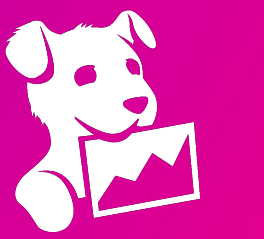
Cfgmgmtcamp 2026

CI/CD Observability, Metrics and DORA: Shifting Left and Cleaning Up!

February 2026

Peter Souter
Sr. Solutions Engineer, Datadog



DATADOG

Introductions

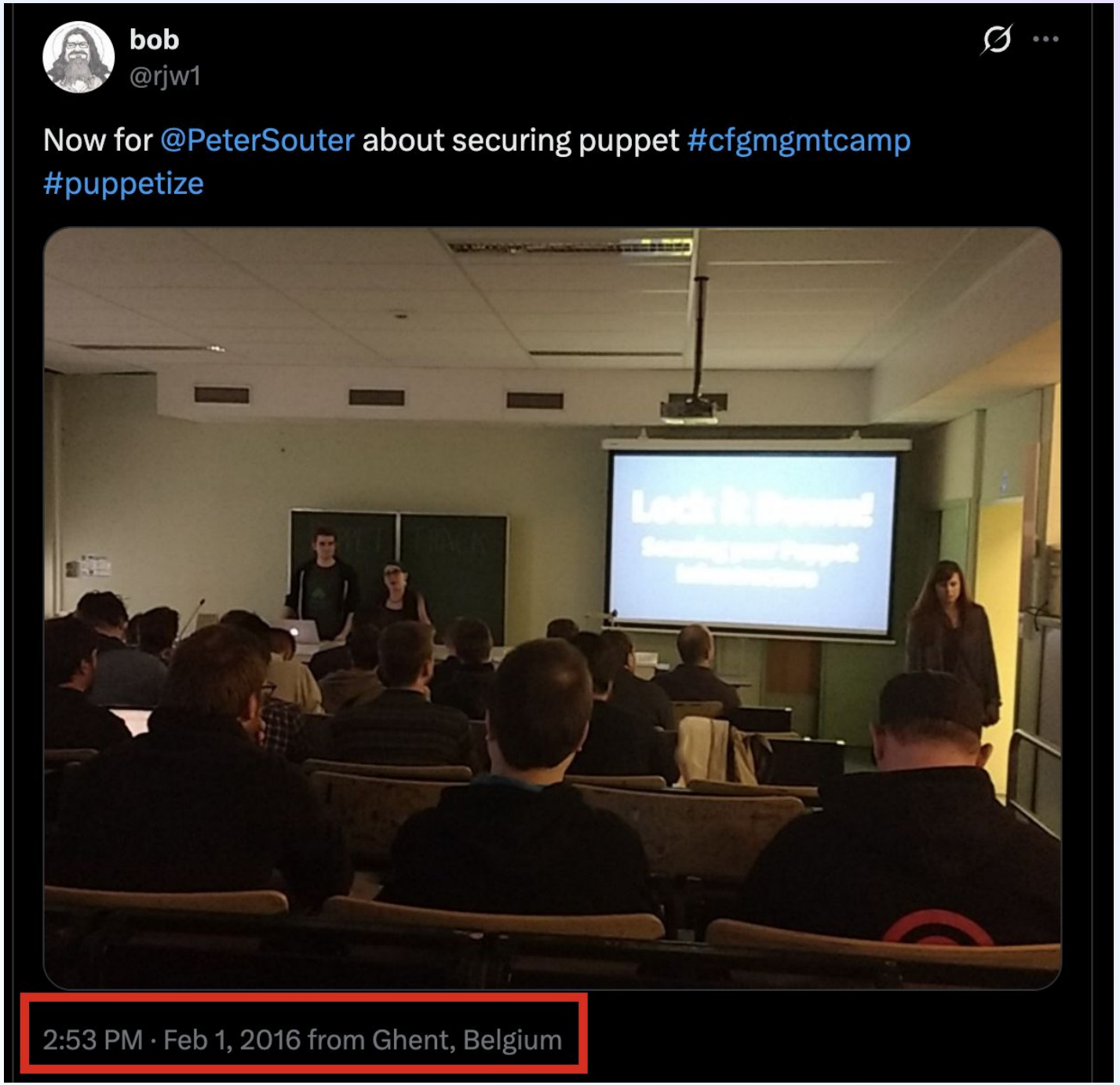
Sr. Solutions Engineer

Based in London, UK

Started at Datadog… March 2021

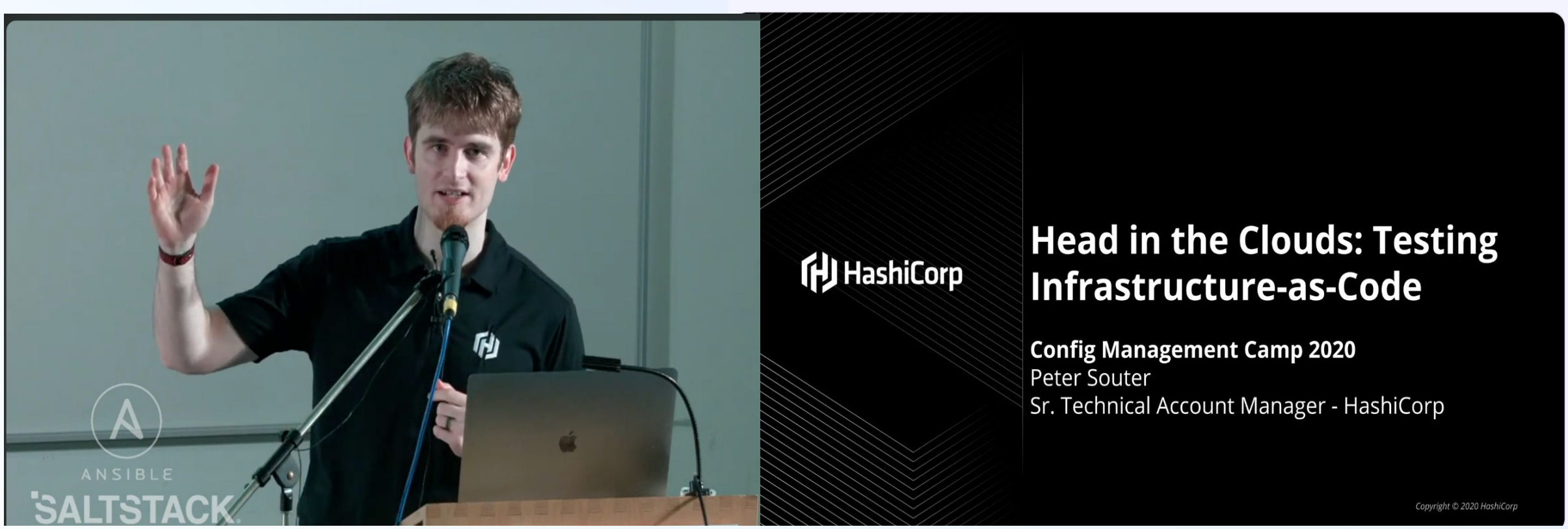
Previously HashiCorp, Puppet

Wanna feel old?


bob
@rjw1
Now for @PeterSouter about securing puppet #cfgmgmtcamp #puppetize
2:53 PM · Feb 1, 2016 from Ghent, Belgium

Last time I was here?

Previously…



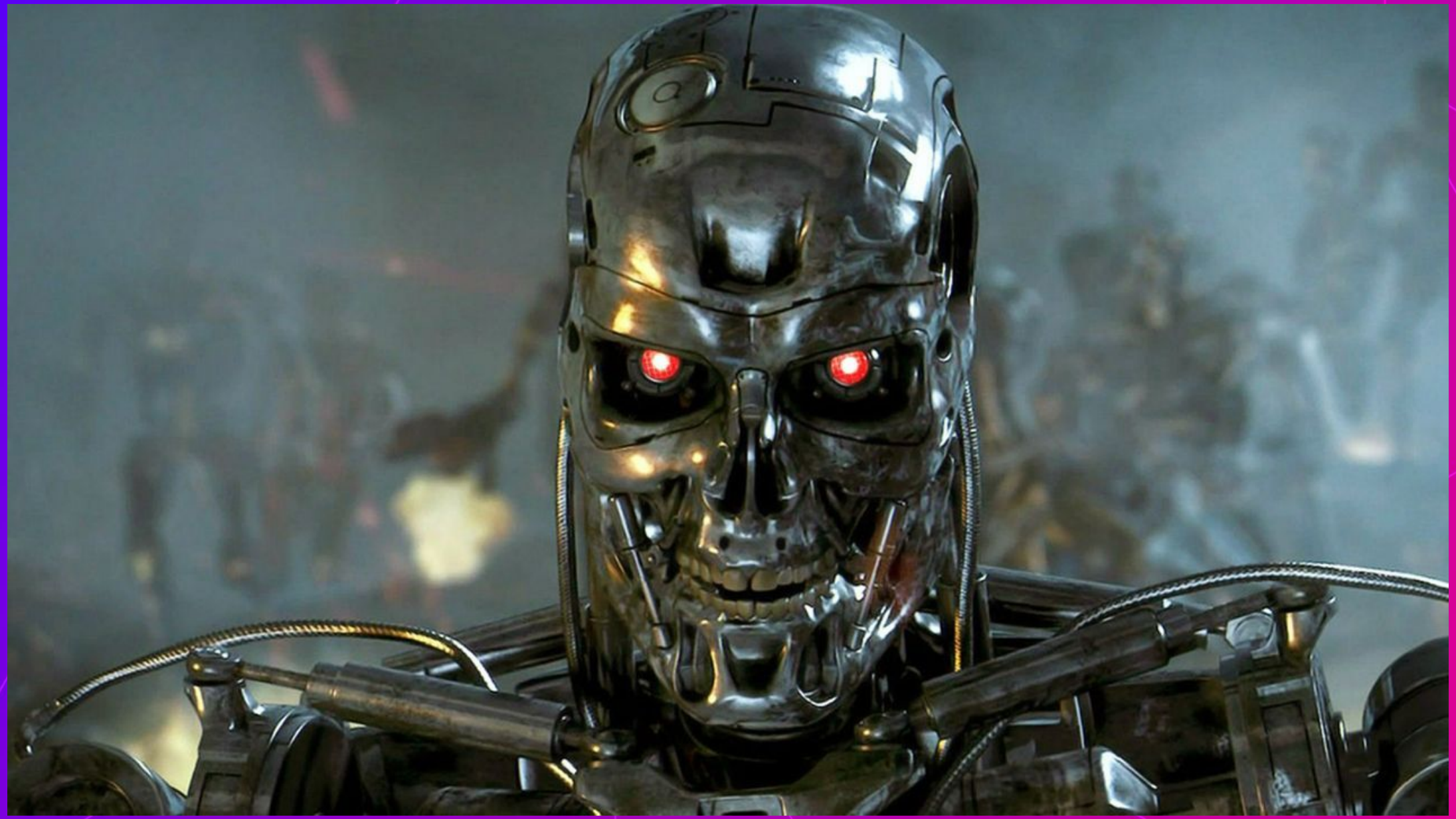
Cfgmgmtcamp 2020 - Head in the Clouds: Testing Infra as Code

February 2020…
Why does that ring a bell?


A GAME BY
MATT LEACOCK
PANDEMIC
CAN YOU SAVE HUMANITY?
PANDEMIC
INTERNATIONAL AWARD-WINNING GAME!
USA
USA
FRANCE
AUSTRALIA
Z-MAN games
CAN YOU SAVE HUMANITY?

DATADOG

A lot can change in 6 years!



DATADOG

But let's set the scene first...

