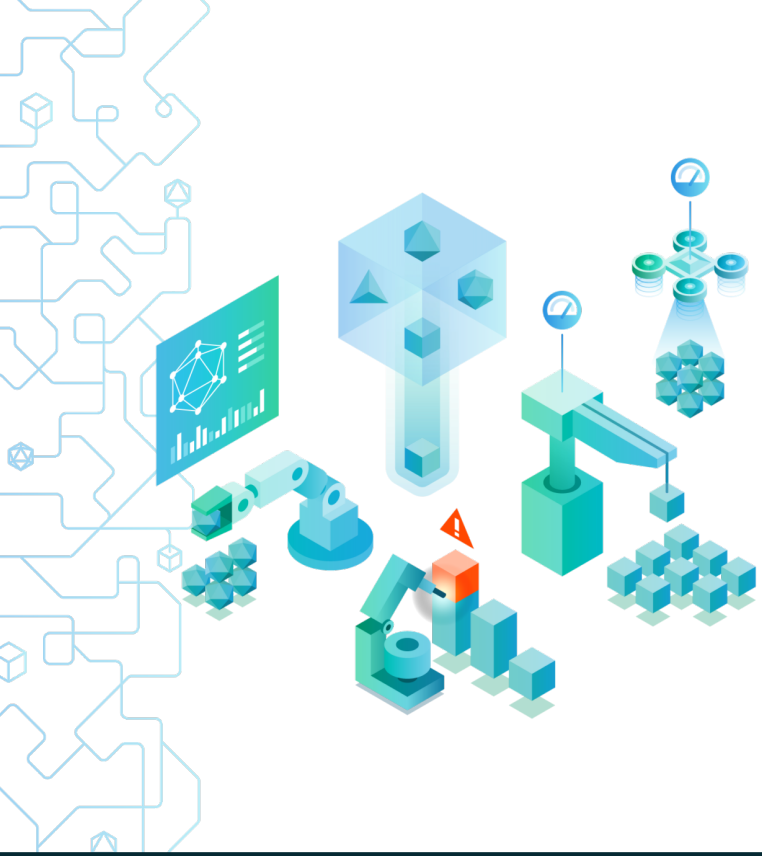


Table 8.3
Environments of the Best and Worst Performers in the Coding War Games

Environmental Factor	Those Who Performed in 1st Quartile	Those Who Performed in 4th Quartile
1. How much dedicated work space do you have?	78 sq. ft.	46 sq. ft.
2. Is it acceptably quiet?	57% yes	29% yes
3. Is it acceptably private?	62% yes	19% yes
4. Can you silence your phone?	52% yes	10% yes
5. Can you divert your calls?	76% yes	19% yes
6. Do people often interrupt you needlessly?	38% yes	76% yes

Though the phrase had not yet been coined, **increased productivity came down to developer experience.**

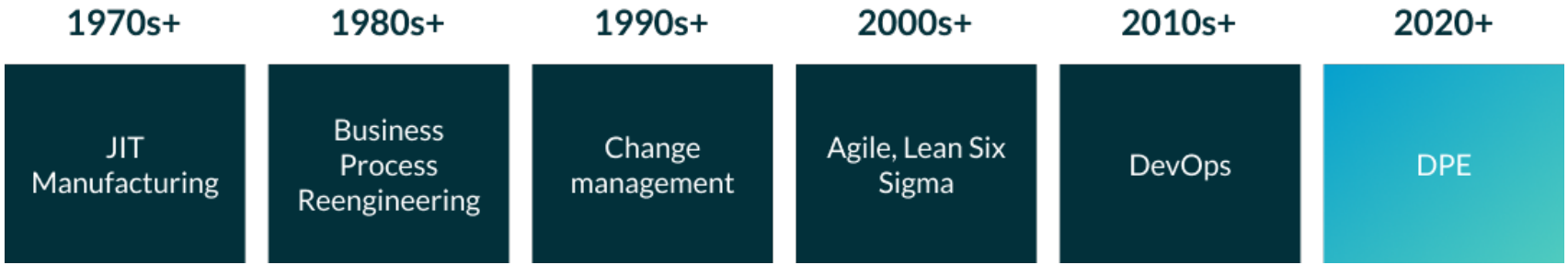


DPE is a new software development practice used by leading software development organizations to maximize developer productivity and happiness.

Gradle is Pioneering DPE

What Problems Does DPE Solve?

- This takes too long!
- This takes too long to fix
- This could have been prevented


1970s+
1980s+
1990s+
2000s+
2010s+
2020+
JIT Manufacturing
Business Process Reengineering
Change management
Agile, Lean Six Sigma
DevOps
DPE

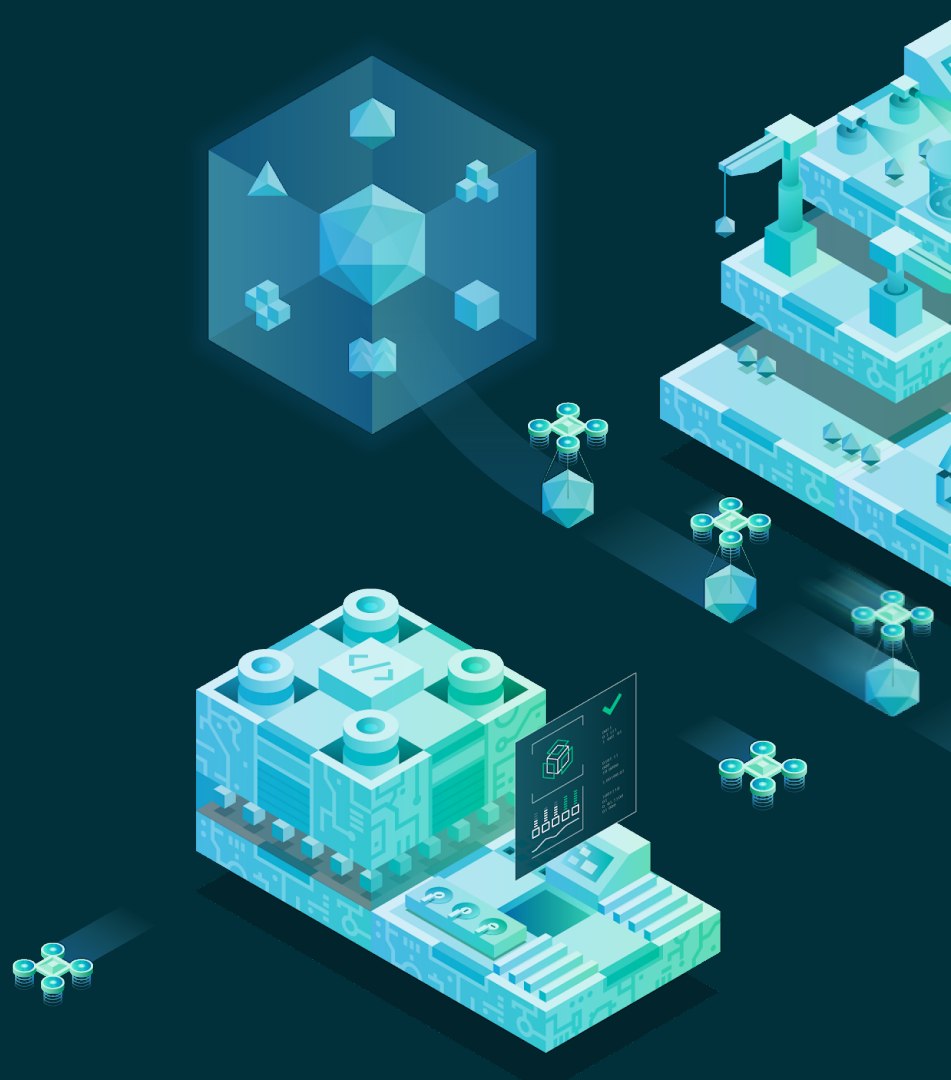
DevOps, 12-Factor, Agile, etc, have still **not captured all bottlenecks**, friction, and obstacles to throughput

