SRE in Enterprise

Site Reliability Engineering

Steve McGhee &
James Brookbank
Google Cloud

https://g.co/cloud/ent-sre

DISCLAIMER:

These are our own personal opinions,
not the opinions of our employer

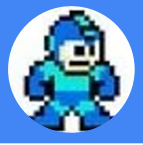
The book is official though

Intros

@JamesBrookbank
linkedin.com/in/jamesbrookbank

James Brookbank is a cloud solutions architect Solution architects help make cloud easier for Google's customers by solving complex technical problems and providing expert architectural guidance. Before joining Google, James worked at a number of large enterprises with a focus on IT infrastructure and financial services.

@stevemcghee
linkedin.com/in/stevemcghee



Steve was an SRE at Google for about 10 years in Android, YouTube and Cloud. He then joined a company to help them move onto the Cloud.

Now he's back at Google as a Reliability Advocate, helping more companies do that.

What are we seeing with SRE in the Enterprise?

"It's only SRE if it comes from the Mountain View region in California, otherwise it's just sparkling operations"

- **Enthusiasm > Successful Adoption** of SRE
- Reliability **isn't the most important thing** for everything
- SRE is often seen as **expensive** or **difficult** to achieve (usually both)
- Not everyone wants *the Google SRE way*
  - but they usually still want something that is **better than today**

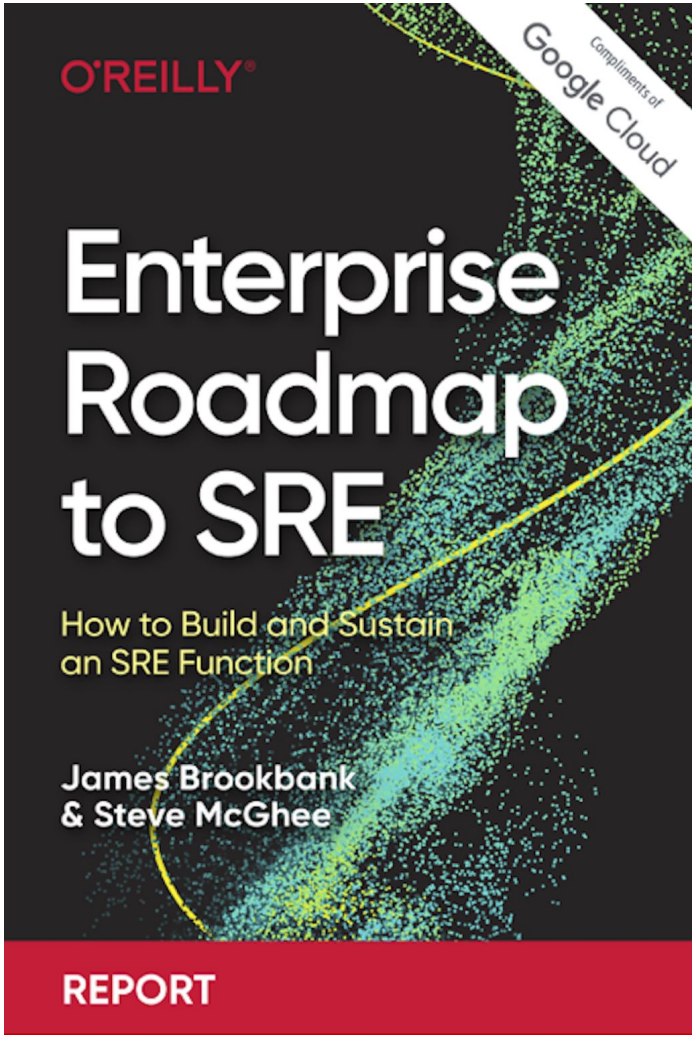
Agenda

Learn about the challenges of adopting site reliability engineering (SRE) in enterprises, and how we recommend cloud customers go about this journey

- Adoption of SRE best practices by cloud customers through evaluating their **existing** environment and architecture
- Identify how SRE guiding **principles** fit into a cloud customers existing organization (e.g. how to embrace risk)
- Adapt SRE **practices** for cloud customers existing team structure and knowledge
- **Nurture** a successful SRE initiative outside of Google

Enterprise Roadmap to SRE

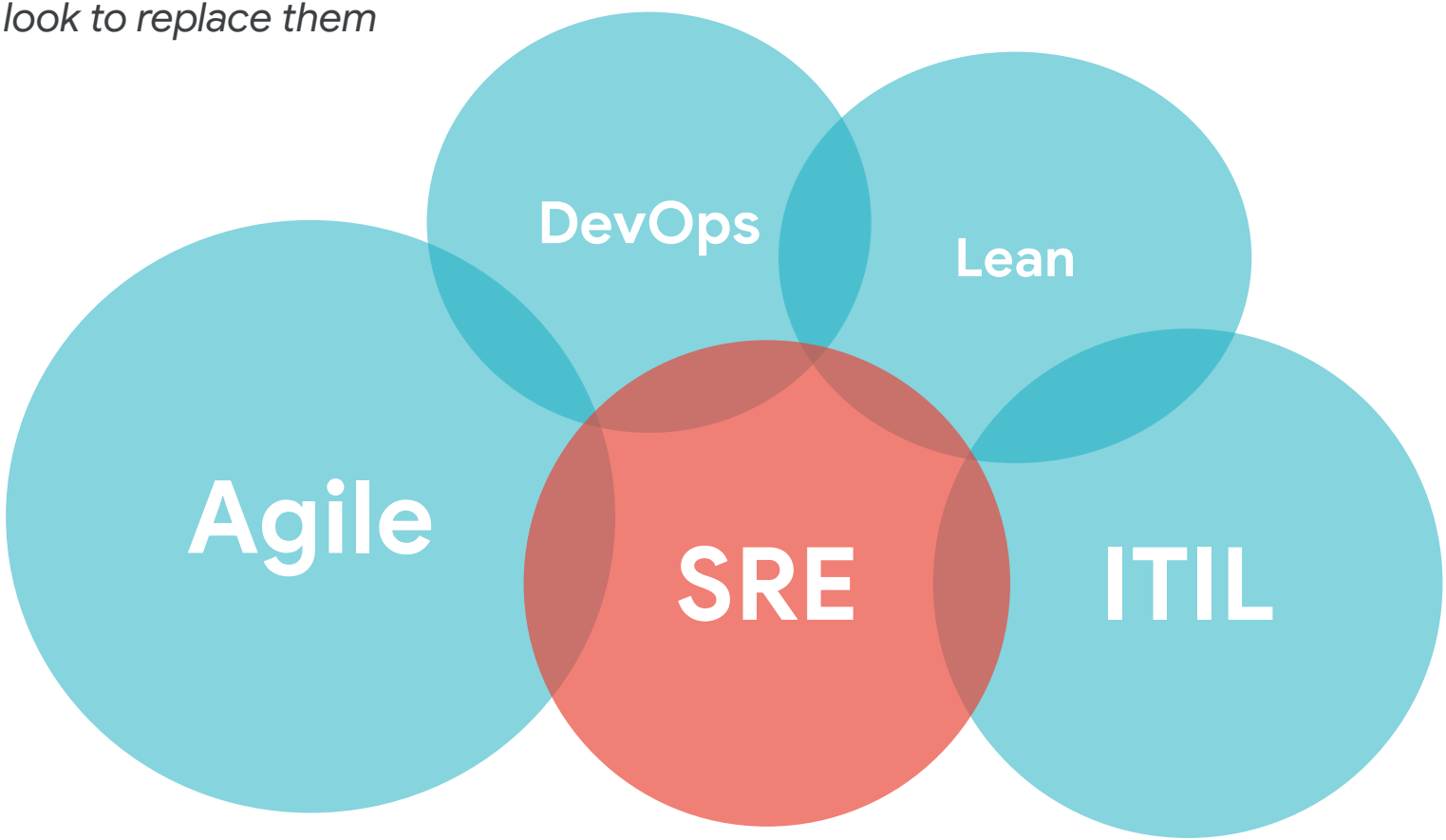
Downloads are free
Physical copies are available!



Section 1
Getting started with Enterprise SRE

SRE overlaps with many other things

It doesn't look to replace them



Sticking points

CABs
NOCs
...etc.

these individual practices aren't faulty on their own.

it's the **centralization** and **top-down organization** that doesn't work @ scale.

DevOps!

