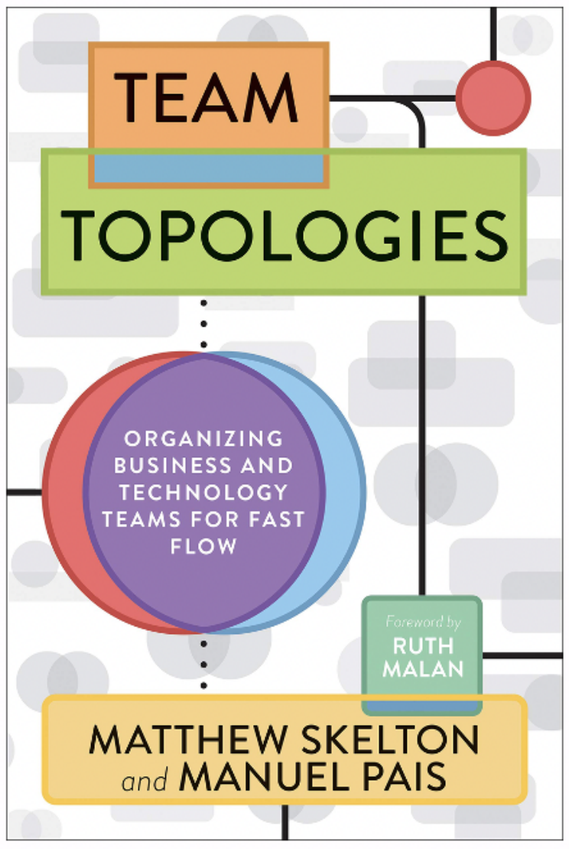
💜A love letter to DORA 💜

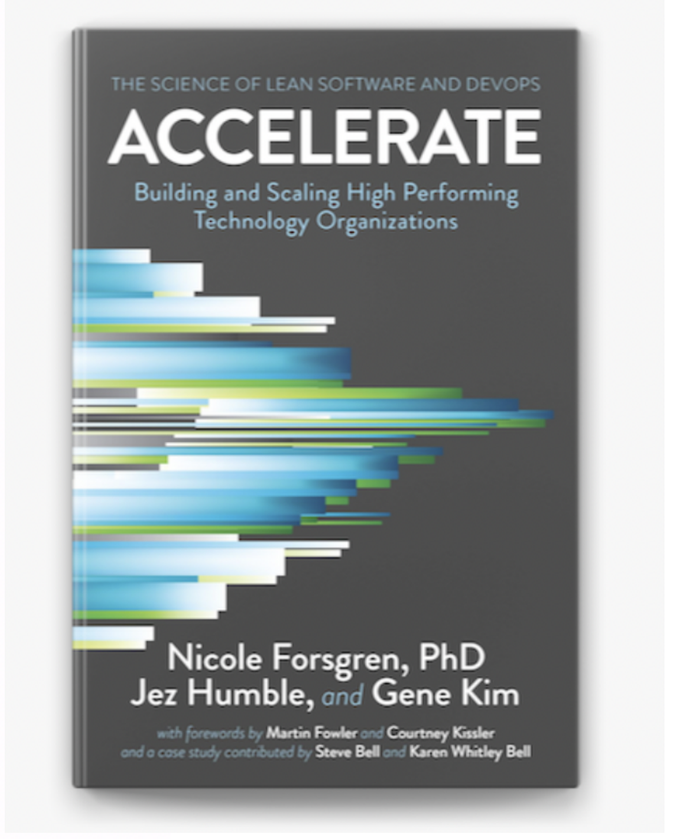
Head of Engineering Practice

cinch


cinch
XXX
£XXX
XXX
£XXX
Hi!
cinch

The 'starting a start-up in 2019' starter pack





Continue the conversation:
http://dora.community

cinch

"What Does Your Business Think of IT?"

	Cost Center	Asset	Partner	Enabler
Focus On	Cost	Return on Investment	Business Value	Speed & Innovation
CIO Reporting	CFO	COO	CTO	CEO
Common Strategy	Outsource IT	Harmonize/ Rationalize	Insourcing	IT = Business
Levers	Cost Cutting	Economies of Scale	Business Case Driven	Economies of Speed

Traditional Structure

Digital Disruption

Source: Gregor Hope "Enterprise Architecture = Architecting the Enterprise" YOW! 2017

Continuous Delivery ltd

Dave Farley - Managing Director, Continuous Delivery Ltd

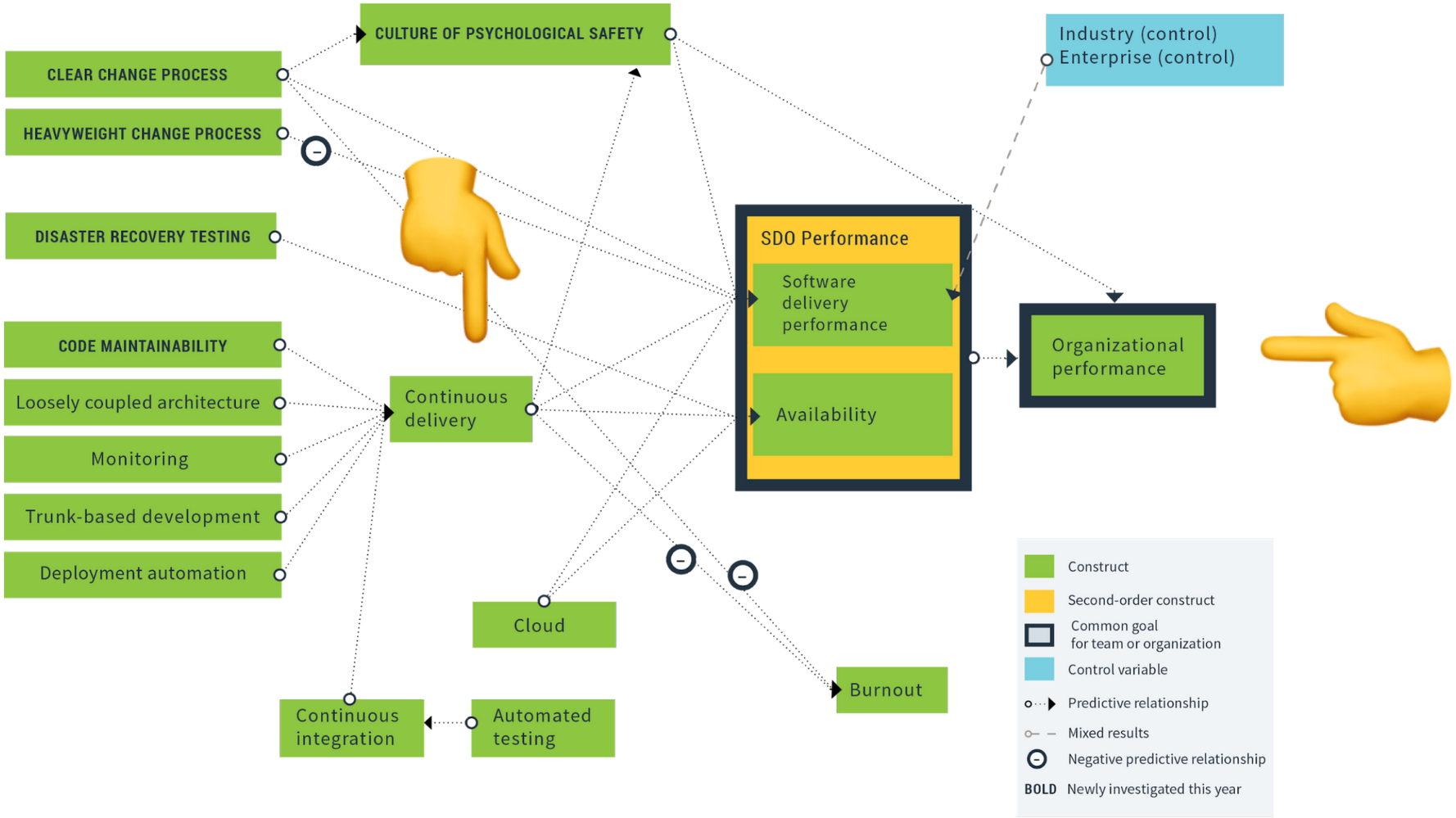
Matt Tuplin
Engineering
Director

"It is now **scientifically proven** that speed **and** stability predict organisational performance"