In the beginning…

DATADOG

DATADOG

Software Delivery Lifecycle (SDLC)

Winston W. Royce

DATADOG

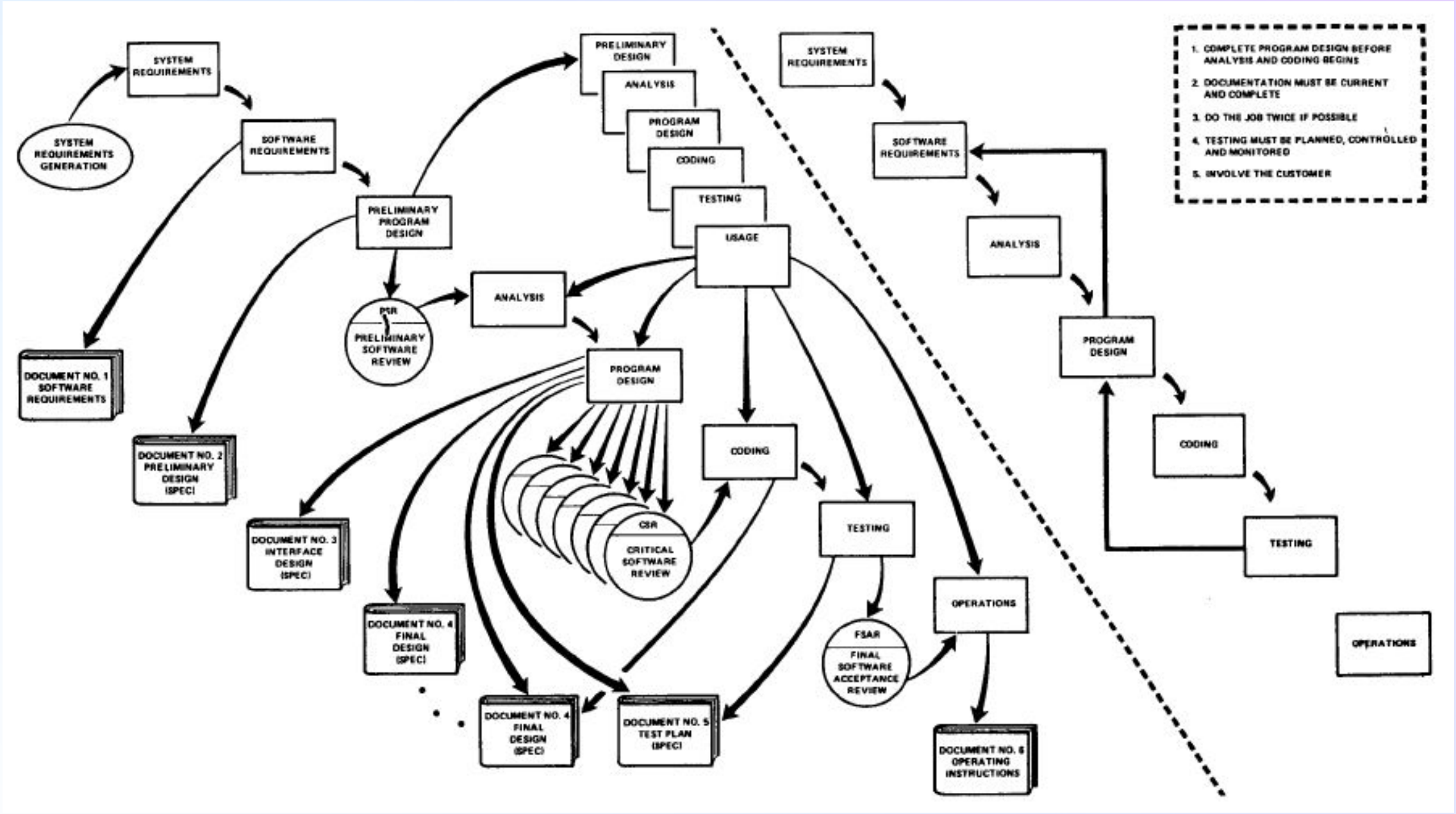

Winston W. Royce

I am going to describe my personal views about **managing large software developments**. I have had various assignments during the past nine years, mostly concerned with the development of software packages for spacecraft mission planning, commanding and post-flight analysis.

In these assignments I have experienced different degrees of success with respect to **arriving at an operational state, on-time, and within costs.**

I have become **prejudiced by my experiences** and I am going to **relate some of these prejudices in this presentation**....

“Managing the development of large software systems” - 1970
https://dl.acm.org/doi/10.5555/41765.41801



"Managing the development of large software systems"

DATADOG

Batch Size

Feedback Loops

Many frameworks and many years later...

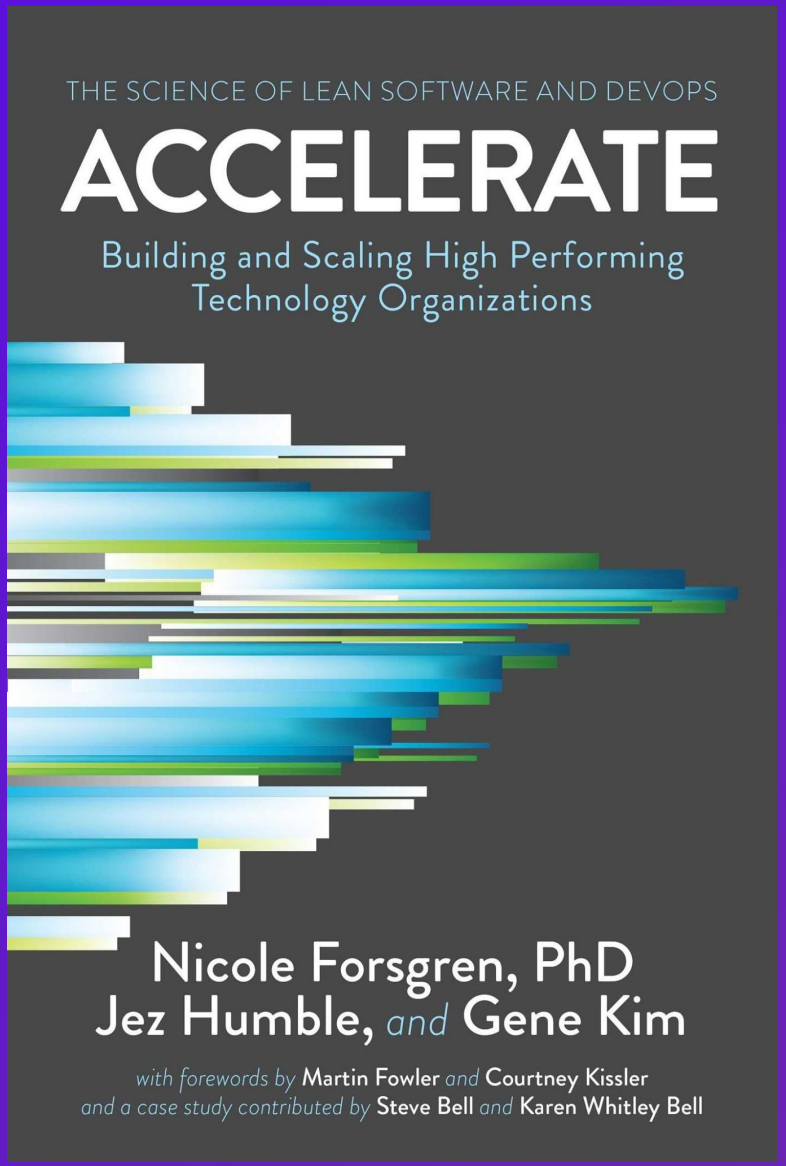
Batch Size

Feedback Loops

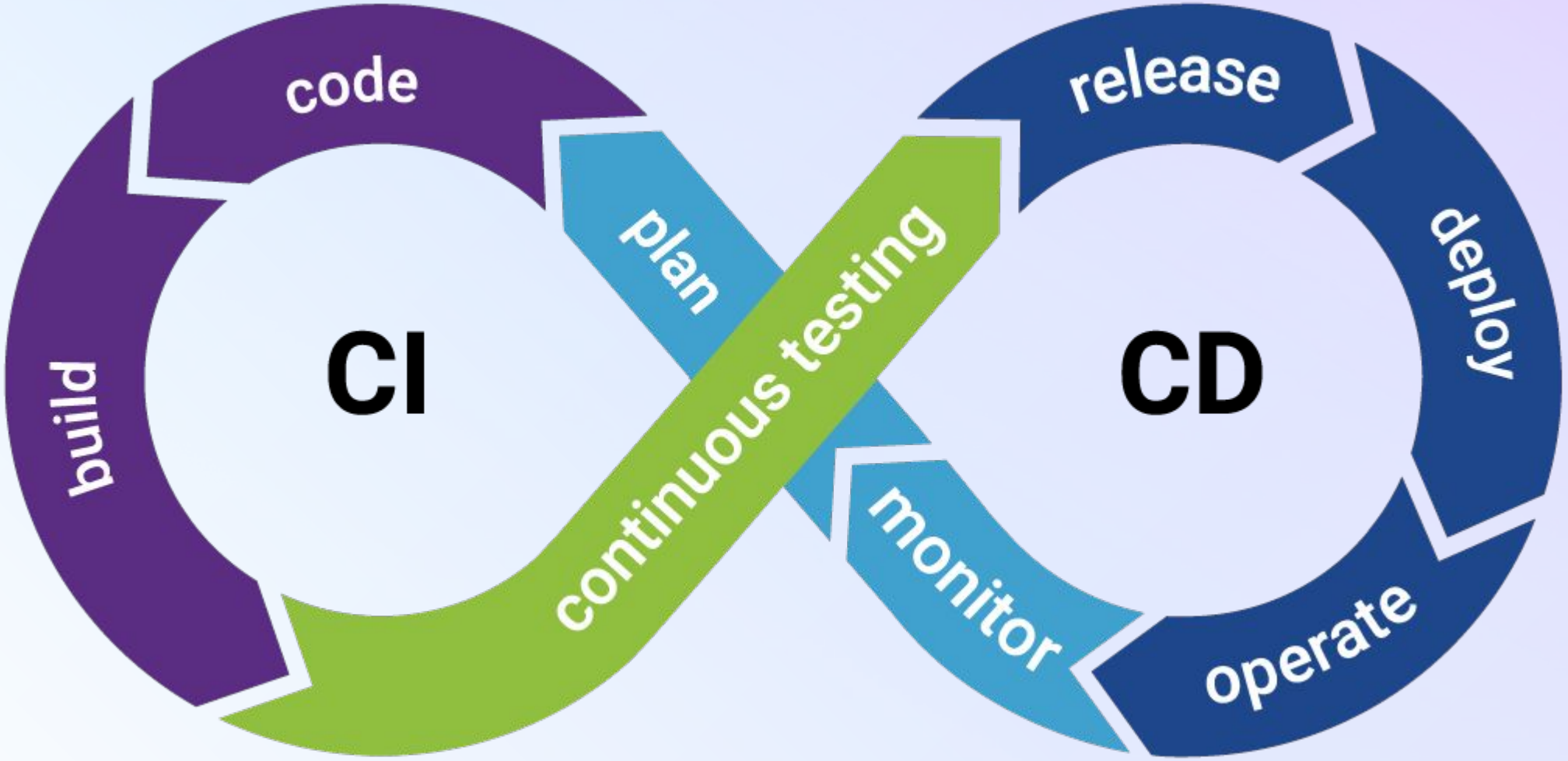
“Reducing batch sizes reduces cycle times and variability in flow, accelerates feedback, reduces risk and overhead, improves efficiency, increases motivation and urgency, and reduces costs and schedule growth,,