Capabilities

Technical
- Trunk-based development
- Cloud infrastructure
- Shifting left on security

...

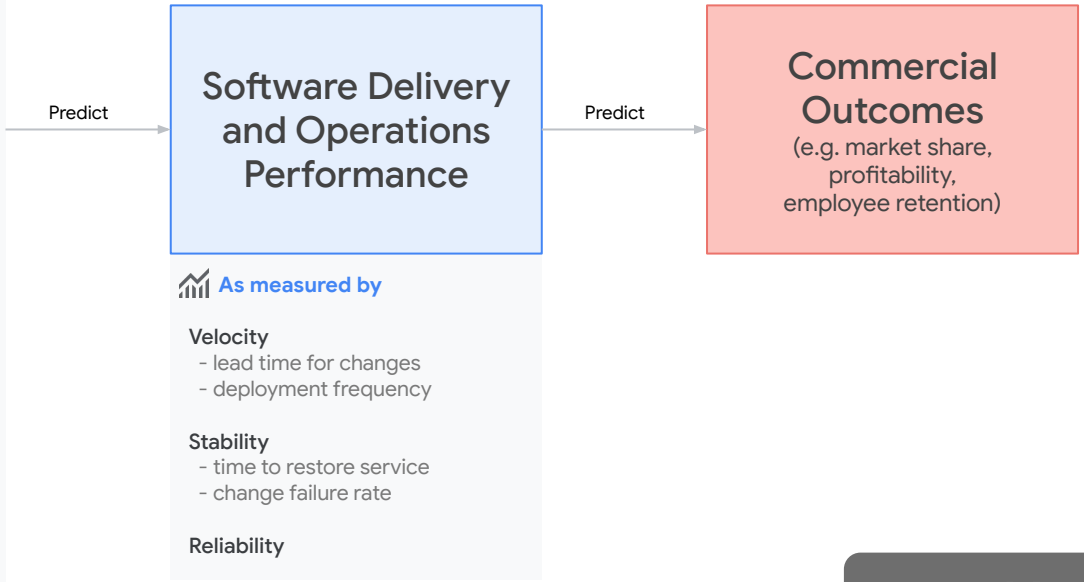
Process
- Work in small batches
- Streamlined change approval
- Visibility of work in value stream

...

Cultural
- Generative, trust-based
- Learning culture
- Transformational leadership

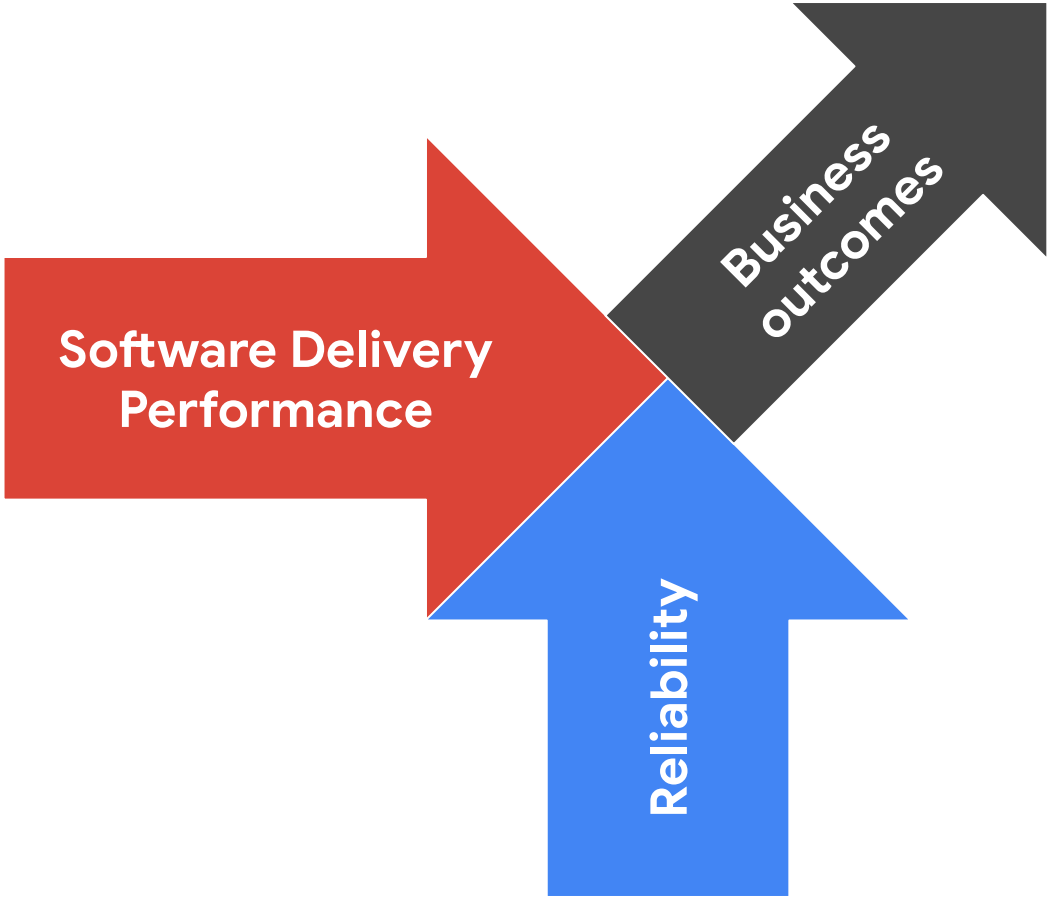
...

DORA
DEVOPS RESEARCH & ASSESSMENT



g.co/devops

2022 State of DevOps



1.8x

more likely to meet or exceed organizational goals

Lessons from DevOps

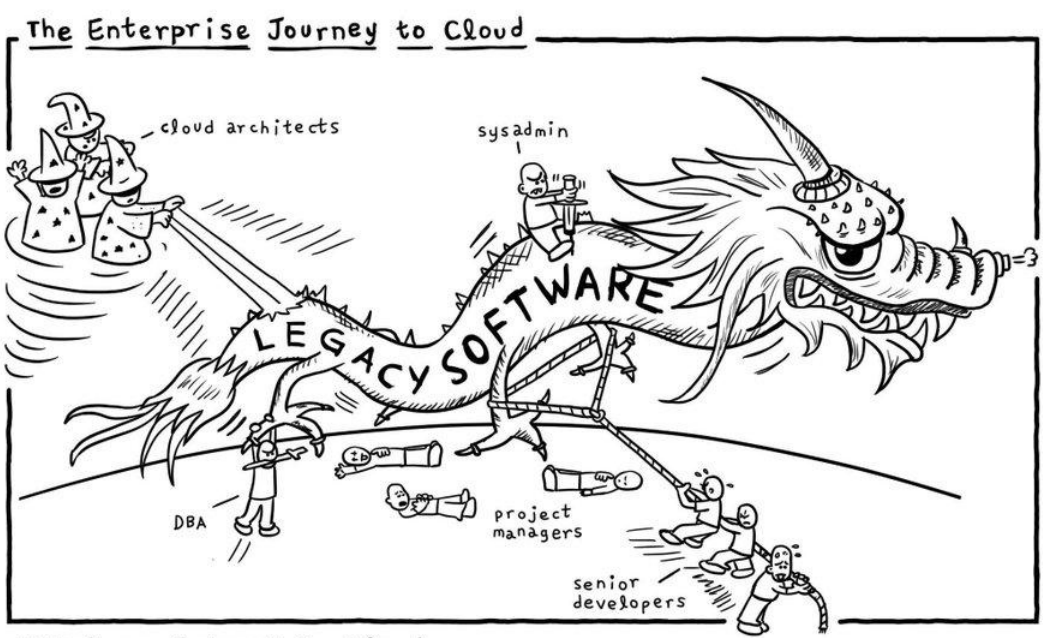
What works? What doesn't?

- **Training centers**
  - ~10% of training should be classroom based
  - Most training should be mentoring or learning by doing e.g. Dojos
- **Centers of excellence**
  - Centers of *enablement* use hands on coaches
  - Learning by doing instead of best practices from the ivory tower
- **Big Bang**
  - Continuous improvement is better than delayed perfection
  - In complex areas we can't predict the future

SRE and Cloud

Your cloud journey **isn't the same** as your SRE journey

But your SRE journey will need on-demand, resource pooled infrastructure with broad network access, rapid elasticity, and measured service...



