What would happen if you ran your test suite 10 times in a row for every build?

What are some benefits your team would realize if your build times were cut in half? If they doubled?

https://conversations.dora.dev/

DORA & SRE & Platforms & Labs

SRE Day SF 2024

smcghee@google.com
@stevemcghee

June 2024

% whoami

- has worked here **17 years, 9 months**
- last joined **5 years, 2 months ago**
- first joined **19 years, 5 months ago**
- has joined **twice**
- has had **7 roles**
- has worked from **8 offices**
- has worked from **2 countries**

Steve McGhee
Reliability Advocate
smcghee@google.com
@stevemcghee

I help companies to build robust, reliable systems with resilient, sustainable teams in the cloud.

SRE on: ads, gmail, android, fiber, youtube, cloud

A Recipe

01 #hook

02 #lecture (SRE, DORA, Platforms)

03 #lab

04 #production

01

#hook

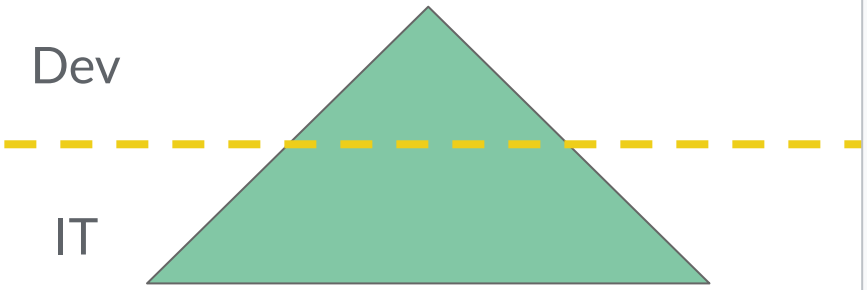
Q: Can you build
99.99% things
on
99.9% things?

Q: Can you build **99.99%** things (apps)
on **99.9%** things (infra)?

Yes.

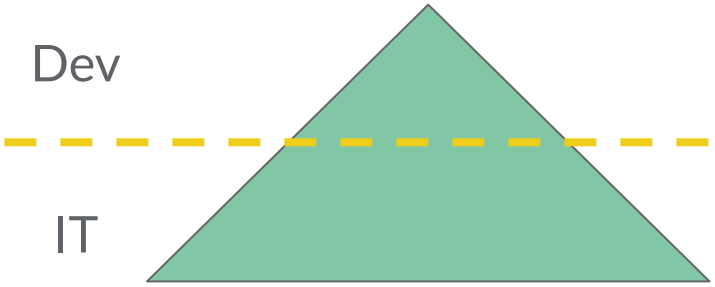
You can build
more reliable things
on top of
less reliable things.

Remember RAID?



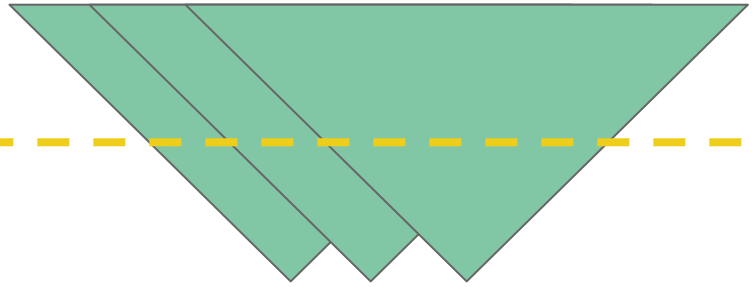
Worked great, for
a long time

Common
mental model



Worked great, for a long time

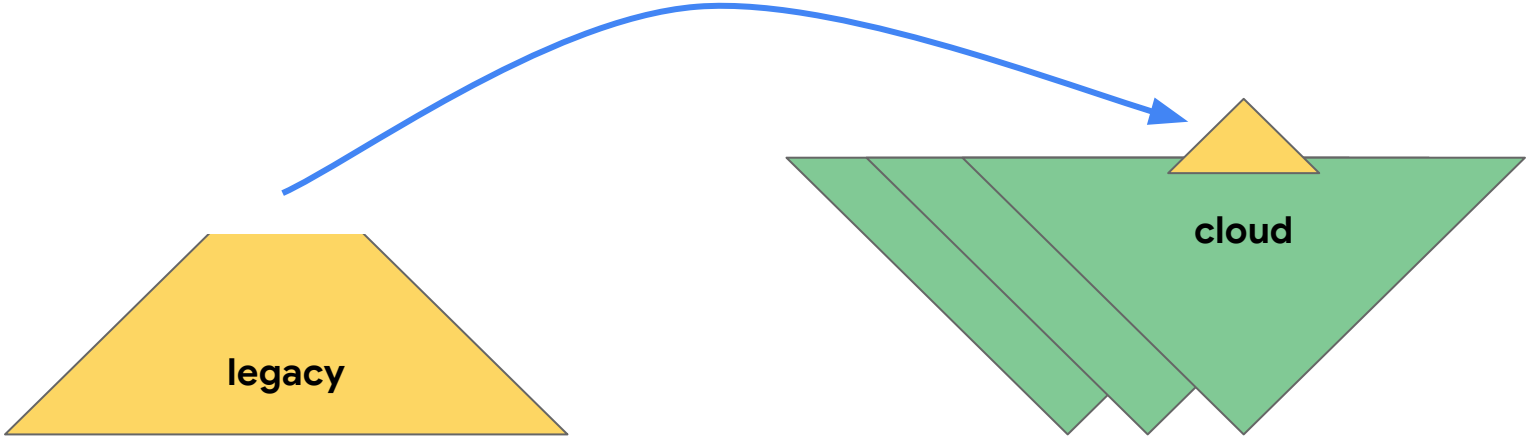
Common mental model



Cloud is here, though.

(because scale, mostly)

((You can't **buy more nines** for your VM in Cloud))



Infrastructure changes **can't fix** the app.

** even though they **used to**.*

Why? Why now?

- Distributed Systems - "Always slightly broken"
- Warehouse **Scale** Computing
- SaaS, global audience
- Consumers expect "*always-on*"

"Operations as a **competitive advantage** (and occasionally a “strategic weapon”).

This advantage is the ability to consistently create and deploy **reliable software** to an **unreliable platform** that **scales horizontally**."

Operations is a competitive advantage... (Secret Sauce for Startups!)

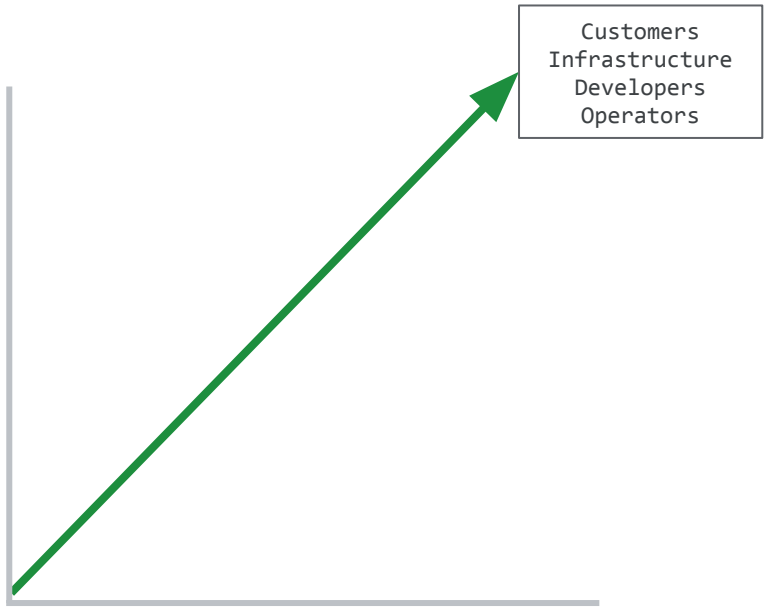
by Jesse Robbins | @jesserobbins | October 23, 2007

02

#lecture

(SRE, DORA, Platforms)

