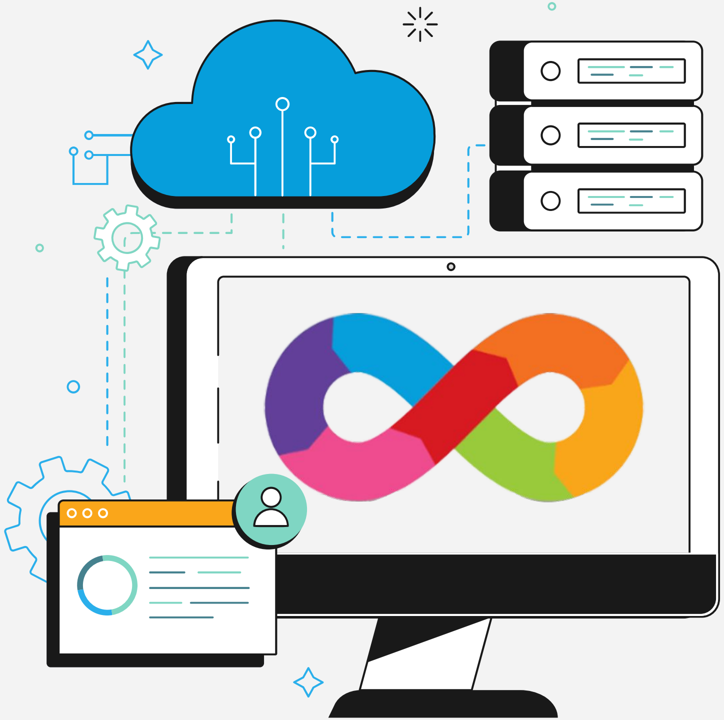
Introduction to DevOps

@ IBA - SMCS

Week 14 - 2

SRE & Platform Engineering



Obaid ur Rehman
Software Architect / Engineering Manager @ Folio3

SRE & PE

- Site Reliability Engineering - SRE
  - What is SRE
  - Why the need for SRE?
  - What is Reliability
  - What does DevOps do and whats SRE about
  - class SRE implements DevOps
  - What else class SRE implements.
  - SLA & error budget
  - SRE Tasks and responsibilities
- Platform Engineering
  - What is PA?
  - Why Platform Engineering.

What is SRE

- Originated at Google with Ben Treynor Sloss who founded a site reliability team after joining the company in 2004.

- He assembled a team of Software engineers to do what was traditionally an operations work.

- Framework for operating large scale systems reliability.

> *"What exactly is Site Reliability Engineering, as it has come to be defined at Google? My explanation is simple: SRE is what happens when you ask a software engineer to design an operations team."*

- Benjamin Treynor Sloss,
Vice President, Engineering, Google

Why the need for SRE?

- Traditionally DevOps is about faster delivery of software.
- Overtime the focus shifted to pushing things out faster
- This compromised stability.
- No dedicated role in DevOps that focused on reliability.
- One must remember that reliability is a key tenant of DevOps

What is Reliability?

- Reliability is the extent to which a resource functions as required upon request.

- For example, if your users require an application to serve a web page within one second, and it does this 99%of the time while also maintaining a 99% availability level, that would be a relatively reliable application.

- In contrast, an application that responds to almost all requests but that suffers from high latency or error rates would not be very reliable, although it might be highly available.

What makes a system un**reliable?**

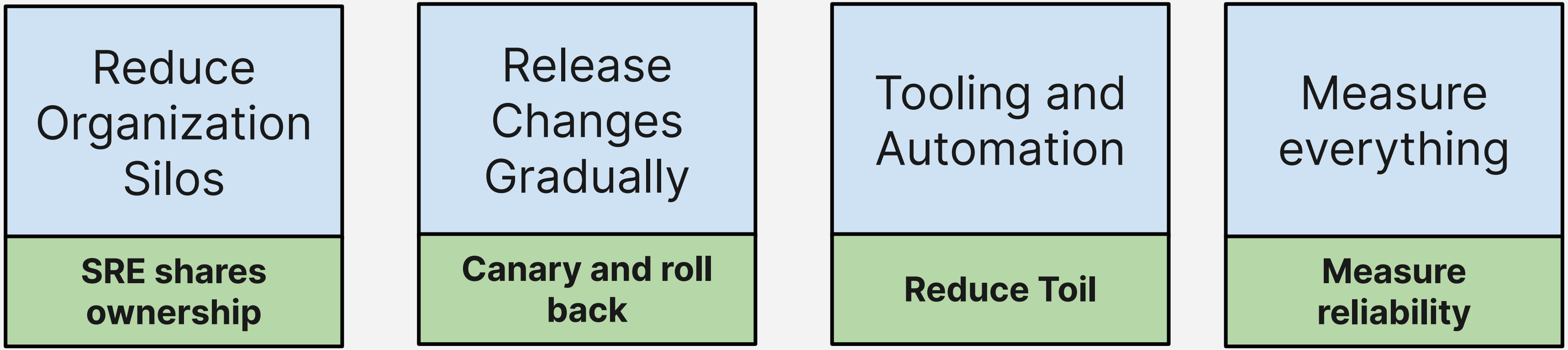
- Pushing out changes, features & **fast.**
- A business need to stay competitive.
- Innovation in product.