Jaz Chana
Technology
Director

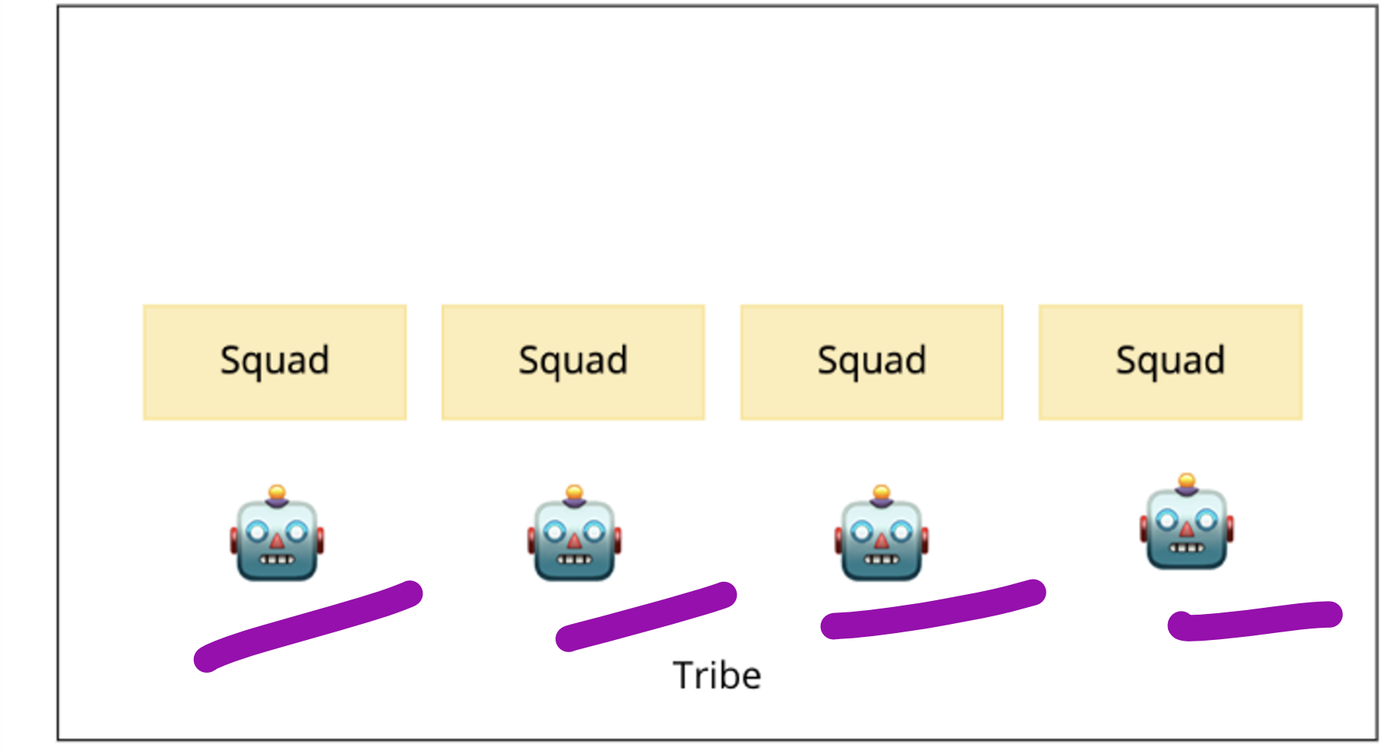
"Let's do it, have **DevOps-first mindset**. Let's start a **book club** on Accelerate"

Continuous Delivery was not "hip" any more. It was facts.