We think SRE is **emergent** from culture

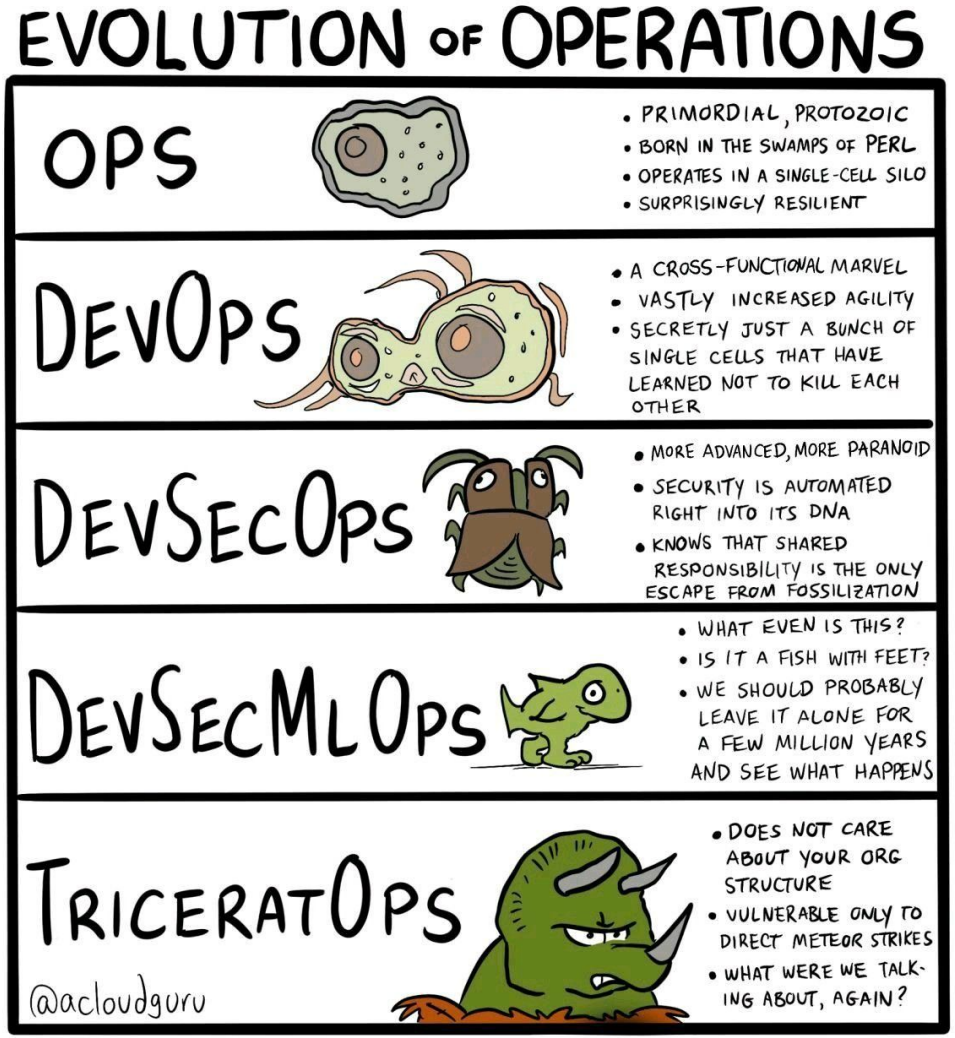
Pathological (power oriented)	**Bureaucratic** (rule oriented)	**Generative** (performance oriented)
Low cooperation	Modest cooperation	High cooperation
Messengers shot	Messengers neglected	Messengers trained
Responsibilities shirked	Narrow responsibilities	Risks are shared
Bridging discouraged	Bridging tolerated	Bridging encouraged
Failure leads to scapegoating	Failure leads to justice	Failure leads to enquiry
Novelty crushed	Novelty leads to problems	Novelty implemented

Westrum, "A Typology of Organizational Cultures" | http://bmj.co/1BRGh5q

Section 2
Why the SRE approach to Reliability?

What is driving the evolution of SRE?

(Spoiler alert: *selection pressure*)



Q: Can you build **99.99%** services on **99.9%** infra?

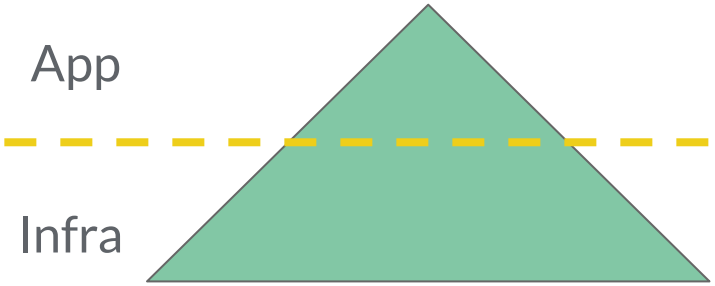
NBC Universal

Q: Can you build **99.99%** services on **99.9%** infra?

Yes.

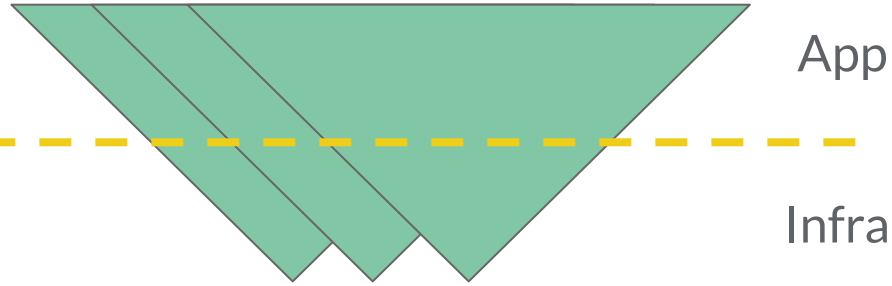
You can build
more reliable things
on top of
less reliable things

a simple example: RAID see: *The SRE I Aspire to Be*, @aknin SREconEMEA 2019



Traditional Model

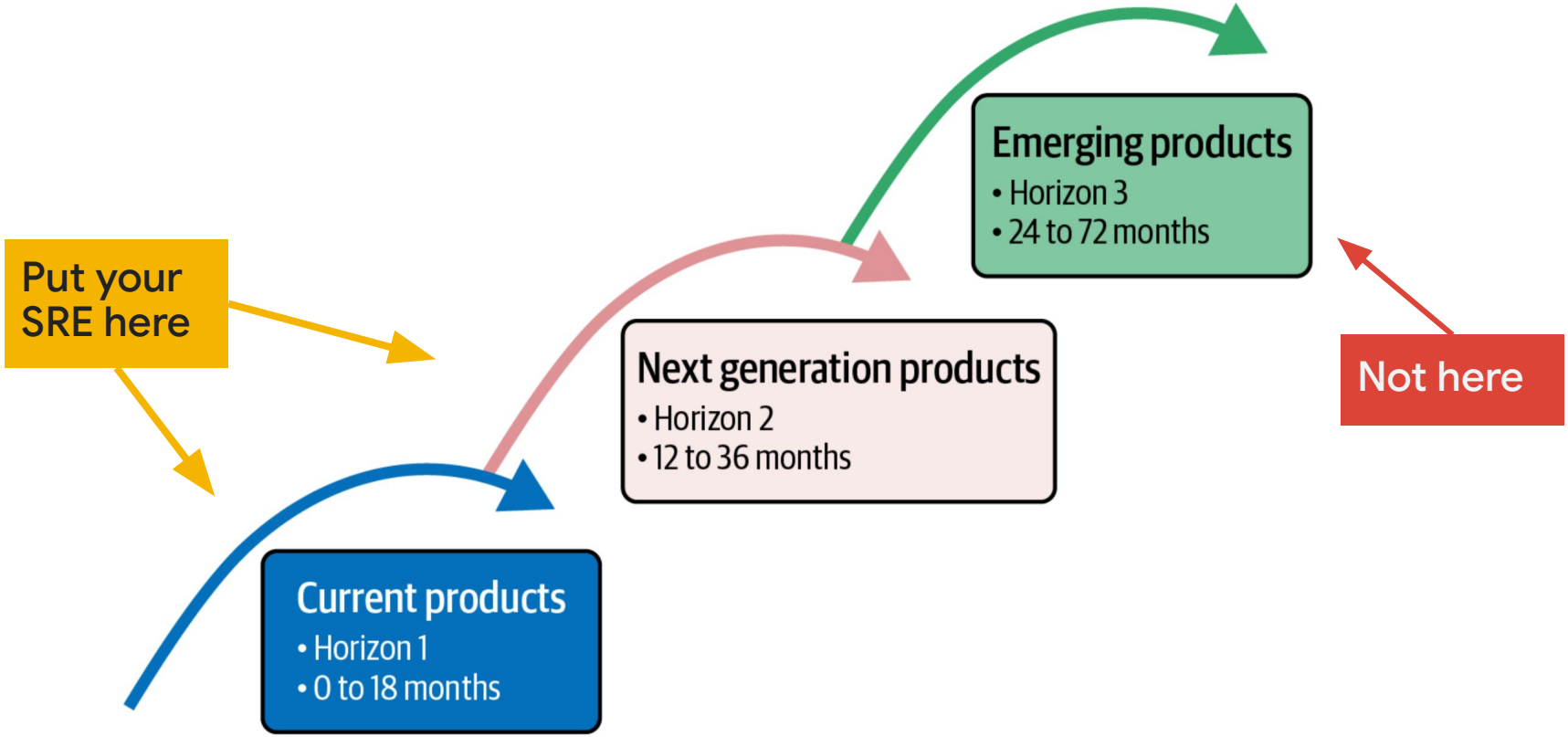
- Inherit reliability from the base
- Lower levels *must* be more reliable
- "scale up"