Lets see what DevOps does

- Reduce Organization Silos
- Release Changes Gradually
- Tooling and Automation
- Measure everything

class SRE implements DevOps

Lets see what SRE does



Reduce Organization Silos

SRE shares ownership

Release Changes Gradually

Canary and roll back

Tooling and Automation

Reduce Toil

Measure everything

Measure reliability

Wait what?

- If SRE implements DevOps what is the difference between them?
- While there is a lot of overlap between both, SRE is a more focused approach to reliability.
- Like we mentioned earlier, DevOps are more focused on deployment, release frequency.
- In smaller team you would be doing both DevOps and SRE.

```
class SRE implements DevOps, [others]
```

What does SRE do (additionally)?

- Automates releasing **FAST** & **SAFE releases** thus maintaining Reliability and reducing **downtime**.

- How does it do that? Well by automating the process of evaluating the impact a change will have on reliability.

- Among other things…

How is the automated evaluation done?

- SLA & Error Budgets.

Error Budget

- An error budget is the amount of acceptable unreliability a service can have before customer happiness is impacted.

- If a service is well within its budget, the developers can take more risks in their releases. If not, developers need to make safer choices.

SLA - Service Level agreement

- A commitment between the user and service provider on how reliable the system is going to be for end-users.

- Expressed in percentages like 99.99% available.

- Sample: Go here for AWS SLAs

- Availability Table: https://sre.google/sre-book/availability-table/

- **100% Reliability?**

> “100% is the wrong reliability target for basically everything.

Benjamin Treynor Sloss

Vice President of 24x7 Engineering, Google

Who decides SLAs?

- Product owners, stakeholders, Business people, Engineers based on competition, industry standards.

- Once SLAs are decided, the objective of SRE is to maintain it.

- Using observability, SRE will monitor SLAs.

SLAS as Brakes

- An SLA can act as the gas pedal and brakes for development.
- If you aren't going to reach the target, SLOs slam on the brakes. Policies like code freezes or bug bashes stop further development, reducing the risk of further issues, and refocusing efforts toward making the current code more reliable.

SLA Tasks and Responsibilities

- **Automation:** Calculating SLAs, Detecting issues.

- **Monitoring & Observability:** To calculate SLAs and to monitor the health and reliability of the system.

- **Support & On-call:** Since system is not 100% reliable, things will go down. SREs remain vigilant and helps identify and fix issues. Making sure the outage is fixed quickly.

SLA Tasks and Responsibilities

- **Postmortem:** Once the issues is resolved, SRE will report on what happened, how it happened, what was done to fix the issue.

