

# **SRE in Enterprise**

Site Reliability Engineering

Steve McGhee & James Brookbank Google Cloud





#### **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 <u>cloud solutions architect</u>
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

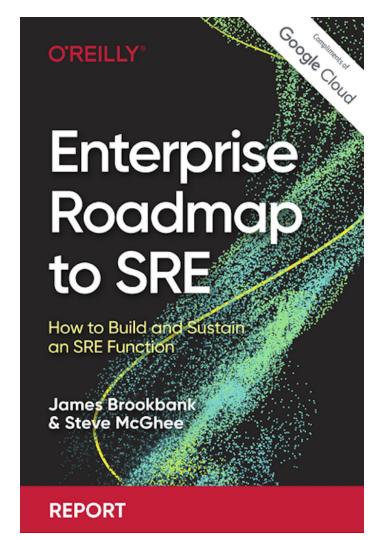
Learn about the challenges of adopting site reliability engineering (SRE) in enterprises, and how we recommend cloud customers go about this journey

- 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

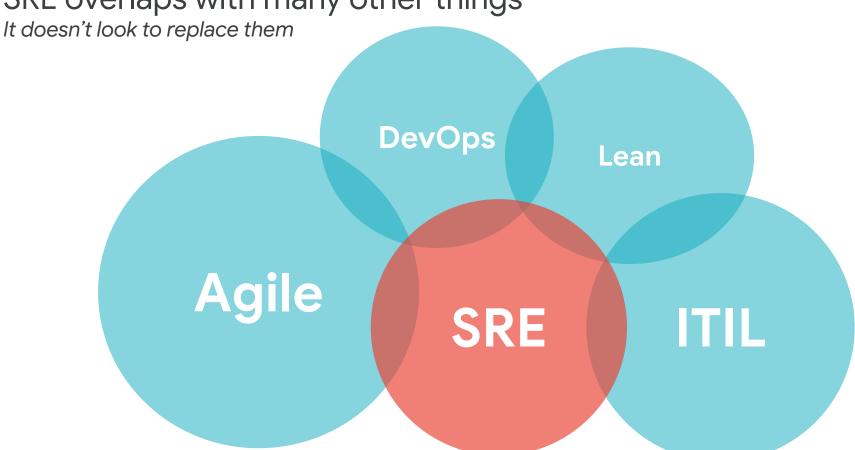


Downloads are free Physical copies are available!



# Getting started with Enterprise SRE

SRE overlaps with many other things



#### 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

• • •



Software Delivery and Operations Performance

Predict

Commercial Outcomes

(e.g. market share, profitability, employee retention)

#### As measured by

#### Velocity

Predict

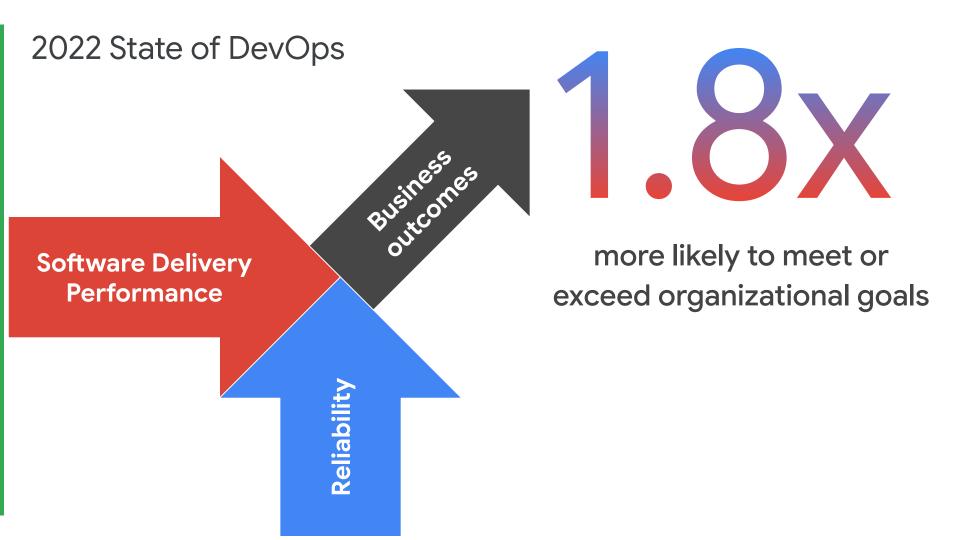
- lead time for changes
- deployment frequency