Many are **hiding in plain sight**, in the developer experience itself

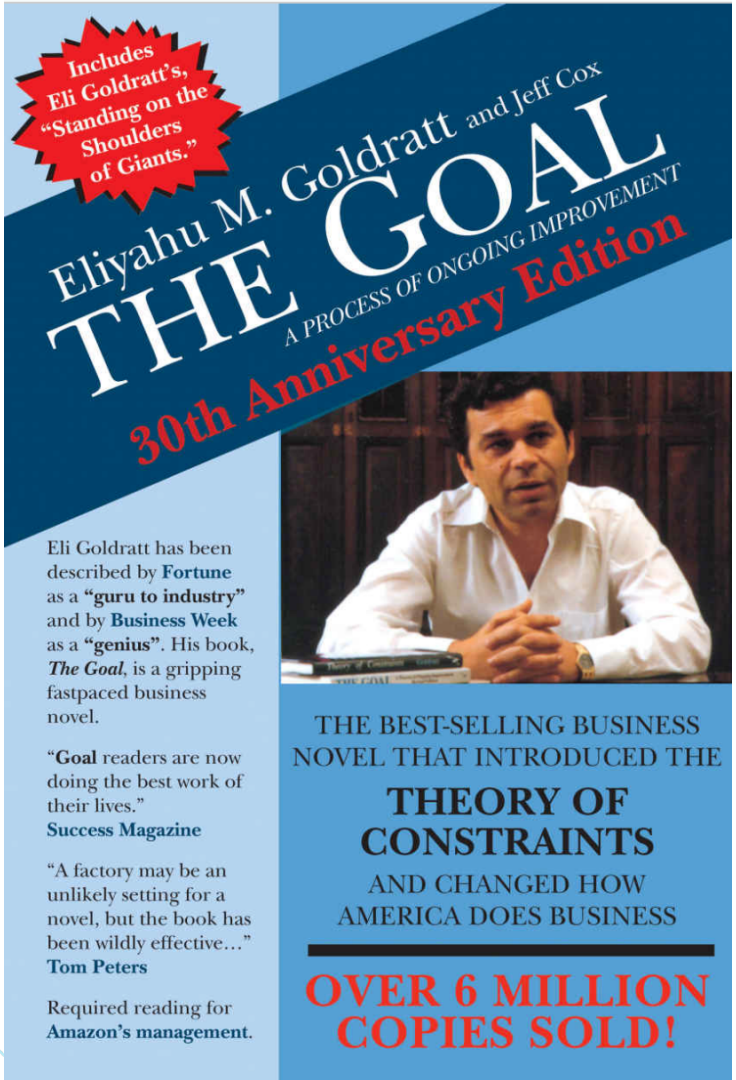
A 10x organization should be **reducing build and test feedback times** and **improving the consistency and reliability** of builds

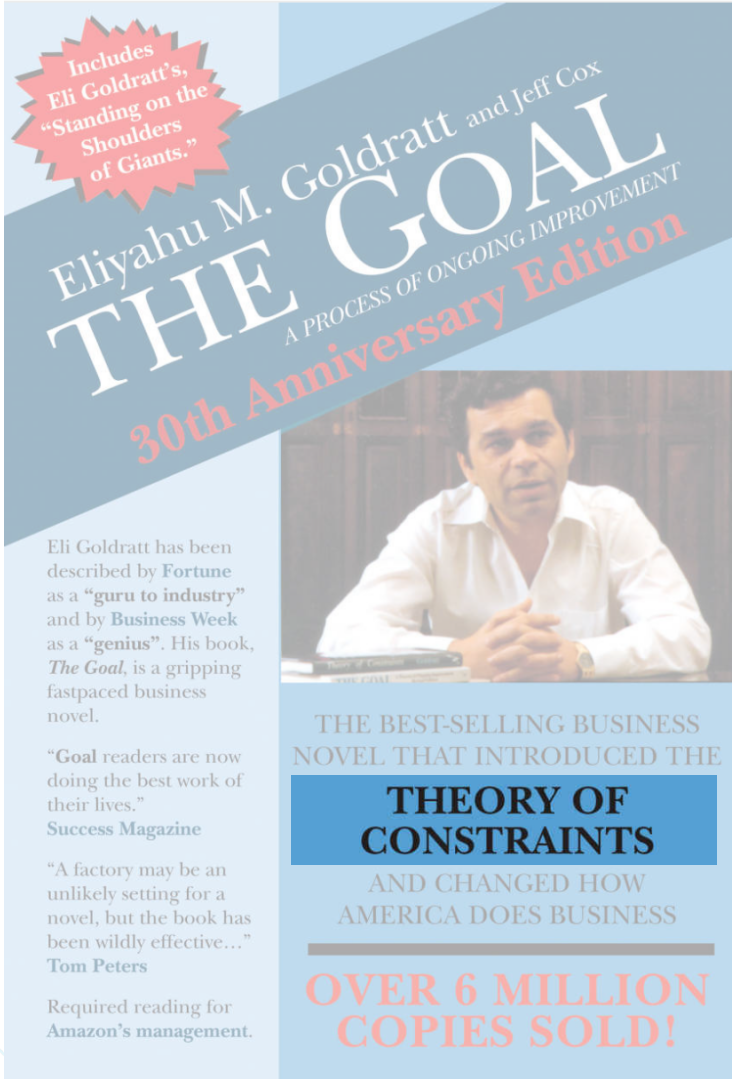
Pain Point:

Waiting for Builds & Tests to Complete



Are you tracking local build and test times?


Includes Eli Goldratt's, "Standing on the Shoulders of Giants."
Eliyahu M. Goldratt and Jeff Cox
THE GOAL
A PROCESS OF ONGOING IMPROVEMENT
30th Anniversary Edition
Eli Goldratt has been described by Fortune as a "guru to industry" and by Business Week as a "genius". His book, The Goal, is a gripping fastpaced business novel.
"Goal readers are now doing the best work of their lives."
Success Magazine
"A factory may be an unlikely setting for a novel, but the book has been wildly effective…"
Tom Peters
Required reading for Amazon's management.
THE BEST-SELLING BUSINESS NOVEL THAT INTRODUCED THE
THEORY OF CONSTRAINTS
AND CHANGED HOW AMERICA DOES BUSINESS
OVER 6 MILLION COPIES SOLD!

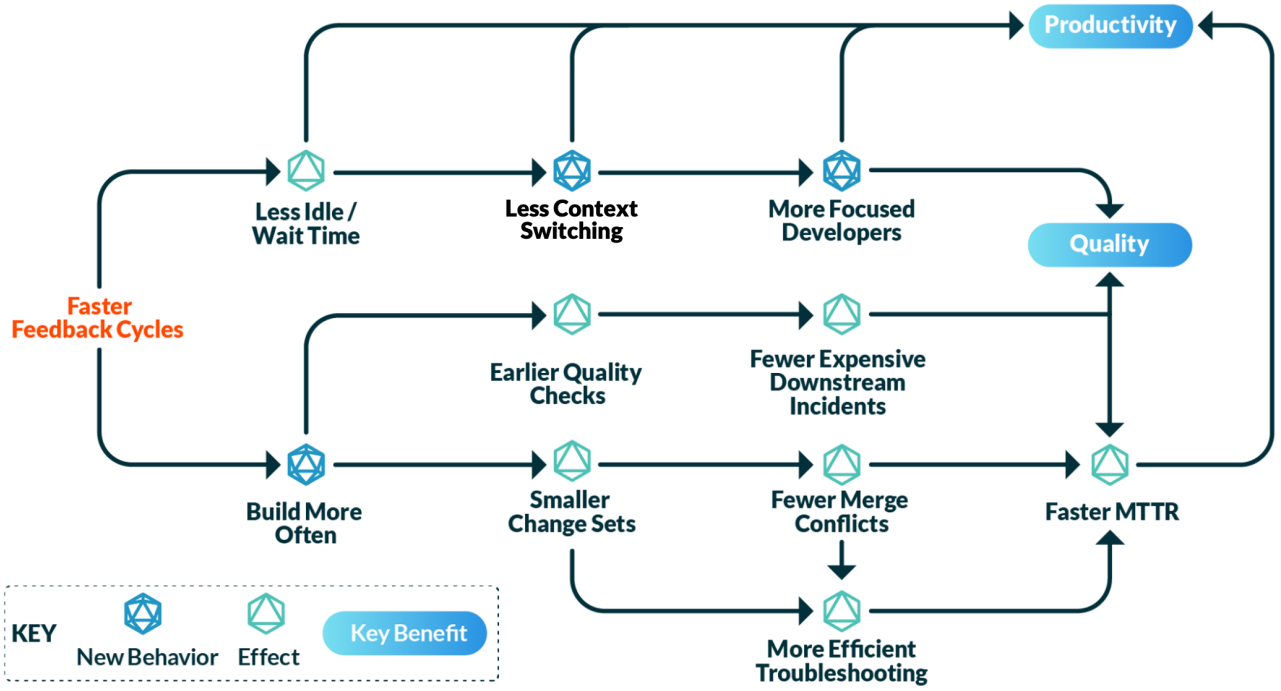

Includes Eli Goldratt's, "Standing on the Shoulders of Giants."
Eliyahu M. Goldratt and Jeff Cox
THE GOAL
A PROCESS OF ONGOING IMPROVEMENT
30th Anniversary Edition
Eli Goldratt has been described by Fortune as a "guru to industry" and by Business Week as a "genius". His book, The Goal, is a gripping fastpaced business novel.
"Goal readers are now doing the best work of their lives."
Success Magazine
"A factory may be an unlikely setting for a novel, but the book has been wildly effective…"
Tom Peters
Required reading for Amazon's management.
THE BEST-SELLING BUSINESS NOVEL THAT INTRODUCED THE
THEORY OF CONSTRAINTS
AND CHANGED HOW AMERICA DOES BUSINESS
OVER 6 MILLION COPIES SOLD!