- Example Postmortem: https://sre.google/sre-book/example-postmortem/

SLA Tasks and Responsibilities

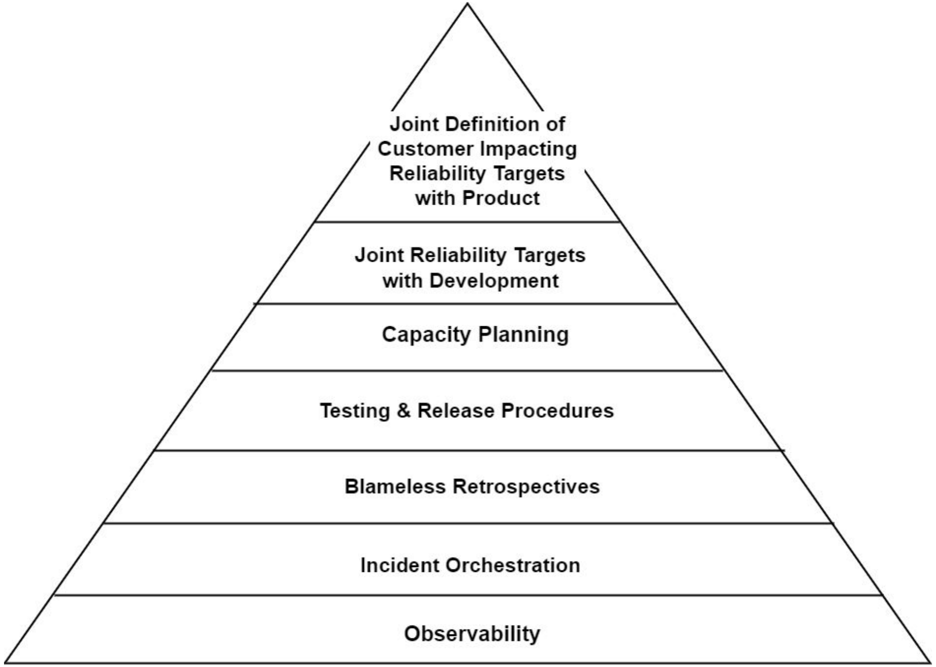
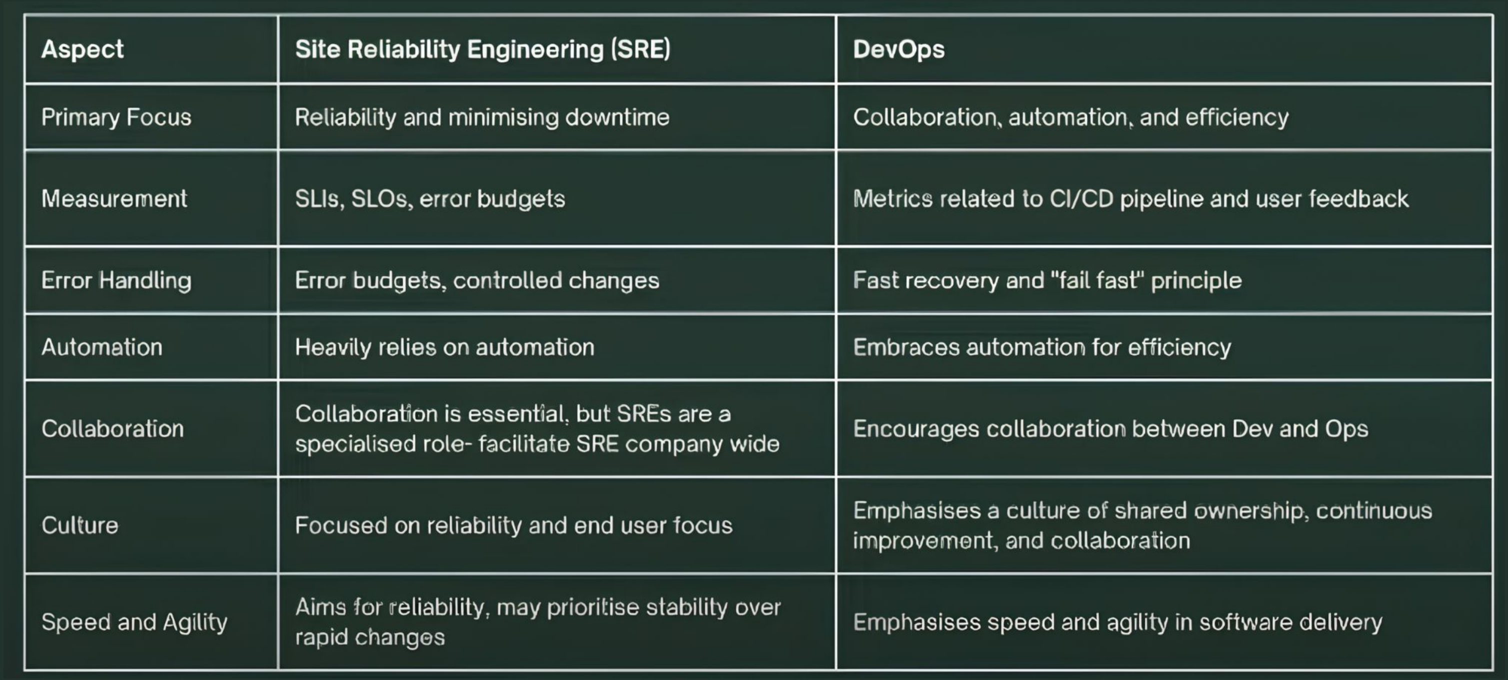