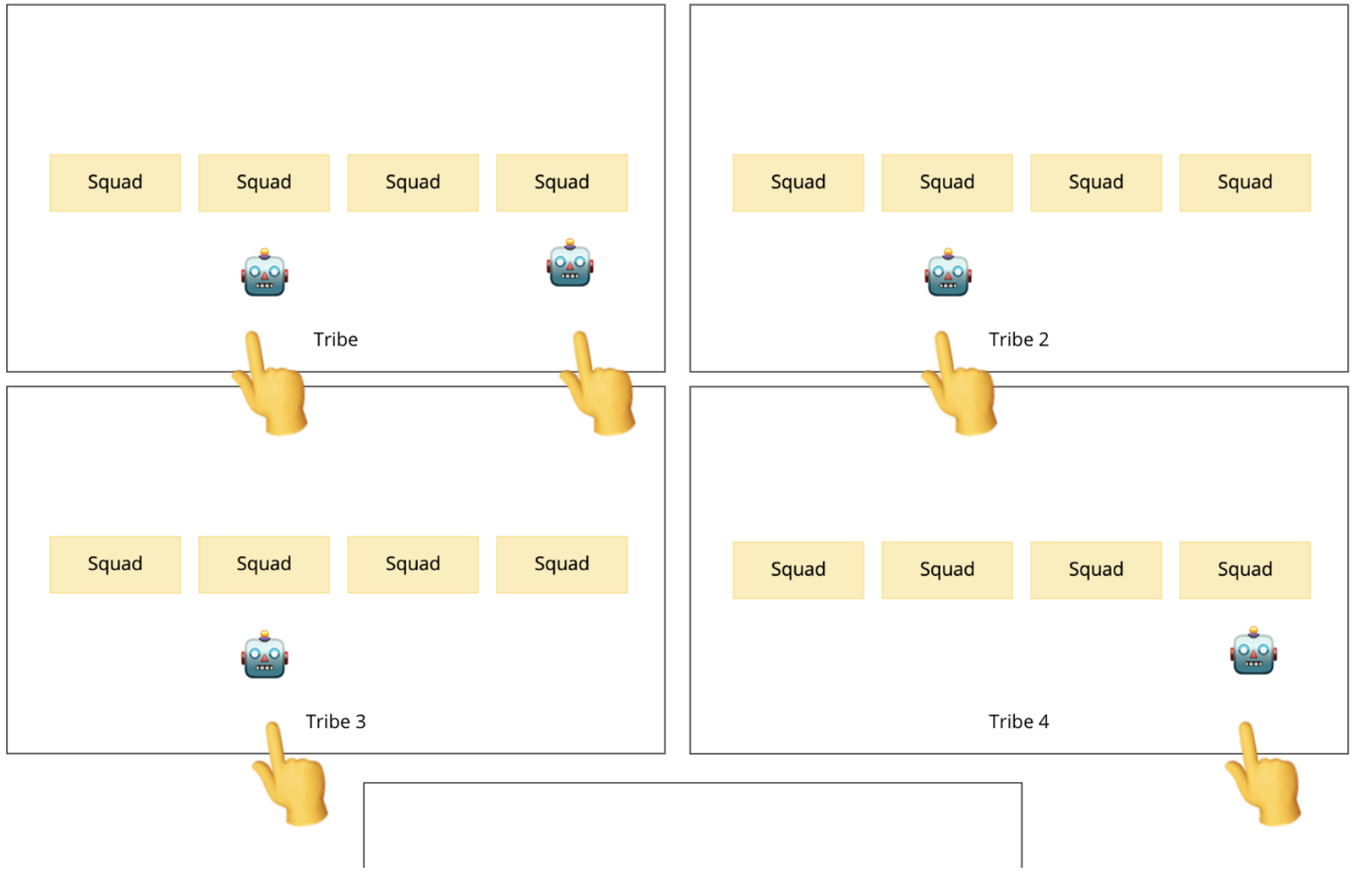


They hired a DevOps guy

Every squad gets an Automation Engineer

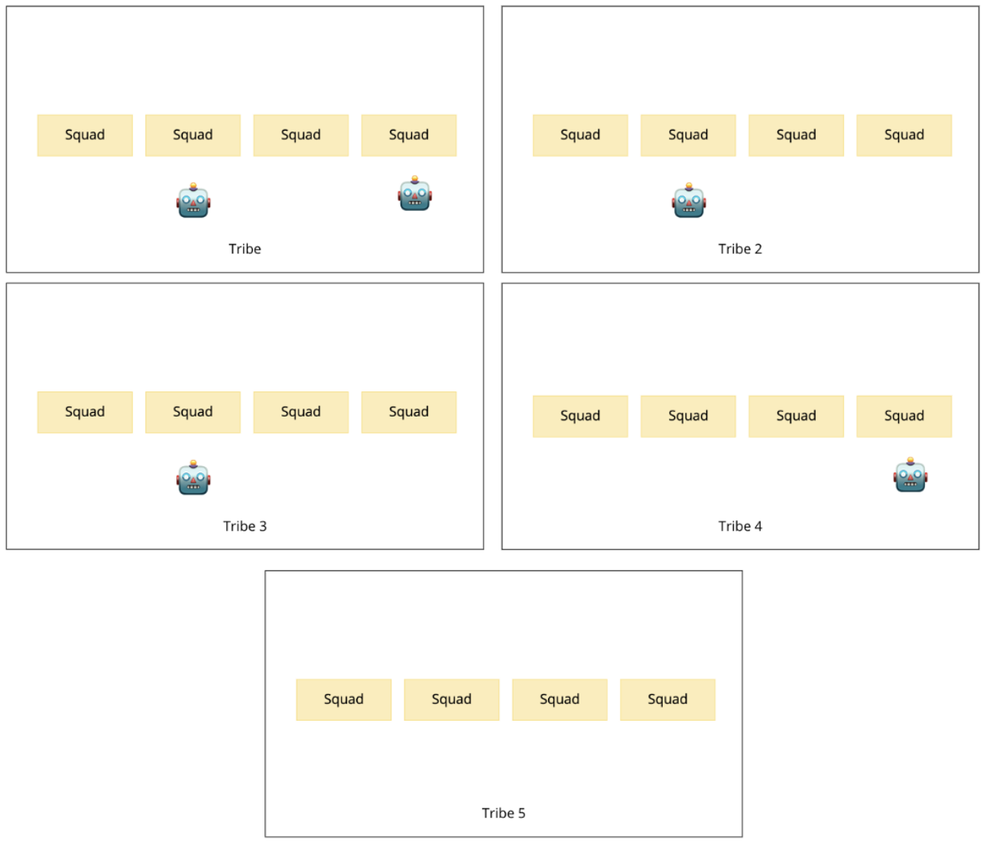


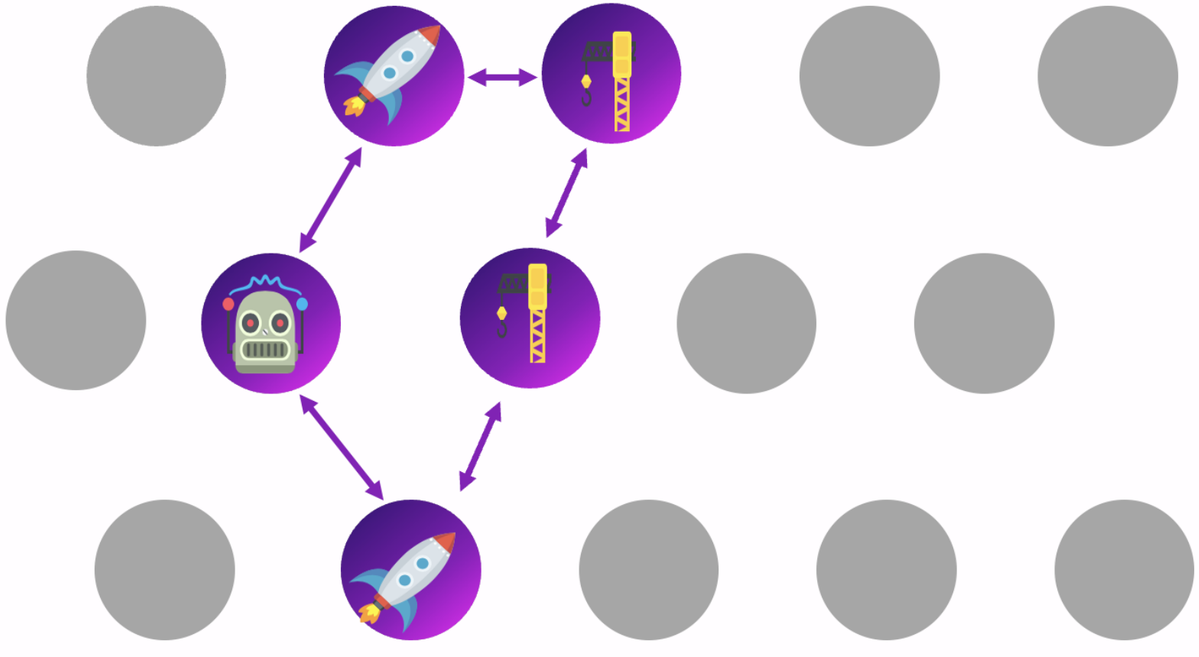
In reality...


Squad
Squad
Squad
Squad
Tribe
Squad
Squad
Squad
Squad
Tribe 2
Squad
Squad
Squad
Squad
Tribe 3
Squad
Squad
Squad
Squad
Tribe 4

Automation Community

DevOps Community





How do we make sure squads

keep improving

their DevOps practices?

1. Delivery Lead Time
2. Deployment Frequency

Speed

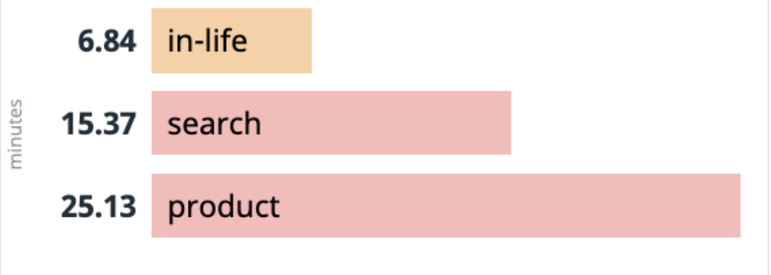
3. Time to restore service
4. Change fail rate

Stability

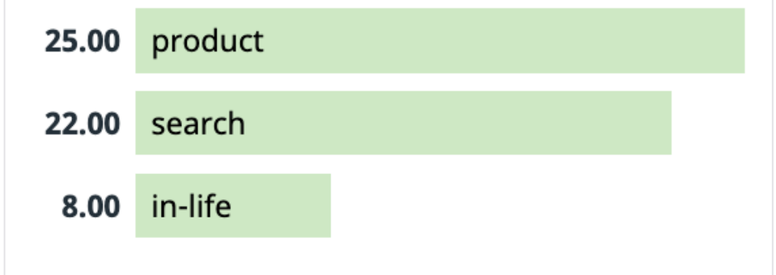
We can measure them!

Leader Board

Leader Board: Average Lead Time



Number of Deployments



Deployments

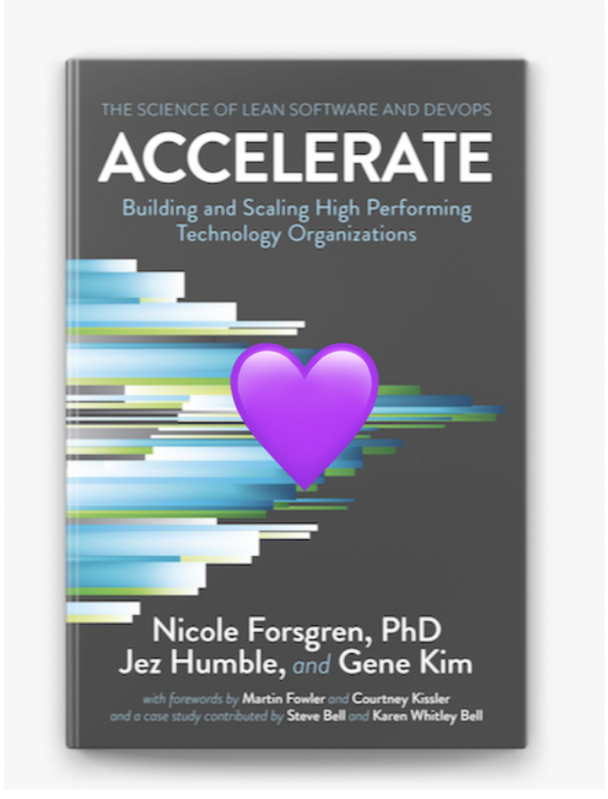
SQUAD	TRIBE	ENV	SERVICE	↑ AVERAGE	TOTAL DEPL
in-life	incremental-	prod	vaps-compor	6m 50s	8
search	ecommerce	prod	vehicle-searc	15m 22s	22
product	ecommerce	prod	product-com	25m 8s	25

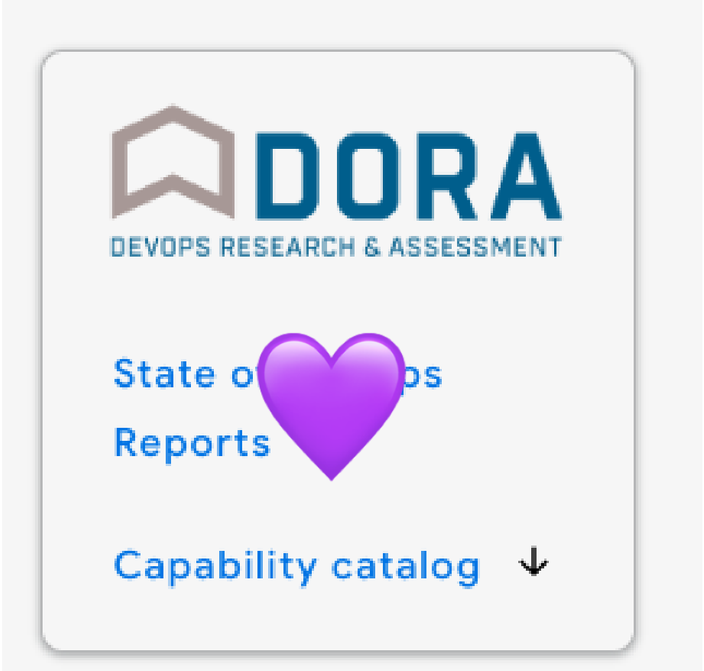
DATADOG

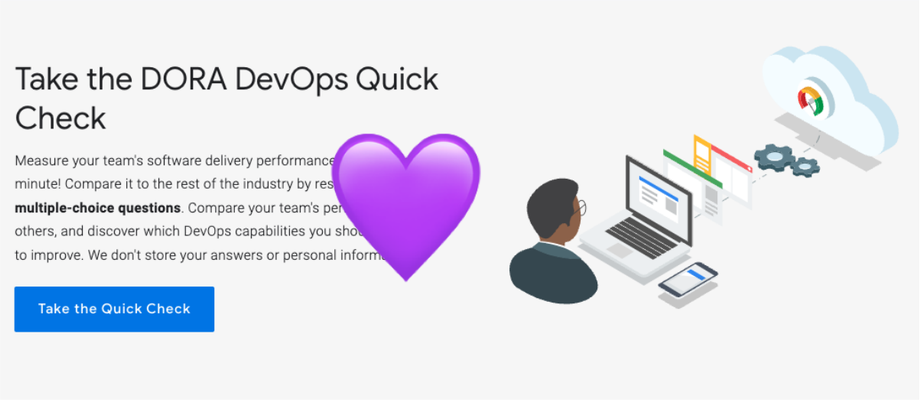
How do we make sure squads

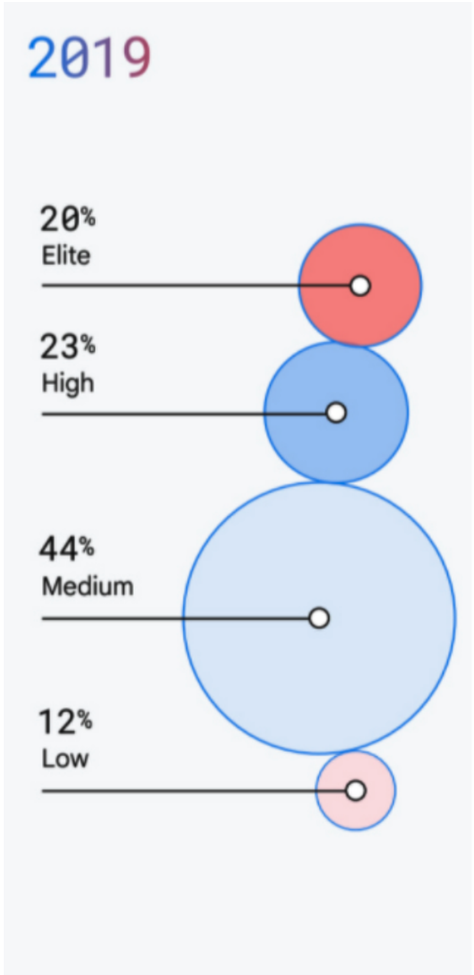
keep improving

their DevOps practices?