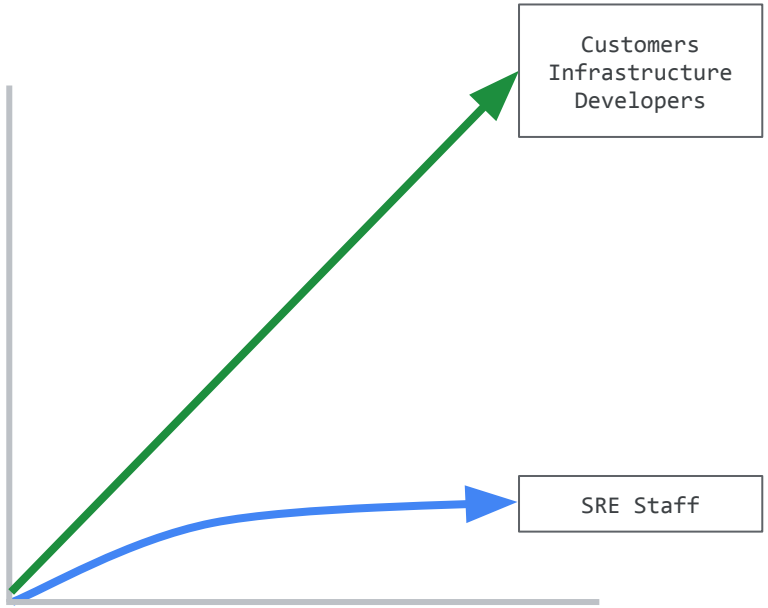
Why SRE? Scaling operations problem



Linear scaling

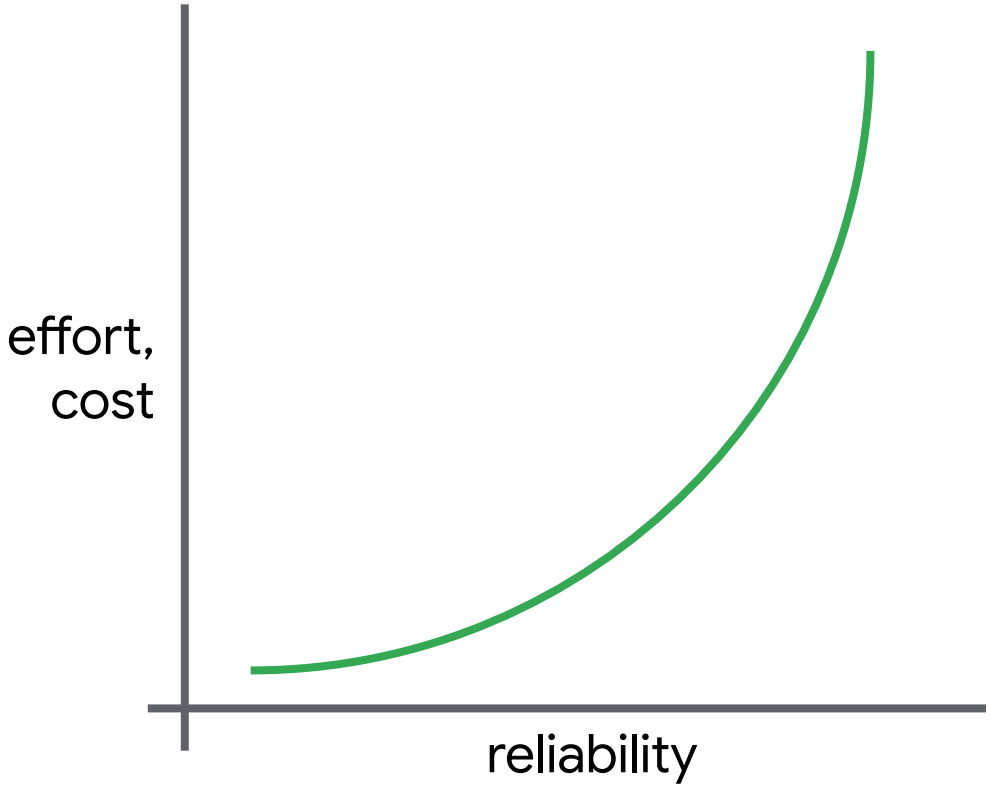
The number of operators needs to scale proportionally with the size and scope of any product they maintain

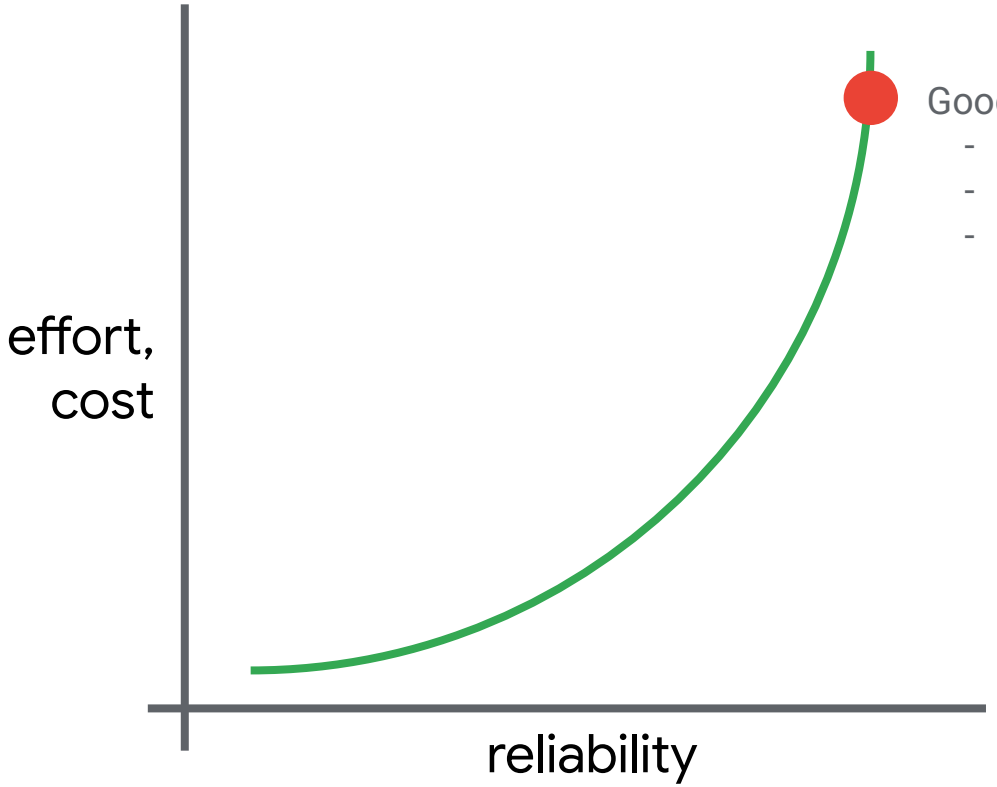
Why SRE? Scaling operations problem



Sublinear scaling solution

- Automation -> self-healing
- Standardized tooling
- Community of practice
- Shared responsibility


effort,
cost
reliability



Google Ads (ca **2003**)

- seconds matter $$$
- high margins
- huuuuuge scale (growing)

Failure Domains
know your abstractions (zones, regions, clusters, etc)

Avoid:

- **Coordinated Failure** - isolated change
- **Cascading Failure** - plan for containment

Via:

- **Gradual Change** - fail early, fail small

Design for Success - in the face of failure

- Provide **"exit paths"** when failure domains ... ***fail***.

  → **"run from your problems"** ;)

- Avoid **coordinated, cascading failures**

Provide **Generic Mitigations** in your platform:

- ❏ drain, spill, rollback,
- ❏ freeze, degrade, hospitalize,
- ❏ upsize, blocklist

https://www.oreilly.com/content/generic-mitigations

"Are we getting better?"

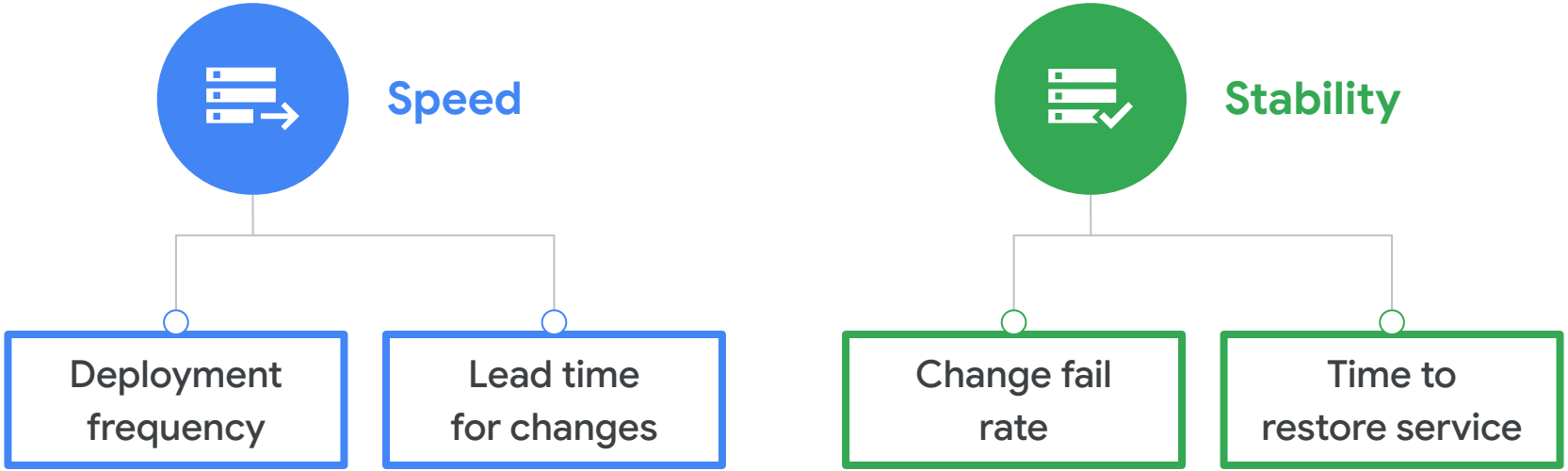
Speed

Stability

DORA
DEVOPS RESEARCH & ASSESSMENT

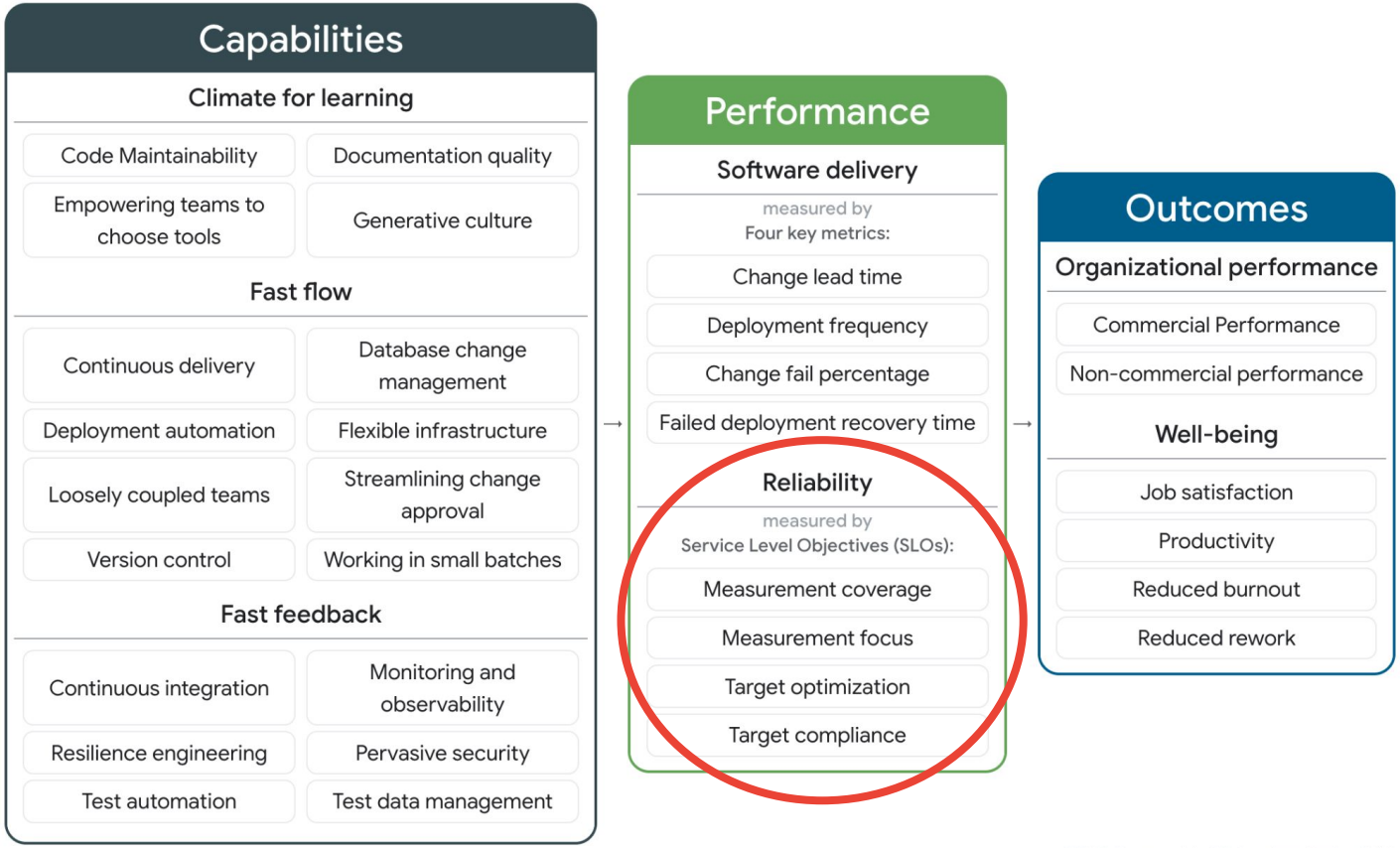
"Are we getting better?"