Joint Definition of Customer Impacting Reliability Targets with Product
Joint Reliability Targets with Development
Capacity Planning
Testing & Release Procedures
Blameless Retrospectives
Incident Orchestration
Observability

Fig. Service Reliability Hierarchy

Further Readings…

The Google SRE Book:
https://sre.google/sre-book/table-of-contents/

DevOps V/s SRE



Aspect	Site Reliability Engineering (SRE)	DevOps
Primary Focus	Reliability and minimising downtime	Collaboration, automation, and efficiency
Measurement	SLIs, SLOs, error budgets	Metrics related to CI/CD pipeline and user feedback
Error Handling	Error budgets, controlled changes	Fast recovery and "fail fast" principle
Automation	Heavily relies on automation	Embraces automation for efficiency
Collaboration	Collaboration is essential, but SREs are a specialised role- facilitate SRE company wide	Encourages collaboration between Dev and Ops
Culture	Focused on reliability and end user focus	Emphasises a culture of shared ownership, continuous improvement, and collaboration
Speed and Agility	Aims for reliability, may prioritise stability over rapid changes	Emphasises speed and agility in software delivery

Platform Engineering

Platform Engineering

- Platform engineering is a practice built up from DevOps principles that seeks to improve each development team’s security, compliance, costs, and time-to-business value through improved developer experiences and self-service within a secure, governed framework.

- Lately, there is a lot of industry excitement around the term platform engineering. In fact, Gartner expects around **80 percent of engineering organizations to have a team dedicated to platform engineering by 2026.**

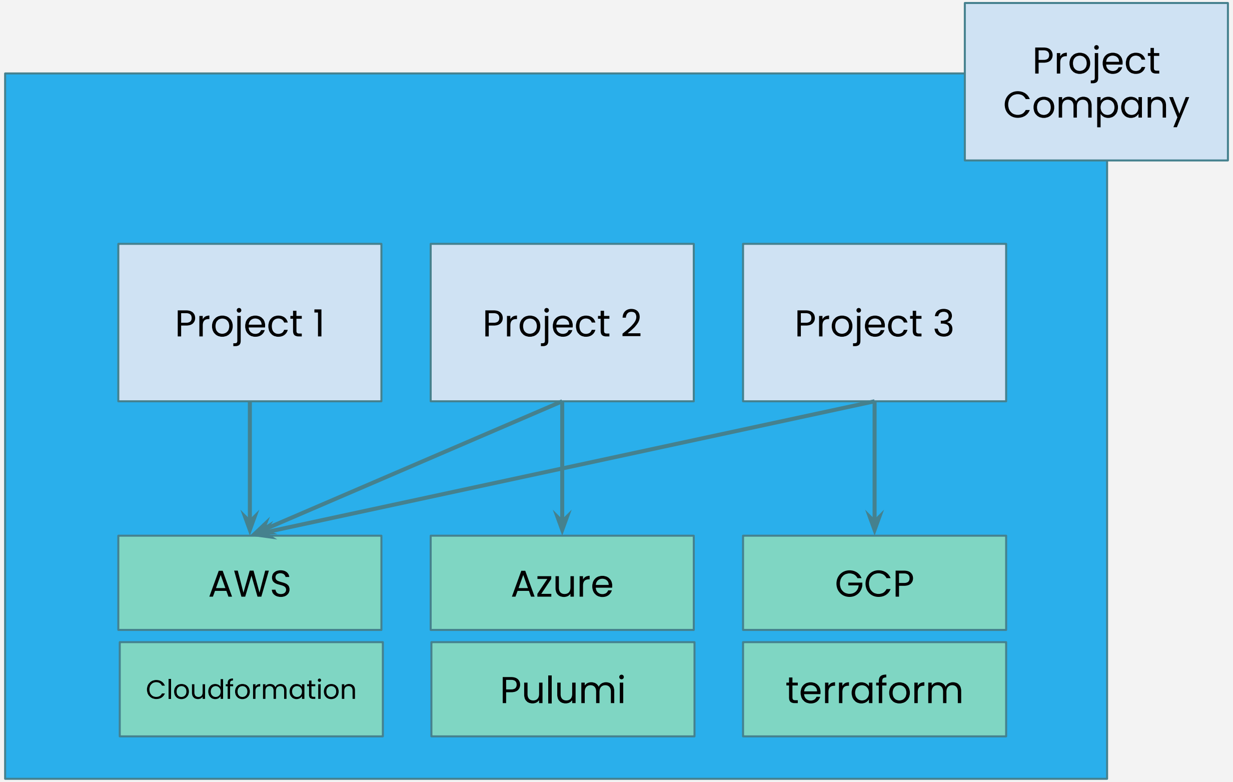
Source: https://learn.microsoft.com/en-us/platform-engineering/overview/

Platform Engineering

- Platform Engineers build an **internal developer platform. (IDPs).** It is focused on a company's internal development practices. You define a set of recommended and supported development paths to production.