THE SCIENCE OF LEAN SOFTWARE AND DEVOPS
ACCELERATE
Building and Scaling High Performing Technology Organizations
Nicole Forsgren, PhD
Jez Humble, and Gene Kim
with forewords by Martin Fowler and Courtney Kissler
and a case study contributed by Steve Bell and Karen Whitley Bell


DORA
DEVOPS RESEARCH & ASSESSMENT
Reports
Capability catalog


Take the DORA DevOps Quick Check
Measure your team's software delivery performance
minute! Compare it to the rest of the industry by res
multiple-choice questions. Compare your team's per
others, and discover which DevOps capabilities you sho
to improve. We don't store your answers or personal inform
Take the Quick Check



👉

Capability catalog

Explore the technical, process, measurement, and cultural capabilities which drive higher software delivery and organizational performance. Each of the articles below presents a capability, discusses how to implement it, and how to overcome common obstacles. You can also learn how to deploy a program to implement these capabilities in our article "How to Transform."

TECHNICAL | PROCESS | MEASUREMENT | CULTURAL

Version control
A guide to implementing the right version control practices for reproducibility and traceability.
Learn more →

Continuous integration
Learn about common mistakes, ways to measure, and how to improve on your continuous integration efforts.
Learn more →

Deployment automation
Best practices and approaches for deployment automation and reducing manual intervention in the release process.
Learn more →

Trunk-based development
Prevent merge-conflict hassles with trunk-based development practices.
Learn more →

We use **maturity** to highlight **capabilities**

💡

☑ **Steps:**

1. Vote on each question as a group attempting to answer where you think your squad is at with these questions (⏱5 mins each)

2. Enter results in DORA DevOps Quick Check

3. Identify capabilities using the next board

💡 **Tip**

Lead time: Think about the journey of a commit from the moment it has been pushed to the moment code is ready to be used in production
Speed: Think about how often you *can* make changes to production - not only how many times you actually do (on demand is the best possible

Restore service: Factor in the time it takes typically to become aware of an incident

Failed Deployments: Think any change that requires manual intervention

Guess!: If you don't have any data to backup your answer, use your best guess based on your knowledge of the domain and code base!

1

Speed

For the primary application or service you work on, what is your **lead time for changes** (that is, how long does it take to go from code committed to code successfully running in production)?

- Lead Time: More than six months
- Lead Time: One to six months
- Lead Time: One week to one month
- Lead Time: One day to one week
- Lead Time: Less than one day
- Lead Time: Less than one hour

2

Speed

For the primary application or service you work on, how often does your organization **deploy code to production** or release it to end users?

- Deployment Frequency: Fewer than once per six months
- Deployment Frequency: Between once per month and once every six months
- Deployment Frequency: Between once per week and once per month
- Deployment Frequency: Between once per day and once per week
- Deployment Frequency: Between once per hour and once per day
- Deployment Frequency: On demand (multiple deploys per day)

3

Stability

For the primary application or service you work on, **how long** does it generally take to **restore service** when a service incident or a defect that impacts users occurs (for example, unplanned outage, service impairment)?

- Time to Restore Service: More than six months
- Time to Restore Service: One to six months
- Time to Restore Service: One week to one month
- Time to Restore Service: One day to one week
- Time to Restore Service: Less than one day
- Time to Restore Service: Less than one hour

4

Stability

For the primary application or service you work on, what percentage of changes to production or releases to users **result in degraded service** (for example, lead to service impairment or service outage) and subsequently **require remediation** (for example, require a hotfix, rollback, fix forward, patch)?

- Change Failure Rate: 0–15%
- Change Failure Rate: 16–30%
- Change Failure Rate: 31–45%
- Change Failure Rate: 46–60%
- Change Failure Rate: 61–75%
- Change Failure Rate: 76–100%

www.devops-research.com
DORA DevOps Quick Check

cinch

DevOps Checkpoints

Created by Apostolis Apostolidis
Last updated: Jan 11, 2023 by Varuna Venkatesh • 1 min read • 90 people viewed

🤷 What is a DevOps checkpoint

The 'DevOps checkpoint' is a squad wide collective workshop. The aim of the workshop is to take a deep dive in **your squad's DevOps maturity**, **continuous delivery** and **Observability** and **Monitoring.** The squad discusses, votes and decides on the most important improvements collectively. Ultimately, at the end of the workshop the squad has a clear vision and agrees on concrete actions to actually implement the improvements.

⏳ How long does it take

The workshop needs around 2 hours to complete. It can be done in 2 parts of 1 hour or in a 2 hour slot.

🏄 How do I arrange a DevOps checkpoint

Schedule some time with your squad. Use the Miro template 'DevOps Checkpoint v7 ' (available to cinch-labs team) as a basis for the workshop. Include your tribe's Principal Engineer to help facilitating. The 'source' for the DevOps Checkpoint Miro template is at DevOps Checkpoint.

🕰️ How often should we perform a DevOps checkpoint

We intend to run these workshops quarterly or biyearly so that we have a clear emphasis on continuous improvement.

🚴 How do I a run a DevOps checkpoint

If you have scheduled the time in, please take a look at Running a DevOps Checkpoint for more detail on how to run the workshop.

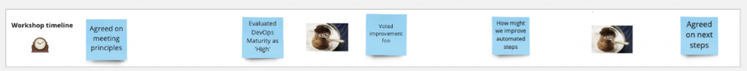
Running a DevOps Checkpoint

Created by Apostolis Apostolidis
Last updated: Jun 10, 2022 by Sergejs Katusenoks • 5 min read • 30 people viewed

The goals and aims of DevOps Checkpoints are outlined but how do I run one for my squad?

Given that you have some time scheduled with the squad, you can facilitate the session.

✅ While going through the session, make sure you update the timeline as it does help with quickly recapping at the end of each session or at the start of the second session. Or simply when you come back to it in 3 months time



Evaluating maturity

Use the workshop to:

✅ what DevOps maturity the squad is at according to DORA research

✅ what capabilities the squad should emphasise based on the maturity estimated from the squad's answers to the 4 'Accelerate' questions

Welcome to your DevOps Checkpoint

DevOps Checkpoint

Agree on meeting principles

What is a DevOps Checkpoint?

- Squad level **self-evaluation** of DevOps practices & patterns to **identify** areas of improvements

- We want to be **continuously improving** so serves as a benchmark on which we can build on

- Feeds into **service readiness** for new services or new features

- **Regular** (quarterly or biyearly) review of **DevOps maturity**

- View all past DevOps checkpoints on Confluence

Agenda

- Deployment Principles & Schematic Approach
- DevOps maturity
- Design current deployment approach
- Identify areas of improvement
- Review & create DevOps Summary

Outcomes from today's session

We have identified our DevOps maturity capabilities we need to work on and we have some collectively agreed improvement actions