#### Stability

- time to restore service
- change failure rate

#### Reliability

g.co/devops



## 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

Communities of practice

Bottom-up or grassroots

Training centers

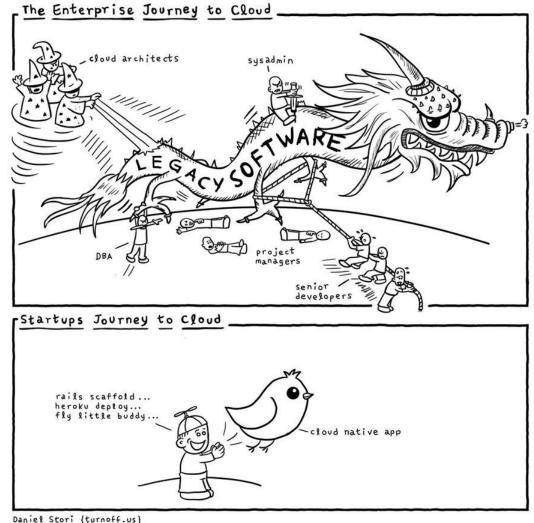
Centers of excellence

A big bang approach

#### 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...



Daniel Stori {turnoff.us}
Thanks to Michael Tharrington

## 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	1
Responsibilities shirked	Narrow responsibilities	į	Risks are shared	i
Bridging discouraged	Bridging tolerated	1	Bridging encouraged	   
Failure leads to scapegoating	Failure leads to justice	1	Failure leads to enquiry	
Novelty crushed	Novelty leads to problems		Novelty implemented	

Why the SRE approach to Reliability?

# What is driving the evolution of SRE?

(Spoiler alert: selection pressure)

## EVOLUTION OF OPERATIONS

OPS



- . PRIMORDIAL, PROTOZOIC
- . BORN IN THE SWAMPS OF PERL
- . OPERATES IN A SINGLE-CELL SILO
- · SURPRISINGLY RESILIENT

DEVOPS (SECTION)

- · A CROSS-FUNCTIONAL MARVEL
- · VASTLY INCREASED AGILITY
- SECRETLY JUST A BUNCH OF SINGLE CELLS THAT HAVE LEARNED NOT TO KILL EACH OTHER

DEVSECOPS TO

DEVSECMLOPS 1/2

- · MORE ADVANCED, MORE PARANOID
- · SECURITY IS AUTOMATED RIGHT INTO ITS DNA
- KNOWS THAT SHARED

  RESPONSIBILITY IS THE ONLY
  ESCAPE FROM FOSSILIZATION
  - . WHAT EVEN IS THIS?
  - . IS IT A FISH WITH FEET?
  - WE SHOULD PROBABLY LEAVE IT ALONE FOR A FEW MILLION YEARS AND SEE WHAT HAPPENS



- DOES NOT CARE
  ABOUT YOUR ORG
  STRUCTURE
- VULNERABLE ONLY TO DIRECT METEOR STRIKES
- . WHAT WERE WE TALK-ING ABOUT, AGAIN?

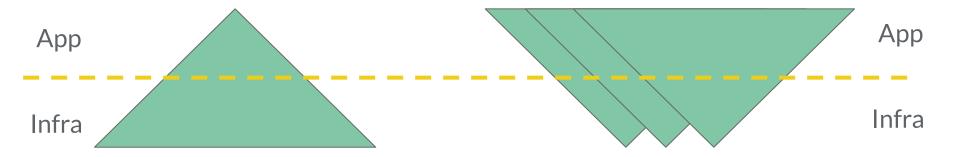
# Q: Can you build **more reliable** services on **less reliable** infra?