- Some say it will eliminate the need for DevOps Engineers.

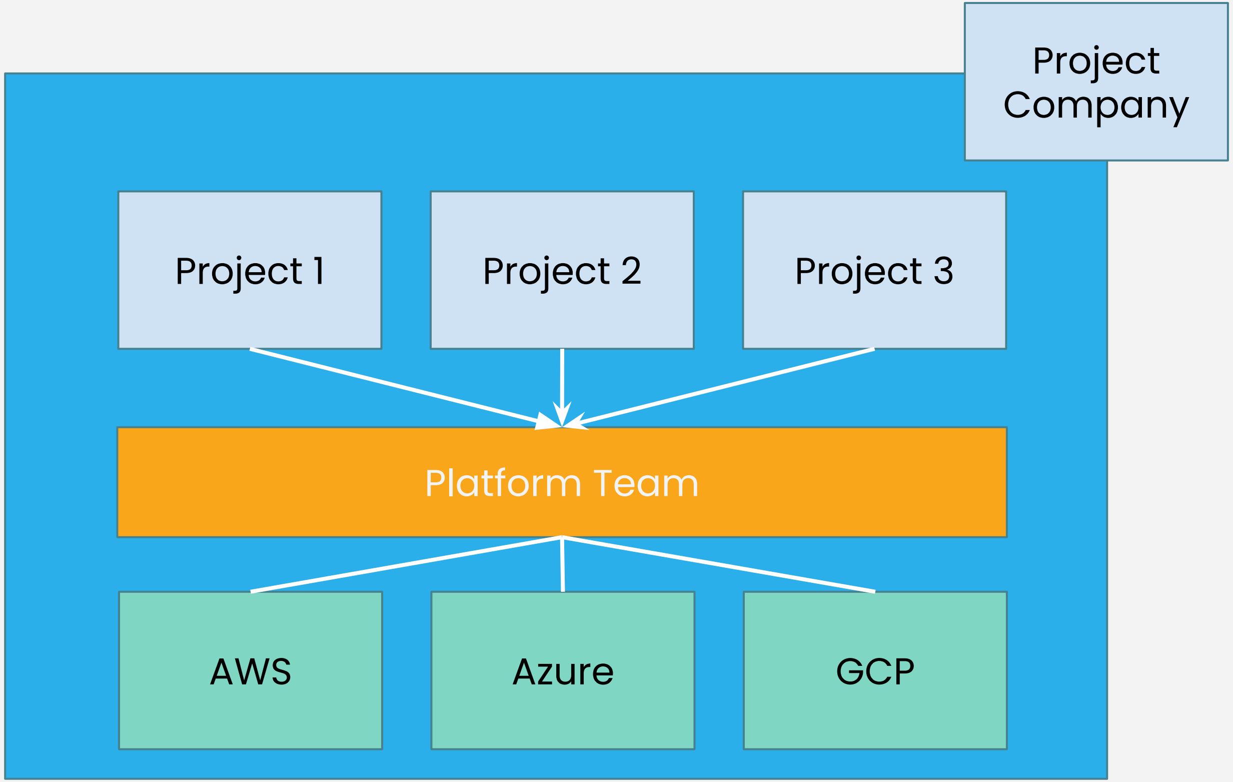
Why Platform Engineering?



What's happening here?

- Complexity for Developers
- Everyone is doing the same thing.
- There is no organization wide consistency.
- Each team working in Silo and there can be no standards around security, best practices etc.

Why Platform Engineering?



Platform Engineering Team

- Platform Engineers provide **internal developer platform. (IDPs).**

- The IDP is kind of a self service platform where teams log into and provision a cluster for e.g

- IDP is a product and application team is the client of the IDP.

More Reading

- https://medium.com/google-cloud/deploy-the-online-boutique-sample-apps-with-score-and-humanitec-d99101001e69

End

Q&A