cinch

Principles for today's session

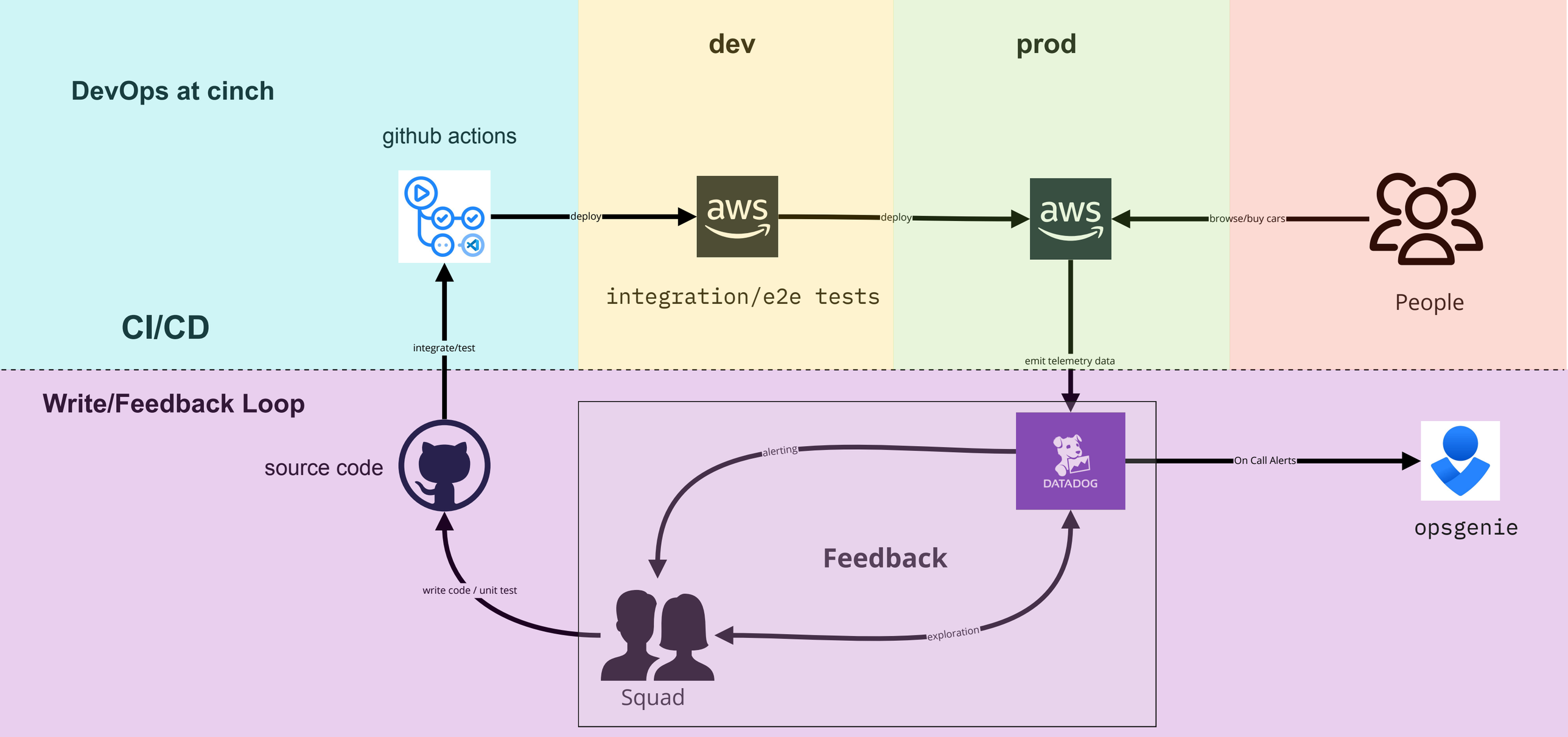
- Everyone’s input is equally valued
- We keep things simple and jargon free
- Share your experiences
- Close down Teams and another things that might distract you
- Keep conversations on track, and make it clear when we're moving onto another topic
- We always respect each other
- Have fun

Deployment Principles

Set the DevOps scene


MAERSK LINE


SUPPORT


DevOps at cinch
dev
prod
github actions
deploy
deploy
browse/buy cars
integration/e2e tests
People
CI/CD
integrate/test
emit telemetry data
Write/Feedback Loop
source code
alerting
DATADOG
On Call Alerts
opsgenie
Feedback
write code / unit test
exploration
Squad

Accelerate metrics

Through six years of research, the DevOps Research and Assessment (DORA) team has identified four key metrics that indicate the performance of a software development team:

- **Deployment Frequency**—How often an organization successfully releases to production
- **Lead Time for Changes**—The amount of time it takes a commit to get into production
- **Change Failure Rate**—The percentage of deployments causing a failure in production
- **Time to Restore Service**—How long it takes an organization to recover from a failure in production

At a high level, Deployment Frequency and Lead Time for Changes measure velocity, while Change Failure Rate and Time to Restore Service measure stability. And by measuring these values, and continuously iterating to improve on them, a team can achieve significantly better business outcomes.

Deployment frequency

Lead time for changes

Change failure rate

Time to restore

How might we strike the right balance between velocity and stability?

How might we minimise time to acknowledge and time to restore?

cinch

CI/CD principles

A few principles that guide what good CI/CD mechanisms look like.

Principles:

- shouldn't feel like a rule
- should feel like suggestions, not commandments
- are things that we believe contribute to good CI/CD mechanisms

😎 A tale of confidence

The below principles should lead to the most important property of a good CI/CD practice: having confidence in changing your software.

🔀 Commit and push often

Smaller changes are less risky and they adhere to "Lean Principles". They also avoid "sunk cost fallacy".

🚀 Be comfortable with your change going straight to prod

Take a Continuous Integration and Continuous Deployment (CI/CD 👀) approach to making changes to our software. Choose Continuous deployment by default.

> CD can also interchangeably stand for Continuous Delivery. When practicing Continuous Delivery, code is integrated and delivered to be evaluated, but not promoted to prod. Continuous Deployment includes promoting atomic changes to production without a manual gate. The differences are explained by this Google document

☠️ Make pipelines visible and transparent

Pipelines are easy to understand and observable. They communicate in the right place, at the right time. When something fails, it is clear what it is. They are easy to measure.

Measurable pipelines allow for behaviour to be observed over time. We can focus on things that matter, like:

- increasing reliability
- reducing cycle time.

💪 Reduce toil

Make our lives easier by reducing toil. Reducing toil helps us to focus on small, frequent, and reliable changes.

▸ What is toil?

💜 Automation is trustworthy

Automated tasks are consistent and predictable. They work a very high percentage of the time. Automated tasks help to improve quality in the delivered product.

🌏 Life as code

Code-ification over click-ifaction. Prefer code declarations over manual configuration.

✒️ Make pragmatic, informed decisions about security

Security is important. It's a balancing act between high levels of security and high levels of throughput. Tooling helps to provide visibility of security concerns.

🏎️ Create timely feedback loops

When running automated tasks - locally or remotely - feedback is timely. Getting quick feedback on the effect of changes is important. It allows focusing on the change itself at the time the change is happening.

DevOps Maturity

Evaluate DevOps Maturity

www.devops-research.com

DORA DevOps Quick Check

☑ **Steps:**

1. Vote on each question as a group attempting to answer where you think your squad is at with these questions (⏱5 mins each)

2. Enter results in [DORA DevOps Quick Check]

3. Identify capabilities using the next board

💡 **Tip** 👉

Lead time: Think about the journey of a commit from the moment it has been pushed to the moment code is ready to be used in production
Speed: Think about how often you *can* make changes to production - not only how many times you actually do (on demand is the best possible. Ask yourselves:
- Can you complete multiple deploys in less than two hours? - Can sustain that rate without negative impacts (e.g., burnout) for weeks?

Restore service: Factor in the time it takes typically to become aware of an incident

Failed Deployments: Think any change that requires manual intervention

Guess!: If you don't have any data to backup your answer, use your best guess based on your knowledge of the domain and code base!

1

Speed

For the primary application or service you work on, what is your **lead time for changes** (that is, how long does it take to go from code committed to code successfully running in production)?

- Lead Time: More than six months
- Lead Time: One to six months
- Lead Time: One week to one month
- Lead Time: One day to one week
- Lead Time: Less than one day
- Lead Time: Less than one hour

2

Speed

For the primary application or service you work on, how often does your organization **deploy code to production** or release it to end users?

- Deployment Frequency: Fewer than once per six months
- Deployment Frequency: Between once per month and once every six months
- Deployment Frequency: Between once per week and once per month
- Deployment Frequency: Between once per day and once per week
- Deployment Frequency: Between once per hour and once per day
- Deployment Frequency: On demand (multiple deploys per day)

3

Stability

For the primary application or service you work on, **how long** does it generally take to **restore service** when a service incident or a defect that impacts users occurs (for example, unplanned outage, service impairment)?

- Time to Restore Service: More than six months
- Time to Restore Service: One to six months
- Time to Restore Service: One week to one month
- Time to Restore Service: One day to one week
- Time to Restore Service: Less than one day
- Time to Restore Service: Less than one hour

4

Stability

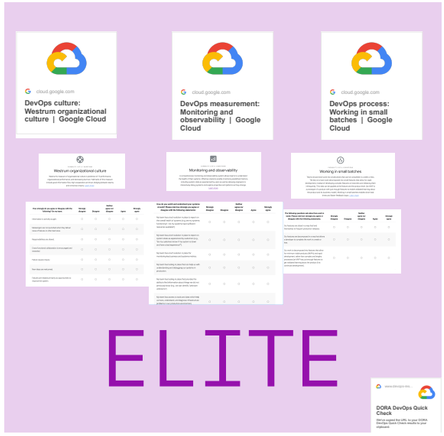
For the primary application or service you work on, what percentage of changes to production or releases to users **result in degraded service** (for example, lead to service impairment or service outage) and subsequently **require remediation** (for example, require a hotfix, rollback, fix forward, patch)?