# 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



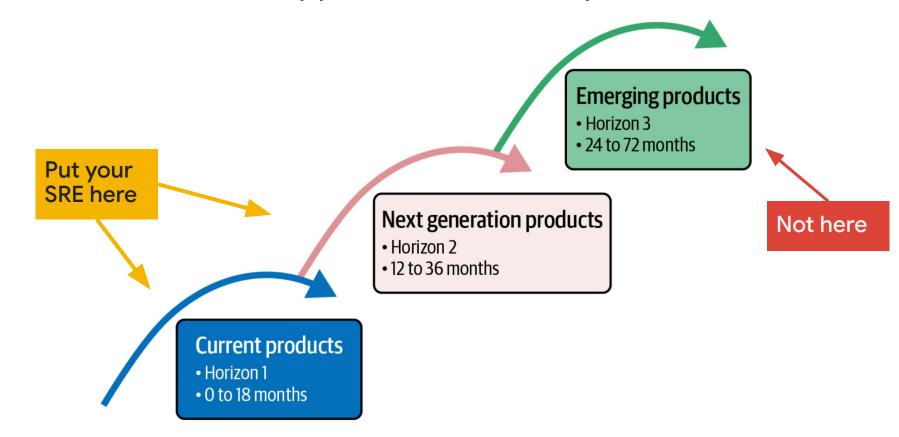
#### **Traditional Model**

- 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?



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** (\$\(\frac{1}{2}\)) in long-term operational efficiency (\$\(\frac{1}{2}\))

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

This is **critical** for your finance team



TV Globo



© Sony Pictures, Marvel

## Start small

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

# Invest in people

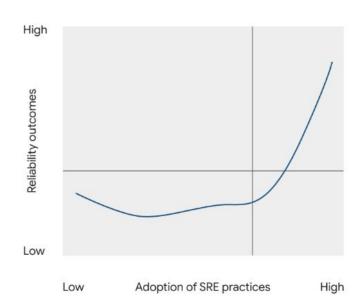
- Staffing and retention
- Hiring feels easy but growing is more sustainable
- Don't fire everyone in ops who can't code
- Value existing employees they know the business!

#### **Embrace risk**

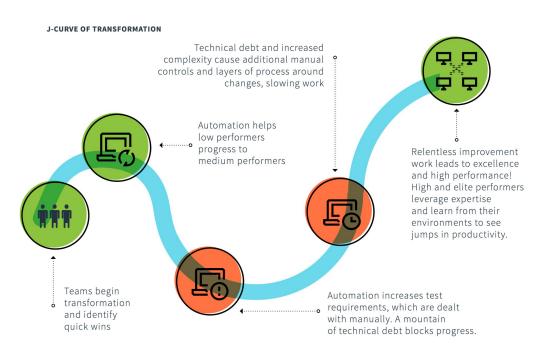
- Create a safe to fail environment
- You can't only take risks that will succeed
- Demonstrating active leadership is important
- Treat failures as unplanned learning opportunities

## SRE & DevOps agree (SoDR 2022)

## Give it time

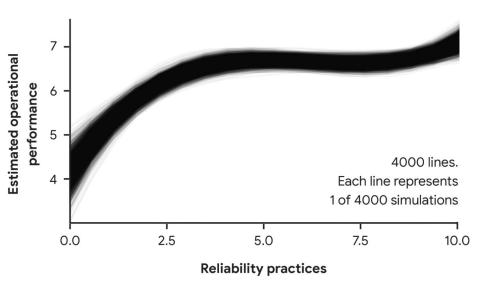


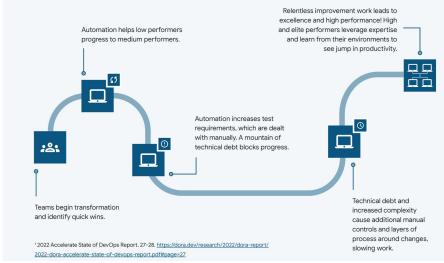
Teams that persist beyond initial steps of SRE adoption see increasing improvement in reliability outcomes