The 4 DORA Metrics



DORA Core Model

View mode: ○ summary ◉ detail



DORA Core model v2.0.0 download as PNG

START HERE

Measure

1. Measure 4 DORA metrics

2. Determine bottleneck(s)

3. Choose **capability** to improve next

Improve

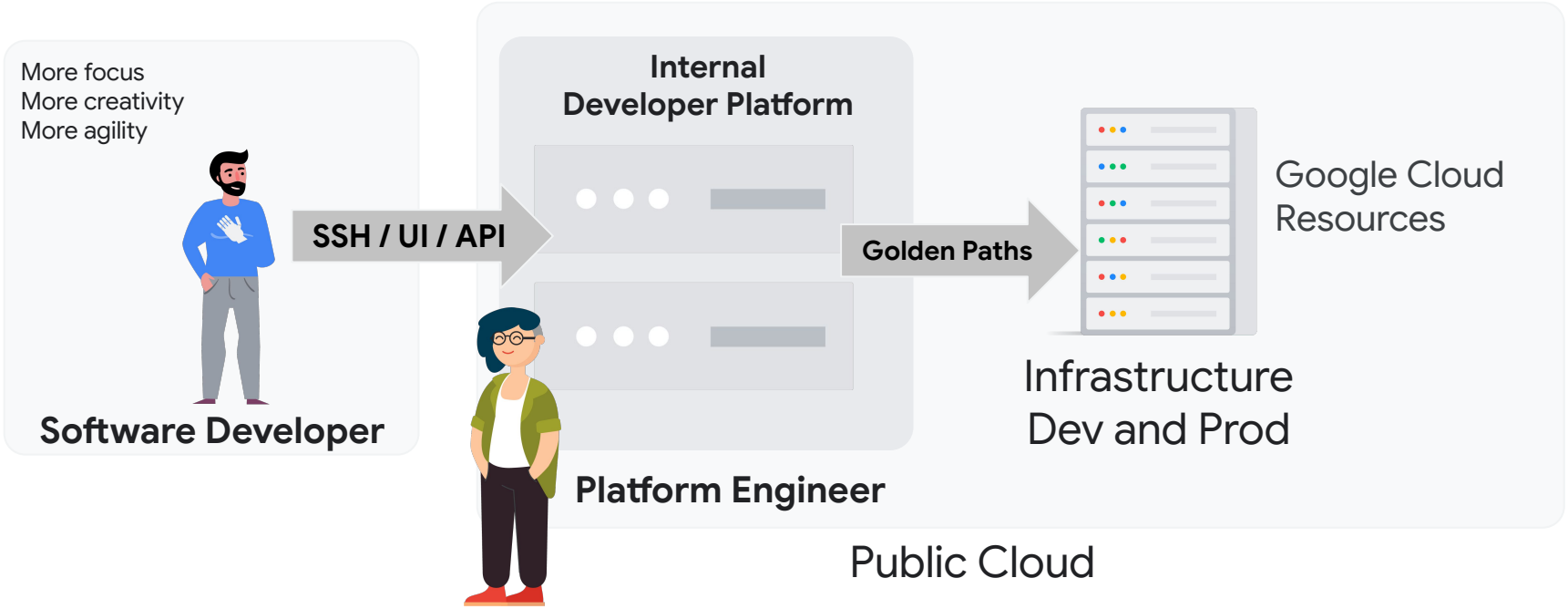
4. Build / buy capabilities

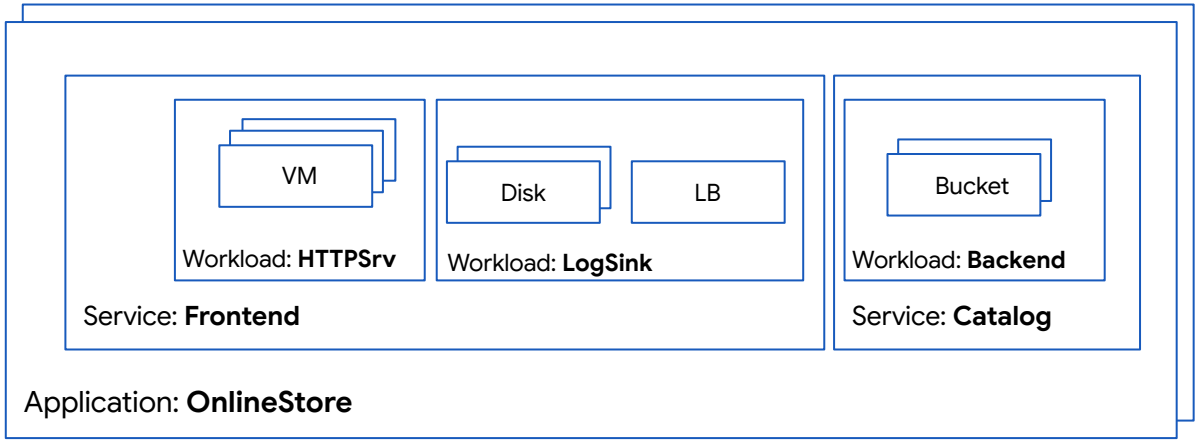
5. Enable capability in platform, document