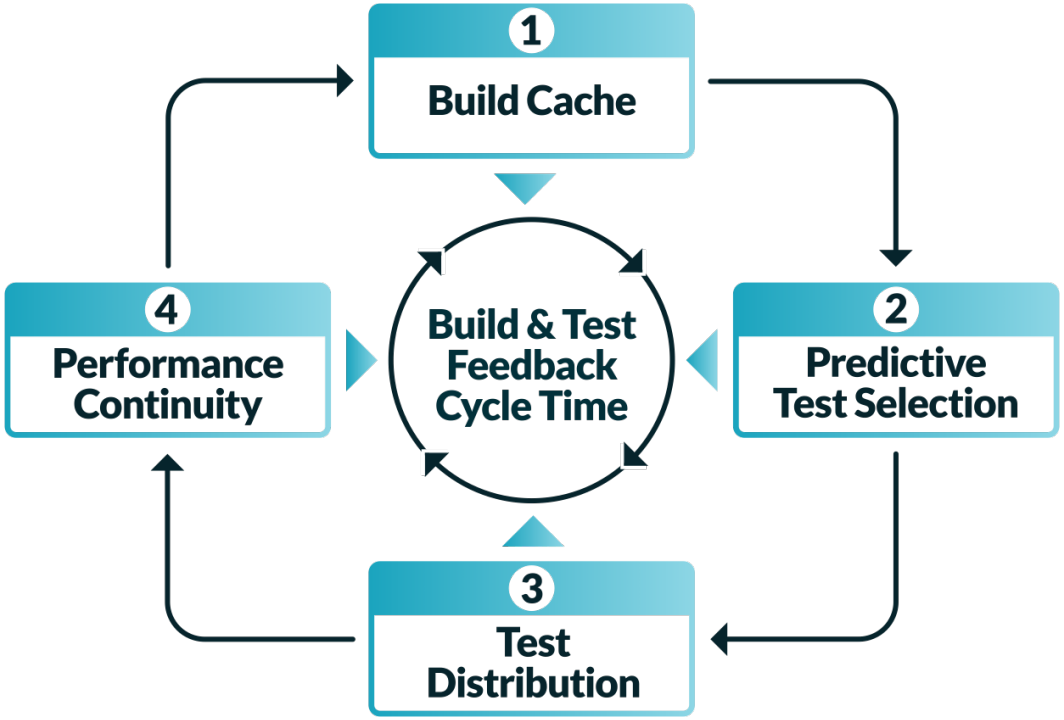
The only initiatives that will positively impact performance are ones which **increase throughput while simultaneously decreasing cost**

Faster Builds Improve Creative Flow

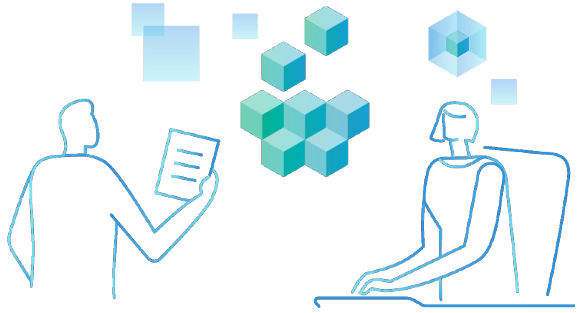
	Team 1	Team 2
No. of Devs	11	6
Build Time	4 mins	1 mins
No. of local builds	850	1010

Very Fast Feedback Is Important

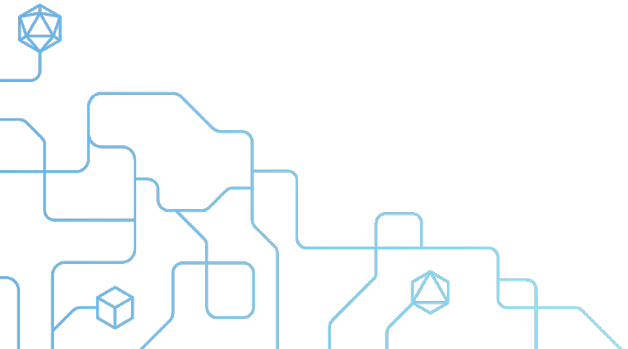




Solution: Acceleration Technologies



Build Caching *Speeds up Builds and Tests*



Build Caching

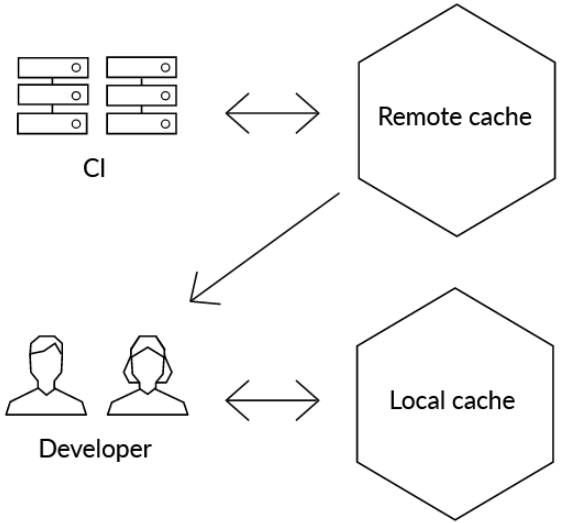
- Introduced to the Java world by Gradle in 2017
- Used by leading technology companies like Google and Facebook
- Can support **both user local and remote caching** for distributed teams

Build Caching

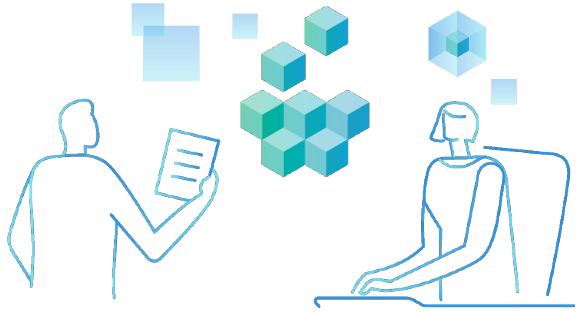
When the inputs have not changed, the **outputs can be reused** from a previous run.

Demo: Build Cache for Maven and Gradle

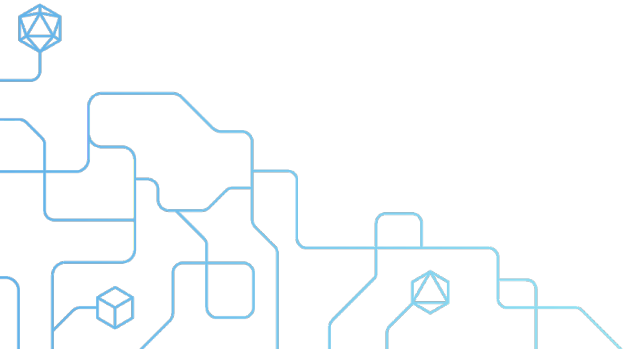
Remote Build Cache



- Shared among different machines
- Speeds up development for the whole team
- Reuses build results among CI agents/jobs and individual developers



Test Distribution *Parallelizes Test Execution*



Existing solutions: Single machine parallelism

Parallelism in Gradle is controlled by these flags:

`--parallel / org.gradle.parallel`

Controls project parallelism, defaults to false

`--max-workers / org.gradle.workers.max`

Controls the maximum number of workers, defaults to the number of processors/cores

`test.maxParallelForks`

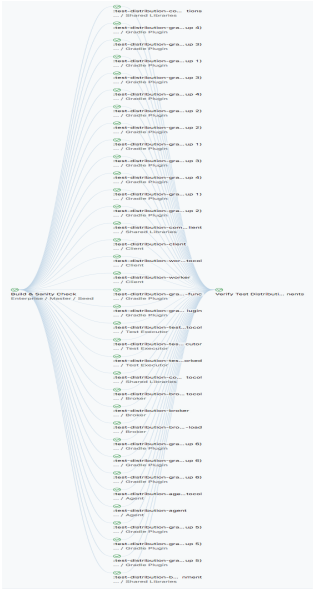
Controls how many VMs are forked by an individual test task, defaults to 1

See https://guides.gradle.org/performance/#parallel_execution for more information