#### SRE & DevOps agree (SoDR 2023)

## Give it time





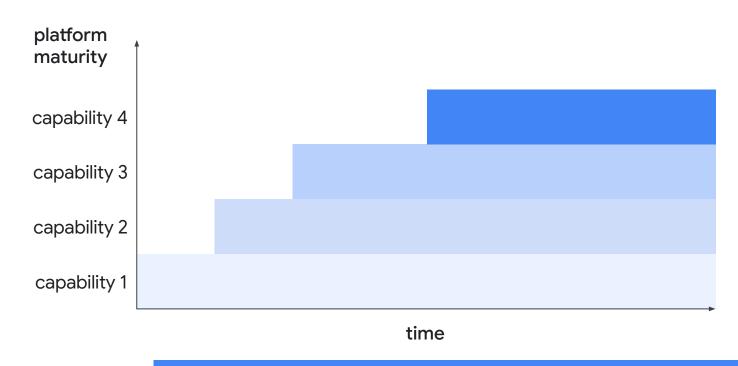
Teams that persist beyond initial steps of SRE adoption see increasing improvement in reliability outcomes

# **SRE Practices**

# SRE Practices

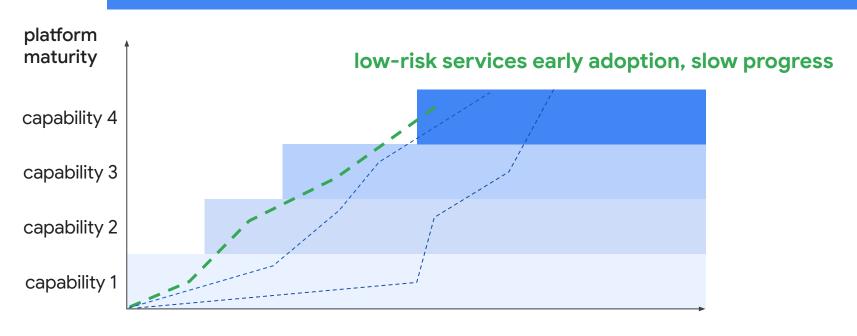
Platform Engineering so hot right now

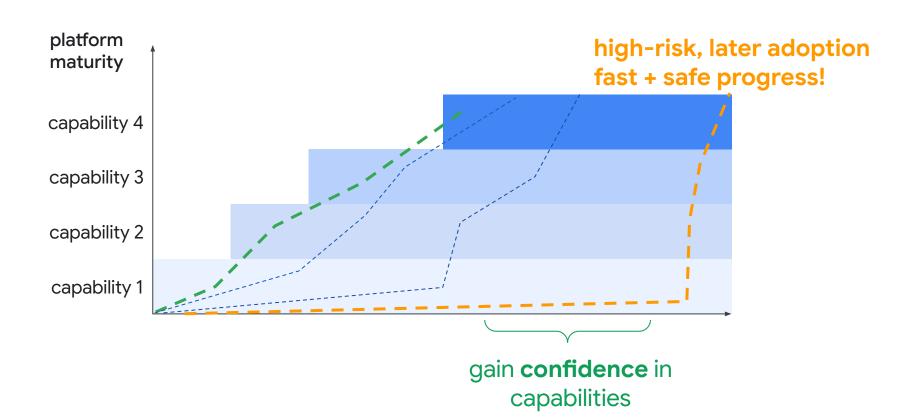
- Build a platform of capabilities!
- Capabilities get built, purchased, added over time
- Services are introduced to the platform, as it makes sense.
- Antipattern: Pick the toughest thing first (since that will totally fix all the problems)