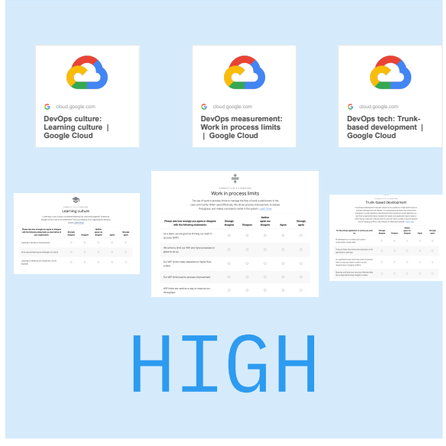
- Change Failure Rate: 0–15%
- Change Failure Rate: 16–30%
- Change Failure Rate: 31–45%
- Change Failure Rate: 46–60%
- Change Failure Rate: 61–75%
- Change Failure Rate: 76–100%

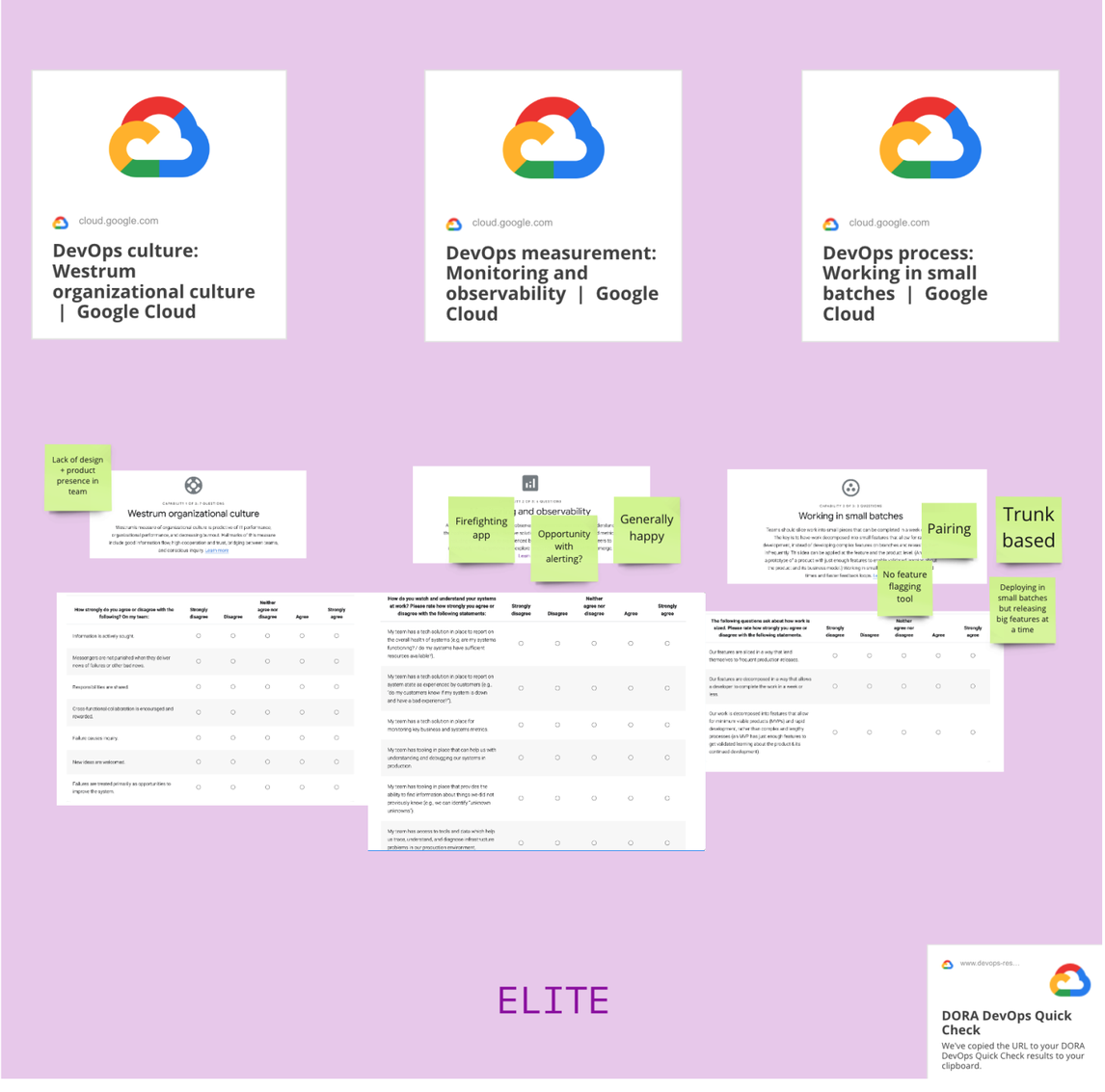
cinch

LOW MATURITY

HIGH MATURITY







IMPROVEMENT AREAS PERFORMANCE COMPARISON ELITE PERFORMERS

Key DevOps capabilities recommended for you

Based on your performance profile, we think you should work on the three capabilities listed below. Based on your responses to the capabilities assessment, **we recommend you focus on monitoring and observability first.**

Monitoring and observability

KEY FOCUS AREA

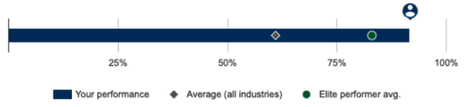
A comprehensive monitoring and observability system allows teams to understand the health of their systems. Effective solutions enable monitoring predefined metrics, including system state as experienced by users as well as allowing engineers to interactively debug systems and explore properties and patterns as they emerge.

View solution

Working in small batches

Teams should slice work into small pieces that can be completed in a week or less. The key is to have work decomposed into small features that allow for rapid development, instead of developing complex features on branches and releasing them infrequently. This idea can be applied at the feature and the product level. (An MVP is a prototype of a product with just enough features to enable validated learning about the product and its business model.) Working in small batches enables short lead times and faster feedback loops.

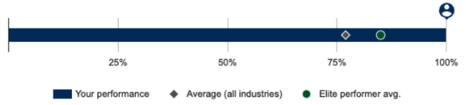
View solution



Westrum organizational culture

Westrum's measure of organizational culture is predictive of IT performance, organizational performance, and decreasing burnout. Hallmarks of this measure include good information flow, high cooperation and trust, bridging between teams, and conscious inquiry.

View solution



Design Current Approach

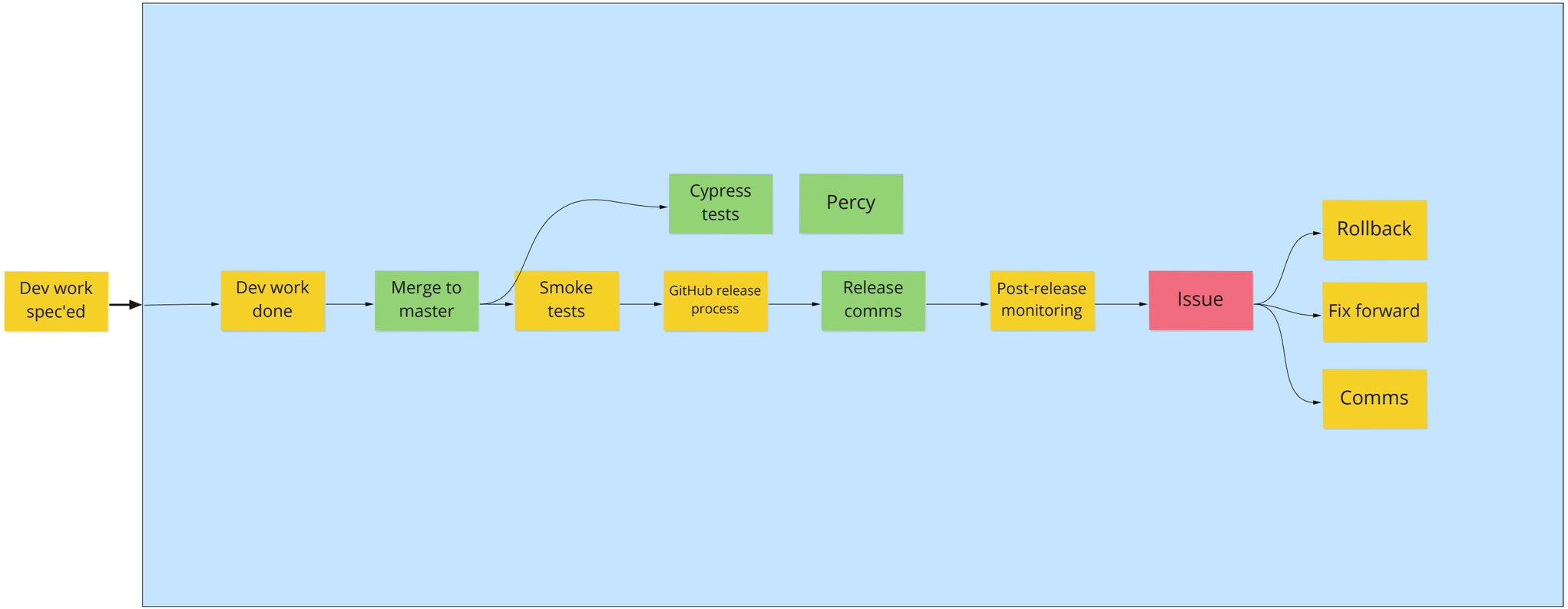
Design Path to Production
Vote for the things to improve

☑ Steps:

1. Design your squad's current deployment process (⏱10 mins)

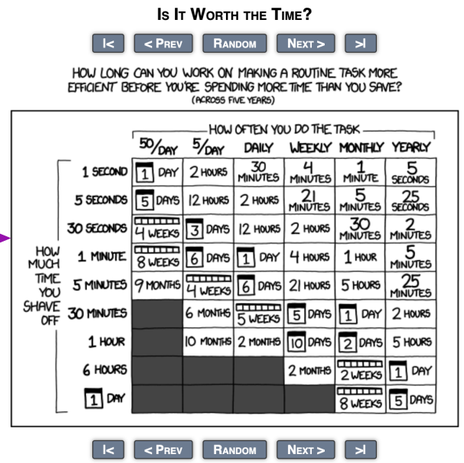
2. Identify areas of improvements by applying the sticky notes to the deployment process (⏱10 mins)