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'
    }
  }
}
```

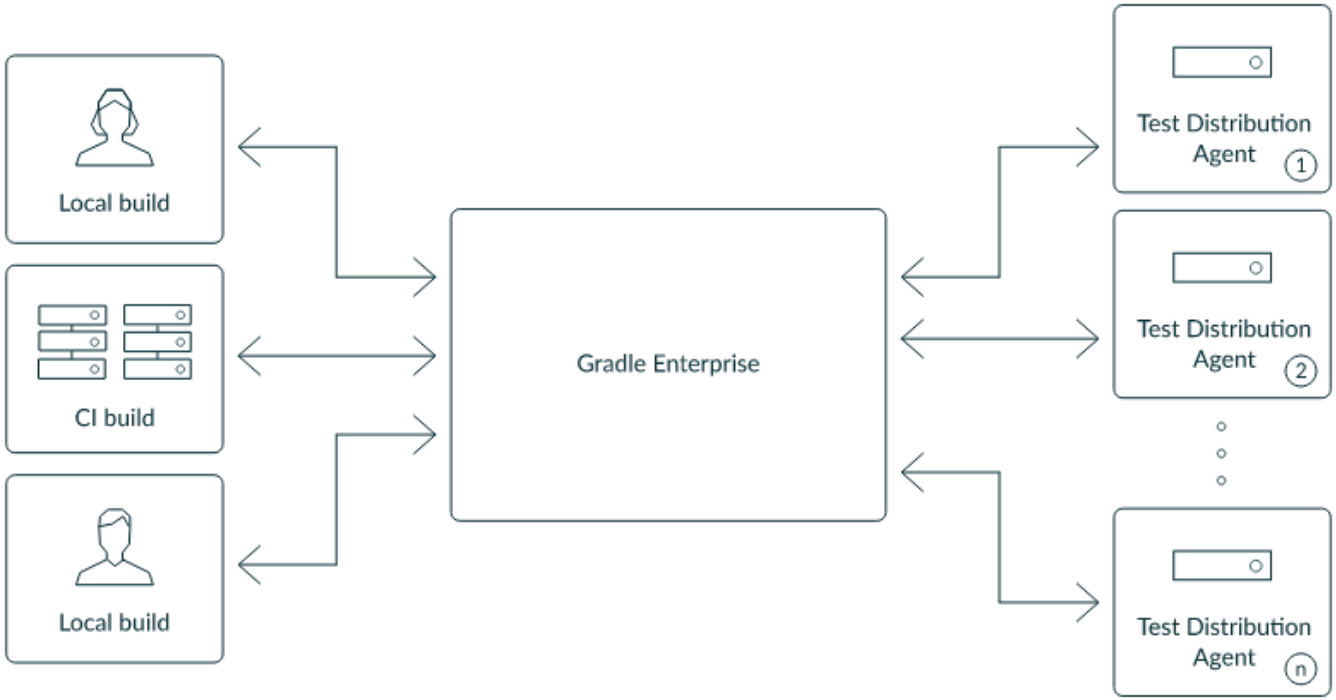


See https://builds.gradle.org/project/Gradle for an example of this strategy

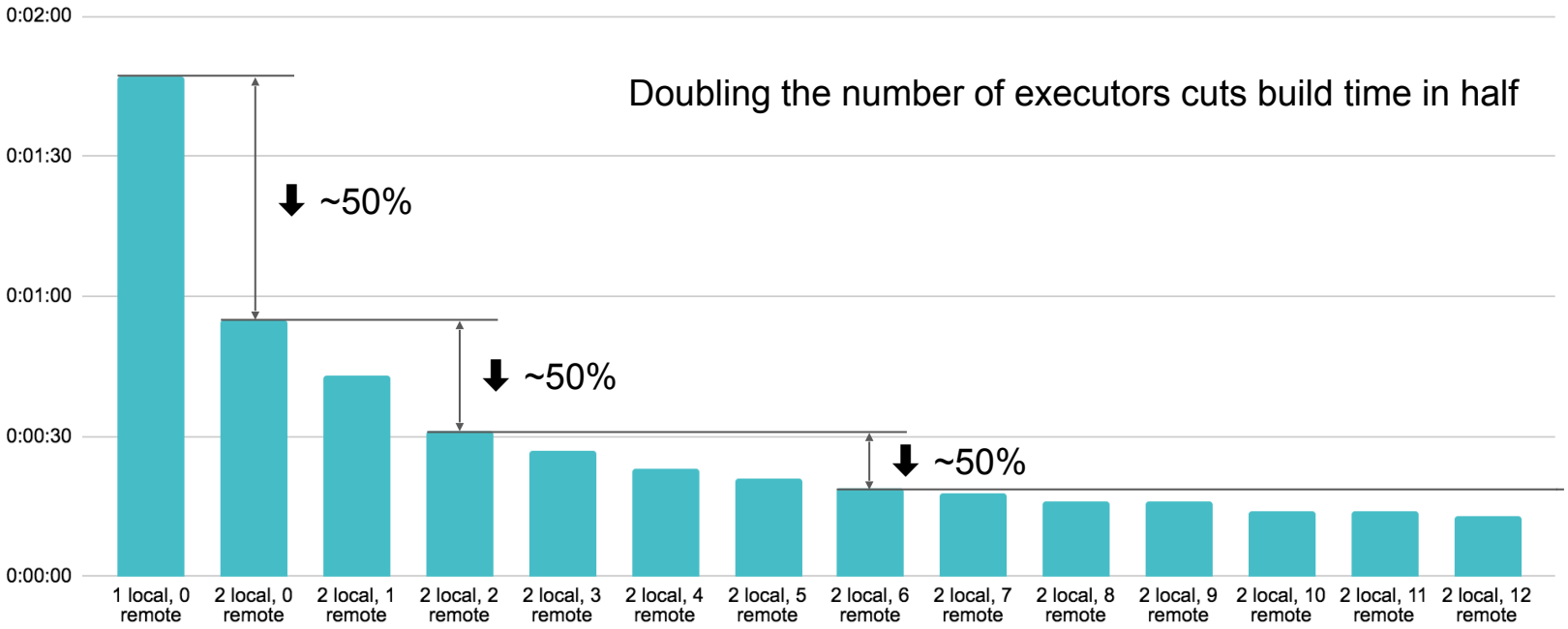
Assessment of existing solutions

- **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, requires manual setup and test partitioning, and result collection/aggregation

Test Distribution in Gradle Enterprise



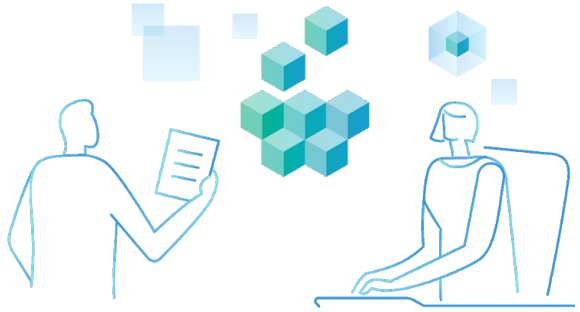
Test Distribution Results



Measurements from the demo project

NETFLIX

Netflix reduced a 62-minute test cycle time down to just under 5 minutes!



Machine learning *leads to greater efficiencies*

Predictive Test Selection

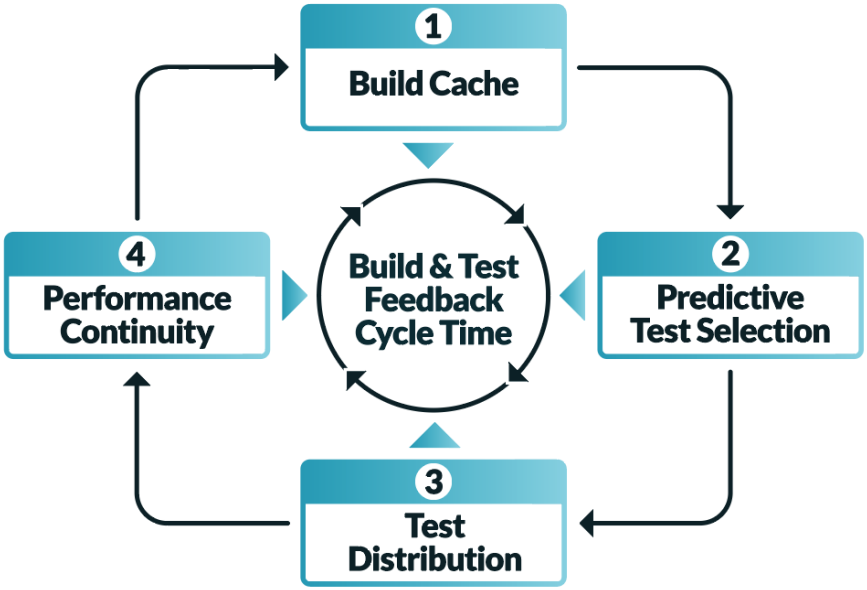
01 Instead of trying to analyze which tests could possibly be impacted by developer changes, Predictive Test Selection looks at the history of changes and what has happened to tests in the past

02 When tests complete, they can either FAIL, SUCCEED, or be FLAKY. Predictive Test Selection will predict the outcome of the test based on the history it is analyzing

03 PTS will recommend skipping tests that are successful, and will only run tests that are likely to provide valuable feedback

https://arxiv.org/pdf/1810.05286.pdf

Force multiplier when used in combination

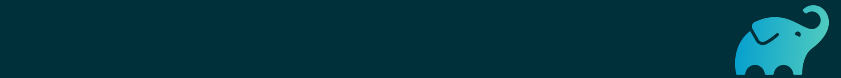


1. **Build Cache**. Avoid unnecessarily running components of builds and tests whose inputs have not changed.
2. **Predictive Test Selection**. Run only the relevant subset of test tasks likely to provide useful feedback.
3. **Test Distribution**. Speed up the execution of the necessary and relevant remaining tests by running them in parallel.