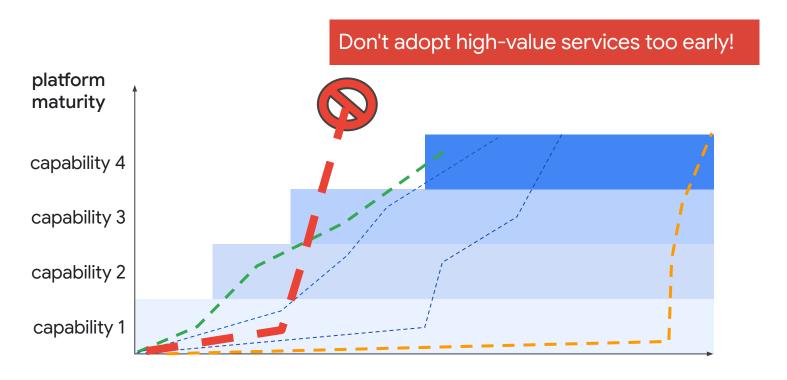


Add capabilities over time: CI/CD, rollbacks, multi-cluster

#### Introduce services as their platform requirements are met





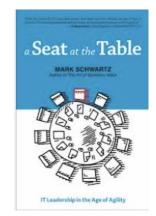


#### **SRE Practices**

- Avoid SRE as <u>dev support</u> / "Developer IT"
  - "hey prod is broken"
  - o "my laptop is broken" 😩
  - o "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. <u>PDCA</u> / <u>OODA</u>)

#### **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

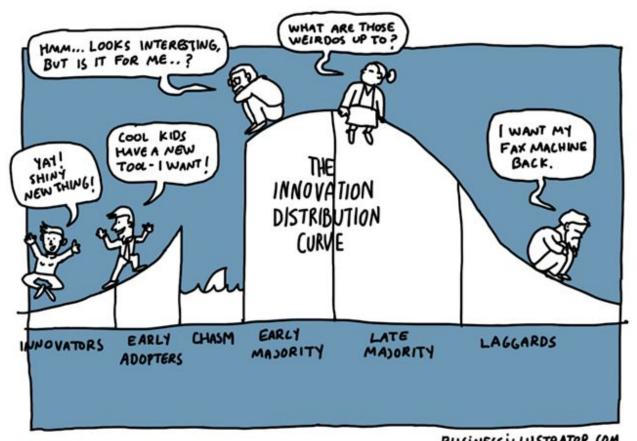
#### **SRE Practices**

So, is it working yet?

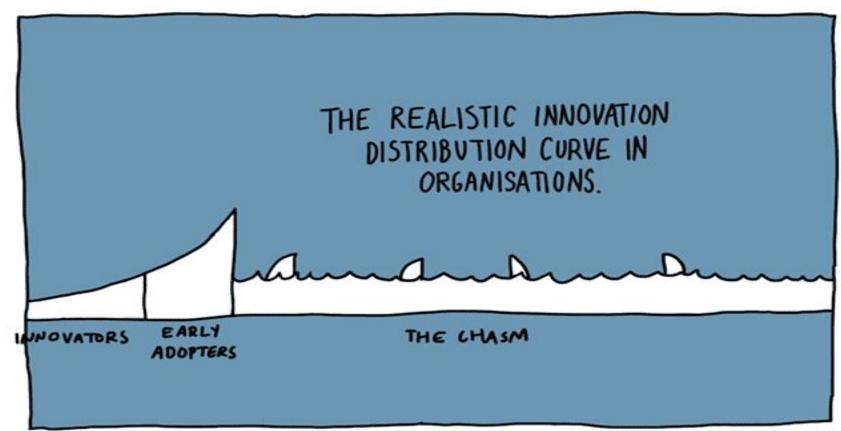
Progress won't be in a dashboard

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?



BUSINESS ILLUSTRATOR.COM



BUSINESSILLUSTRATOR.COM

"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

## Psychological Safety

Team members feel safe to take risks and be vulnerable in front of each other.

## 2 Dependability

Team members get things done on time and meet Google's high bar for excellence.

# 3 Structure & Clarity

Team members have clear roles, plans, and goals.

#### Meaning

Work is personally important to team members.

Impact

Team members think their work matters and creates change.