6. Gather early dev feedback

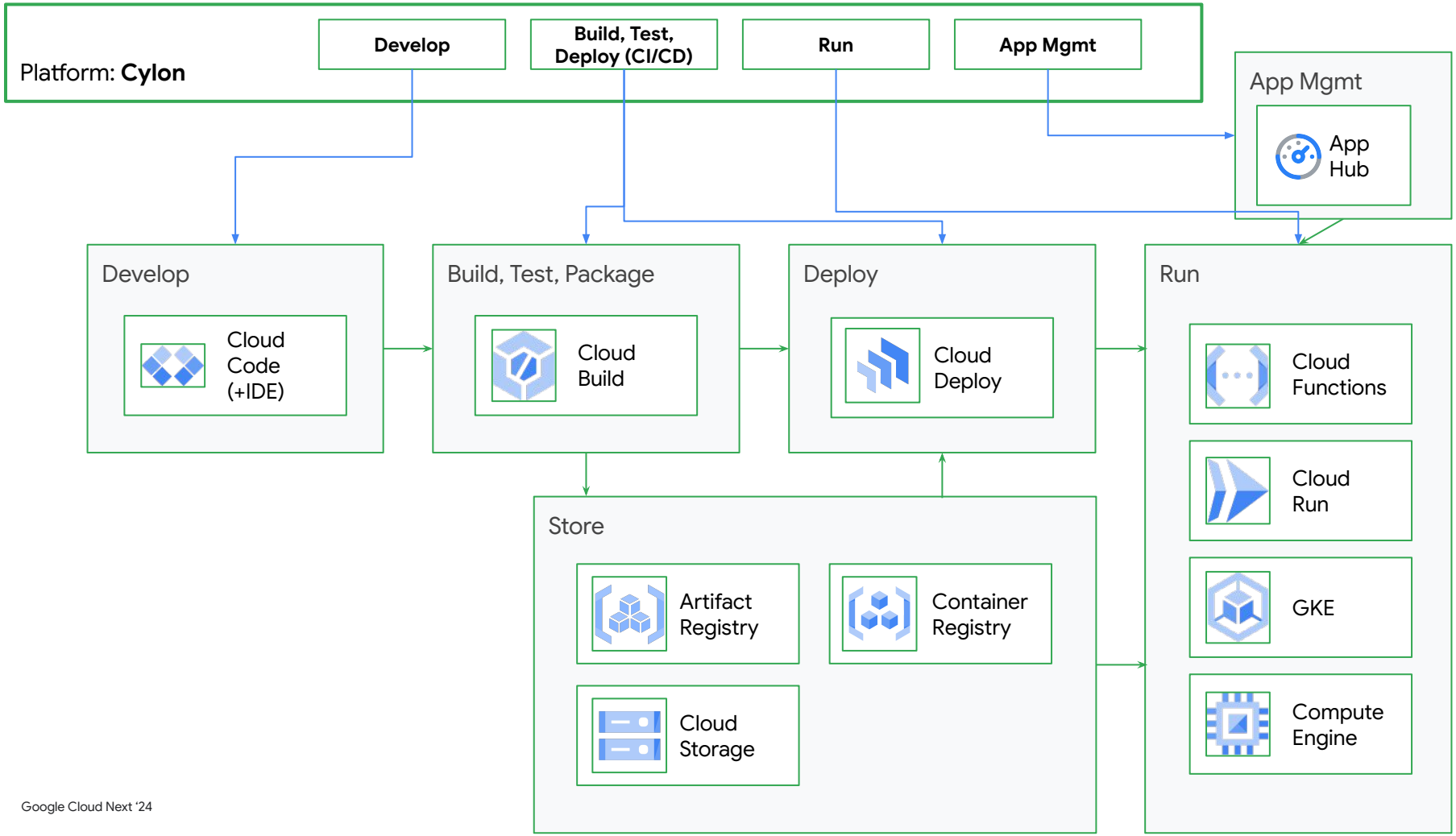
7. Release next version of platform

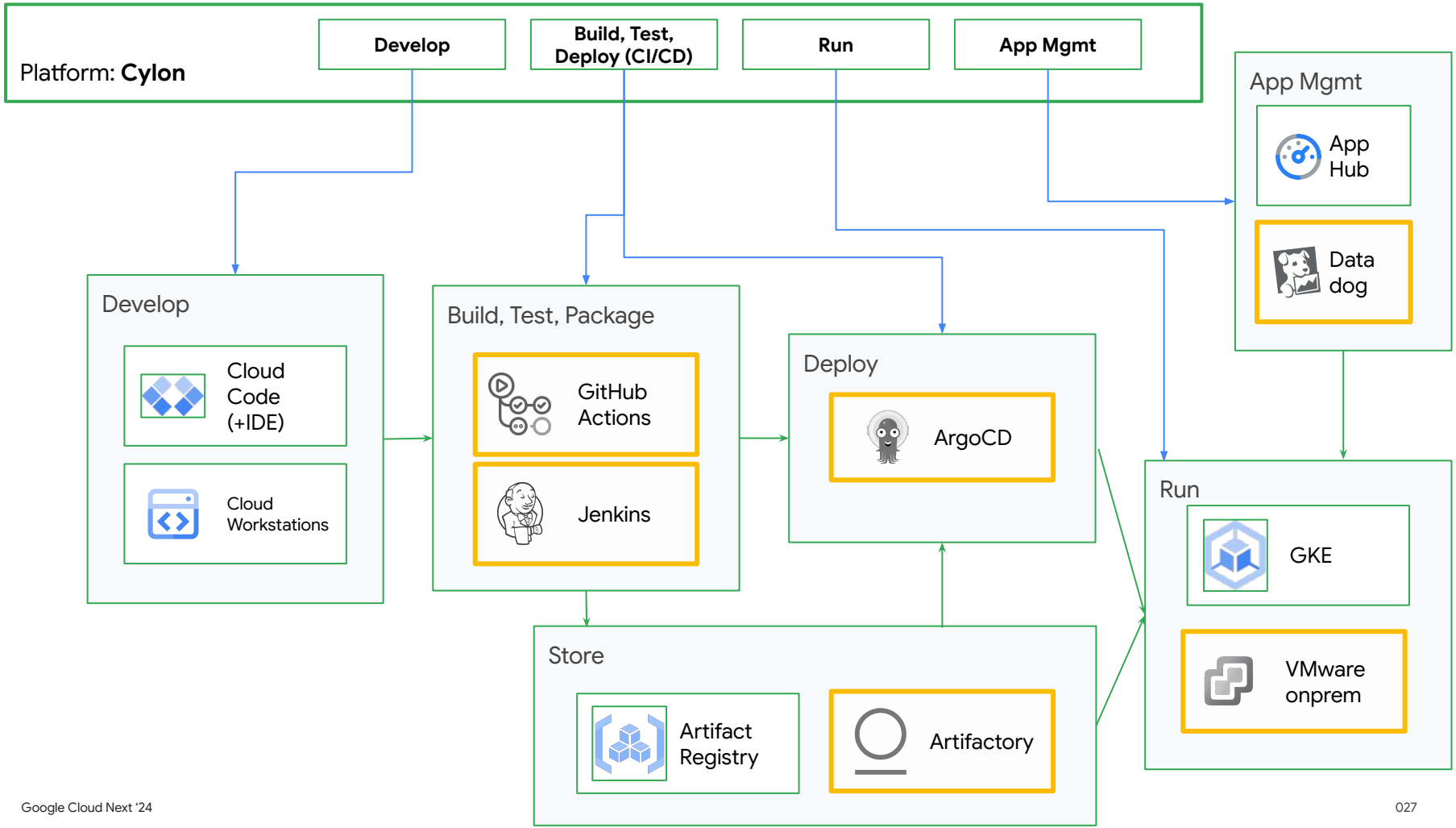
Platform Engineering




VM
Workload: HTTPSrv
Disk
LB
Workload: LogSink
Service: Frontend
Bucket
Workload: Backend
Service: Catalog
Application: OnlineStore


Platform: Cylon
Develop
Build, Test, Deploy (CI/CD)
Run
App Mgmt

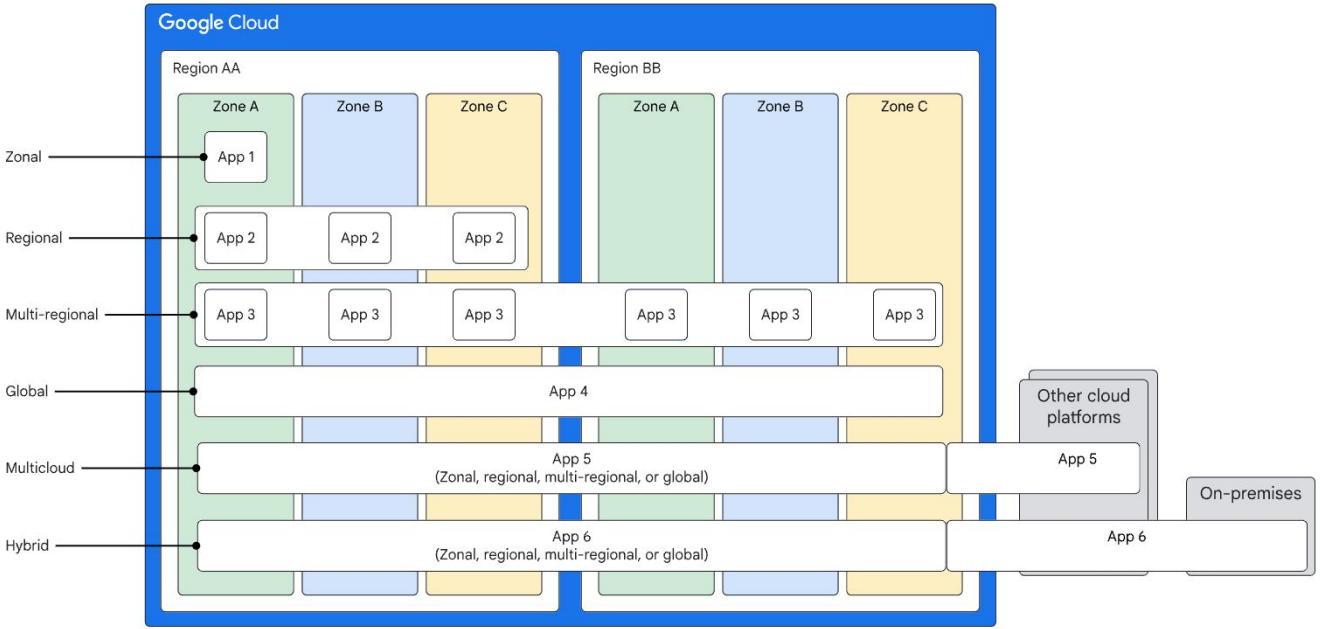

Platform: Cylon
Develop
Build, Test, Deploy (CI/CD)
Run
App Mgmt
App Mgmt
App Hub
Develop
Cloud Code (+IDE)
Build, Test, Package
Cloud Build
Deploy
Cloud Deploy
Run
Cloud Functions
Cloud Run
GKE
Compute Engine
Store
Artifact Registry
Container Registry
Cloud Storage


Platform: Cylon
Develop
Build, Test, Deploy (CI/CD)
Run
App Mgmt
App Mgmt
App Hub
Data dog
Develop
Cloud Code (+IDE)
Cloud Workstations
Build, Test, Package
GitHub Actions
Jenkins
Deploy
ArgoCD
Run
GKE
VMware onprem
Store
Artifact Registry
Artifactory