Cloud is here, though.

- Cost-effective base at scale
- Software must improve availability
- "scale out"

When to use the SRE approach to Reliability?



Why the SRE approach to Reliability for your Enterprise?

- ✅ **R9y as product differentiator**
- ✅ **Critical risk mitigation**
- ✅ **Hyperscale services**

However! – not every service needs SRE

Why the SRE approach to Reliability?

Cost reduction...?

Yes! ... But also no.

SRE is a **strategic investment** ($↑)
in long-term operational efficiency ($↓)

Cost optimization is **global**, not **local**.

This is **critical** for your finance team

TV Globo

Section 3
SRE Principles

SRE Principles

SRE Principles

Start small

- Build practices incrementally
- More advanced capabilities need to have foundational ones first
- Prevent **organization destroying mistakes**

Invest in people

- Staffing and retention
- Hiring feels easy but growing is more sustainable
- **Don’t fire everyone in ops who can’t code**
- Value existing employees – they know the business!

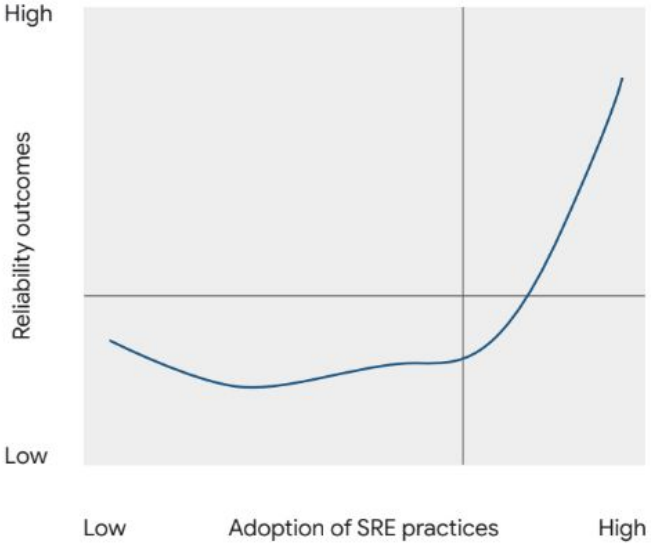
SRE Principles

Embrace risk

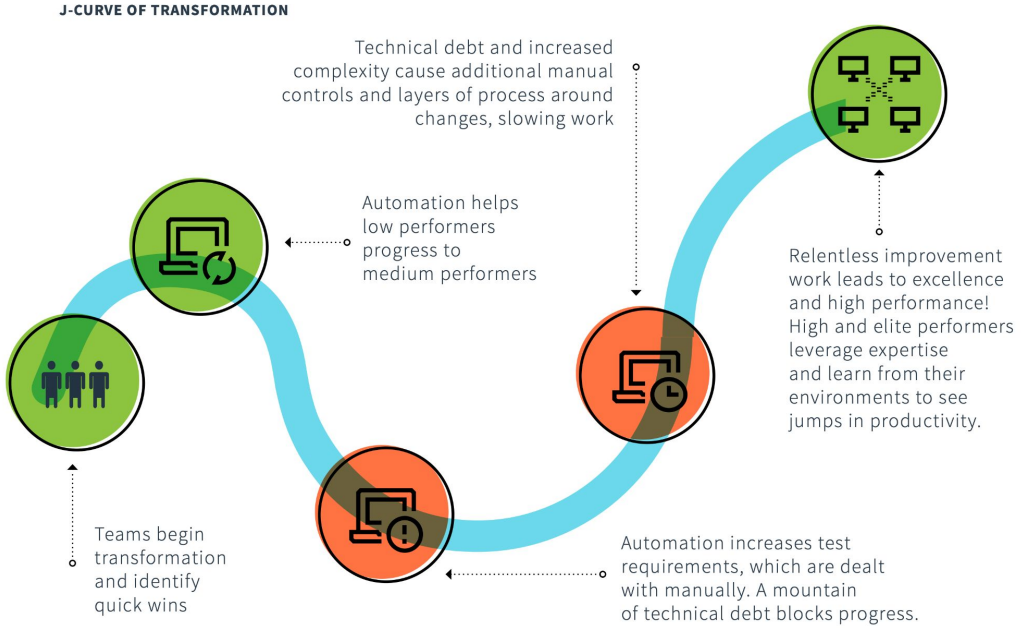
- Create a **safe to fail** environment
- **You can’t only take risks that will succeed**
- Demonstrating active leadership is important
- Treat failures as **unplanned learning opportunities**

SRE & DevOps agree (SoDR 2022)

Give it time



Teams that persist beyond initial steps of SRE adoption see increasing improvement in reliability outcomes



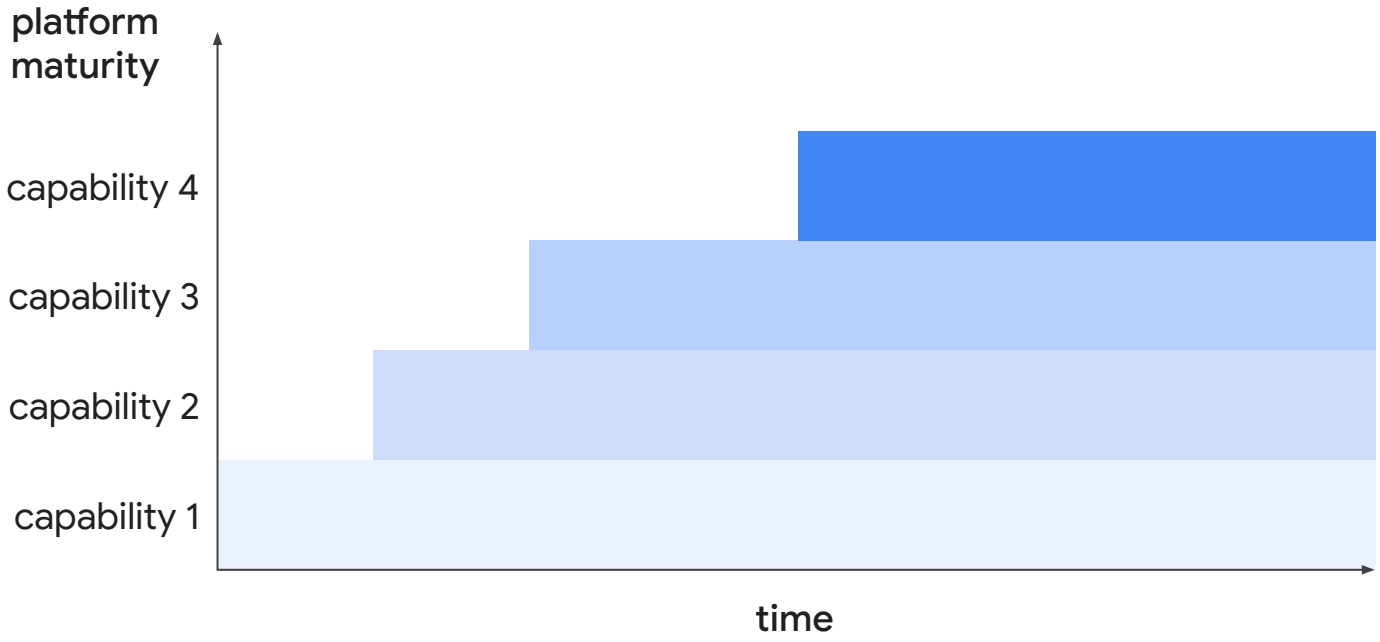
DORA State of DevOps 2018

Section 4
SRE Practices

SRE Practices

- Build a **platform** of **capabilities**!
- Capabilities get built, purchased, added over time
- Services are introduced to the platform, **as it makes sense**.
- Antipattern: **Pick the toughest thing first** since that will fix all the problems