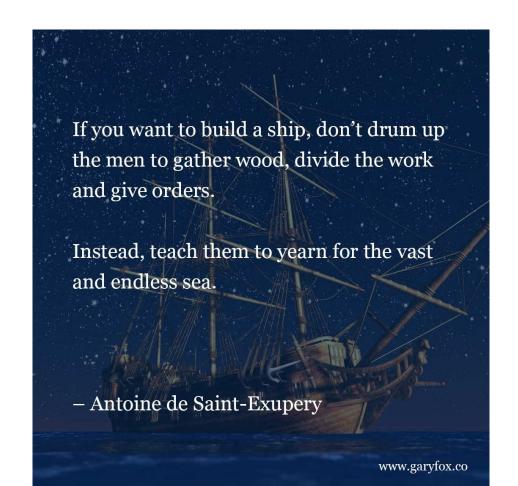


#### Hints and tips!

- Strive for sublinear scaling
- We Building and retaining sustainable teams (grow teams organically)
- SRE is dynamic and evolves over time
- A 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

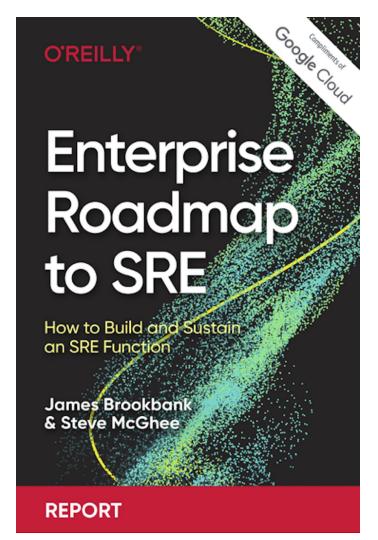


### Enterprise Roadmap to SRE



Copies are available!

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



## Fin / Q&A



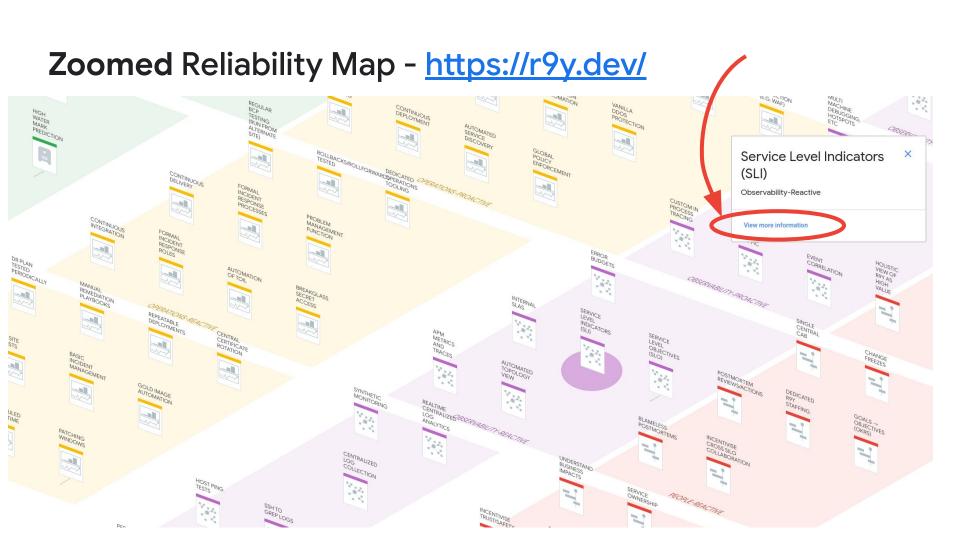
Bonus - What's next?