Nicole Forsgren, Phd, Jez Humble, and Gene Kim

Accelerate - pg. 49

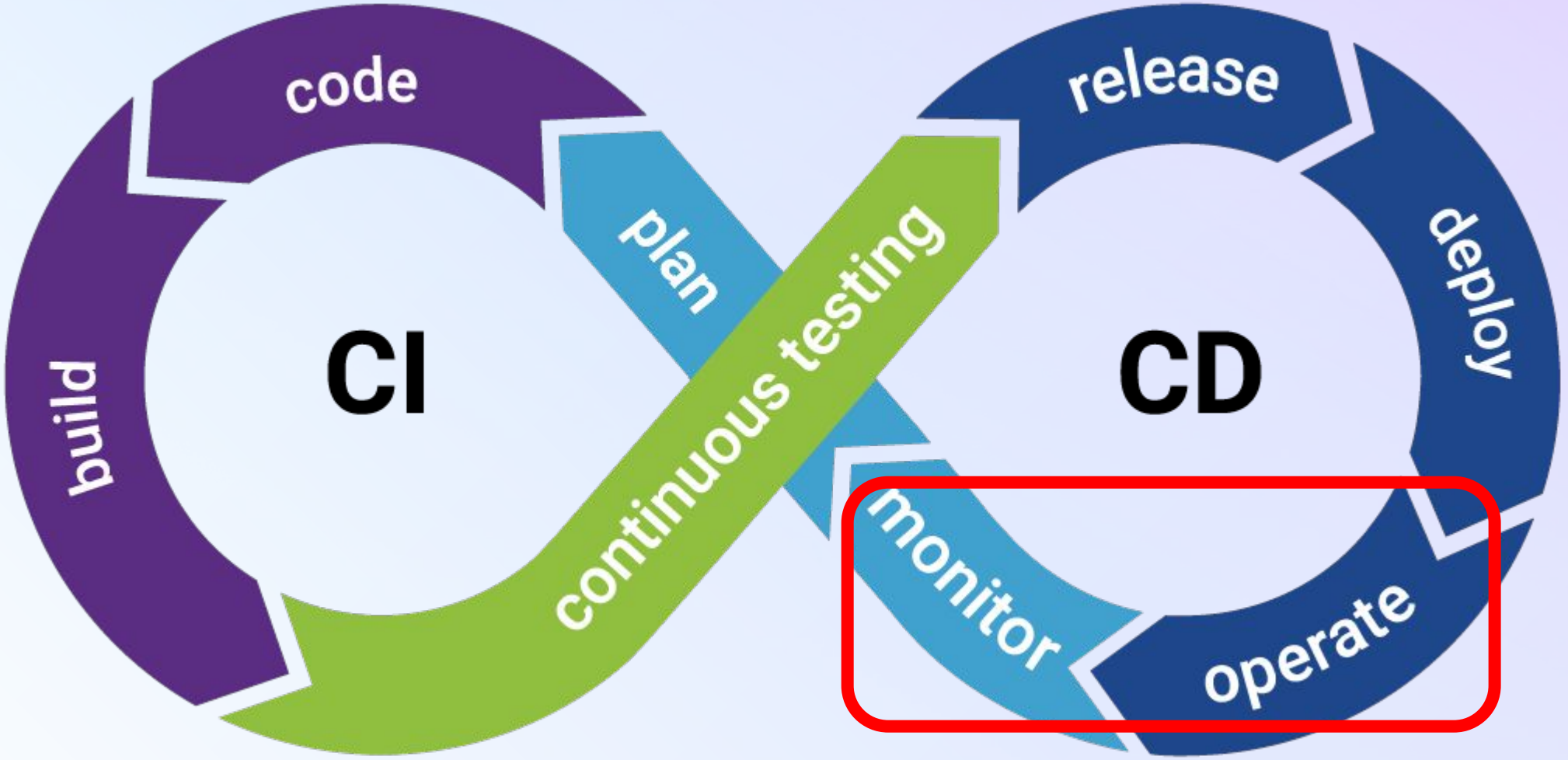


The Challenge?



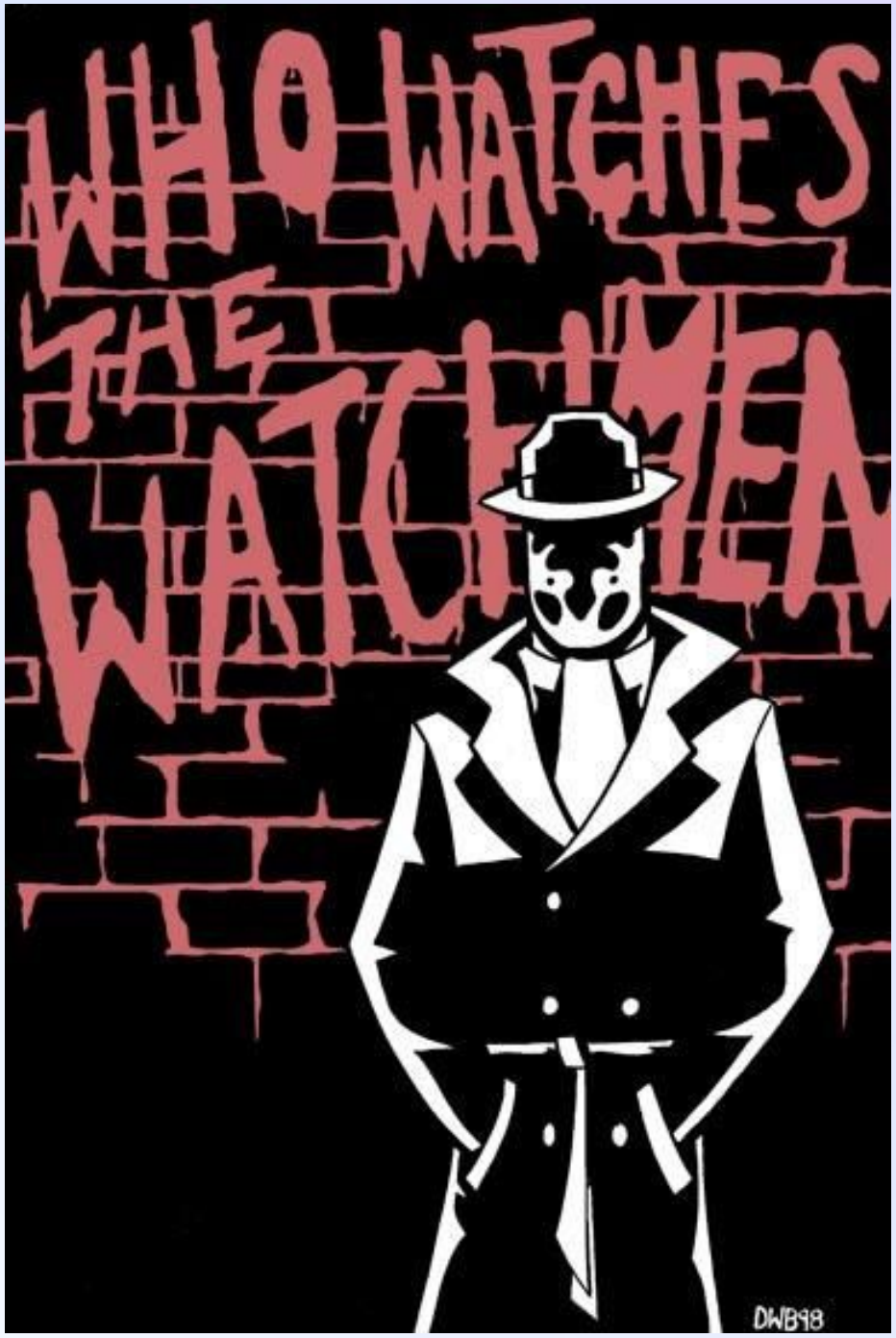
https://www.abtasty.com/resources/ci-cd/

DATADOG



https://www.abtasty.com/resources/ci-cd/

Quis custodiet ipsos custodes?



DATADOG

Question to think about in the audience...

Who “owns” CICD in your org?

The most common answer:
No-one / Everyone!

If everyone owns it…
No one does!

That’s a governance smell!

Re-jig how you think of your SDLC…

“A product with your engineers as the customer”

What feedback are we getting from our customers?

“Ugh, our pipelines are too slow”

“Ooft, yeah the tests are flaky, run it again…”

“Production releases are getting blocked by our deployment processes”

How did we get here?

1. Pipelines left the data center

2. Testing scaled faster than observability

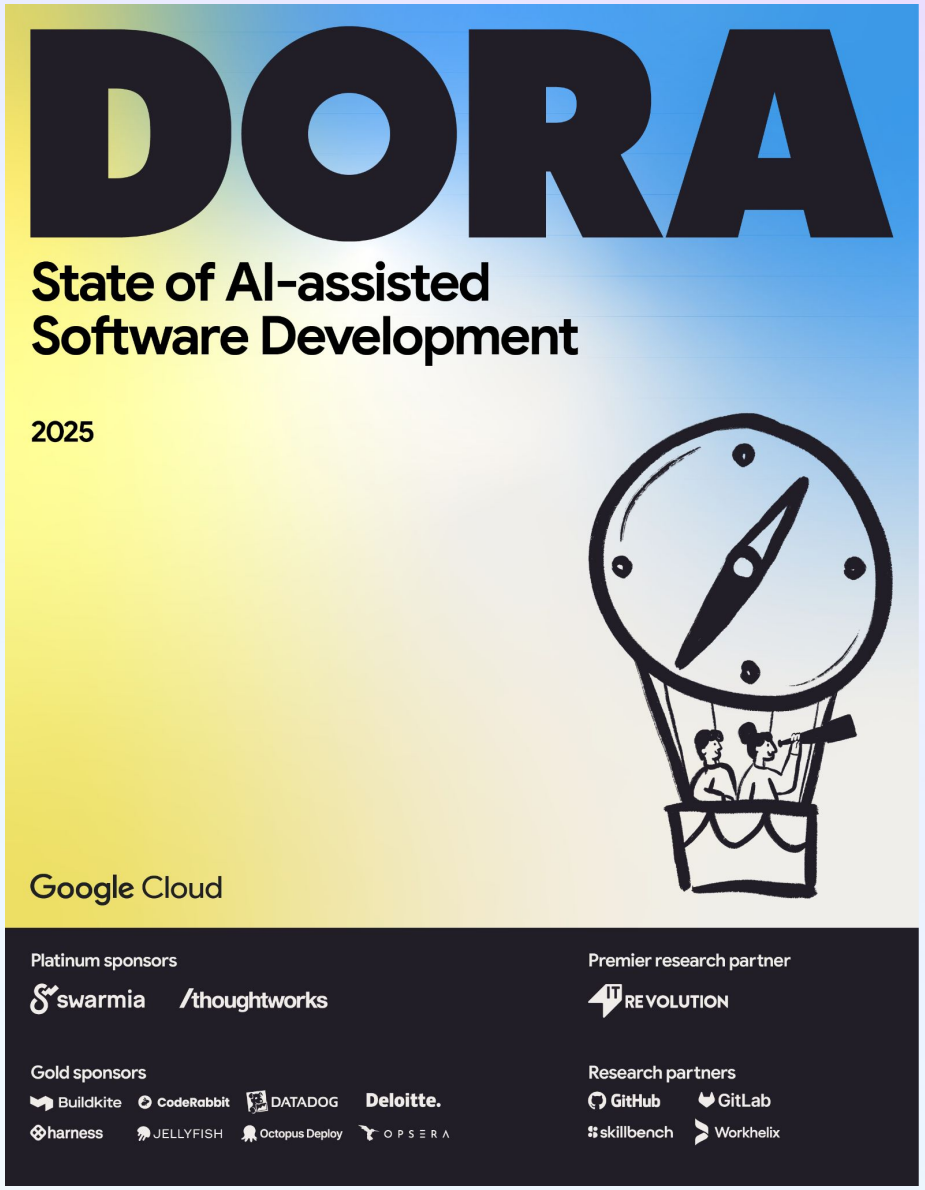
3. Teams optimized production telemetry over CICD

Every technological trend of the past decade has put greater strain on the CI system: agile, DevOps, containers, cloud native, high-performance engineering


NEW CHALLENGER
APPROACHING

GenAI is the latest part of this trend!