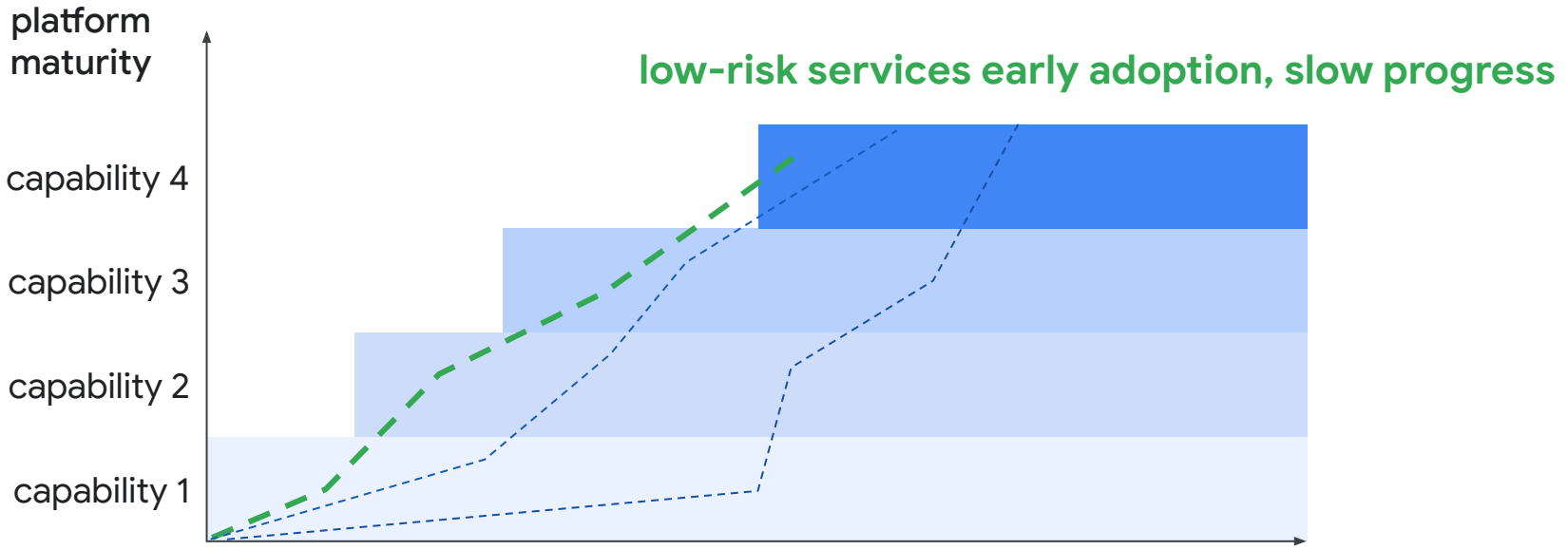
Building Platforms



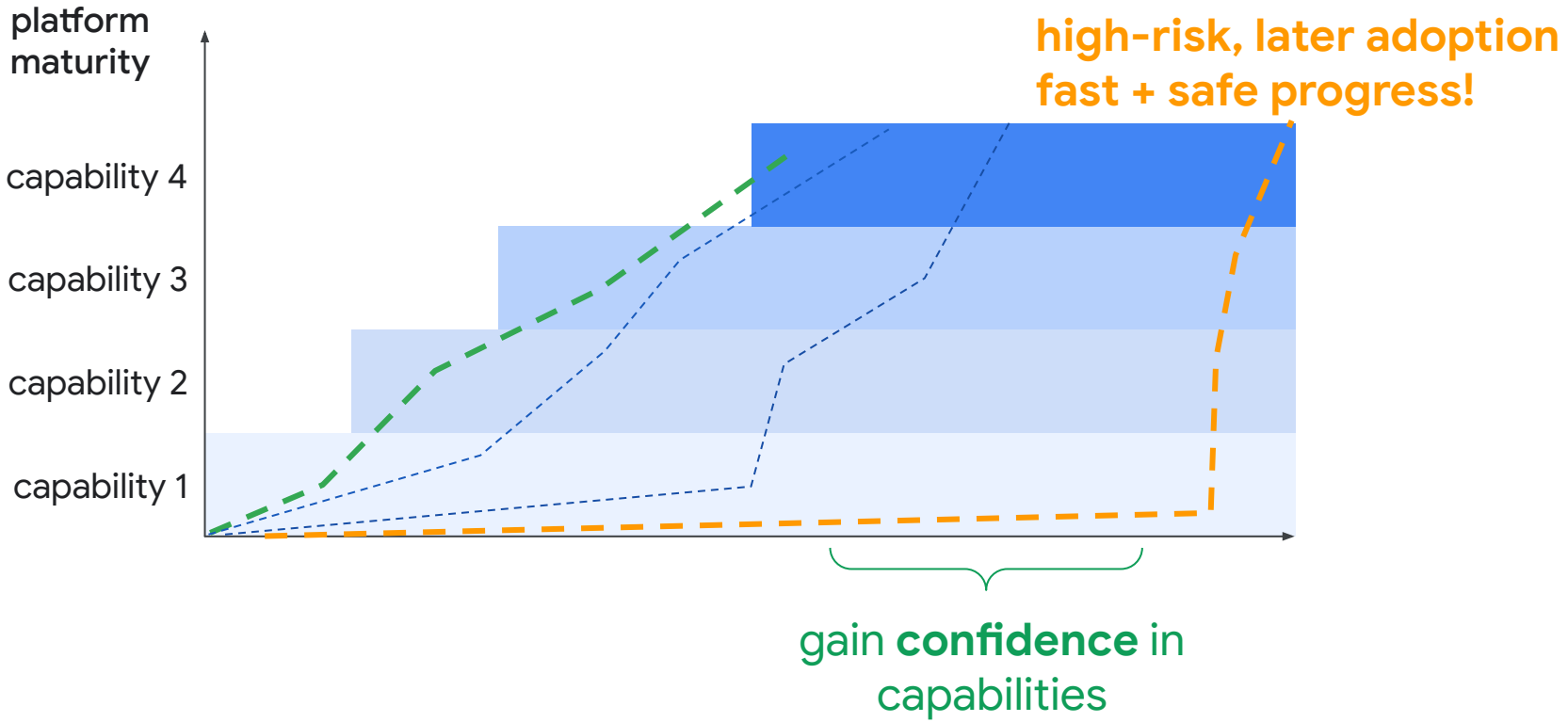
Add capabilities over time: CI/CD, rollbacks, multi-cluster