### **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!





## Big Picture Reliability Map - <a href="https://r9y.dev/">https://r9y.dev/</a>

• I DEMO			DETERMINISTIC			REACTIVE			PROACTIVE			AUTONOMIC		1.7
LOCAL DEVELOPMENT	MONOLITH	DEVI		CODE REVIEW		PRE MERGE HOOKS	ACTIVE PASSIVE CLUSTERS	MICROSERVICES	LEFTSHIFT RELIABILITY DESIGN	GRACEFUL SERVICE DEGRADATION (INDIVIDUAL CUJS)	LEFT SHIFT PERFORMANCE TESTING	GRACEFUL SERVICE DEGRADATION (UNIVERSAL)	BOUNDED CONTEXT	PROTOBUFS
		SMOKE TESTS OF	AUTOMATED UNIT TESTING	MULTI SERVICE DEVELOPMENT	EVEL OP	DISTRIBUTED SYSTEMS AWARENESS	DEPLOYMENTS IN PLACE	FEATURE FLAGG		ACTIVE ACTIVE MULTI CLUSTER	BASIC CHAOS TESTING	SERIOUS DESIGN/DOMAIN DRIVEN DESIGN	DESIGN AROUND UNIVERSAL FAILURE DOMAINS	SHARDED DATA
		MANUAL TESTS	CODE VERSION CONTROL	■ PUNCTIONAL TESTS	SEMI AUTOMATED INTEGRATION	DATA VERSIONING	TRAFFIC SHIFTING	INSTRUMENTATION FOR N PROCESS TRACES	BACKWARDS VERSION COMPATIBLITY BY DEFAULT	CANARY DEPLOYMENTS	DEVELO			
		Menistr			ACTIVE			ACTIVE		LEFT SHIFT QA TESTING (SDET)	EZE TESTING RIVER	MULTI CLUSTER ROLLOUT POLICY	UNIVERSAL SMART RETRIES	SHARDED SERVING
				MANUAL INTEGRATION TESTS	REGULAR RELEASE CADENCE		CONTAINERS		BLUE GREEN DEPLOYMENTS	Fuzz testing	DISTRIBUTED SYSTEMS (NO ACTIVE/PASSIVE)	AUTOMATIC ASSURED CAPACITY AND PERFORMANCE TESTING	ANDON CORD/BIG RED BUTTON	CODE QUALITY THRESHOLD (CODE REUSE PREFERRED)
											MIC	LOW CONTEXT ARCHITECTURE, DESIGN, CODING, OPERATIONS	LANGUAGE READABILITY	ONLY CUSTOMIZE COMPONENTS NEED CUSTOMIZATION DESIGN FOR CHAOS
														FORMAL METHODS ( TLA+)
LOCAL DATA STORAGE		SINGLE ZONE	DNS/SIMPLE LB		BASIC LINEAR CAPACITY PROJECTION	ADVANCED LOADBALANCING	■ IAC	UNDERSTAND INFRASTRUCTURE 2 FAILURE DOMAINS 2	AUTO FAILOVER	FAILURE TESTING IN PROD	■ N+1 AS STANDARD	N+2 THINKING		■ N+2 GLOBAL PLANN
■ PET HOST			■ >1COMPUTER	DISTRIBUTED STORAGE	ROCCU	ALTERNATE SITE REPLICATION	CATTLE INFRASTRUCTURE	CONTAINER ORCHESTRATOR	■ AUTO SCALING	ELIMINATE SPOFS (HARDWARE & SOFTWARE)	SERVICE DISCOVERY		■ DRAIN/SPILL (N/S 6 E/W)	
					W. Real	BASIC LOADTESTING	■ MULTIZONE	HOLTZ-WINTER CAPACITY PROJECTIONS	FAILURE INJECTION	■ N+1 REGIONAL PLANNING	■ L7 GLOBAL LB			
					TVE		HIGH WATER MARK PREDICTION	COACTIVE	ASSURED CAPACITY LOAD TESTING	REAL WORLD TRAFFIC LOAD TESTING	L4 REGIONAL LOAD     BALANCING			
											■ MULTI REGION			
OFF-HOST BACKUP	RPO/RTO DEFINED	DR PLAN	RPORTO REFINED	DR PLAN SIMULATED/TABLETOP	DR PLANTESTED PERIODICALLY	CONTINUOUS	CONTINUOUS DELIVERY	REGULAR BCP TESTING (RUN FROM ALTERNATE SITE)	% BASED TRAFFIC STEERING	ACTIVE ACTIVE DATASTORES	INTERNAL RATE LIMITING		AUTONOMOUS RESPONSE SYSTEMS	AUTOMATIC ROLLBA