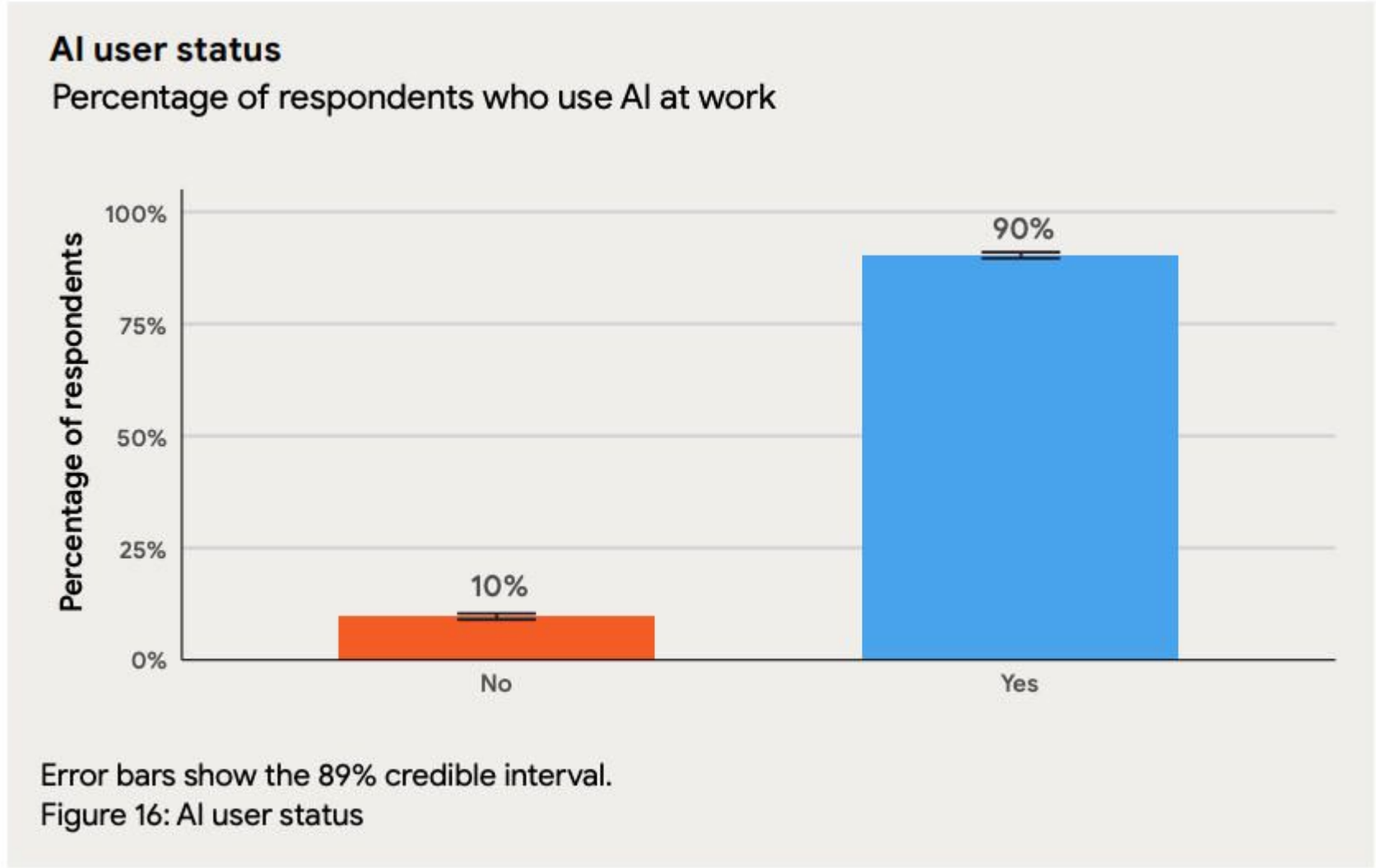
https://dora.dev/dora-report-2024/



https://dora.dev/dora-report-2025/

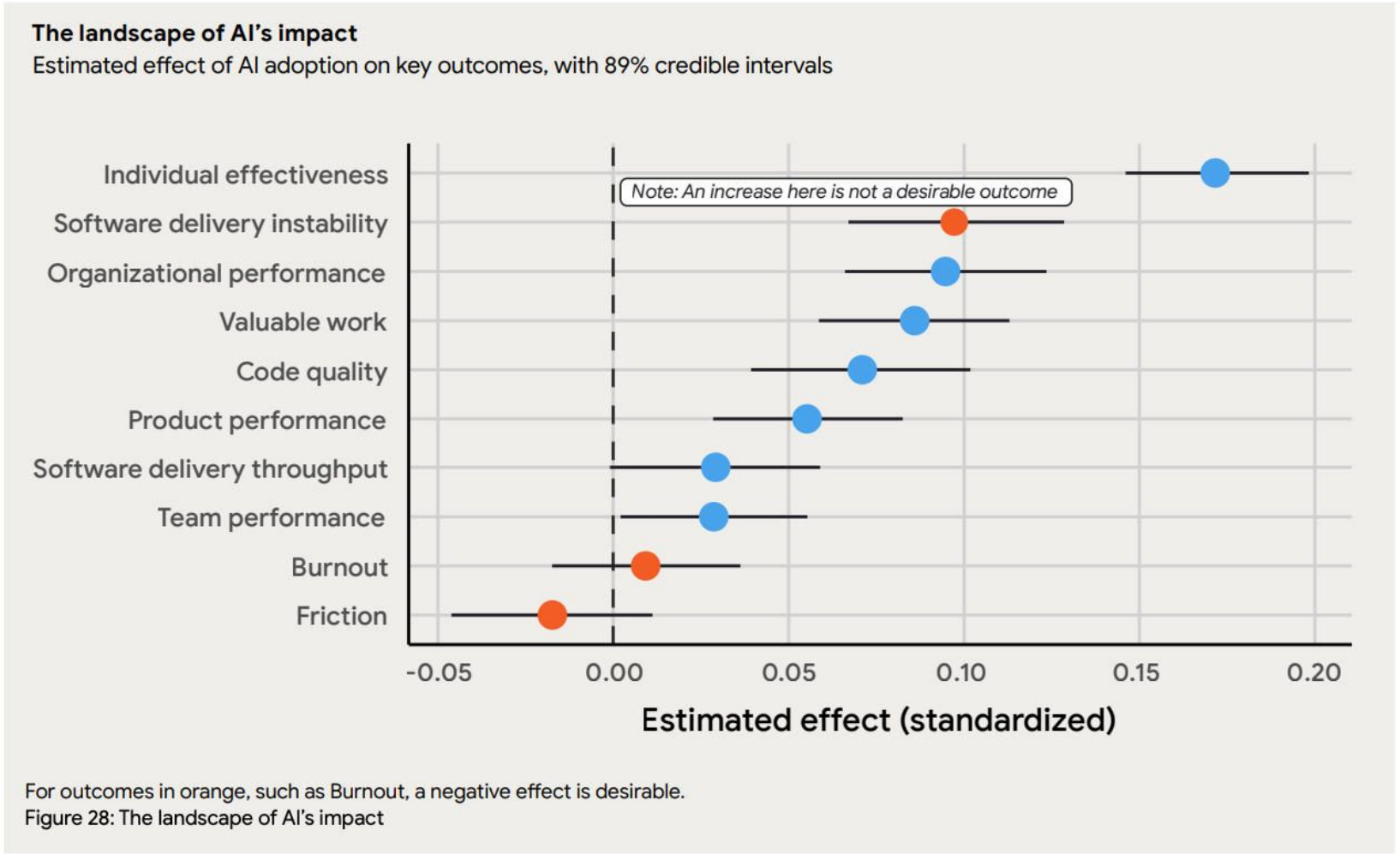
GenAI and the Productivity Paradox..

GenAI: The Productivity Paradox



GenAI: The Productivity Paradox

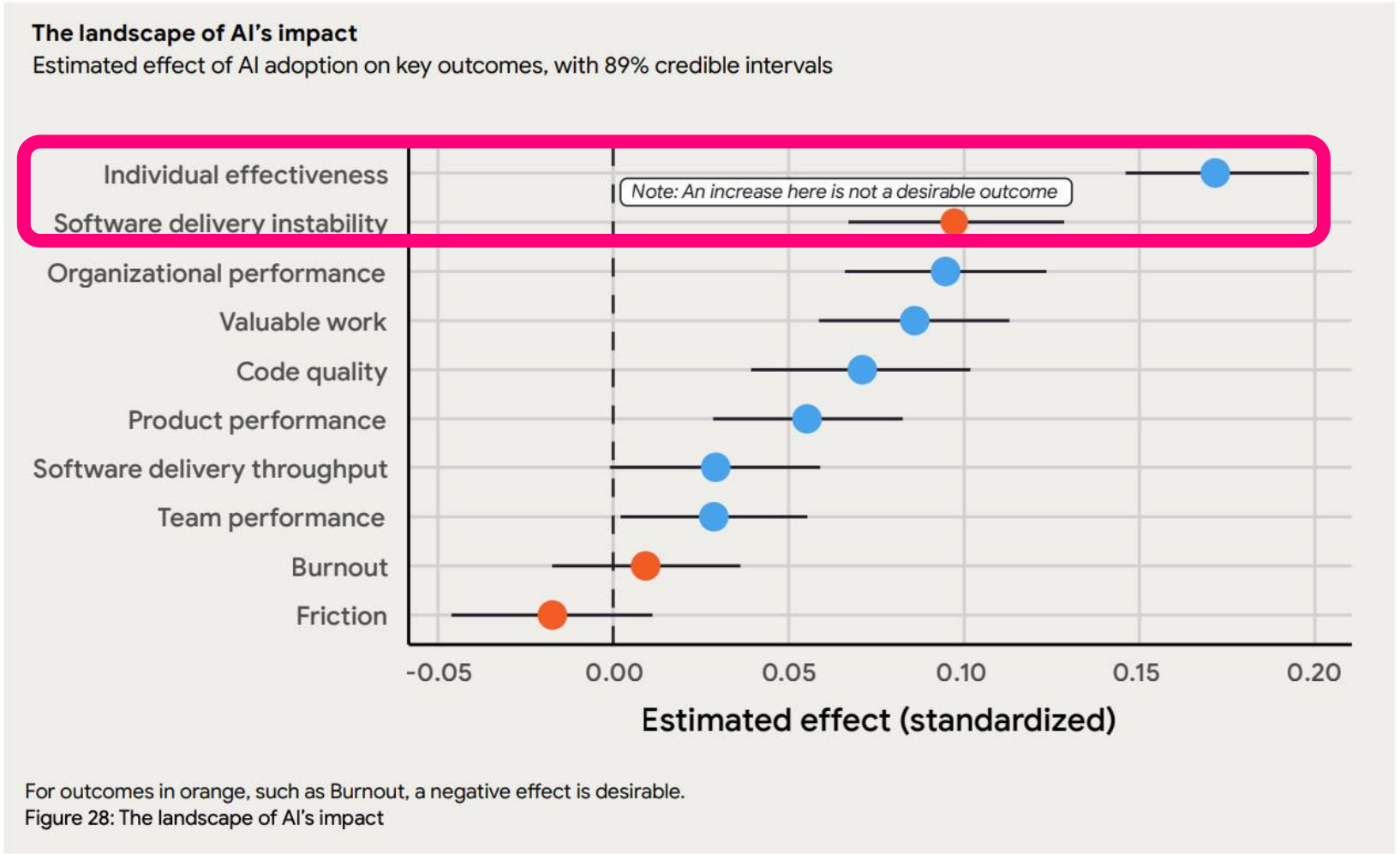
Figure 28 visualizes the relationships AI adoption has with these outcomes.[23]



For outcomes in orange, such as Burnout, a negative effect is desirable.
Figure 28: The landscape of AI's impact