4. **Performance Continuity**. Sustain Test Distribution and other performance improvements over time with data analytic and performance profiling capabilities.

Is the build and test cycle fast enough?

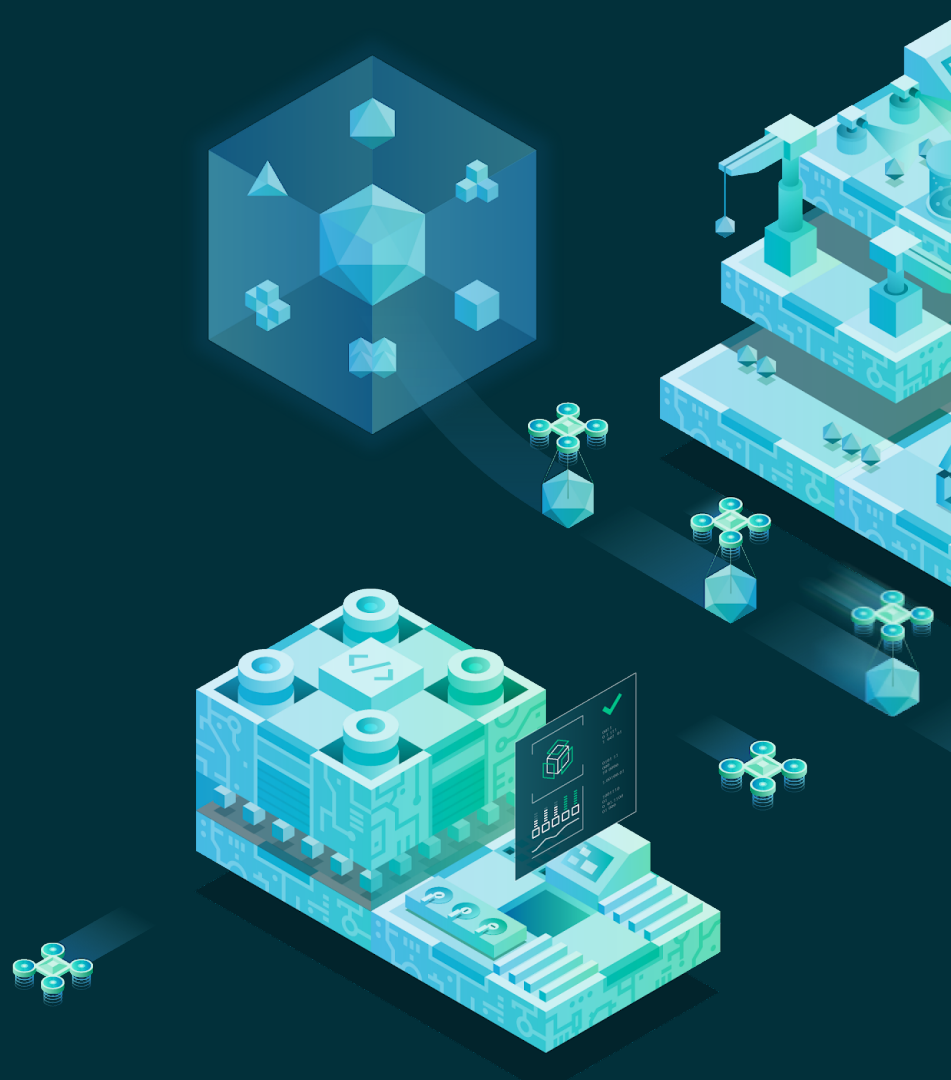
Is the build and test cycle ~~fast enough~~?

Is the build and test cycle **as fast as it can possibly be?**



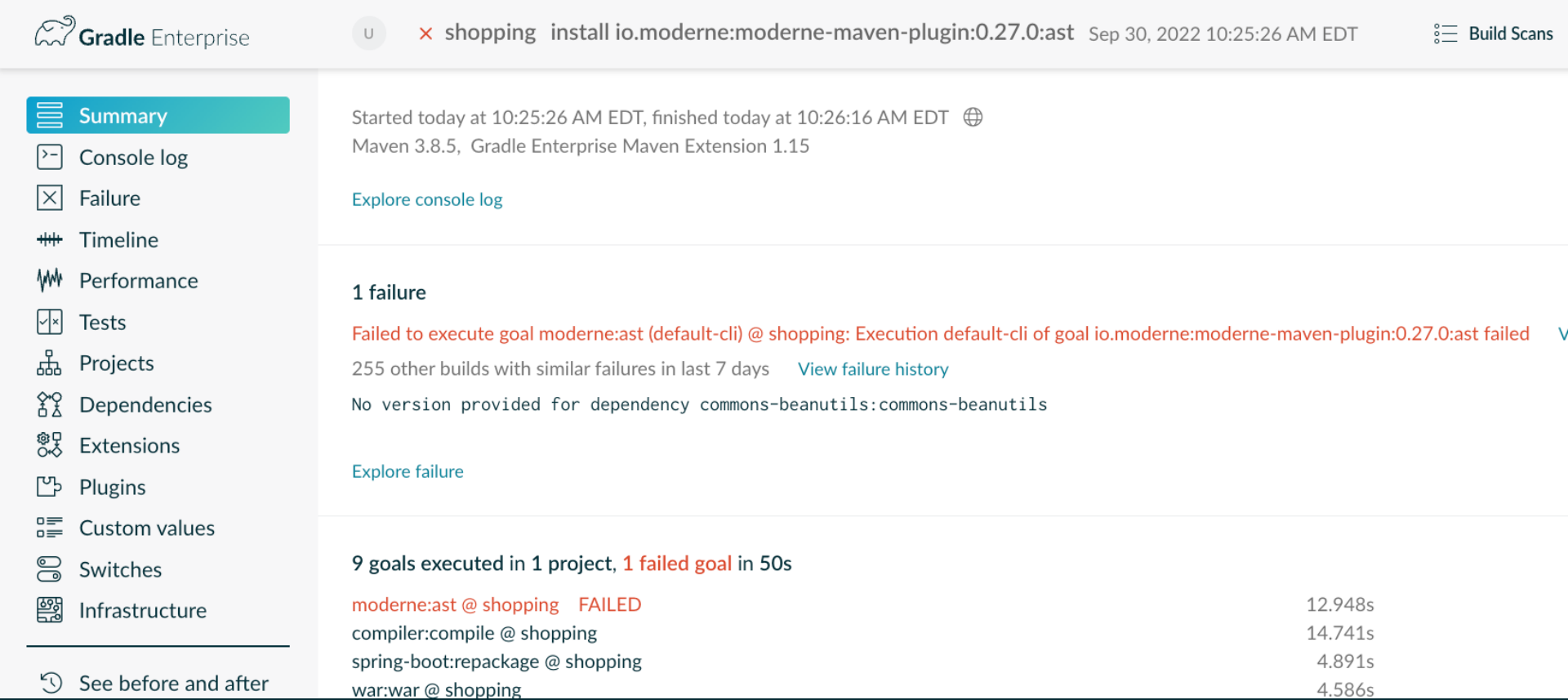
Pain Point:

Inefficient troubleshooting of broken builds



“ You can observe a lot by just watching.”

Yogi Berra, Catcher and Philosopher



Build Scan: scans.gradle.com

Learn more

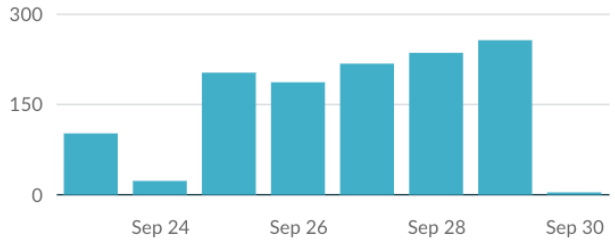
https://bit.ly/grdl-scan

Failures › Execution failed for task *
> There were failing tests. See the report at:

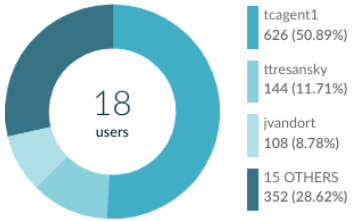
Non-verification | Verification | All failures

Builds with matching failures

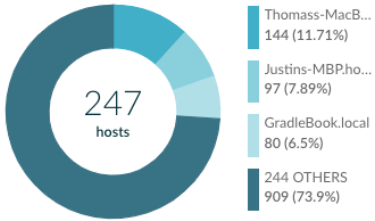
1.23K (2% of 59K total builds)



Affected users



Affected hosts



Top Tags

CACHED	100%
CI	54%
PTS	52%
Check	50%
LOCAL	46%

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	jvandort	Justins-MBP.home

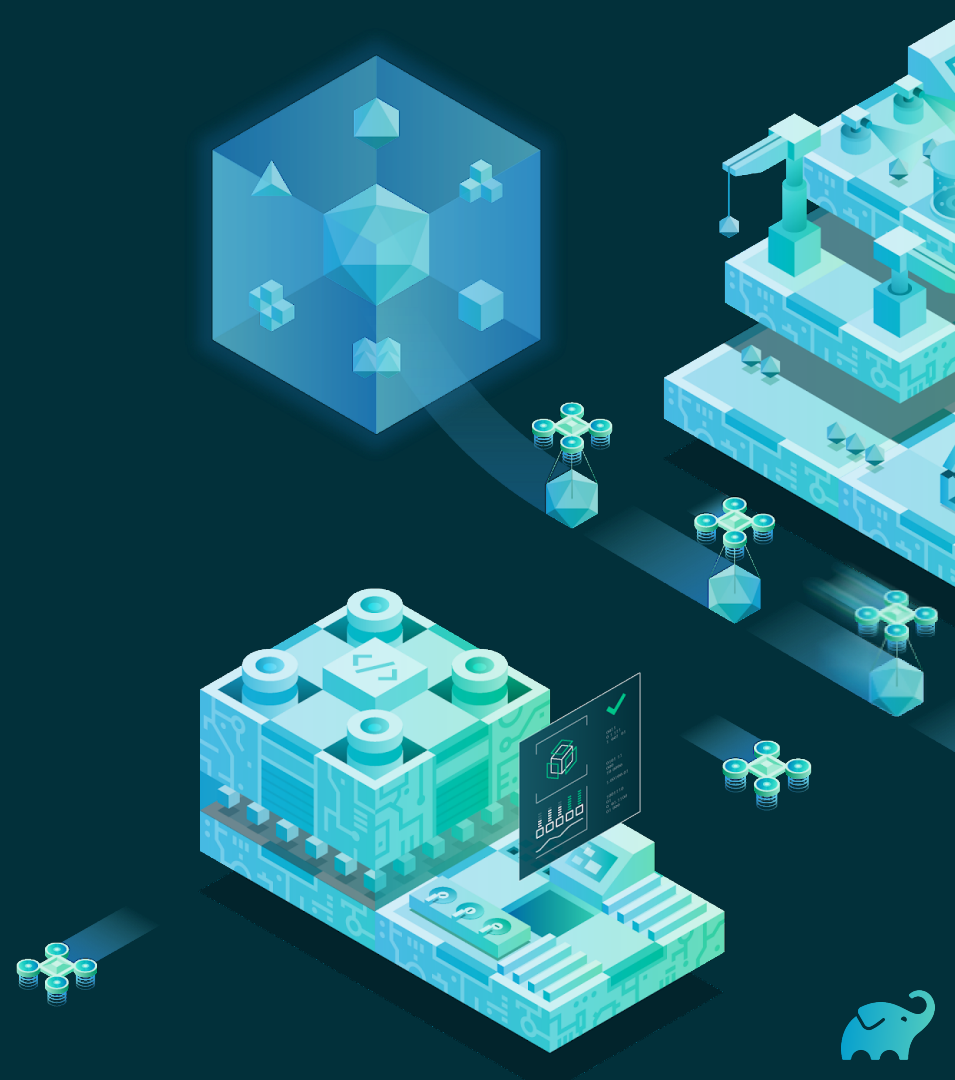
CACHED | LOCAL | IDEA | dirty

Execution failed for task ':core:embeddedIntegTest'.
> There were failing tests. See the report at: file:///Users/jvandort/work/gradle/subprojects/core/build/reports/tests/embeddedIntegTest/index.html

DPE Organizations Track Failure Rates

Pain Point:

Flaky Tests & Other Avoidable Failures



Flaky builds and tests are maddening

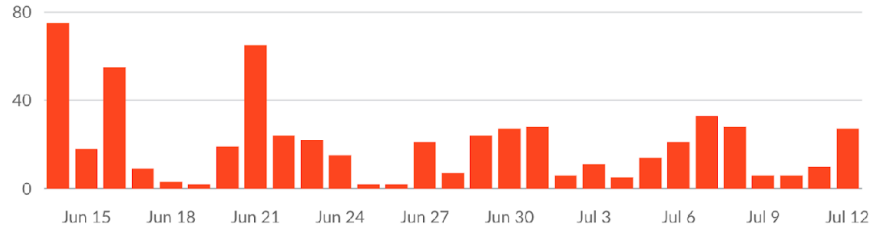
The test is flaky. What do you do now?

- Try it again
- Re-run it
- Re-run it again
- Ignore it and approve PR
- All of the above

Tests › Find tests by class name (use * to match zero or more characters)

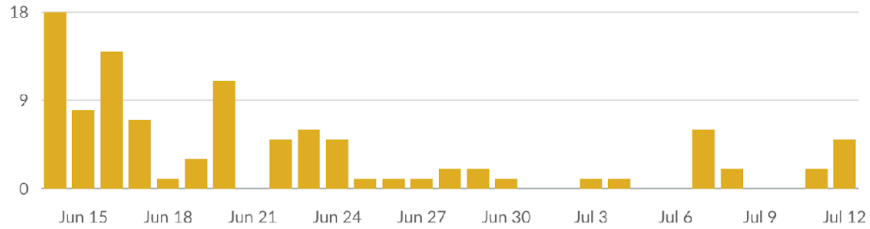
Builds with failed tests