MANUALLY CREATED MACHINES	MANUAL VIMIMAGES	CUSTOM VMS VIA SEMI-	ITIL STYLE NOC	DR SITE EXISTS	MANUAL REMEDIATION PLAYBOOKS	FORMAL INCIDENT RESPONSE ROLES	FORMAL INCIDENT RESPONSE PROCESSES	ROLLBACKS/ROLLFORWARD TESTED	CONTINUOUS DEPLOYMENT	EXTERNAL RATE LIMITING	PRODUCTION CHANGELOG	PROACTIVE DDOS COUNTERMEASURES	■ LOAD PREDICTION	
	MANUAL REMEDIATION	MINNE	SCHEDULED DOWNTIME	BASIC INCIDENT MANAGEMENT	REPEATABLE DEPLOYMENTS	AUTOMATION OF TOIL	PROBLEM MANAGEMENT FUNCTION	DEDICATED OPERATIONS TOOLING	AUTOMATED SERVICE DISCOVERY	DATA COLLECTION AUTOMATION	MOSTLY AUTOMATED REMEDIATION			
		STIC	PATCHING WINDOWS	GOLD IMAGE AUTOMATION	CENTRAL CERTIFICATE (1)	BREAKGLASS SECRET ACCESS		ICTIVE	GLOBAL POLICY ENFORCEMENT	VANELA DDOS PROTECTION	DIRT TESTING			
											PRODUCT SPECIFIC DOOS PROTECTION (E.G. WAF)			
HOST METRICS AND LOGGING		FER HOST ALARMS OF STATEMENT OF	■ HOST PING TESTS		SYNTHETIC MONITORING	APM METRICS AND TRACES	INTERNAL SLAS	TRAGIN	CUSTOM IN PROCESS TRACING	CROSS SERVICE TRANSACTION TESTING	MULTI MACHINE DEBUGGING, HOTSPOTS ETC	ANOMALY DETECTION	OBERVABILITY INTEGRATION ACROSS TOOLS	
			SSH TO GREP LOGS	CENTRALIZED LOG COLLECTION	REALTIME CENTRALIZED OF ANALYTICS	AUTOMATED TOPOLOGY VIEW	SERVICE LEVEL INDICATORS (SU)		RECORD AND REPLAY TRAFFIC	ADVANCED VIZUALIZATIONS (HEATMAPS,		■ NEAR MISS DETECTION		
							SERVICE LEVEL OBJECTIVES (SLO)		EVENT CORRELATION	FLAMEGRAPHS)				
■ HIGH CONTEXT BEHAVIOURS			■ RCA/S WHYS	INCENTIVISE TRUST/SAFETY	UNDERSTAND BUSINESS NAME OF THE PROPERTY OF T	BLAMELESS POSTMORTEMS	POSTMORTEM REVIEWS/ACTIONS	SINGLE CENTRAL CAB	HOLISTIC VIEW OF R9Y AS HIGH VALUE	RELIABILITY EXECUTIVE/SPONSOR EXISTS	RELIABILITY HAS A SEAT AT THE TABLE O	R9Y IS A PRODUCT DIFFERENTIATOR	R9Y CAN STOP FEATURE LAUNCH	PROACTIVE RISK AND SCALING ANALYSIS
	MANAGING PET CONFIGURATION DRIFT	*FOP	■ MEASURE EVERYTHING	DATA DRIVEN DECISIONS	SERVICE OWNERSHIP	INCENTIVISE CROSS SILO COLLABORATION	■ DEDICATED R9Y STAFFING	CHANGE FREEZES	VERTICAL SCALE IS AN ANTIPATTERN	SRE SWE ROLES INTRODUCED	■ EMPOWERED ROY STAFF	ROY EMBEDDED IN HIGH LEVEL STRATEGY AND OPERATIONS	ADVANCED COST OPTIMIZATION	FOCUS ON PREVENT AND NEAR MISSES INSTEAD OF OUTAGE
		■ TODO LISTS	■ WATERFALL PROJECTS/PMO	■ SMART GOALS	TIVE		■ GOALS -> OBJECTIVES (OKRS)	ARCHITECTURE REVIEWS	HIGH PERFORMING STAFF (PROMOTION AND HIRING)	REACTIVE RISK AMALYSIS	BASIC COST OPTIMISATION			
									INTRODUCING DEDICATED SRES	TOIL BUDGETS	DECREASED RELIANCE ON 3RD PARTY SAAS			