Building Platforms

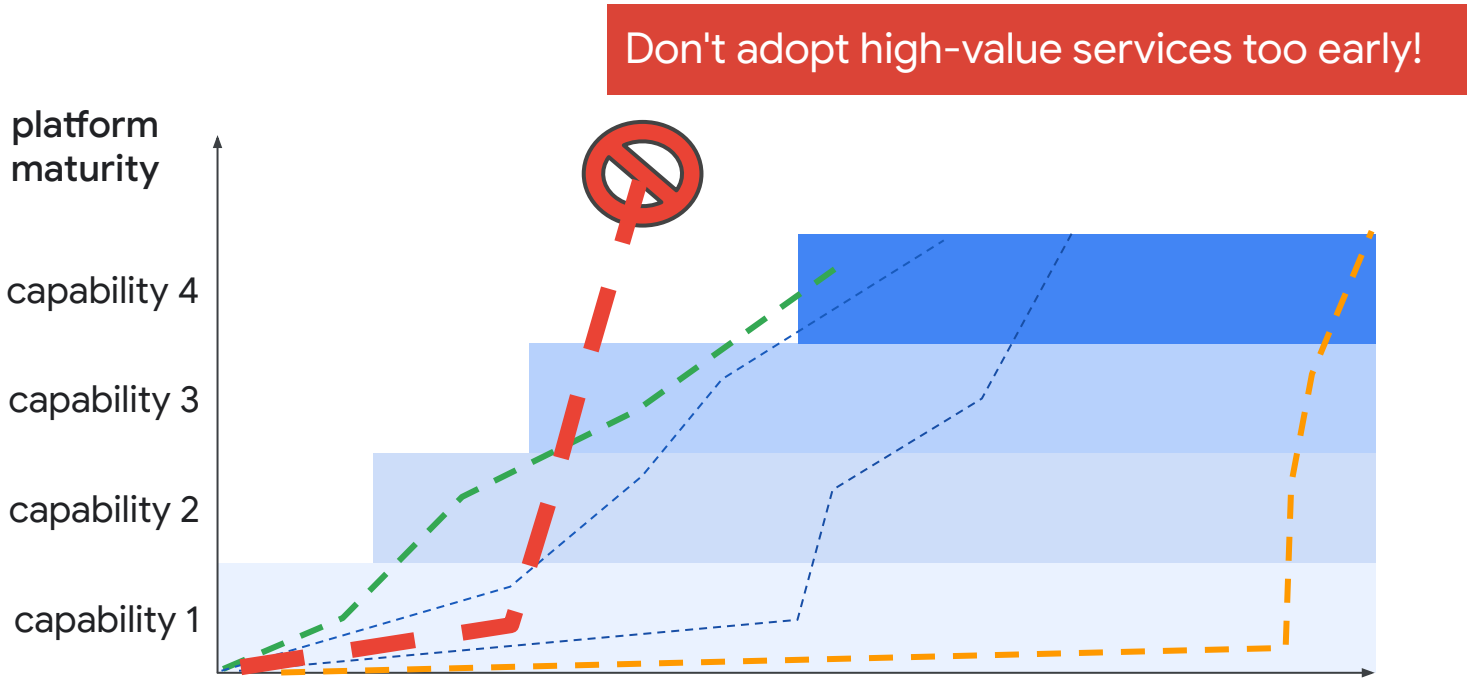
Introduce services as their platform requirements are met



Building Platforms



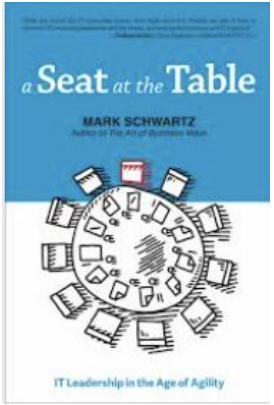
Building Platforms



SRE Practices

- Avoid SRE as dev support / "Developer IT"
  - "hey prod is broken" 😞
  - "my laptop is broken" 😫
  - "the printer is broken" 😈
- Target **mid term planning** (6 months to 2 years)
- Just getting started? Learn from Incidents:
  - ⇒ Incident Response, Postmortems and review, **repeat**.
- **Cause-based** alerting vs **symptom-based**
- Use feedback loops to make this **intentional** (e.g. PDCA / OODA)

SRE Practices



- Consider a **Chief Reliability Officer** (CRO)
  - Seat at the table for strategic reliability decisions
  - **Sponsorship matters!**

- Compare with a **Chief Information Security Officer** (CISO)
  - "Security is everyone's responsibility"
  - Enterprises have CISOs **to nurture and champion** efforts

- An **investment**, **not a cost center**

- **Sponsor abandonment** ⇒ team failure

SRE Practices

So, is it working yet?

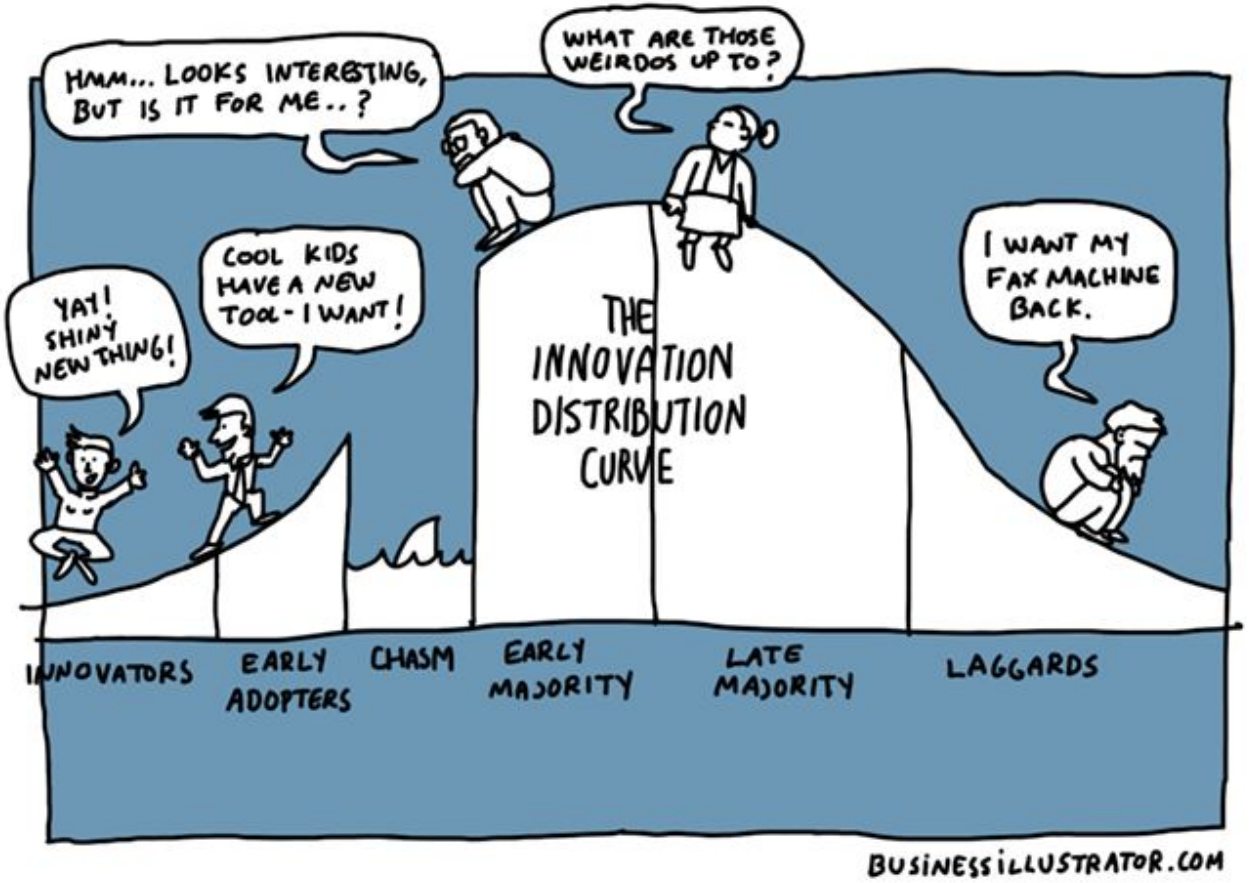
Progress **won't be in a dashboard**

Need to **use proxy metrics** to evaluate:

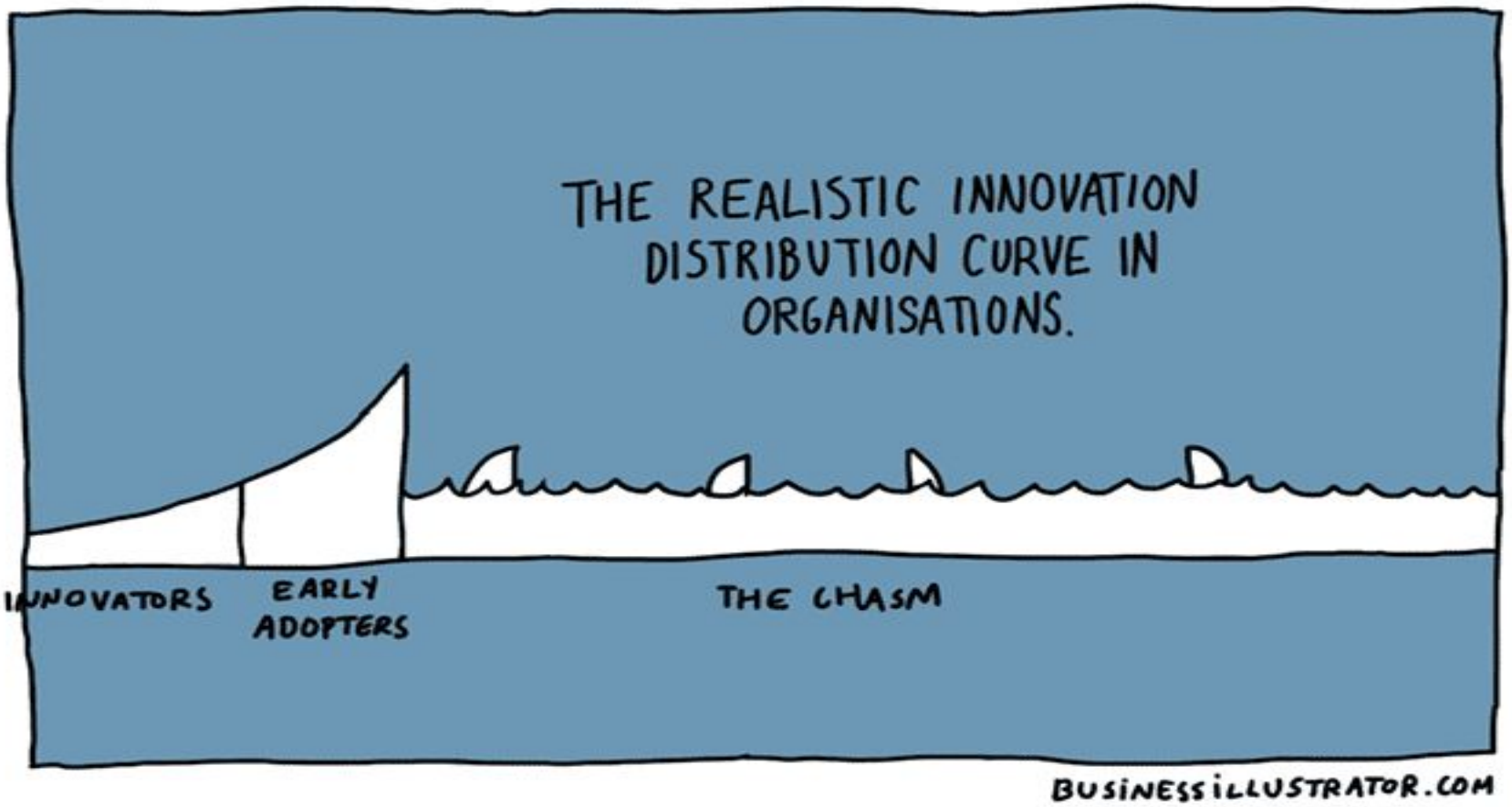
- ⚖️ Can you **enforce consequences** when error budget is exhausted?
- 🦸 Is individual **heroism** still being praised?
- 💥 Are you **correlating funding with outages**?
- 🥳 Is **success celebrated** or treated as table stakes?

Section 5
Actively Nurturing Success

Actively Nurturing Success



Actively Nurturing Success



"Culture eats strategy for breakfast"

Google did the research on what makes the magic happen.

It's **not just**: free food and ping pong

Those don't **cause** the right culture.
They **come from** the right culture.

Follow the re:Work model to adapt your culture

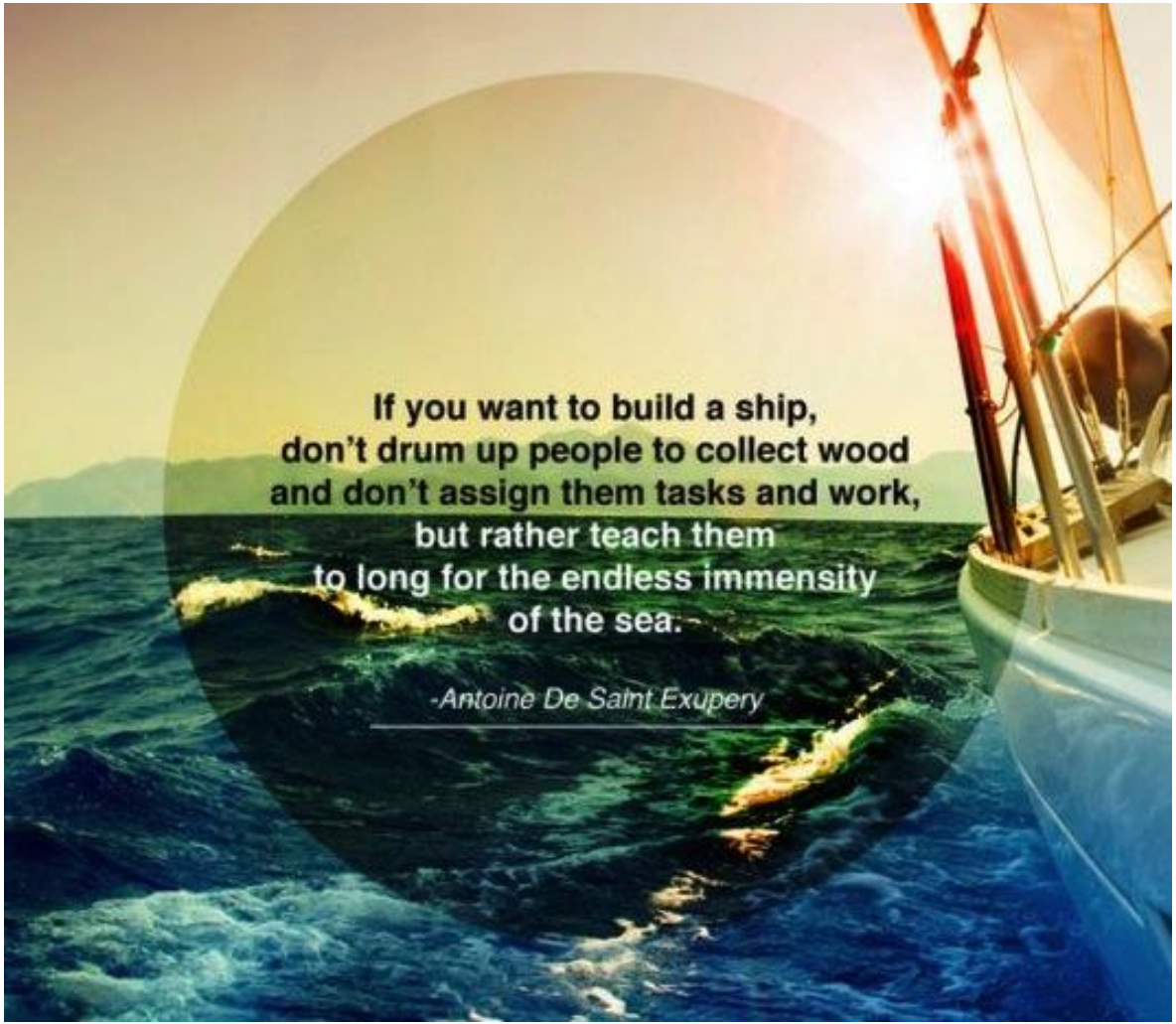
Actively Nurturing Success

Hints and tips!