585 (20% of 2.92K builds that executed tests)



Builds with flaky tests

103 (4% of 2.92K builds that executed tests)



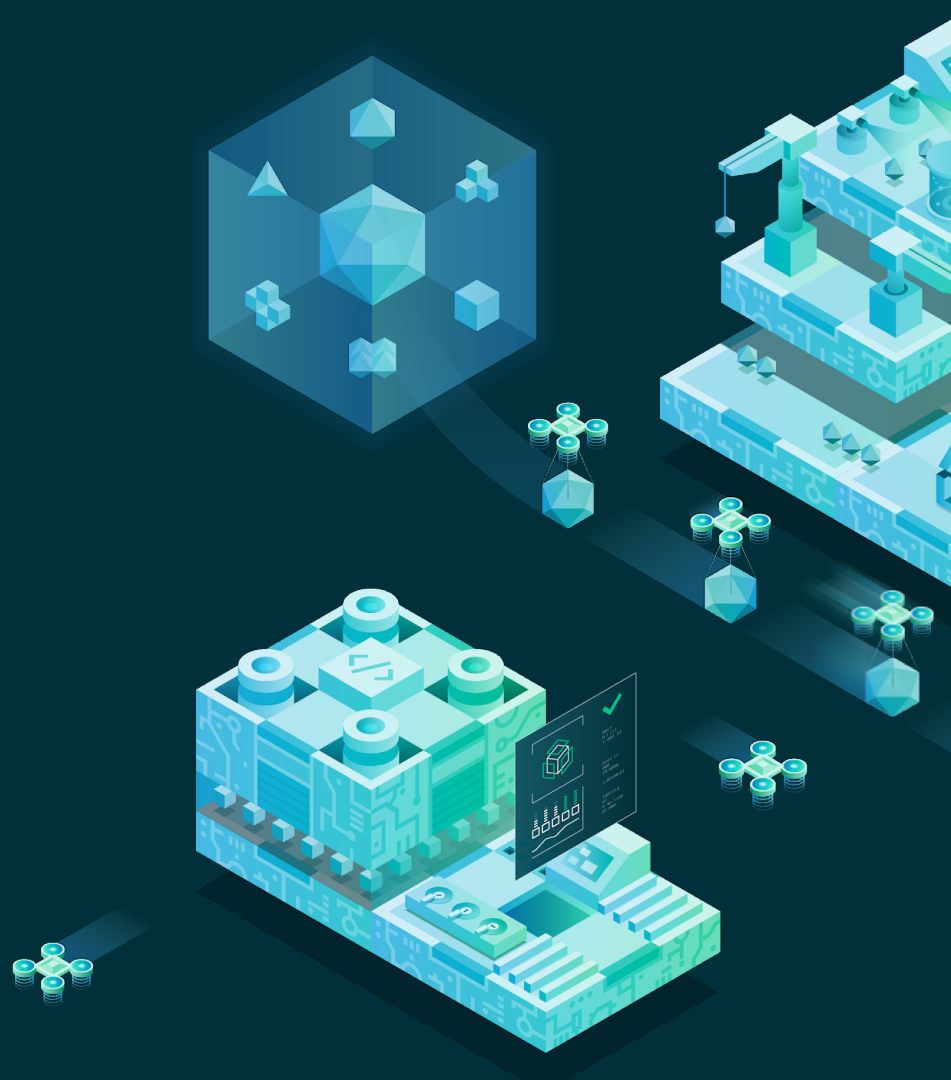
Test classes by failure count (top 50)

Name	Outcome trend	Failed	Flaky	Passed	Mean execution time
org.springframework.boot.image.paketo.PaketoBuilderTests		70 (16%)	3 (1%)	377 (84%)	6 min 41 sec
org.springframework.boot.loader.JarLauncherTests		57 (17%)	0 (0%)	269 (83%)	6.8 sec
org.springframework.web.context.request.ServletWebRequestHttpMethodsTests		32 (12%)	0 (0%)	233 (88%)	1.0 sec

Identify and Track Flaky Tests

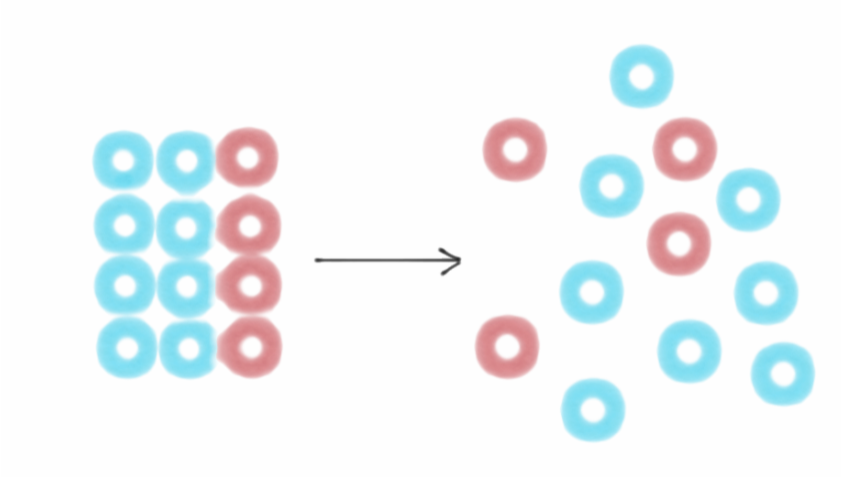
Pain Point:

No Metric/KPI Observability



Without focus, problems can sneak back in

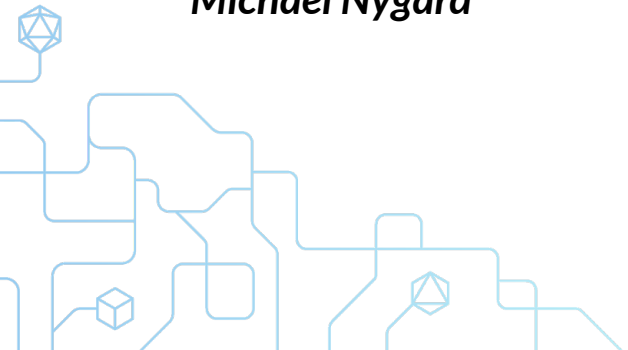
Continuous Improvement: It doesn't really matter what you improve as long as you are constantly improving something, because…



…entropy denotes that if you aren't doing anything, **you're always getting worse.**

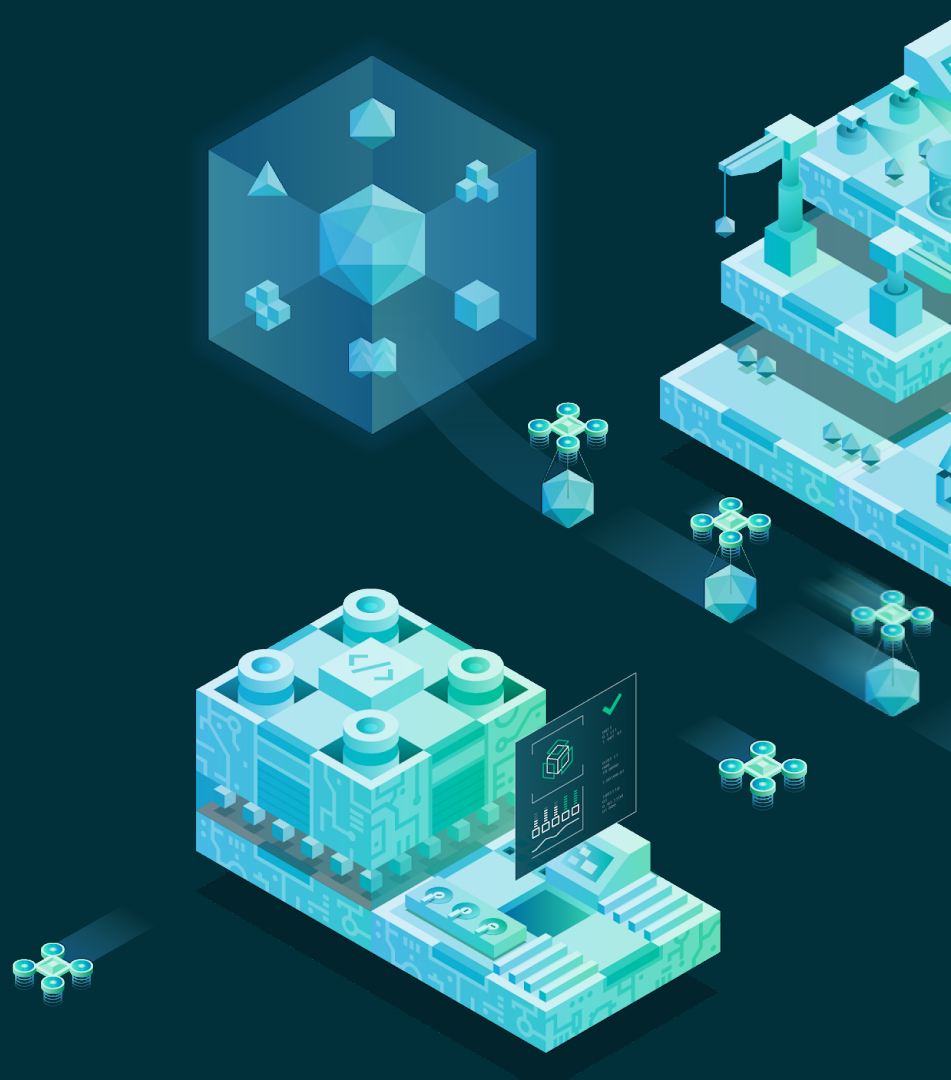
“The tools, services, and environments that developers need to do their jobs should be treated with production-level SLAs. The development platform is the production environment for the job of creating software”

Release It! Second Edition
Michael Nygard

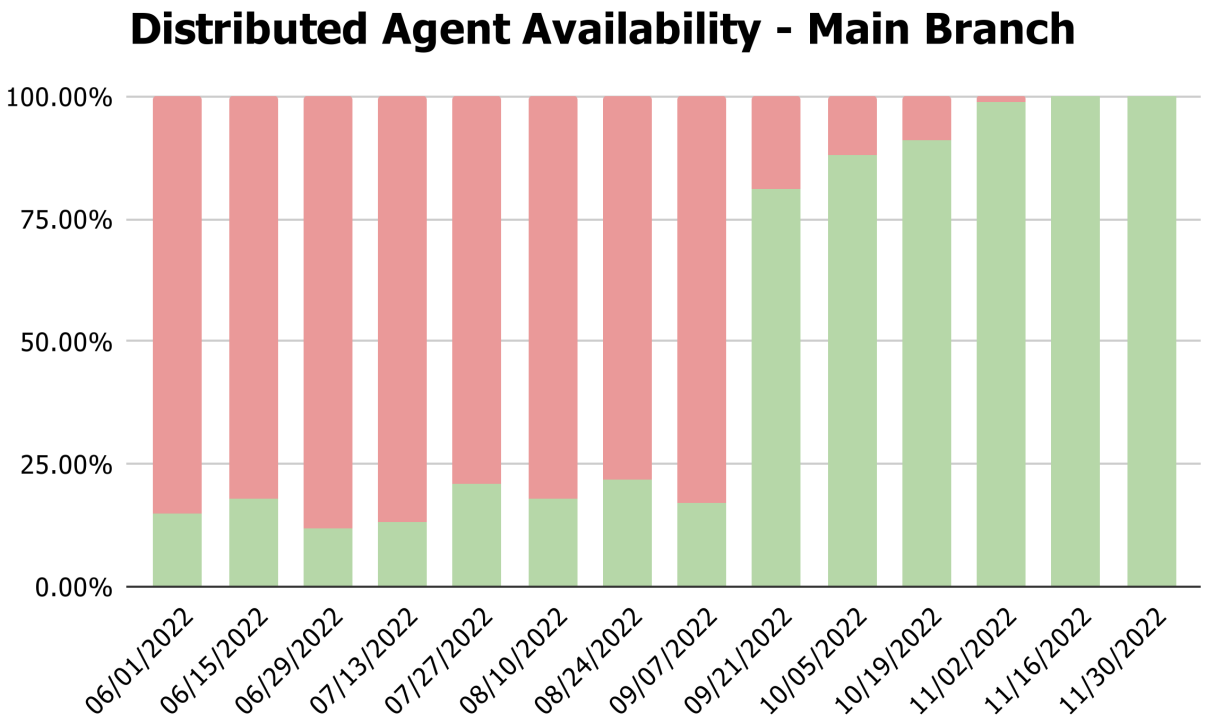


Pain Point:

Inefficient use of CI Resources



All Of This Will Improve CI



In Summary

In Summary

- 10x Developers might be a myth, but **10x Organisations are real**

In Summary

- 10x Developers might be a myth, but **10x Organisations are real**
- Developer Productivity is **deeply linked to Developer Experience**

In Summary

- 10x Developers might be a myth, but **10x Organisations are real**
- Developer Productivity is **deeply linked to Developer Experience**