GenAI: The Productivity Paradox

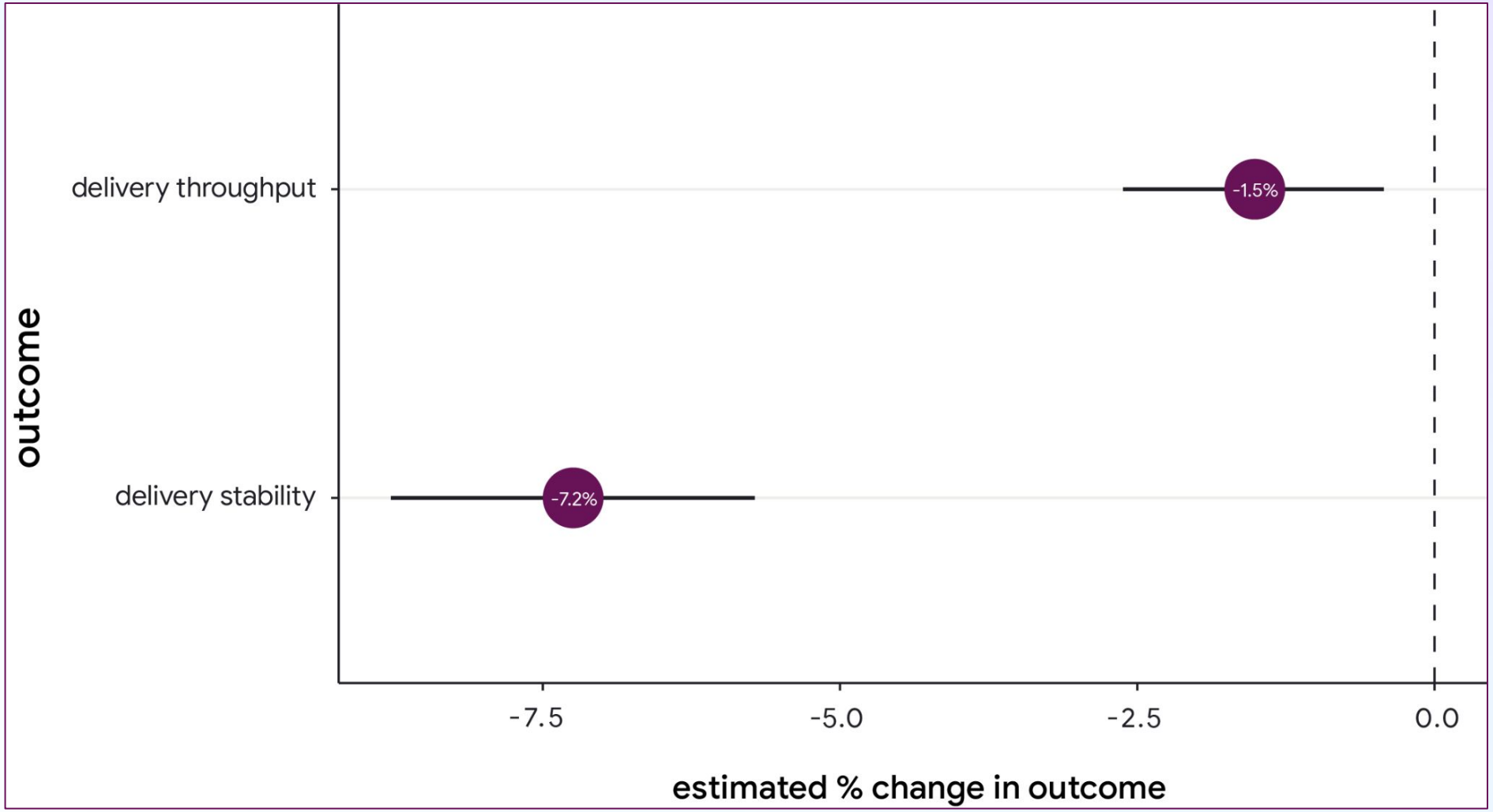
Figure 28 visualizes the relationships AI adoption has with these outcomes.[23]



For outcomes in orange, such as Burnout, a negative effect is desirable.
Figure 28: The landscape of AI's impact

GenAI: The Productivity Paradox

“With +25% AI adoption, delivery throughput reduces by −1.5% and **delivery stability by −7.2%”**

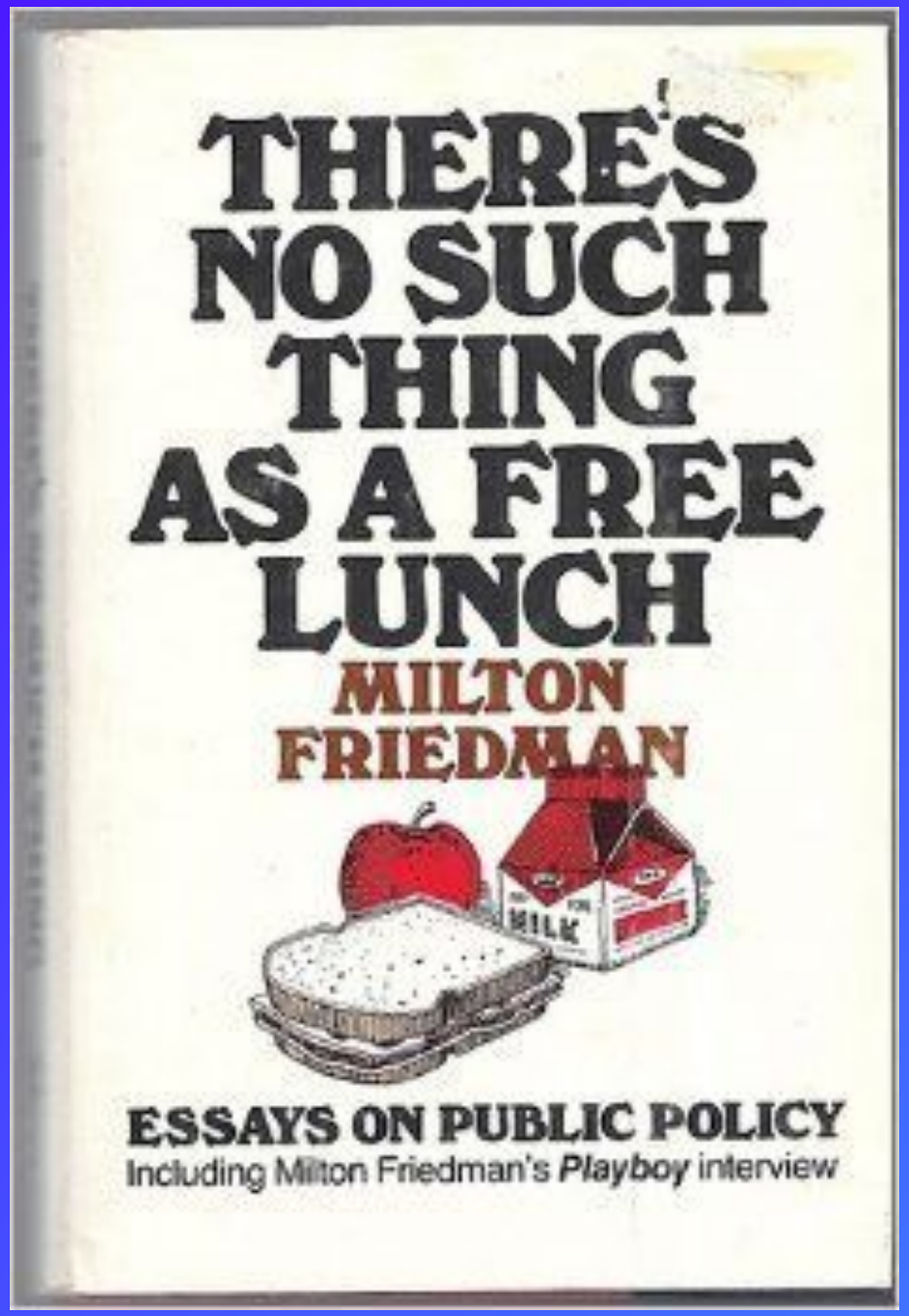


“

The field has forgotten one of DORA’s most basic principles - the importance of small batch sizes

”

DORA 2024 Report: Impact of Generative AI


THERE'S NO SUCH THING AS A FREE LUNCH
MILTON FRIEDMAN
MILK
ESSAYS ON PUBLIC POLICY
Including Milton Friedman's *Playboy* interview

How do we fix it?

“It’s everyone’s problem…”

Nope, not at scale…

“

At a larger scale it’s **impossible to keep flakiness from entering the system** by relying on test authors to do the correct thing.

In earlier days, we relied on the **collective consciousness**, i.e. developers realizing that a test is flaky and proactively investigating it. This worked on a small team (~10 devs), but **at a larger scale there’s a bystander effect.**

”

Ultimately, developers want to **merge the PR they are working on** and **not investigate an unrelated flaky test.**

Arpita Patel - Staff Software Engineer
https://slack.engineering/handling-flaky-tests-at-scale-auto-detection-suppression/

Fixing things holistically...

...and bring that level of visibility to your tests and pipelines in earlier stage environments

Shifting Left

Take visibility historically only seen in production...

What data do we need?

“Production releases are getting blocked by our deployment processes”

“Tests are flaky”

“Our pipelines are too slow”

How slow?
How flaky?
Where and which ones?
How long has this been happening?
Why is this happening?
Where do we need to focus effort to fix it?

Qualitative Measures

Surveys

Pro: Quick, Easy to use and setup

Cons: Susceptible to bias, Difficult to measure over time, Not automatic

---

Recommendations:

Use surveys to find focus areas

Measure broad overview measures (engineer satisfaction etc)

“How do you feel about the SDLC process currently?” 1-5 Scale

Feedback sessions

Guilds, “Ride Alongs”, Embedded DevEx Team members

Quantitative Measures

Systems Data

Pros: Alleviate Bias, Easy to measure over time, Exposes bottlenecks, Helps quantify experimental success

---

Cons: Difficult to set up, storage and engineering costs

Recommendations:

Tiger teams

Use existing tooling to consume from your platforms

Dedicated Technical Ownership of tooling and reporting

DevEx
Enterprise Software Factory (ESF)
SDLC Centre of Excellence

DATADOG

PR build P95 and **Queue Time**

Developer feedback speed

Pipeline Success Rate

Trust in automation

Merge-to-deploy lead time

Flow through the value stream

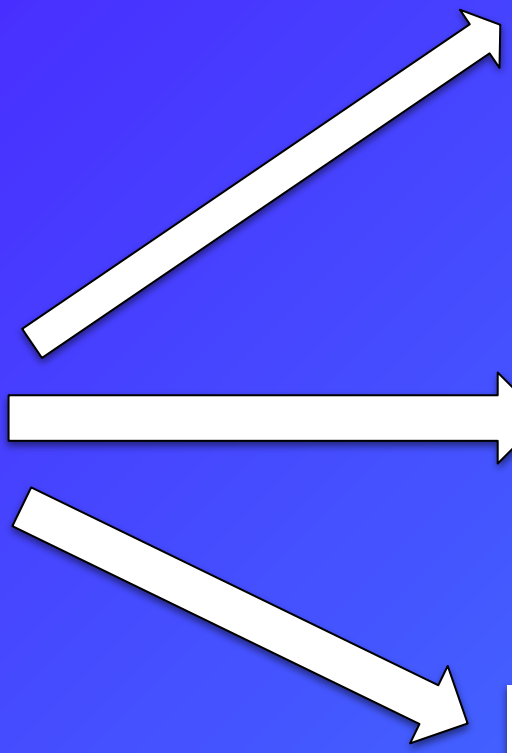
There’s no Silver Bullet: There’s a lot of factors that can lead to slow and blocking pipelines

Just having tangible metrics that you can compare over time is already a big win!

“Our pipelines are too slow”

“Staging to Prod has increased by 20 minutes in the last 6 months because of extra acceptance flows…”

“Staging to Prod has increased by 20 minutes in the last 6 months because of extra acceptance flows…”



Parallelise?

Refactor?

Increase infra resources?

Turn vibes into actionable insights!

DATADOG

How does this look in real companies?

Flaky Tests:
A great target area!

Flaky tests are the worst of both worlds: Time wasting and reducing confidence

Solutions?