- ↗ Strive for **sublinear scaling**
- 🦋 Building and retaining sustainable teams (grow teams organically)
- 🐒 SRE is dynamic and **evolves** **over time**
- 🕰️ High reliability levels take (much) longer than you think
- 🚜 Understand the dedicated org model **isn't supposed to be a silo**
- ☀️ Communities need water and sunlight to thrive
- 💰 Promotion/training/**compensation** match other roles (esp dev)

Conclusion

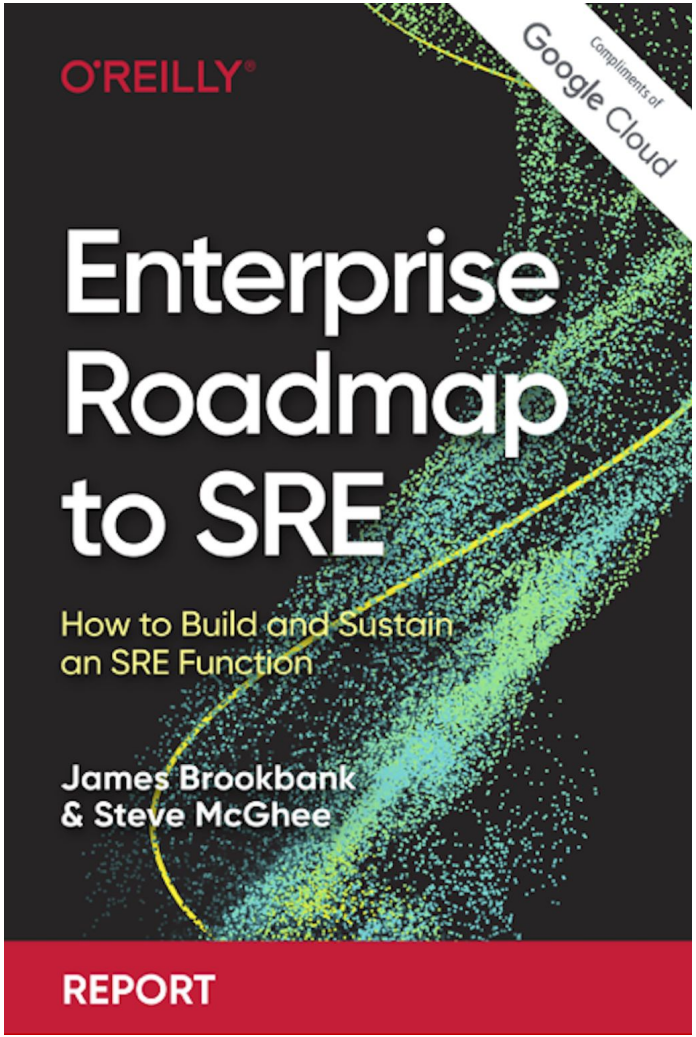
Conclusion


If you want to build a ship,
don't drum up people to collect wood
and don't assign them tasks and work,
but rather teach them
to long for the endless immensity
of the sea.
-Antoine De Saint Exupery

Enterprise Roadmap to SRE

Copies are available!

https://g.co/cloud/ent-sre



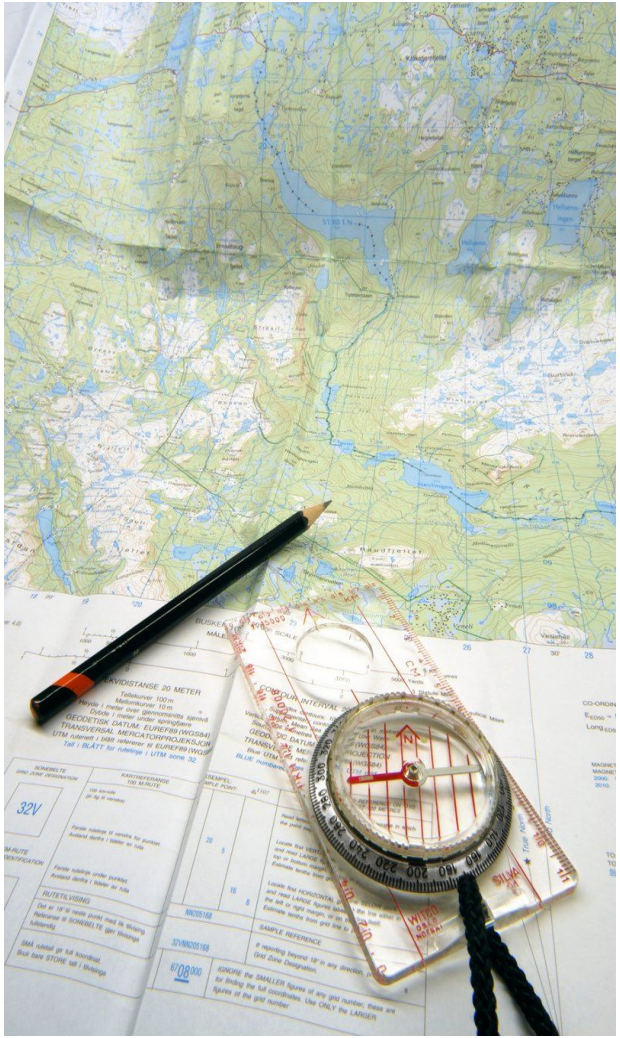
Fin / Q&A

Bonus – What's next?

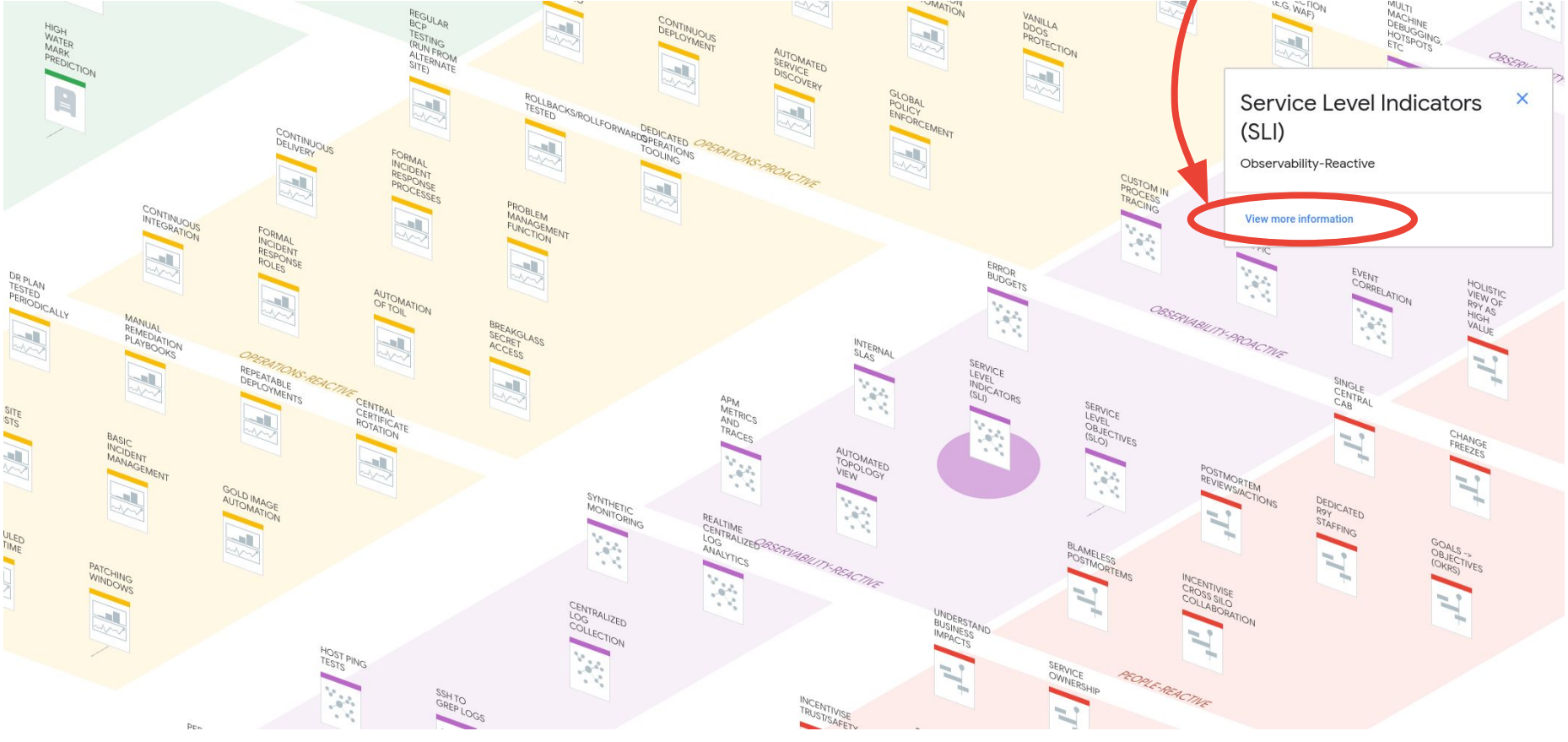
Reliability Mapping

- An SME-constructed map of **reliability capabilities**
- Divided into **Eras** (demarcated by availability nines)
- And **Streams/Personas** e.g. Dev, Infra, Observability

This is in preview!



Zoomed Reliability Map - https://r9y.dev/



Big Picture Reliability Map - https://r9y.dev/