- If you do nothing about productivity, **life will get worse**

In Summary

- 10x Developers might be a myth, but **10x Organisations are real**
- Developer Productivity is **deeply linked to Developer Experience**
- If you do nothing about productivity, **life will get worse**
- Fast feedback, efficient troubleshooting, and reliable cycles are key

In Summary

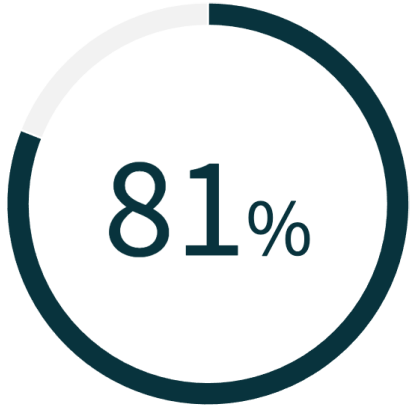
- 10x Developers might be a myth, but **10x Organisations are real**
- Developer Productivity is **deeply linked to Developer Experience**
- If you do nothing about productivity, **life will get worse**
- Fast feedback, efficient troubleshooting, and reliable cycles are key
- Start with **observation**, and then **take action** on data

In Summary

- 10x Developers might be a myth, but **10x Organisations are real**
- Developer Productivity is **deeply linked to Developer Experience**
- If you do nothing about productivity, **life will get worse**
- Fast feedback, efficient troubleshooting, and reliable cycles are key
- Start with **observation**, and then **take action** on data
- **Proactively solve problems for the whole team**

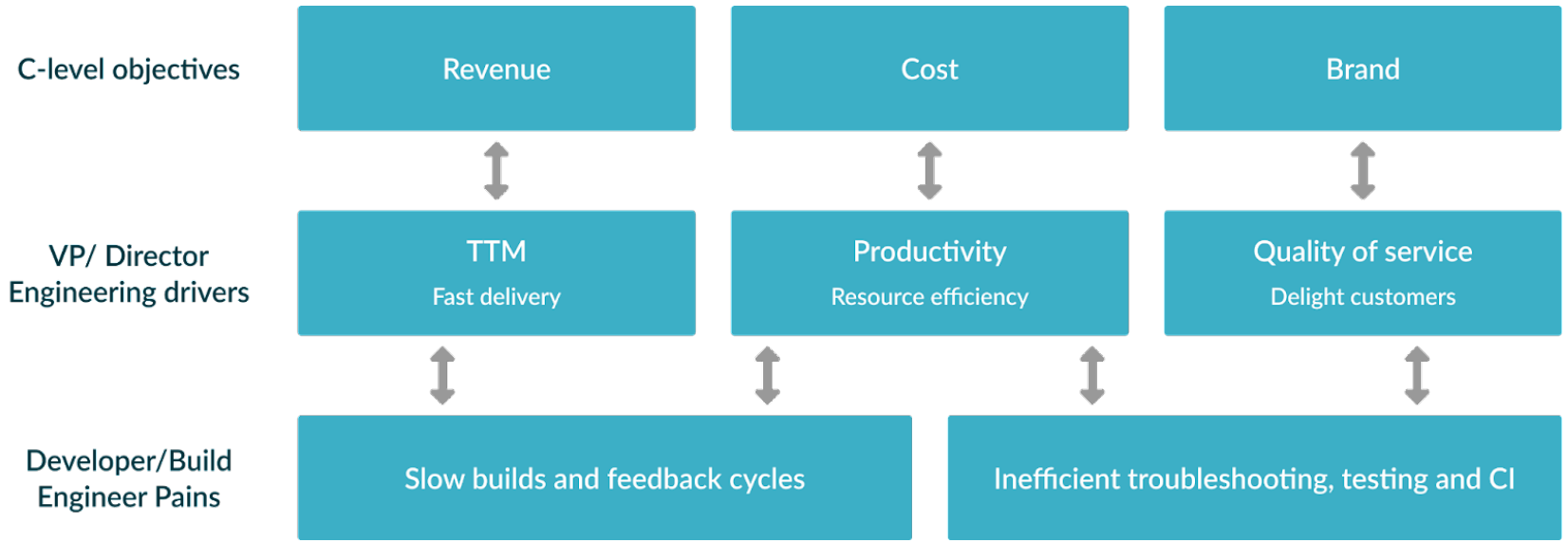
DPE Fosters Developer Joy

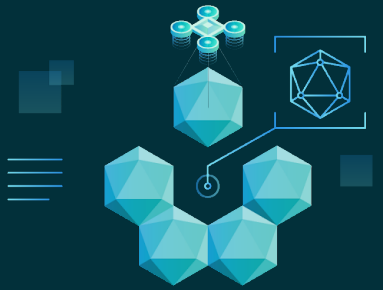
81% of surveyed IT professionals would Agree that DPE's impact on their toolchain makes their job more enjoyable.



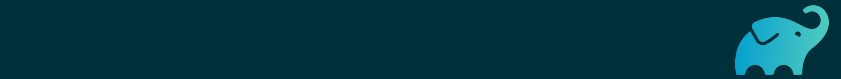
Source: TechValidate. TVID: 066-EEE-DB1

DPE Transforms Every Business Layer





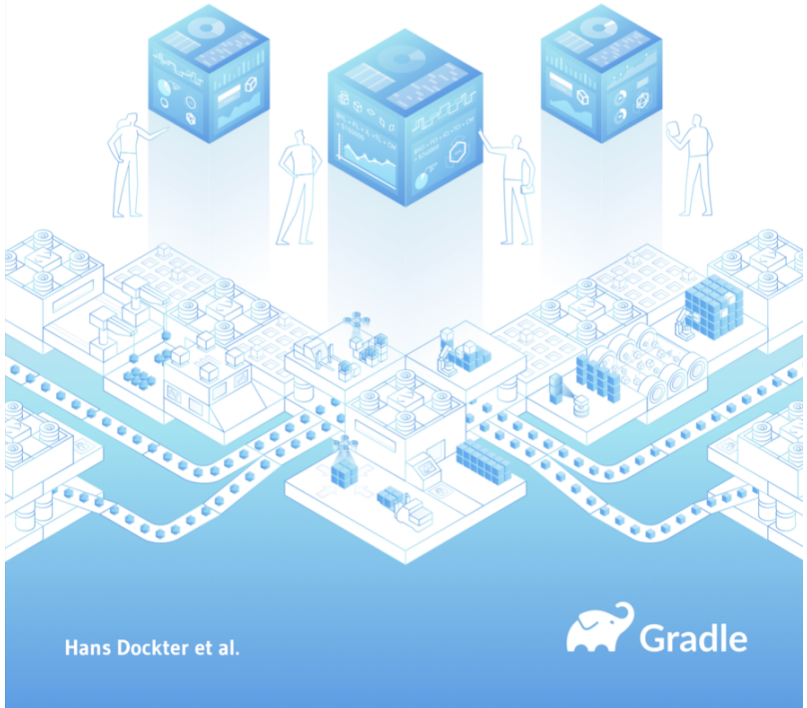
Next Steps



Build speed challenge

https://bit.ly/speed-build

There's a Book for This



Gradle Enterprise

Solutions Pricing Customers Learning Center Company Gradle.org Free Training

Customers

OSS Projects Revved Up by Gradle Enterprise