5 Application Archetypes

goo.gle/app-archetypes

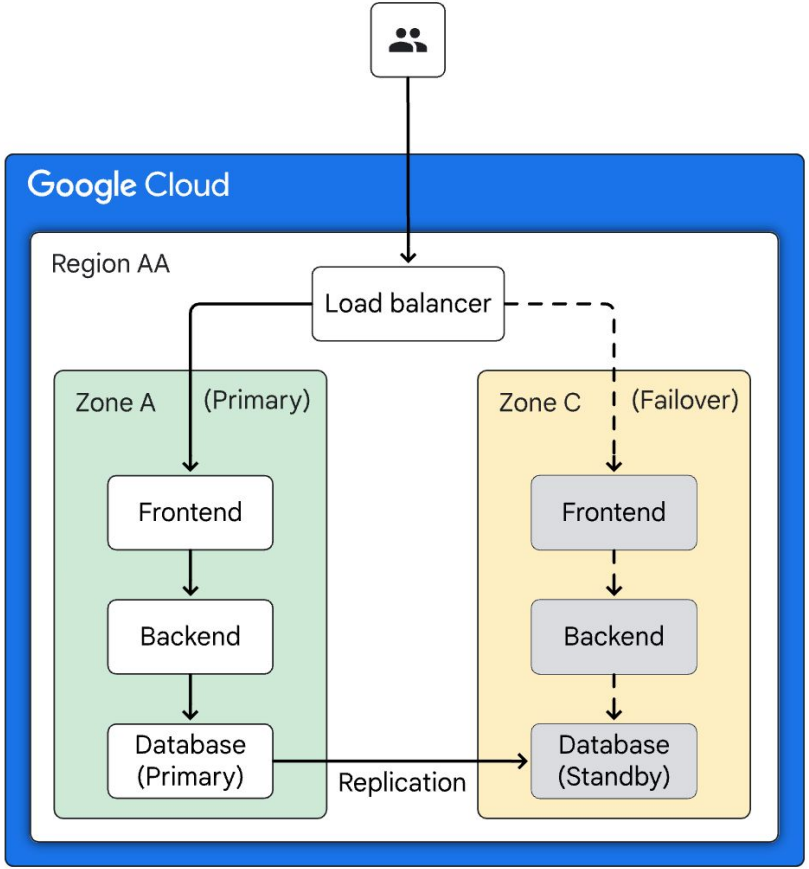
Cloud Architecture Center



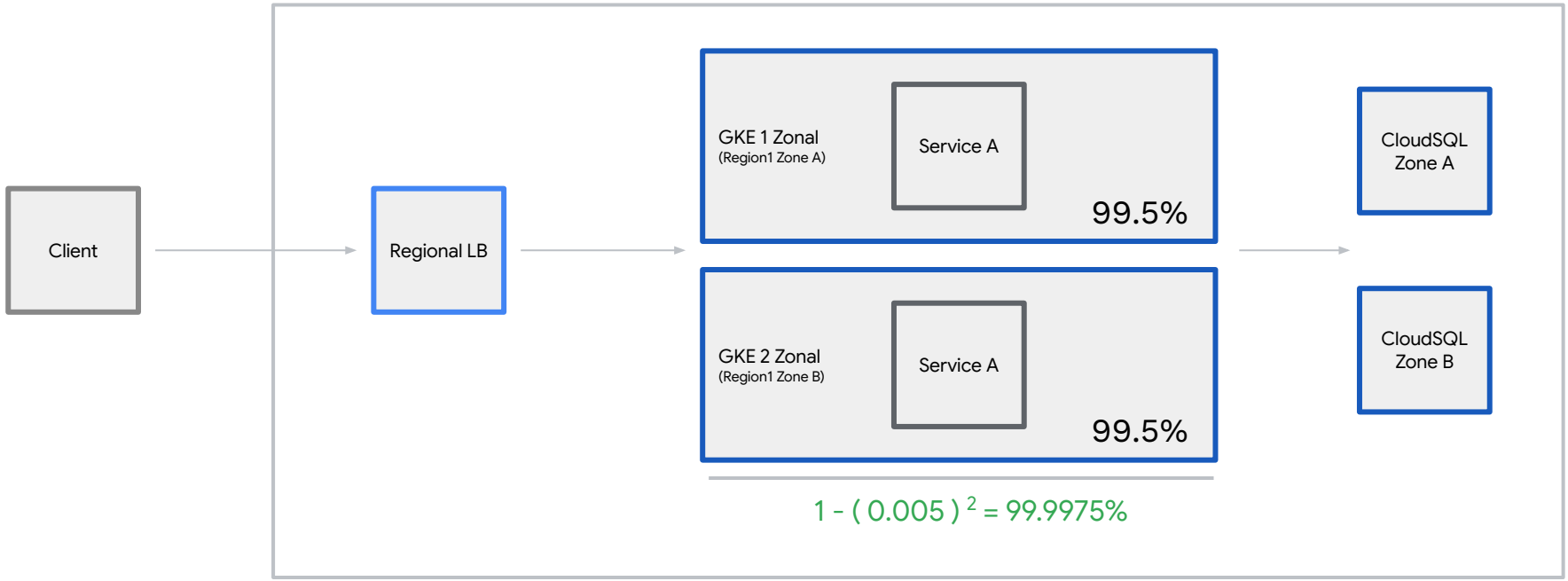
Archetype 1

Active Passive Zones

- **Survives zone** failure.
- **Fail-Ops**: Change LB backend, promote read replica
- **Cost**: 2x serving + 2x data (1 replica)
- **Complexity**: Low
- **App Refactoring**: None (lift and shift)
- **Type**: COTS, licensing

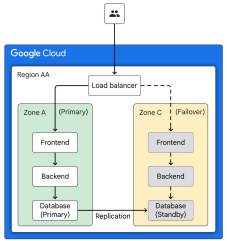


GKE clusters with Cloud SQL HA

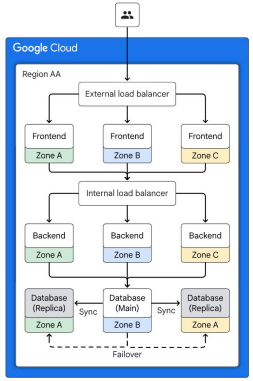


GCP SLAs

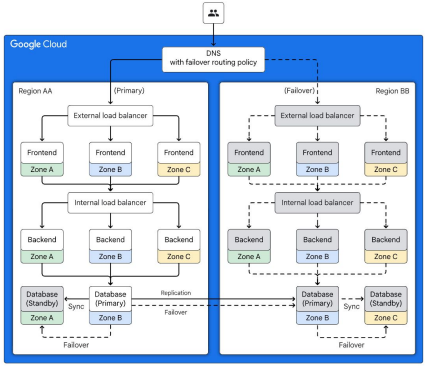
99.98% (Ceiling)
<2h / year !