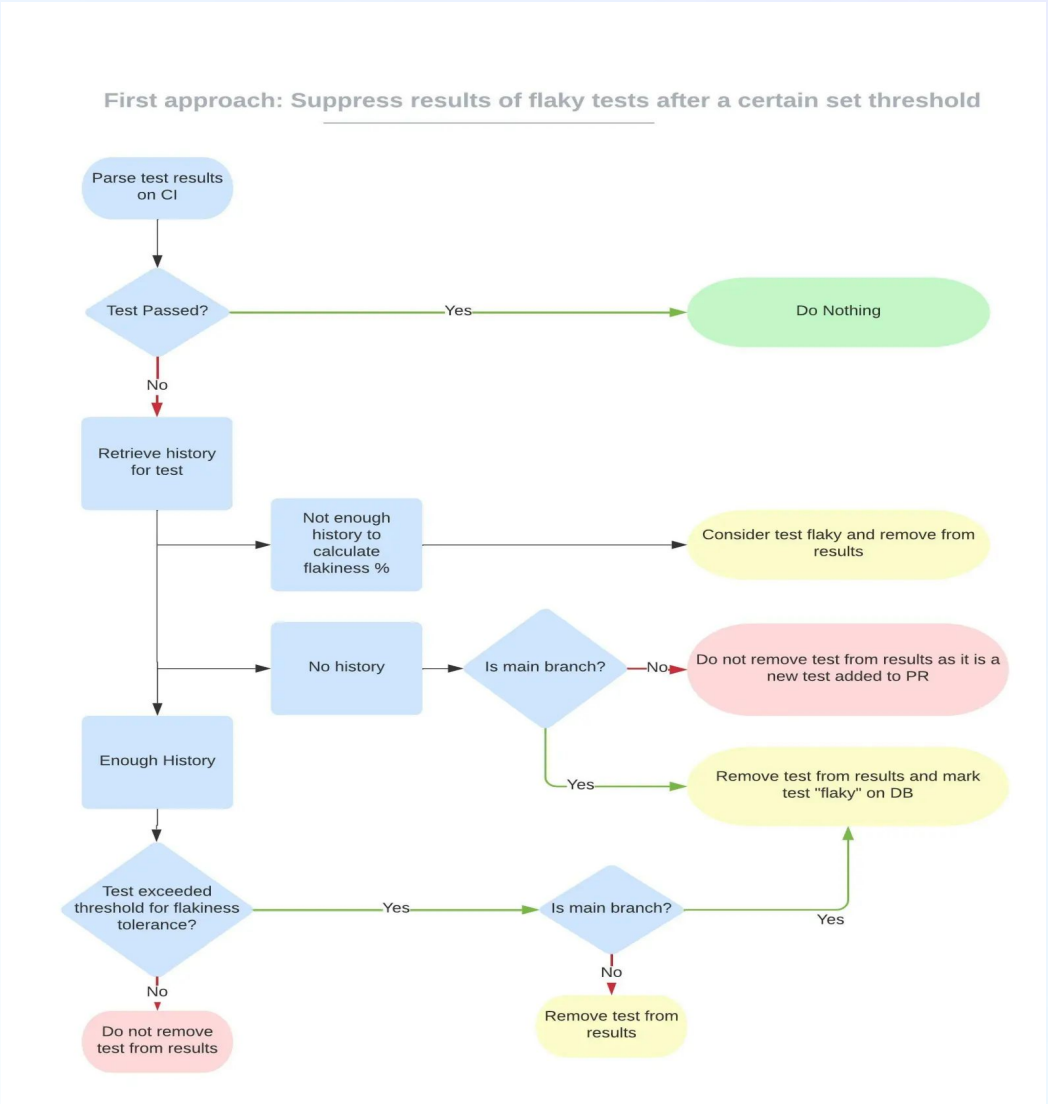
- Auto-Retry
- Quarantine
- Isolate and Refactor

Case Study: Slack

"For the mobile codebases, we have **120+** developers creating **550+ pull requests (PRs)** per week. There are **16,000+** automated tests on Android and **11,000+** automated tests on iOS with the testing pyramid consisting of E2E, Functional, and Unit tests. **All tests run on each commit to a GitHub PR and on every PR merged to the main branch**. Moreover, developers are responsible for writing and maintaining all the automated tests associated with their product group."

https://slack.engineering/handling-flaky-tests-at-scale-auto-detection-suppression/

Case Study: Slack



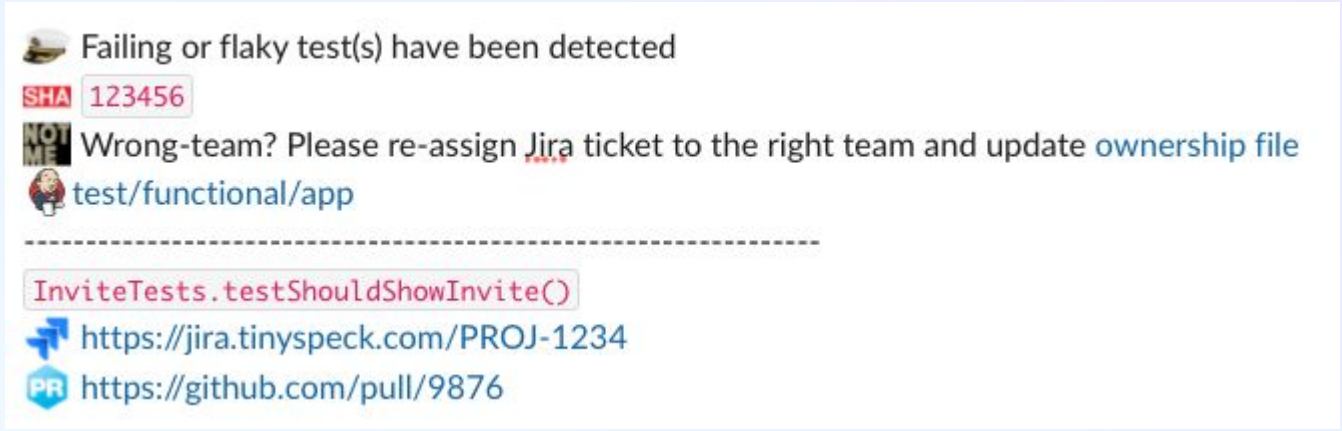
DATADOG

Case Study: Slack

```
"""
Modify the test file to disable test based on platform: iOS or Android
"""
def disable_test(test_name, jira_ticket):
    test_file_path = get_file_path_for_test(test_name)
    with in_place.InPlace(test_file_path) as test_file:
        for line_num, line in enumerate(test_file, 1):
            # Regex to detect test name
            test_found = re.search(test_name + "`?\(", line, re.MULTILINE)
            if test_found:
                if self.platform == "ios":
                    disable_ios_test()
                elif self.platform == "android":
                    disable_android_test()
            test_file.write(line)

"""
This function disables a test by renaming it and adds a Jira ticket to the comment
Example input: func testShouldShowInvite() {
Example output: // https://jira.com/PROJ-123
                func disabled_testShouldShowInvite() {
"""
def disable_ios_test(jira_ticket):
    ...

"""
This function disables a test by renaming it and adds a Jira ticket to the comment
Example input: fun testShouldShowInvite() {
Example output: @Ignore('https://jira.com/PROJ-123')
                fun testShouldShowInvite() {
"""
def disable_android_test(jira_ticket):
    ...
```



https://slack.engineering/handling-flaky-tests-at-scale-auto-detection-suppression/

DATADOG

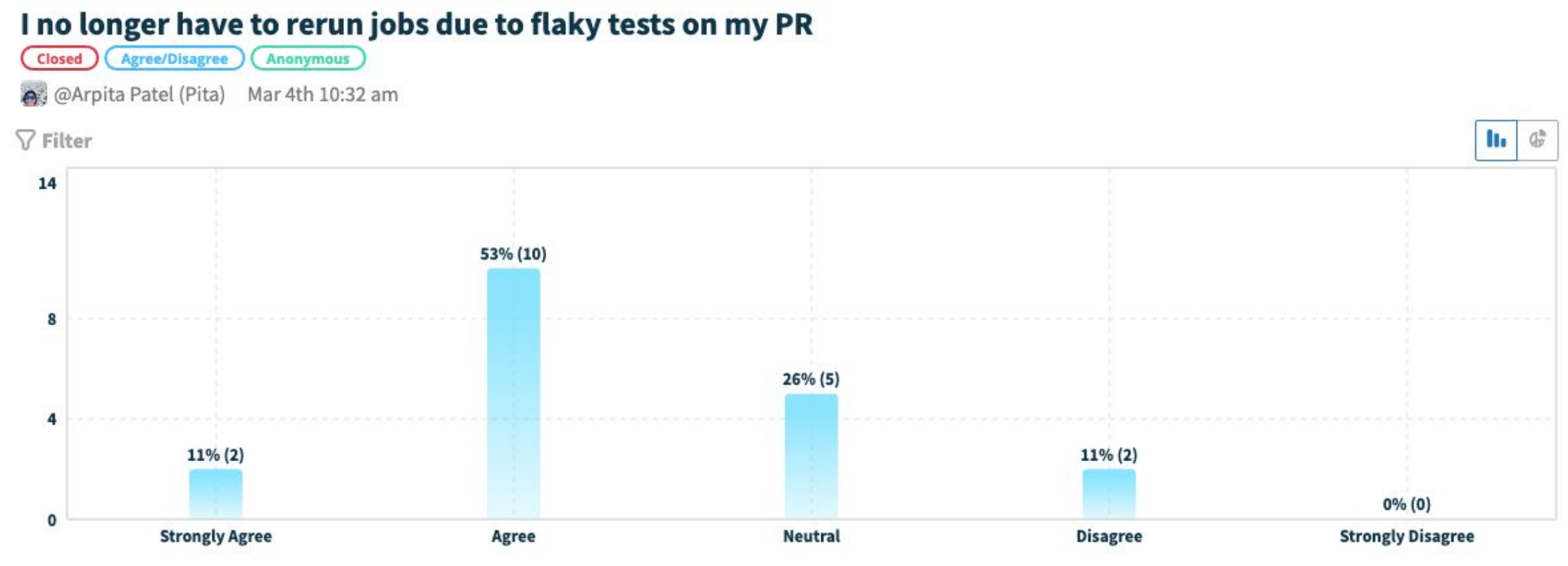
Case Study: Slack

> *Improved the main branch stability to 96%*: From 19.82% on July 27, 2020 to 96% on Feb 22, 2021. Areas negatively affecting stability were 3rd-party services and merge conflicts.

https://slack.engineering/handling-flaky-tests-at-scale-auto-detection-suppression/

DATADOG

Case Study: Slack

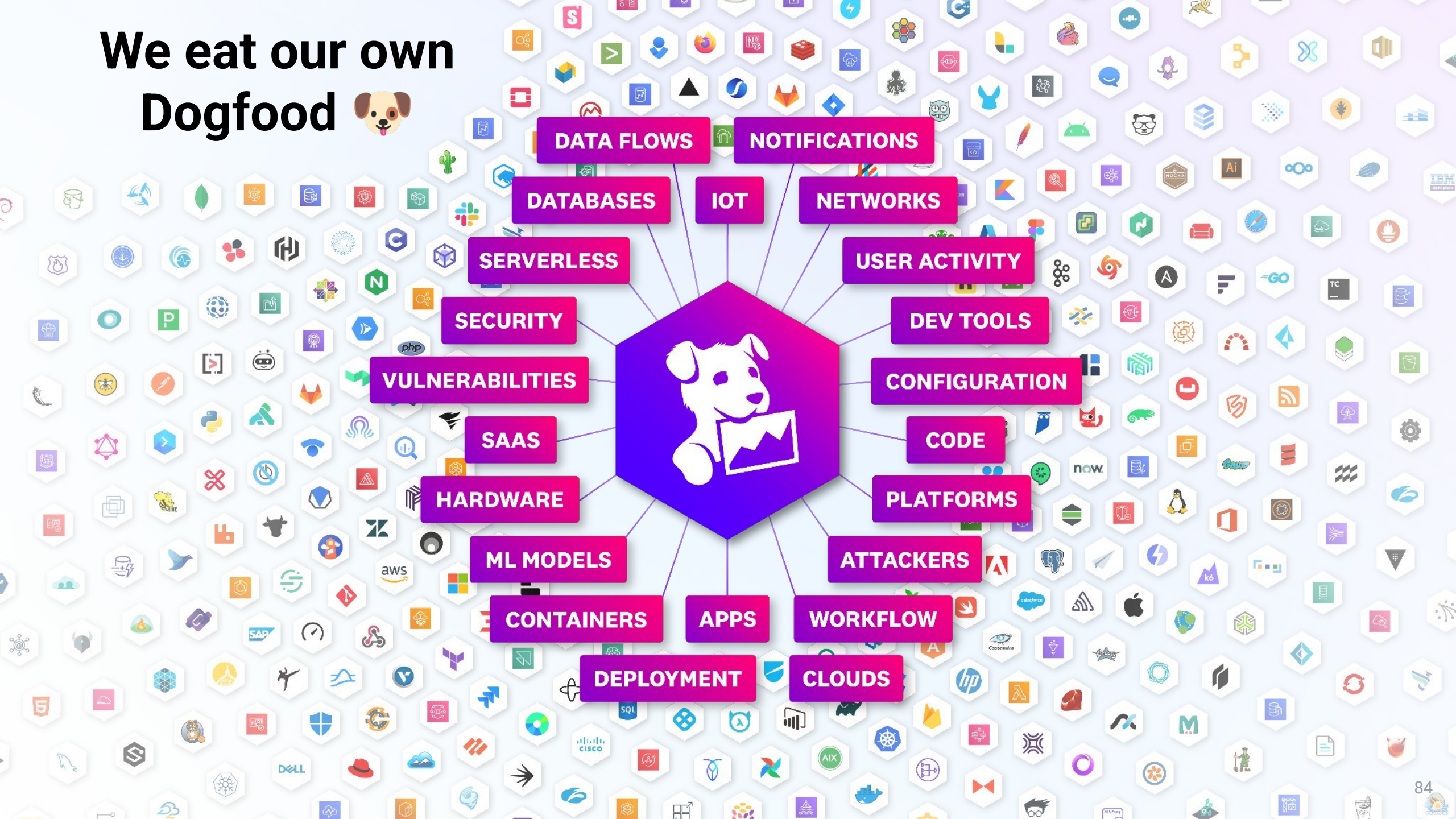


https://slack.engineering/handling-flaky-tests-at-scale-auto-detection-suppression/

DATADOG

Another example…
Datadog itself!