Gradle is proud to support key open-source projects with FREE instances of Gradle Enterprise.

Gradle Enterprise is quickly emerging as a *de facto* standard tool for build and test data analytics and as a source of acceleration technology for many of the most important open source projects. Here is a roundup of OSS projects that rely on Gradle Enterprise to improve build and test feedback cycle times and make troubleshooting more efficient by combining root cause analysis data with failure analytics.

Spring

spring

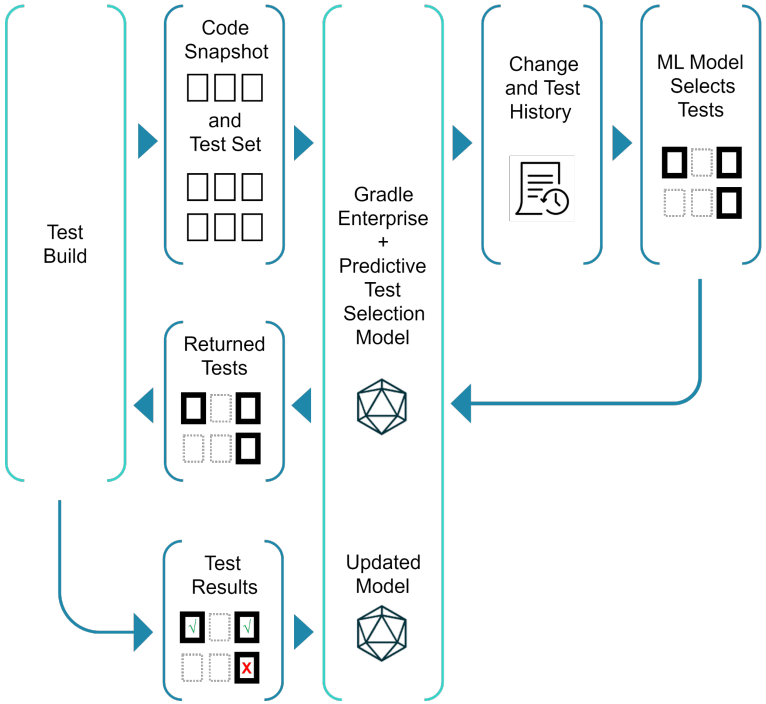
Spring is the world's most popular Java framework for making Java programming quicker, easier, and safer for everybody. Gradle Enterprise is being used by the Spring Boot, Spring Framework, and Spring Security projects. After the Spring Boot project migrated from Maven to the Gradle Build Tool and further optimized the Build Cache effectiveness with Gradle Enterprise, CI builds now take roughly 20 minutes on average, 3-4 times faster than before. Local builds are taking an average of 2.5 minutes, which is 20-30 times faster than before. The team has also started looking at Gradle Enterprise Failure Analytics to address flaky tests and other avoidable failures.

SCAN ME

https://bit.ly/dpe-4me

Thank you!

How it works…



1. When a test run starts, the build tool submits a test input snapshot and test set to a machine learning model.
2. PTS automatically develops a test selection strategy by learning from historical code changes and test outcomes from your Build Scan data to predict a subset of relevant tests, which are then executed by your build.
3. Code change and test results data are processed immediately after a Build Scan is uploaded to PTS and updates the test selection strategy based on new results.