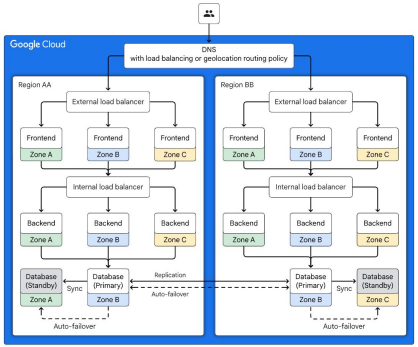
Active Passive Zones



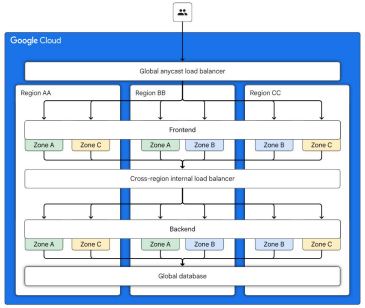
Multi-Zonal



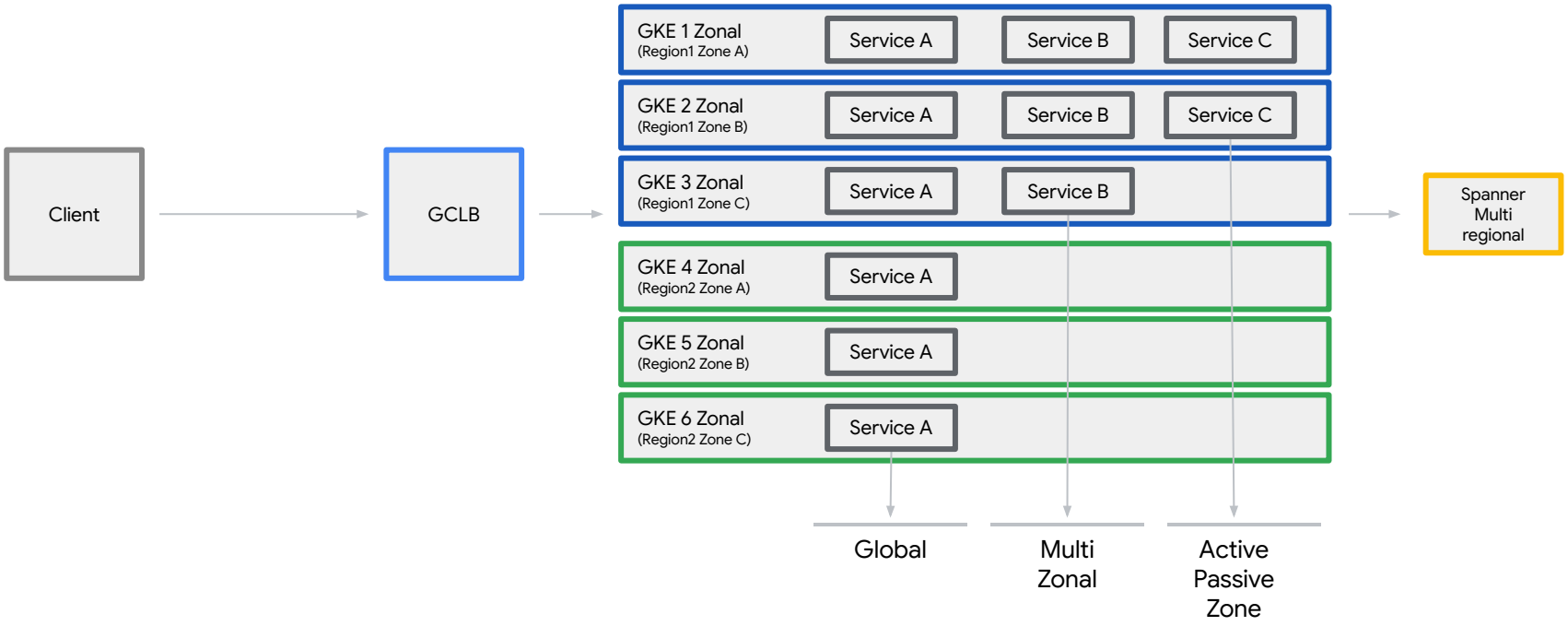
Active Passive Regions

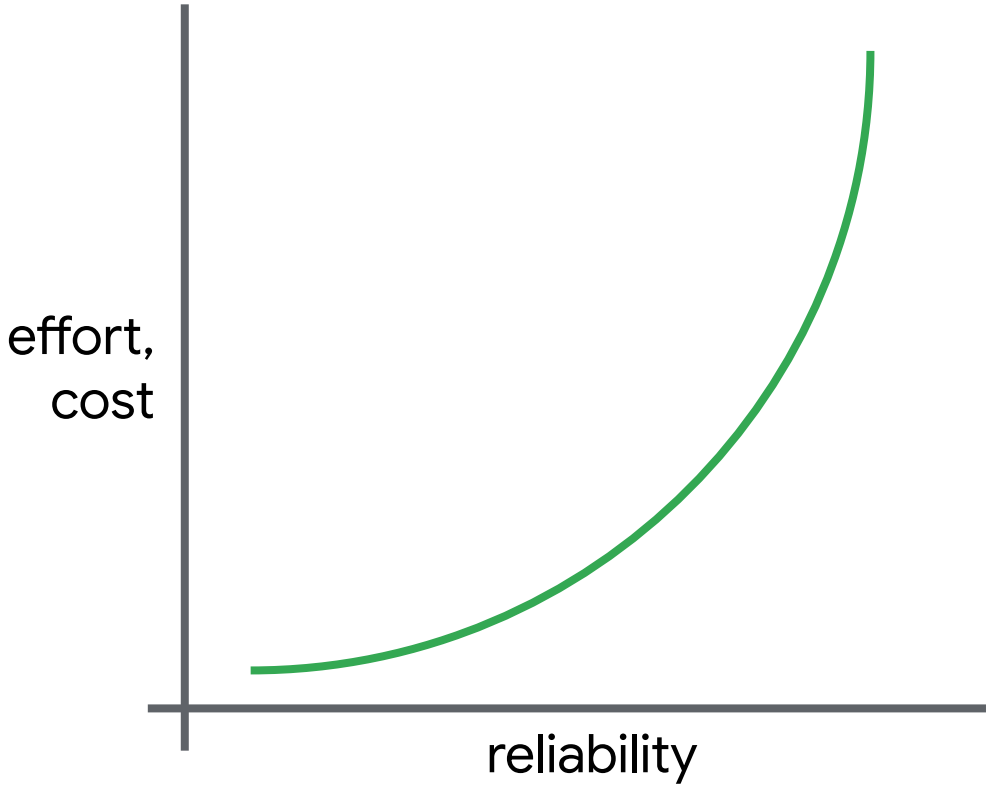


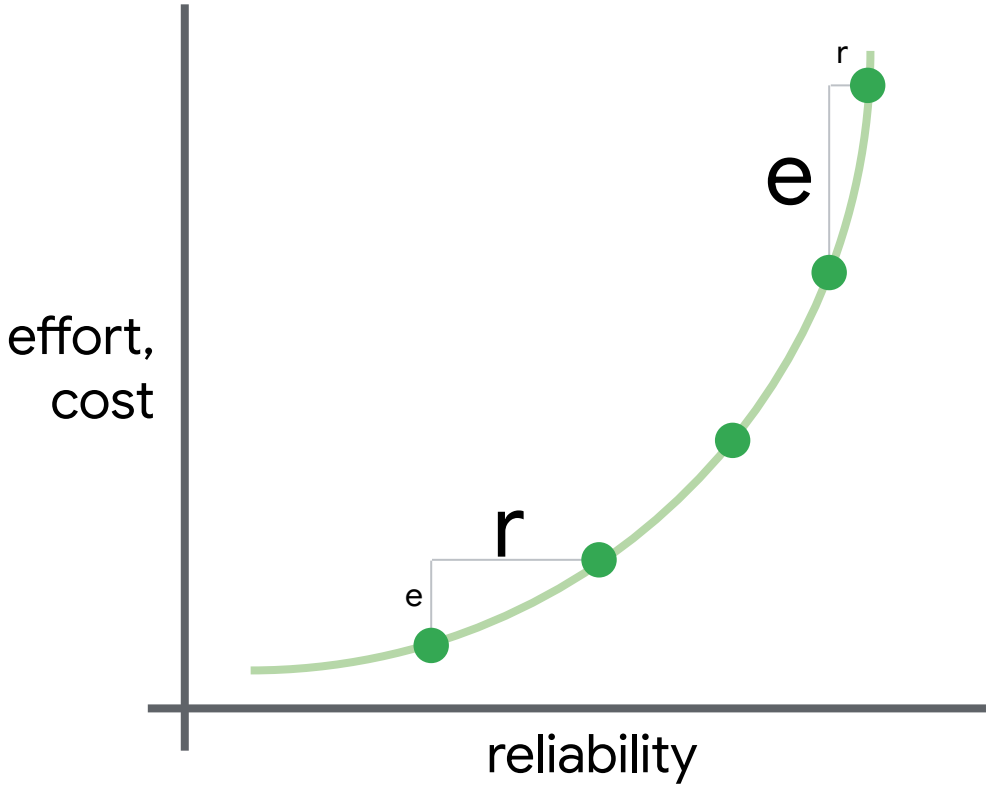
Isolated Regions



Global


Client
GCLB
GKE 1 Zonal
(Region1 Zone A)
Service A
Service B
Service C
GKE 2 Zonal
(Region1 Zone B)
Service A
Service B
Service C
GKE 3 Zonal
(Region1 Zone C)
Service A
Service B
GKE 4 Zonal
(Region2 Zone A)
Service A
GKE 5 Zonal
(Region2 Zone B)
Service A
GKE 6 Zonal
(Region2 Zone C)
Service A
Spanner
Multi
regional
Global
Multi
Zonal
Active
Passive
Zone


effort,
cost
reliability


r
e
effort,
cost
r
e
reliability

03

#lab

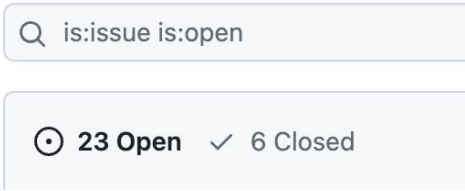
Lab != Prod – A place to play

It's hard to try new things in prod!

It's hard to build a new place beyond a `hello_world`!

So we built an **MVP of a platform** for you to play with.

reliable-app-platforms Public

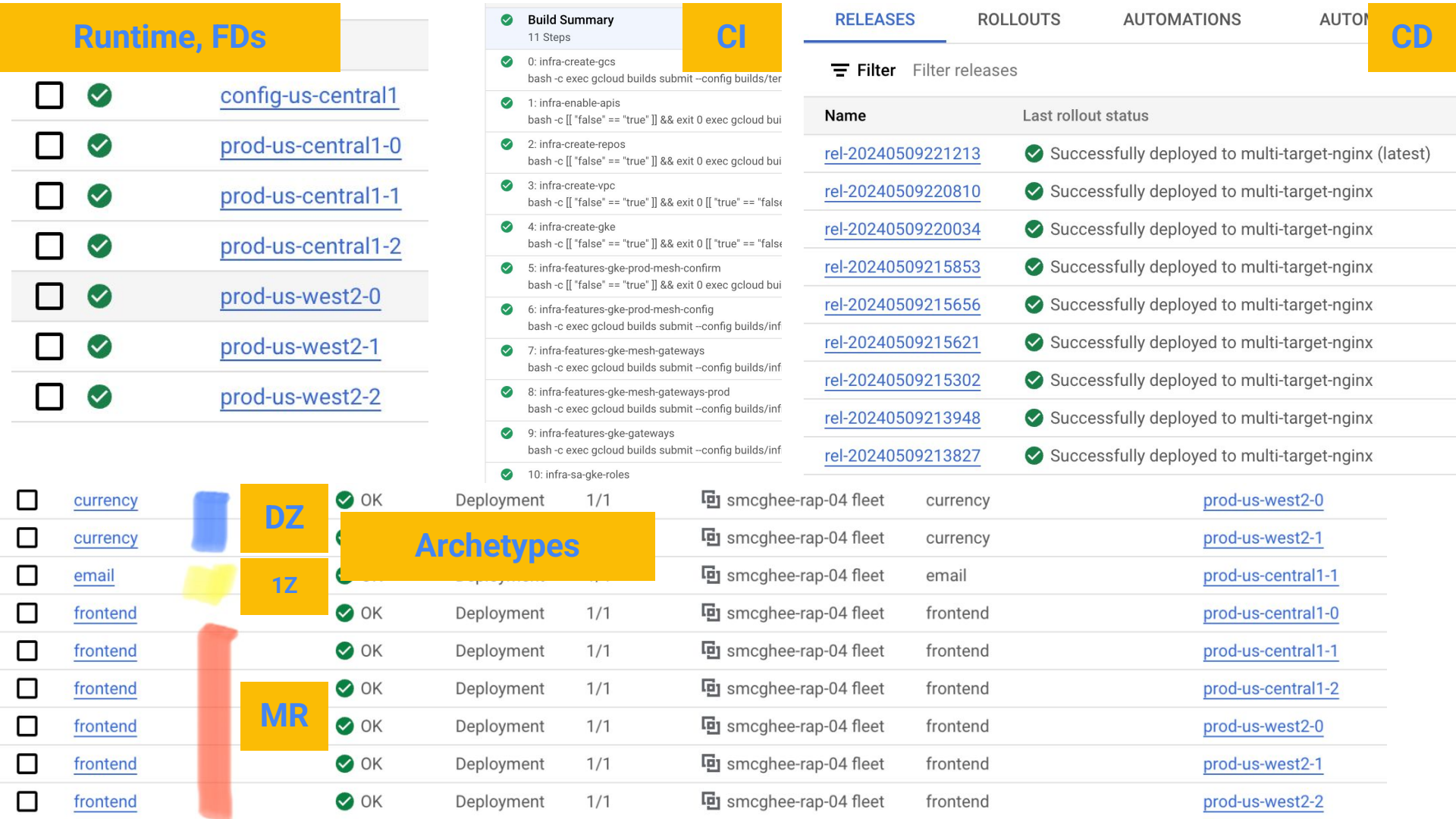


Clone, Fork, Contribute:

goo.gle/reliable-app-platforms

An MVP of a **Platform:**

- CI/CD (Cloud Build, Cloud Deploy)
  - Archetypes
- Multicluster (GKE)
- Observability (Cloud Observability)
- Still Evolving !