3. Vote on the improvements you think are the most valuable? (⏱5 mins)



💡 Tip 👉

Deploy to prod: think about small batches, how do I know that our 'thing' still works as expected
Testing in prod: what do you do after a deployment to prod (manual or automated)
Manual vs Automated: indicate what is manual and what is automated using colour coding
Be ambitious: what are the bits you always wanted to improve?
CI/CD principles: think about the principles outlined on the cinch blueprint



Manual work

Automated work

Failed Deployment

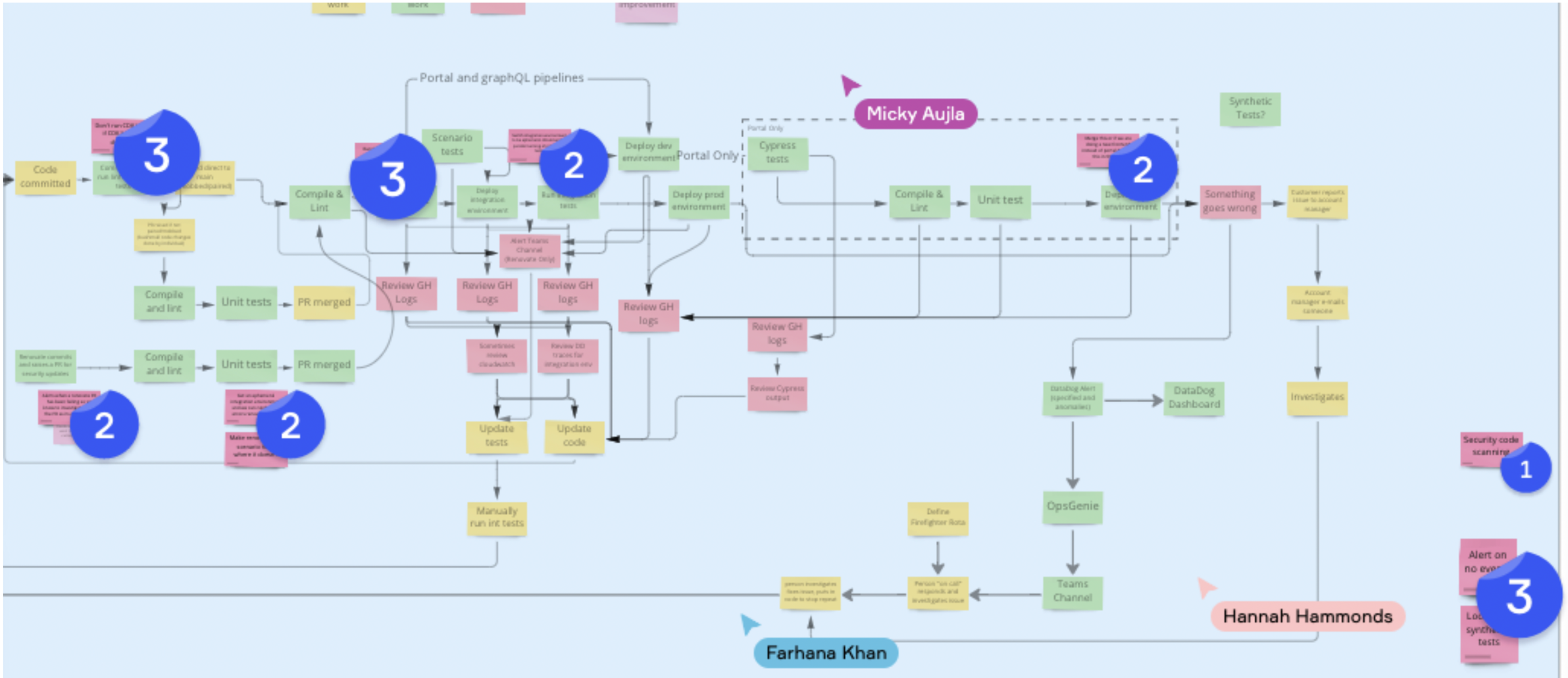
Improvement

Manual work

Automated work

Failed Deployment

cinch


Portal and graphQL pipelines
Micky Aujla
Synthetic Tests?
Code committed
Compile & Lint
Scenario tests
Deploy dev environment
Portal Only
Cypress tests
Compile & Lint
Unit test
Something goes wrong
Deploy prod environment
Compile and lint
Unit tests
PR merged
Review GH Logs
Review GH logs
Review Cypress output
DataDog Dashboard
Investigates
OpsGenie
Teams Channel
Manually run int tests
Update tests
Update code
Hannah Hammonds
Farhana Khan

DevOps Checkpoint Summary

<Month> 2022

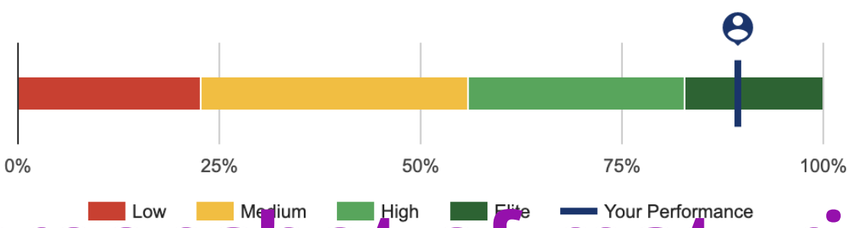
DevOps maturity is... <Maturity>!

Your software delivery performance

Your performance:

Elite

You're performing better than 89% of State of DevOps Survey respondents



SAMPLE - Enter screenshot of maturity here

Share results | Start over

IMPROVEMENT AREAS | PERFORMANCE COMPARISON | ELITE PERFORMERS

cinch | **<Squad Name>**

Based on this maturity, the capabilities the squad should be focusing on are:

Elite capabilities

add a link to this from your DORA check tool results

IMPROVEME

ELITE PERFORMERS

Key DevOps capabilities for elite performers

When we analyzed the responses of other **elite performers**, we found that working on these three capabilities had the most impact on improving their software delivery performance.

Help me prioritize

Rank your responses to a series of 3-12 questions for each of these three capabilities, to determine how you perform and which to consider focusing on first.

SAMPLE - Enter screenshot of capabilities here

Westrum organizational culture
Learn more →

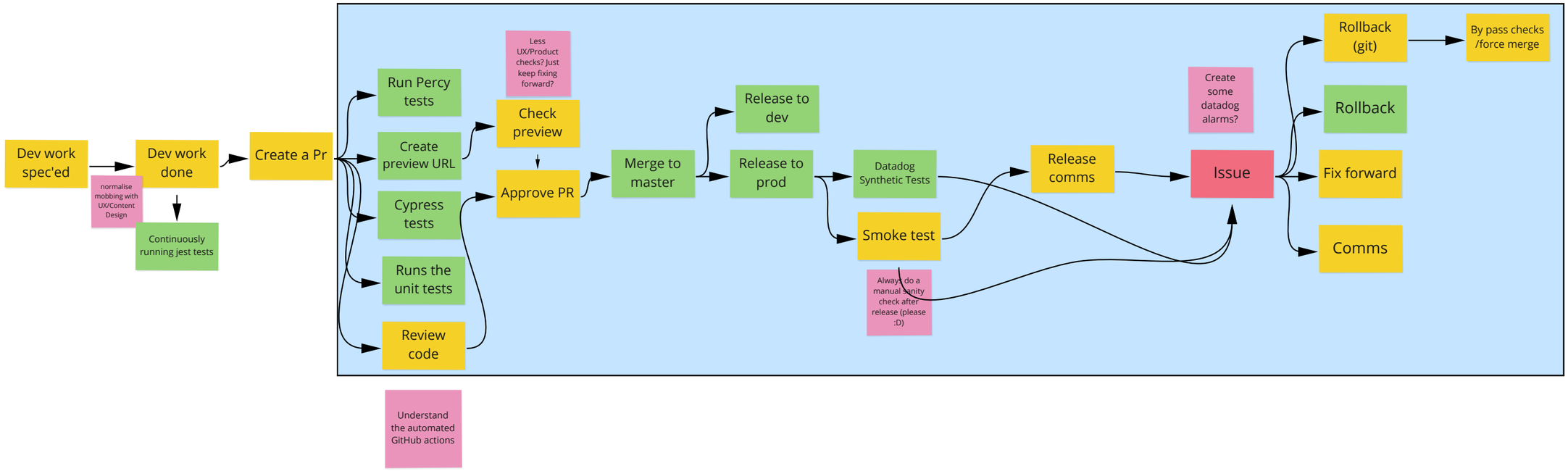
Monitoring and observability
Learn more →