We eat our own Dogfood 🐶



Case Study: Datadog

Datadog

- Main codebase for the platform itself: **Monorepo with 400+ devs**
- Long running branches, **lots of cross-team challenges**
- A lot of **custom dashboards within Datadog's internal setup** for **SDLC challenges**

DATADOG

Case Study: Datadog

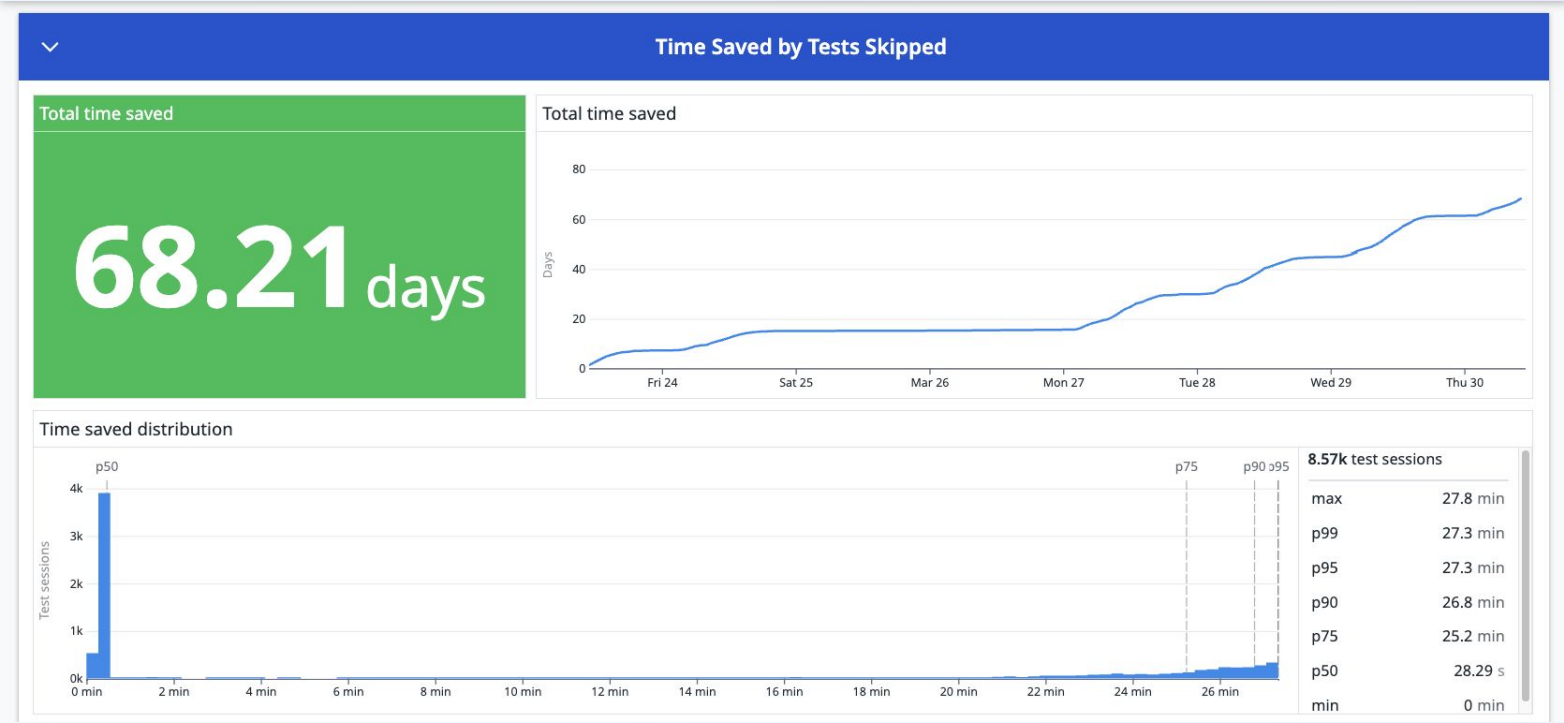
Datadog

- Team “**dogfooded**” features we later product-ised and released as SDLC improvements for customers:
  - **“Test Impact Analysis”** to only run smaller-slice tests runs on branches
  - **OOTB Health Dashboards** for Tests and Pipelines
  - Automatic detection of **nature of failures** (Are they “real” or infra related?)

DATADOG

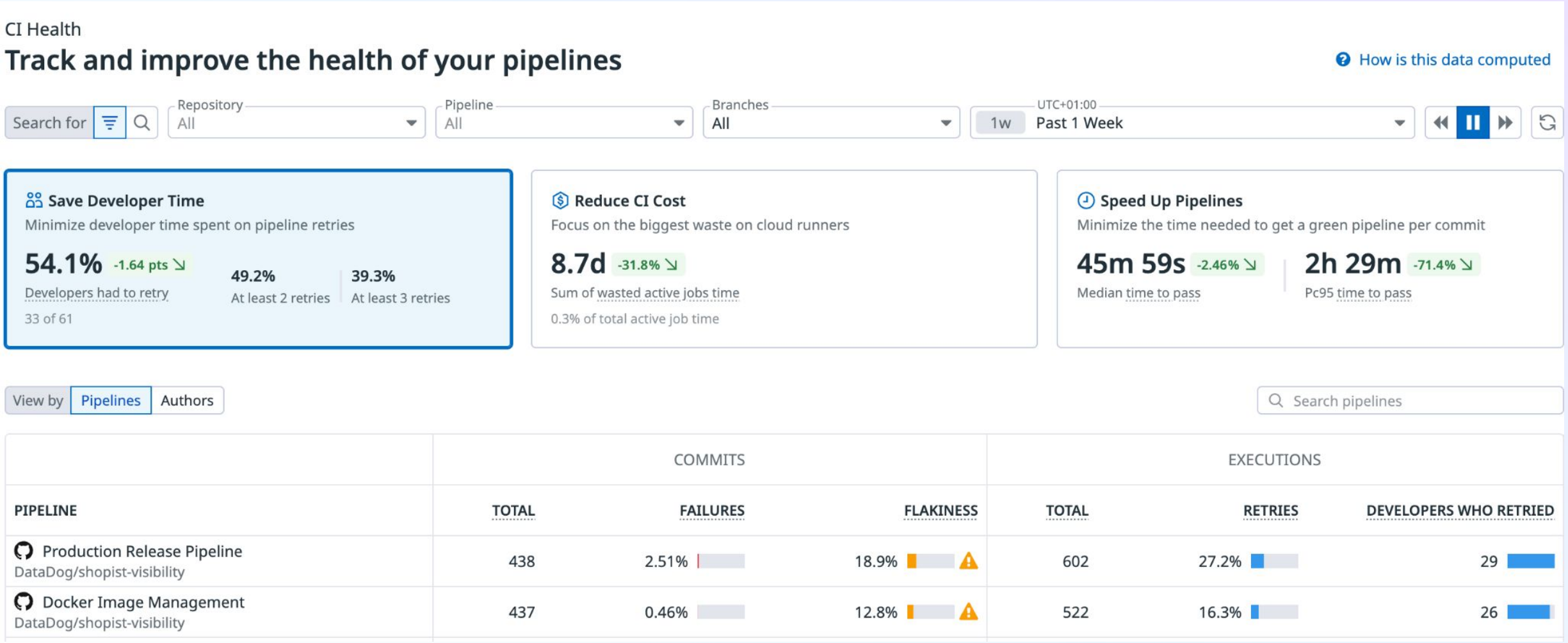
Case Study: Datadog

- **“Test Impact Analysis”**
- Results:
  - **~15 min median duration**
  - Enabled for **all feature branches**
  - **6,500+ hours** saved per month
  - **~19+ hours** saved per dev
  - Significant CI **cost savings**