Runtime, FDs
config-us-central1
prod-us-central1-0
prod-us-central1-1
prod-us-central1-2
prod-us-west2-0
prod-us-west2-1
prod-us-west2-2
CI
Build Summary
11 Steps
0: infra-create-gcs
bash -c exec gcloud builds submit --config builds/ter
1: infra-enable-apis
bash -c [["false" == "true"]] && exit 0 exec gcloud bui
2: infra-create-repos
bash -c [["false" == "true"]] && exit 0 exec gcloud bui
3: infra-create-vpc
bash -c [["false" == "true"]] && exit 0 [["true" == "false
4: infra-create-gke
bash -c [["false" == "true"]] && exit 0 [["true" == "false
5: infra-features-gke-prod-mesh-confirm
bash -c [["false" == "true"]] && exit 0 exec gcloud bui
6: infra-features-gke-prod-mesh-config
bash -c exec gcloud builds submit --config builds/inf
7: infra-features-gke-mesh-gateways
bash -c exec gcloud builds submit --config builds/inf
8: infra-features-gke-mesh-gateways-prod
bash -c exec gcloud builds submit --config builds/inf
9: infra-features-gke-gateways
bash -c exec gcloud builds submit --config builds/inf
10: infra-sa-gke-roles
RELEASES
ROLLOUTS
AUTOMATIONS
AUTON
CD
Filter
Filter releases
Name
Last rollout status
rel-20240509221213
Successfully deployed to multi-target-nginx (latest)
rel-20240509220810
Successfully deployed to multi-target-nginx
rel-20240509220034
Successfully deployed to multi-target-nginx
rel-20240509215853
Successfully deployed to multi-target-nginx
rel-20240509215656
Successfully deployed to multi-target-nginx
rel-20240509215621
Successfully deployed to multi-target-nginx
rel-20240509215302
Successfully deployed to multi-target-nginx
rel-20240509213948
Successfully deployed to multi-target-nginx
rel-20240509213827
Successfully deployed to multi-target-nginx
DZ
1Z
MR
Archetypes
currency OK Deployment 1/1 smcghee-rap-04 fleet currency prod-us-west2-0
currency OK Deployment 1/1 smcghee-rap-04 fleet currency prod-us-west2-1
email OK Deployment 1/1 smcghee-rap-04 fleet email prod-us-central1-1
frontend OK Deployment 1/1 smcghee-rap-04 fleet frontend prod-us-central1-0
frontend OK Deployment 1/1 smcghee-rap-04 fleet frontend prod-us-central1-1
frontend OK Deployment 1/1 smcghee-rap-04 fleet frontend prod-us-central1-2
frontend OK Deployment 1/1 smcghee-rap-04 fleet frontend prod-us-west2-0
frontend OK Deployment 1/1 smcghee-rap-04 fleet frontend prod-us-west2-1
frontend OK Deployment 1/1 smcghee-rap-04 fleet frontend prod-us-west2-2

04

#production

Separate Apps from Platforms

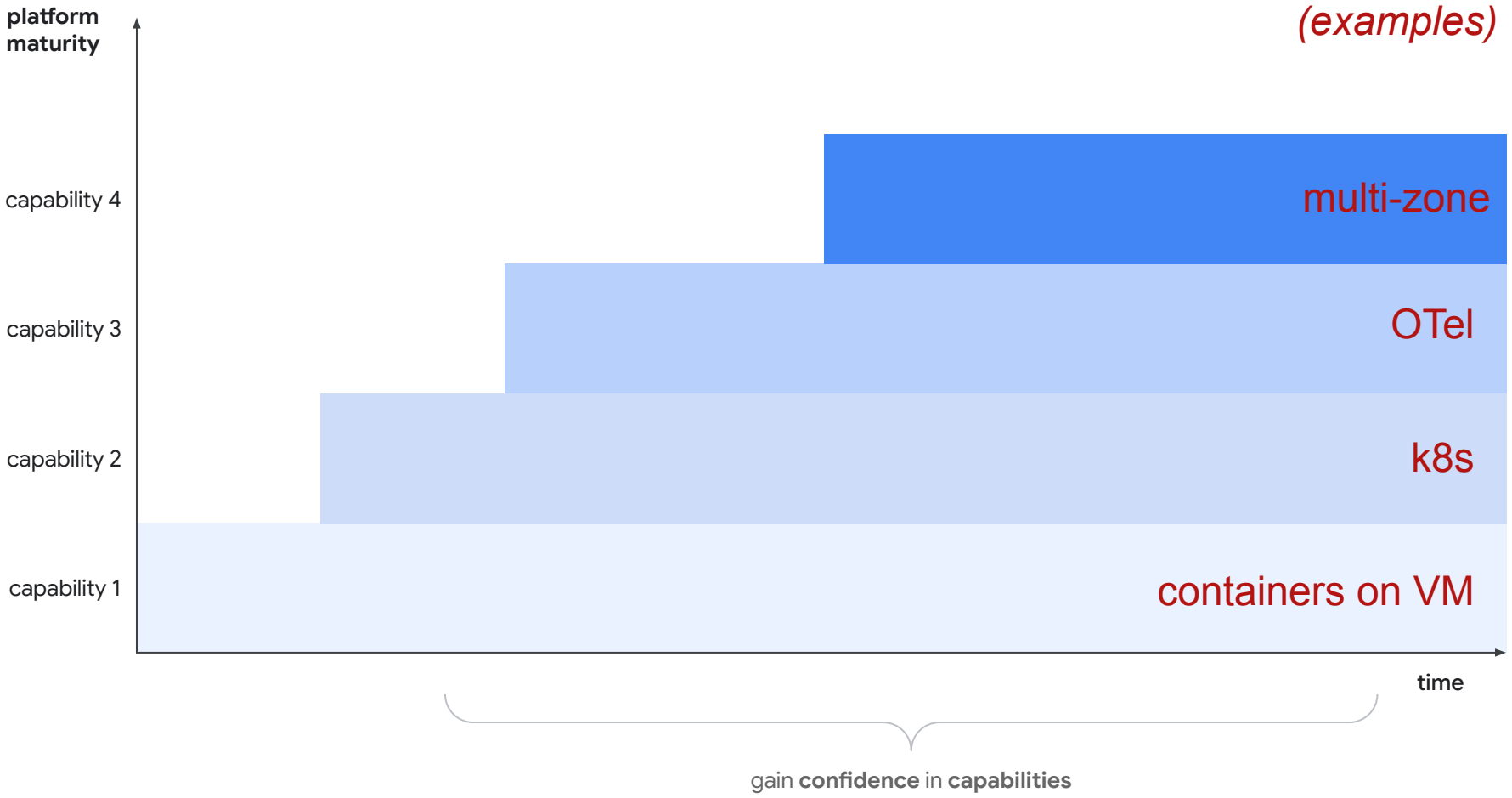
You don't need to handle every app/service/product from the very start.