Working in small batches
Learn more →

cinch | <Squad Name>

Software delivery flow & top improvements identified

SAMPLE - Copy delivery flow here



cinch | <Squad Name>

Top improvements identified

1.	Improvement 1
2.	Improvement 2
3.	Improvement 3
4.	Improvement 4
5.	Improvement 5

SAMPLE - Enter improvements here

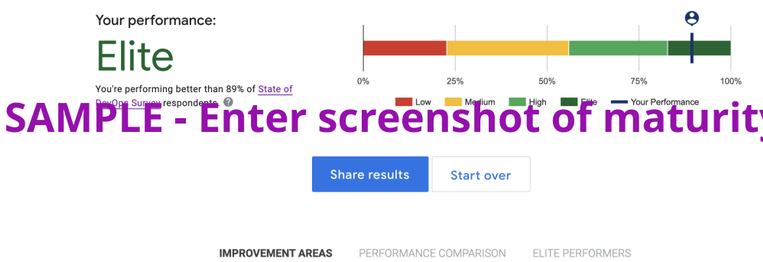
cinch | **<Squad Name>**

DevOps Checkpoint Summary
<Month> 2022

Go to workshop - ENTER LINK HERE

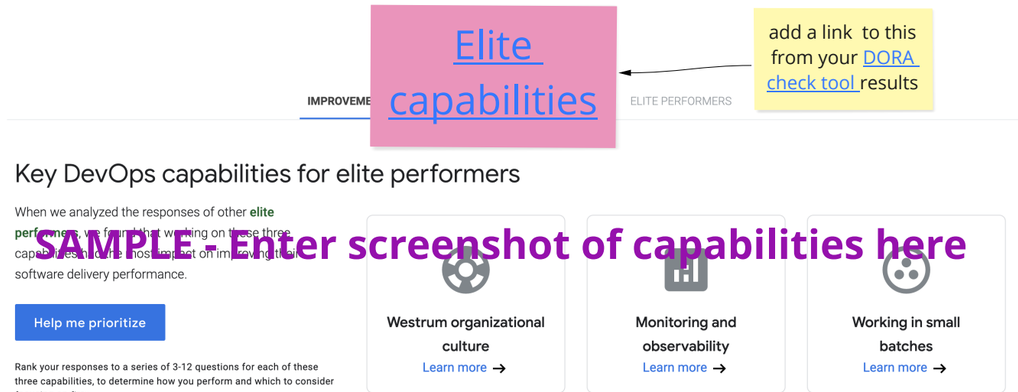
DevOps maturity is... <Maturity>!

Your software delivery performance



SAMPLE - Enter screenshot of maturity here

Based on this maturity, the capabilities the squad should be focusing on are:

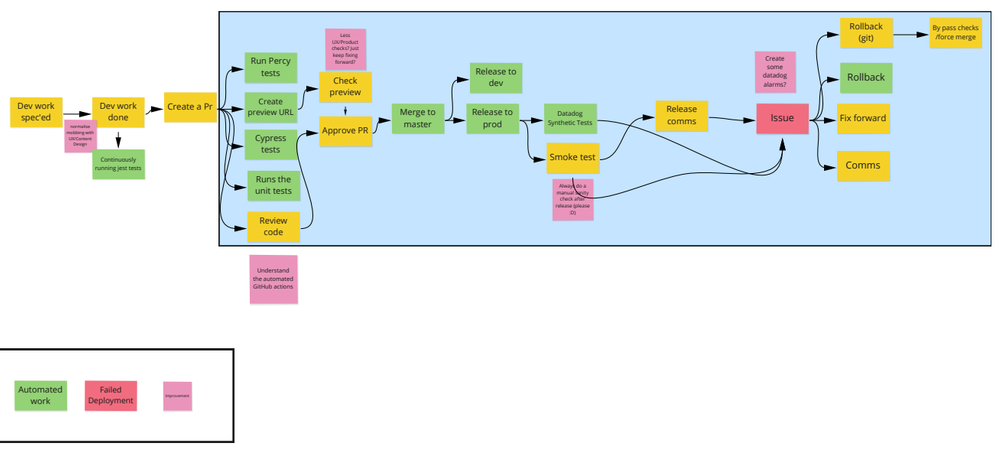


SAMPLE - Enter screenshot of capabilities here

cinch | <Squad Name>

Software delivery flow & top improvements identified

SAMPLE - Copy delivery flow here



cinch | <Squad Name>

Top improvements identified

1. Improvement 1
2. Improvement 2
3. Improvement 3
4. Improvement 4

SAMPLE - Enter improvements here

5. Improvement 5

cinch | <Squad Name>

Partner Tooling | 2023-01 | DevOps Checkpoint

Created by Varuna Venkatesh
Last updated: Jan 11, 2023 • 1 min read • 2 people viewed

👥 Squad	Partner Tooling
⏳ Workshop Date	10 Jan 2023
👤 Facilitator	@Varuna Venkatesh
🏄 Miro Board	Board Link
🌻 DevOps Maturity	Elite
👌 Capabilities	Monitoring and observability

DevOps Checkpoint Summary
Jan 2023

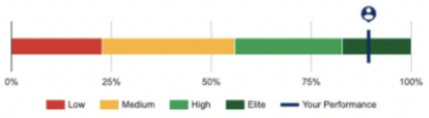
DevOps maturity is... Elite!

Your software delivery performance

Your performance:

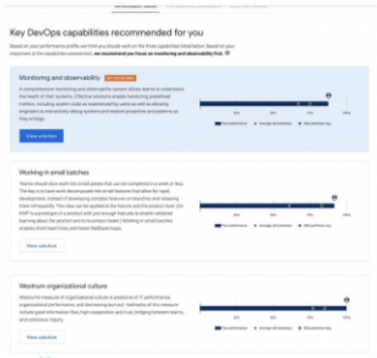
Elite

You're performing better than 89% of State of DevOps Survey respondents.



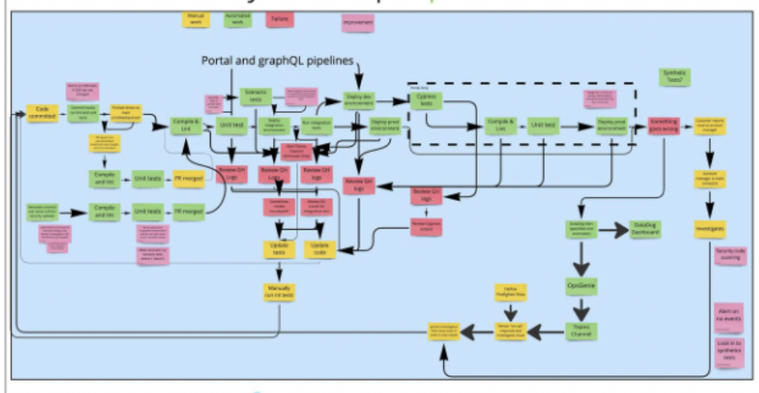
cinch | Partner Tooling

Based on this maturity, the capabilities the squad should be focusing on are:



cinch | Partner Tooling

Software delivery flow & top improvements identified



cinch | Partner Tooling

Top improvements identified

Alert on no events

Commit Hook - Don't run CDK tests if CDK has not changed

Run CDK tests in parallel with unit tests

Alerts when a renovate PR has been failing so we know to investigate or kill the PR as its not needed

Switch integration environment to be ephemeral. Would enable parallel running of integration tests + catch more errors renovate causes

Partner Portal - Merge Dev & Prod deployment pipelines

cinch | Partner Tooling

Past DevOps checkpoints

Created by Apostolis Apostolidis
Last updated: Feb 14, 2022 • 1 min read • 25 people viewed

Title	Workshop Date	Suggestions	DevOps Maturity	Miro Board	Capabilities
[redacted] 2022-02 | DevOps Checkpoint	02 Feb 2022		Medium	https://miro.com/app/board/[redacted]=/?invite_link_id=[redacted]	CI, CD and Architecture
[redacted] | 2022-08-11 | DevOps Checkpoint	11 Aug 2022		High (aspirational as not in prod yet)	https://miro.com/app/board/[redacted]=/?share_link_id=[redacted]	Learning Culture, Work in Process Limits & Trunk
[redacted] | 2022-02 | DevOps Checkpoint	01 Mar 2022		High	Hello Miro	Learning Culture, WIP Limits, Trunk-based Development
[redacted] | 2022-02 | DevOps Checkpoint	15 Feb 2022		High	https://miro.com/app/board/[redacted]=/?invite_link_id=[redacted]	
[redacted] | 2022-02 | DevOps Checkpoint	09 Feb 2022		High	https://miro.com/app/board/[redacted]=/?invite_link_id=[redacted]	
[redacted] | 2021-10 | DevOps Checkpoint	15 Oct 2021		High	Hello Miro	Learning Culture, WIP Limits, Trunk-based Development
[redacted] | 2023-01 | DevOps Checkpoint	10 Jan 2023		Elite	Board Link	Monitoring and observability
[redacted] | 2022-06 | DevOps Checkpoint	16 Jun 2022		Elite	https://miro.com/app/board/[redacted]=/?share_[redacted]	Westrum Organisational Culture, Monitoring and Observability, Working in Small Batches
[redacted] | 2022-06-09 | DevOps Checkpoint Summary	09 Jun 2022		Elite	https://miro.com/app/board/[redacted]=/	Westrum organizational culture, Monitoring and Observability, Working in small batches
[redacted] | 2021-10 | DevOps Checkpoint	19 Oct 2021		Elite	https://miro.com/app/board/[redacted]/	Westrum organisational culture, monitoring and observability, working in small batches

Speed is more about ability to deploy

Teams love feeling "elite"

Operational Performance always felt "missing"

Stability metrics are hard to measure

Improve the thing, don't measure it

The bottleneck is rarely in CI/CD

What should we be measuring in 2023?

How to "measure" operational performance?

Should we actually slow down?

How does any of this change for serverless?

Thank you!