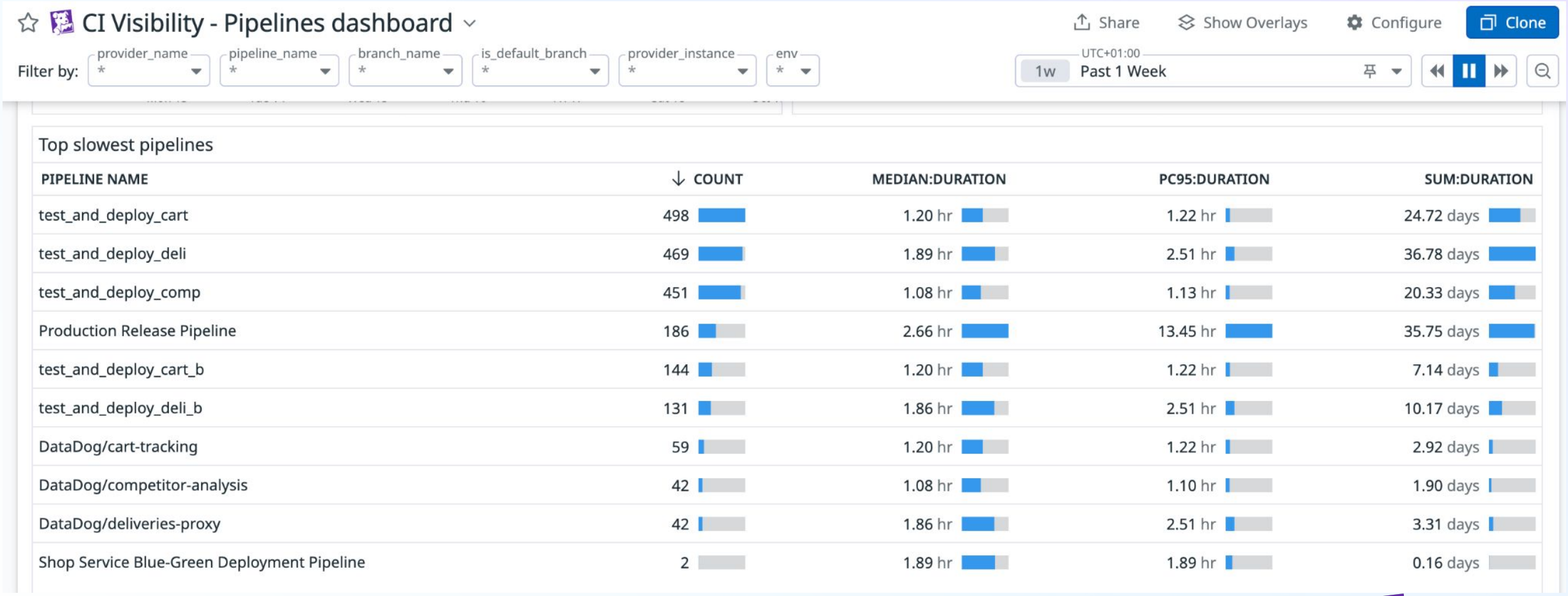
DATADOG

Case Study: Datadog



DATADOG

Case Study: Datadog

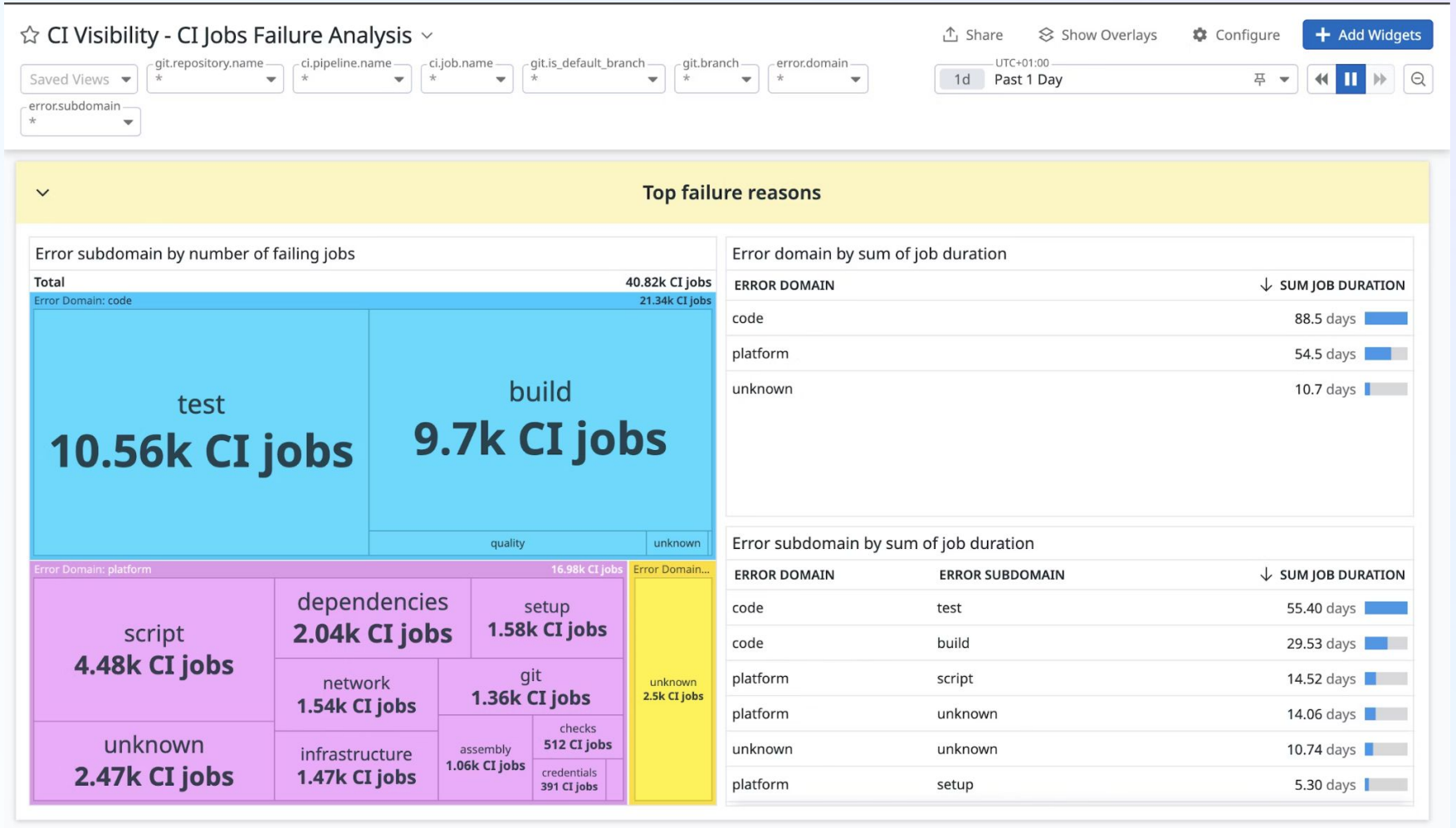


Top slowest pipelines

PIPELINE NAME	COUNT	MEDIAN:DURATION	PC95:DURATION	SUM:DURATION
test_and_deploy_cart	498	1.20 hr	1.22 hr	24.72 days
test_and_deploy_deli	469	1.89 hr	2.51 hr	36.78 days
test_and_deploy_comp	451	1.08 hr	1.13 hr	20.33 days
Production Release Pipeline	186	2.66 hr	13.45 hr	35.75 days
test_and_deploy_cart_b	144	1.20 hr	1.22 hr	7.14 days
test_and_deploy_deli_b	131	1.86 hr	2.51 hr	10.17 days
DataDog/cart-tracking	59	1.20 hr	1.22 hr	2.92 days
DataDog/competitor-analysis	42	1.08 hr	1.10 hr	1.90 days
DataDog/deliveries-proxy	42	1.86 hr	2.51 hr	3.31 days
Shop Service Blue-Green Deployment Pipeline	2	1.89 hr	1.89 hr	0.16 days

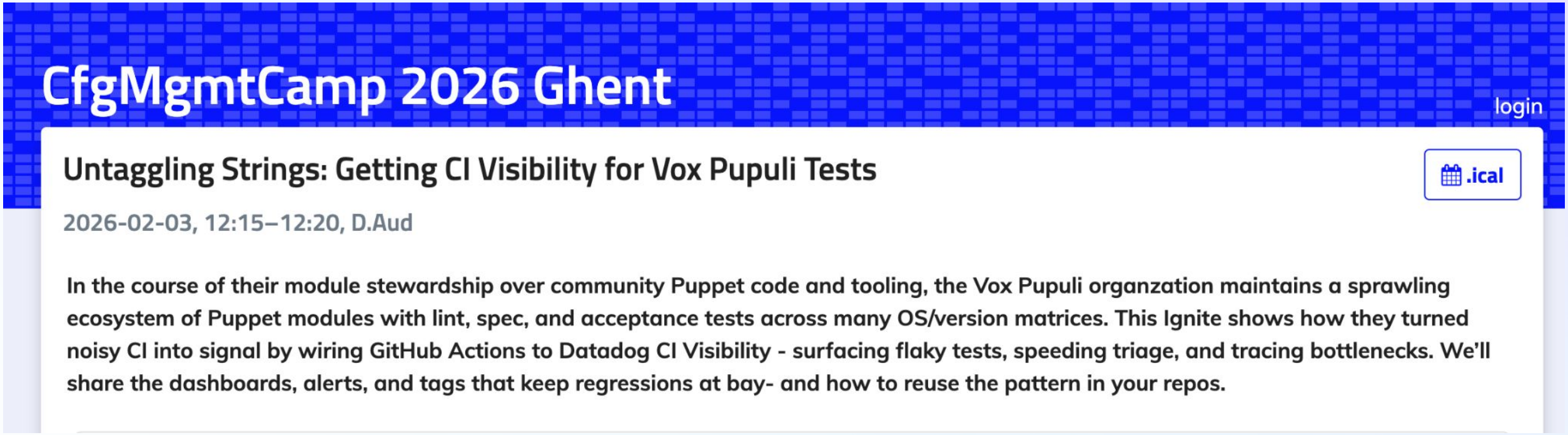
DATADOG

Case Study: Datadog



DATADOG

Case Study: Vox Pupuli



🔥🔥 **Come see my ignite tomorrow!**

DATADOG

Remember to get continuous human feedback and celebrate wins!

- Retrospectives
- Playbacks
- Lunch and Learns
- Awards and Praise

Ok, that’s great, but that’s still very focused on the lens of CICD...

It’s a big part of SDLC but not the be-all-and-end-all…

How do we measure our SDLC in the aggregate?

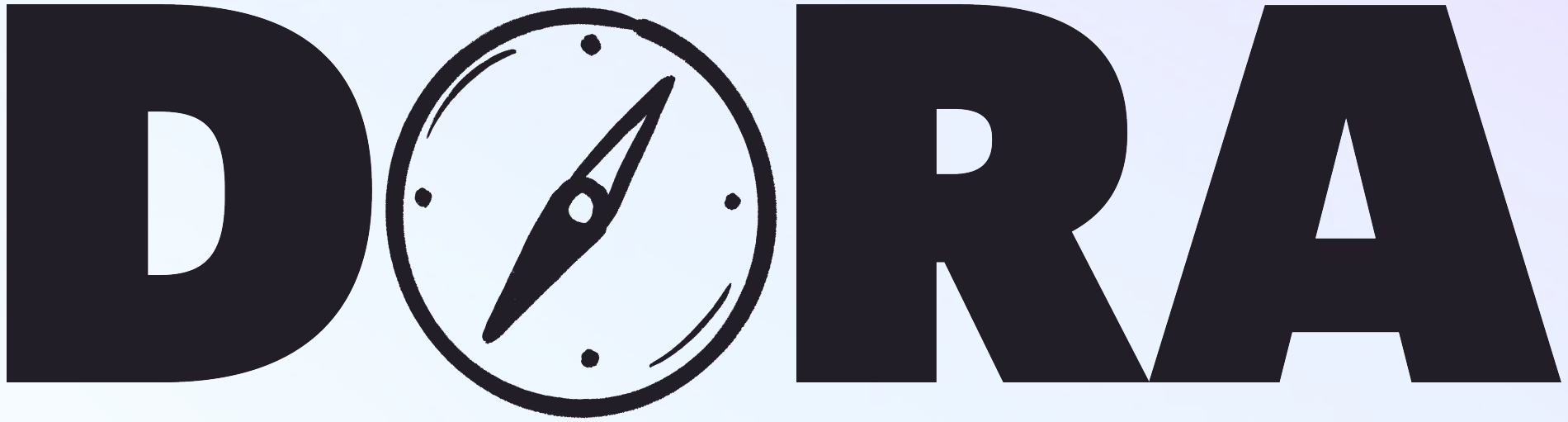
A Framework: DORA

DATADOG

DATADOG

DATADOG


DORA
DEVOPS RESEARCH & ASSESSMENT



DORA.dev

DORA Core Model

dora.dev/research

Capabilities

Climate for learning

- Code Maintainability
- Documentation quality
- Empowering teams to choose tools
- Generative culture

Fast flow

- Continuous delivery
- Database change management
- Deployment automation
- Flexible infrastructure
- Loosely coupled teams
- Streamlining change approval
- Version control
- Working in small batches

Fast feedback

- Continuous integration
- Monitoring and observability
- Reliability engineering
- Pervasive security
- Test automation
- Test data management

→

Performance

Software delivery

Four key metrics:

- Change lead time
- Deployment frequency
- Change fail percentage
- Failed deployment recovery time

Reliability

measured by
Service Level Objectives (SLOs):

- Measurement coverage
- Measurement focus
- Target optimization
- Target compliance

→

Outcomes

Organizational performance

- Commercial Performance
- Non-commercial performance

Well-being

- Job satisfaction
- Productivity
- Reduced burnout
- Reduced rework

What are DORA Metrics?

4 key metrics to measure the velocity and stability of software delivery

VELOCITY

Deployment Frequency
How often an organization successfully releases to production

Change Lead Time
Amount of time it takes a commit to get into production

STABILITY

Change Failure Rate
Percentage of deployments causing a failure in production

Time to Restore
Amount of time it takes an organization to recover from a failure in production

So... are you ELITE?

DORA Metrics Performance Categories

	Deployment frequency	Lead time for changes	Change failure rate	Time to restore service
Low performers	Less than once every six months	Between six months and one month	From 60% to 46%	Between one month and one week
Medium Performers	From once every six months to once every month	Between one month and one week	From 46% to 30%	Between one week and one day
High Performers	From once a month to once a week	Between one week and one day	From 30% to 15%	Less than one day
Elite Performers	On-demand (multiple deploys per day)	Less than one hour	From 15% to 0%	Less than one hour

How do you start?

Set Baselines

Start conversations and map out flows

Have buy-in and commitment from everyone

Do the work!

And of course… take feedback, loop and repeat!

Batch Size

Feedback Loops

Important Note:
These are system outcomes, not individual KPIs!

Don’t turn these into quotas!

Culture over dashboards!

So, what have we learnt?

CI/CD is a Product - Give it an Owner and SLOs

Measure What Matters in your SDLC process

Use DORA as a baseline SDLC Standard

Smaller Batches + Faster Feedback = Safer, Faster Delivery

Batch Size

Feedback Loops

Batch Sizes and Feedback Loops… everywhere!

Make the Work Visible!

Q&A

Thank You For Listening!

Further Links and Resources

- **DORA**
  - https://dora.dev/dora-report-2024/
  - https://dora.dev/dora-report-gen-ai/
  - https://dora.dev/research/ai/
  - https://dora.community
- **Datadog Software Delivery**
  - https://www.datadoghq.com/product/ci-cd-monitoring/
  - https://www.datadoghq.com/knowledge-center/dora-metrics/
- **Relevant Books and Studies**
  - **A Framework for Automating the Measurement of DevOps Research and Assessment (DORA) Metrics** - https://ieeexplore.ieee.org/document/10336287
  - **Accelerate** - By Nicole Forsgren, Jez Humble, Gene Kim - https://itrevolution.com/product/accelerate/

DATADOG