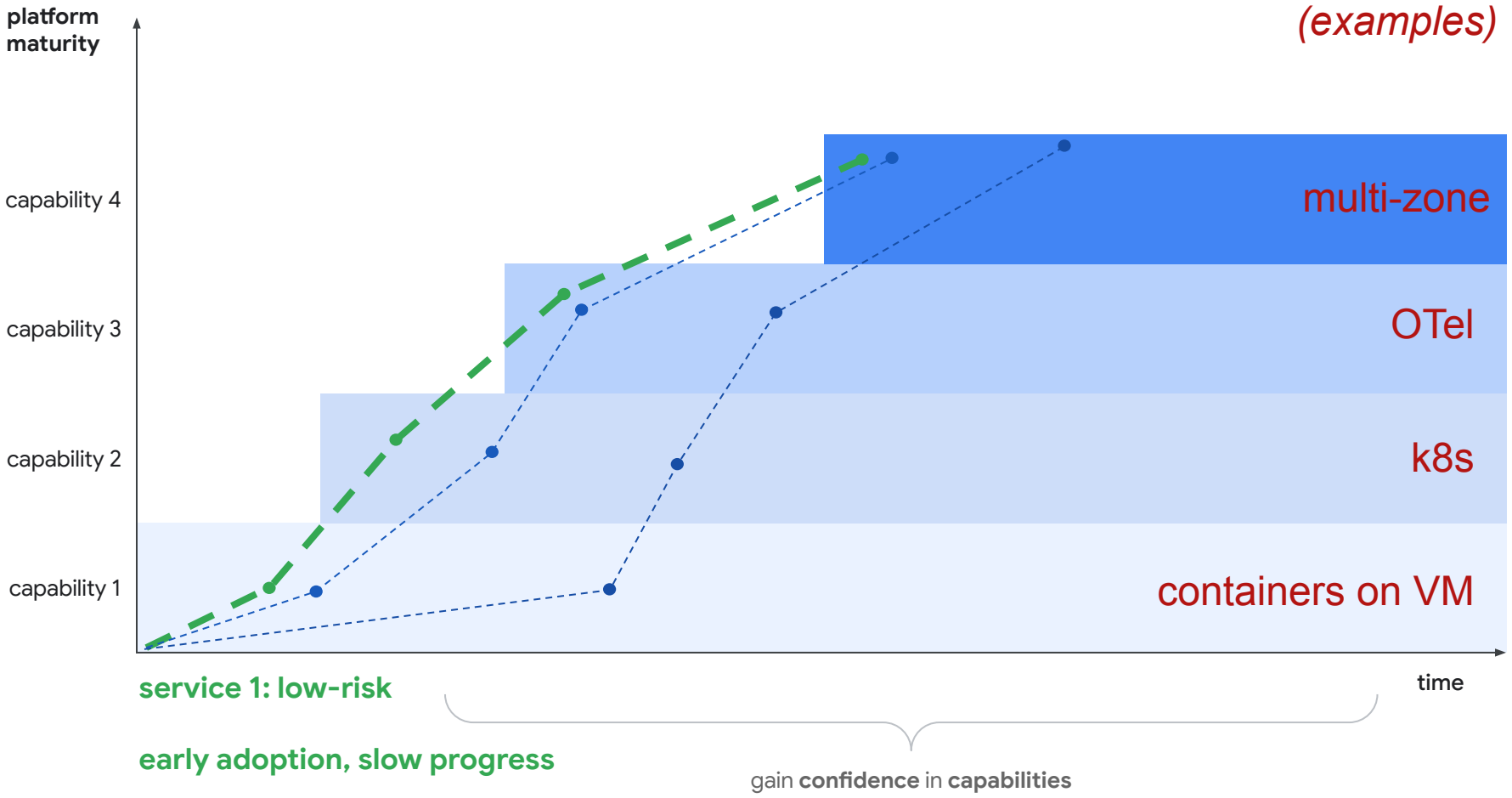
DO

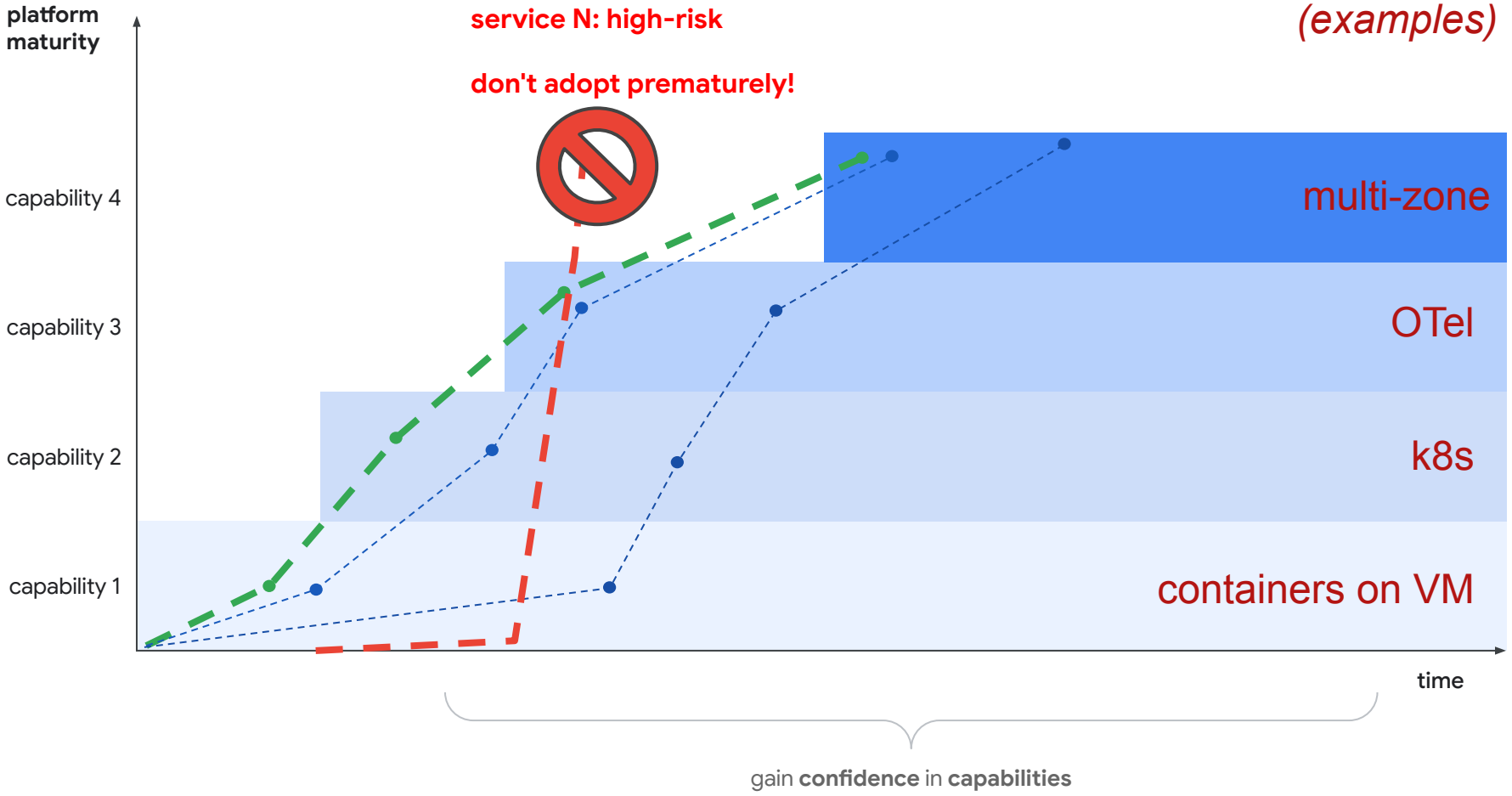
- Let teams adopt the platform at their own pace
- Celebrate early-adopters publicly, **share wins**
- Listen to dev teams **as your customers**

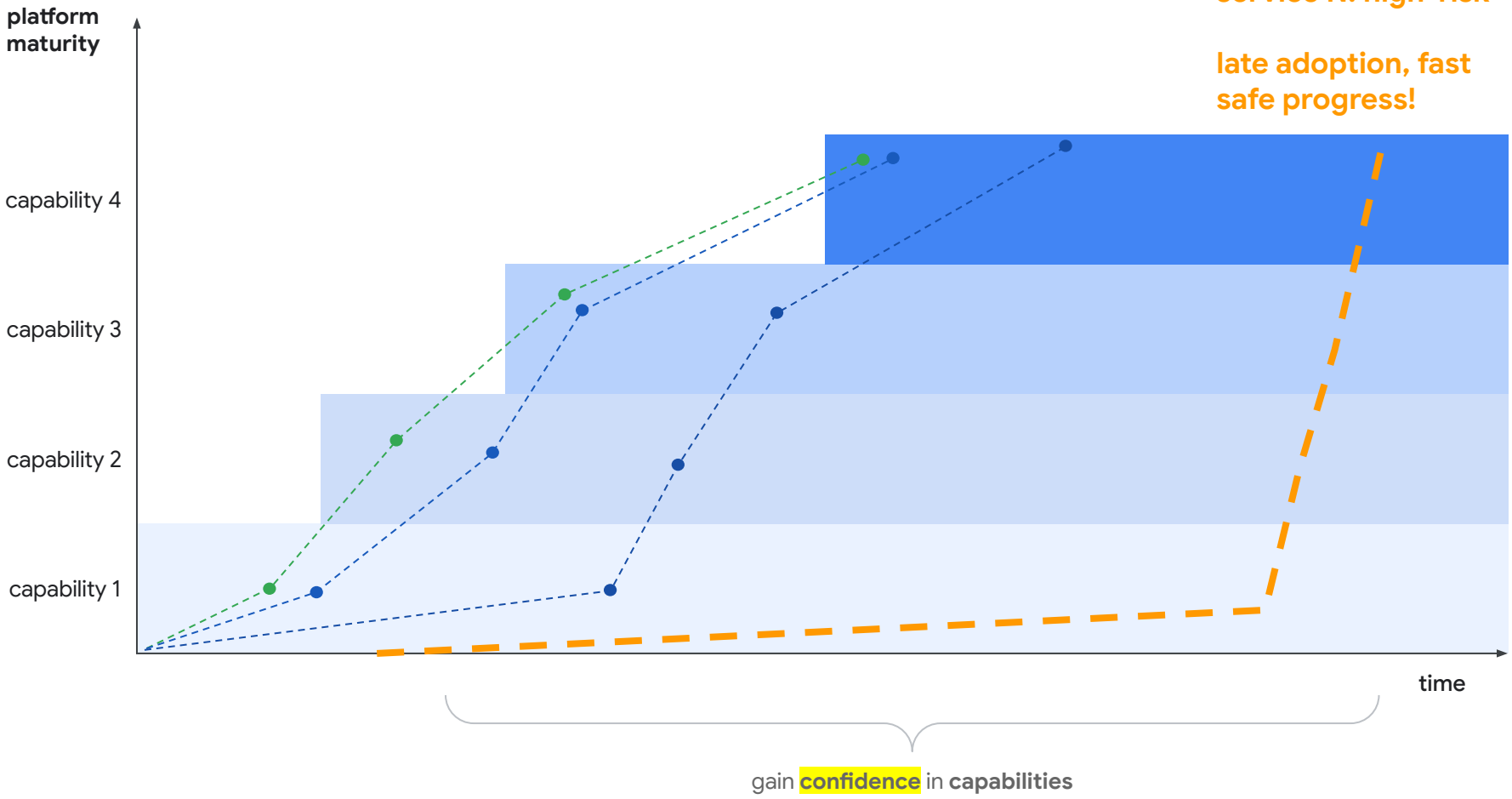
DON'T

- Don't force / demand / set % adoption targets!
- Don't start with the most critical apps!
- Don't waterfall !


platform maturity
(examples)
capability 4
multi-zone
capability 3
OTel
capability 2
k8s
capability 1
containers on VM
time
gain confidence in capabilities


platform maturity
(examples)
capability 4
capability 3
capability 2
capability 1
multi-zone
OTel
k8s
containers on VM
time
service 1: low-risk
early adoption, slow progress
gain confidence in capabilities


platform maturity
service N: high-risk
don't adopt prematurely!
(examples)
capability 4
capability 3
capability 2
capability 1
multi-zone
OTel
k8s
containers on VM
time
gain confidence in capabilities


platform maturity
capability 4
capability 3
capability 2
capability 1
time
service N: high-risk
late adoption, fast safe progress!
gain confidence in capabilities

Parting Shots

- DORA metrics, GBGB
- SLOs ("front door")
- Gradual Change @ Failure Domains
- Capabilities through Platforms
  - "SRE adjacency"
- Practice in the #Lab
- Learn, Write it down

05

Questions?

Thank You!

01 #hook

02 #lecture

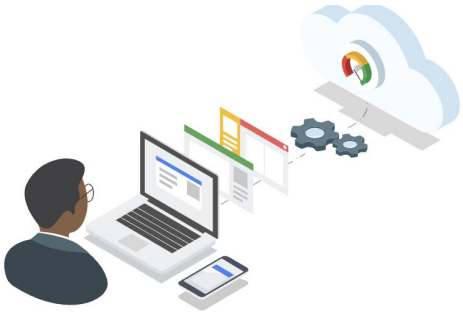
03 #lab

04 #production

05 Q&A

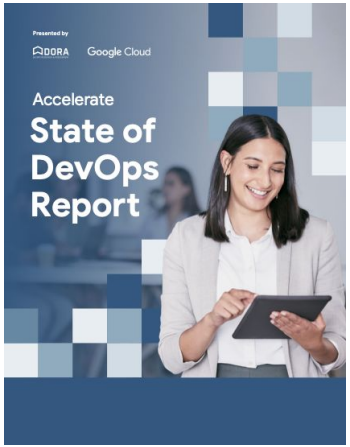
Next steps with DORA

Take the Quick Check



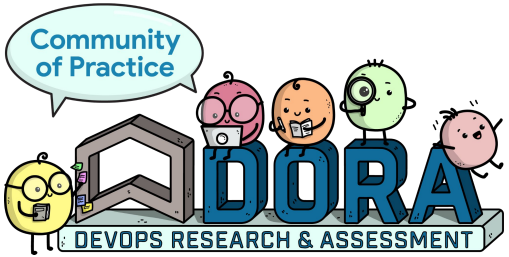
dora.dev/quickcheck

Read the Research



dora.dev/report

Join the Community



dora.community