Max Körbächer x Liquid Reply

Why Developer Platforms and a dedicated Platform Engineering teams is a must have

Cloud Native Rejekts 17.04.2023

Hi!

Max Körbächer - Founder of Liquid Reply

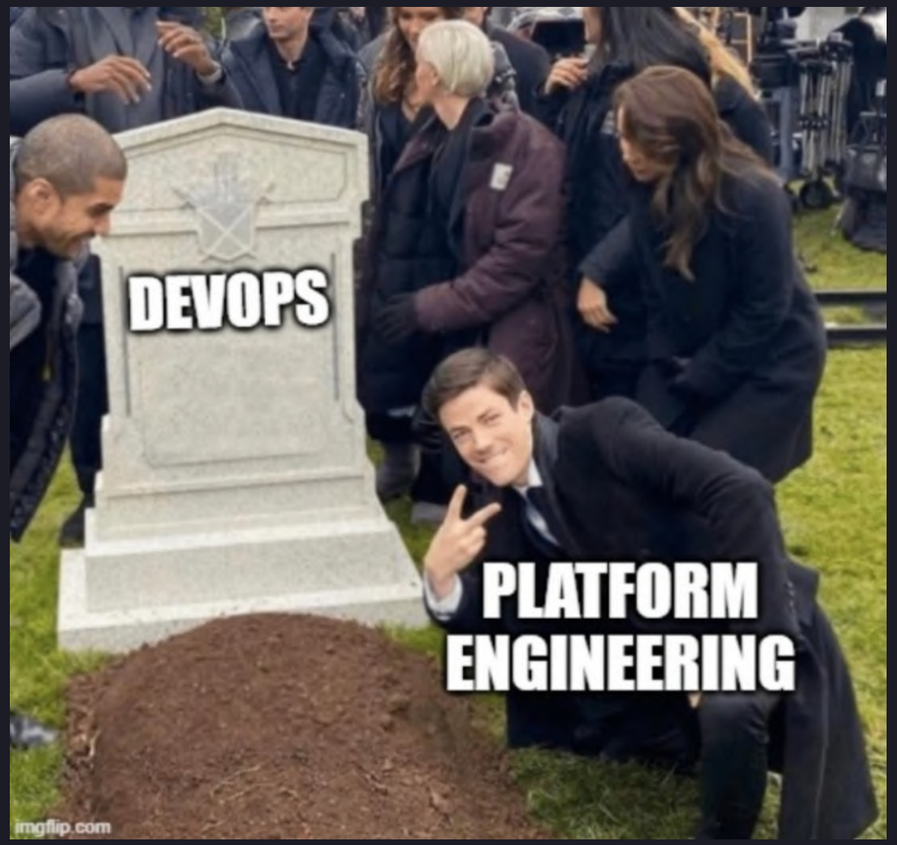
We help customers to adopt cloud native and cloud agnostic principles & implement Kubernetes focused platforms.

CNCF Ambassador, Co-Chair CNCF TAG Environmental Sustainability, LF Europe Advisory Board Member

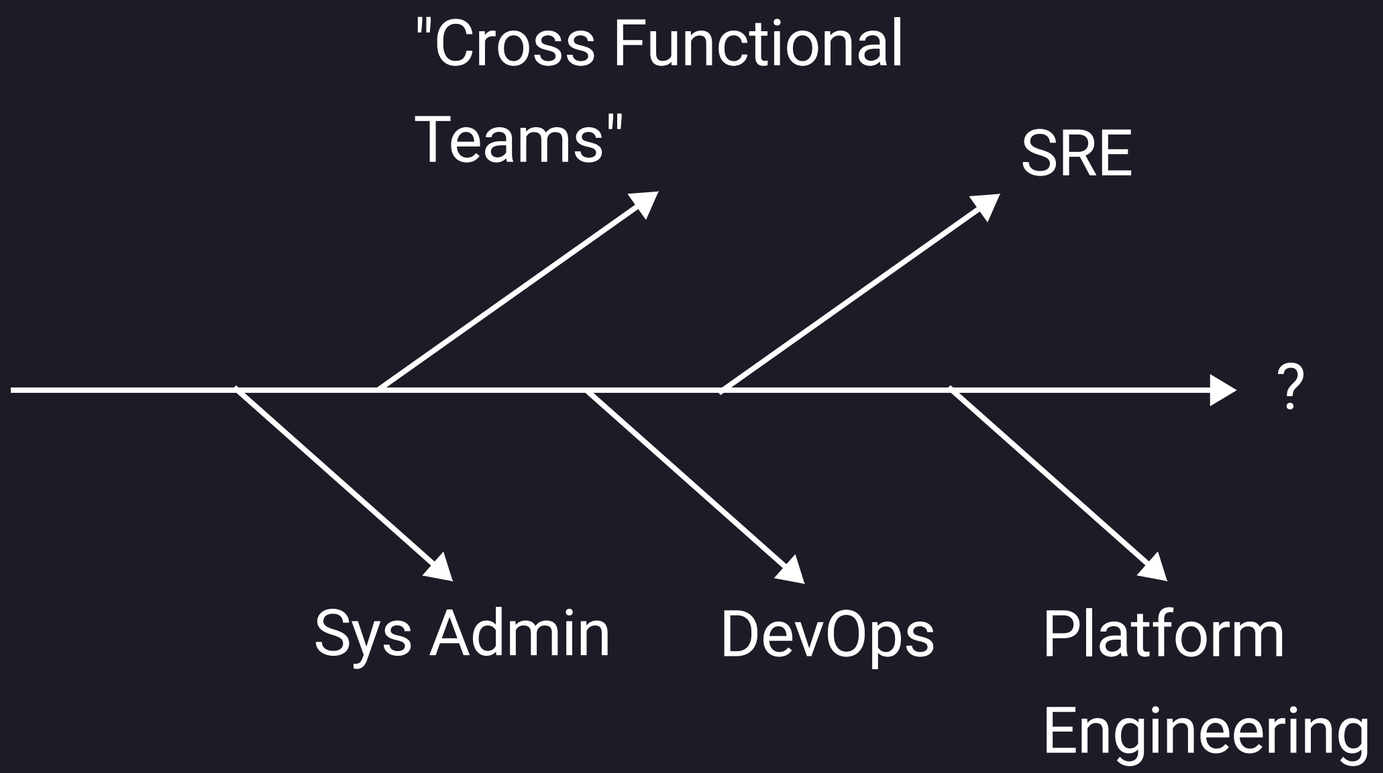
Who of you would see themself as a Platform Engineer?

Who has a Platform Engineer Team/ Internal Developer Platform in the company?

Is DevOps dead yet?



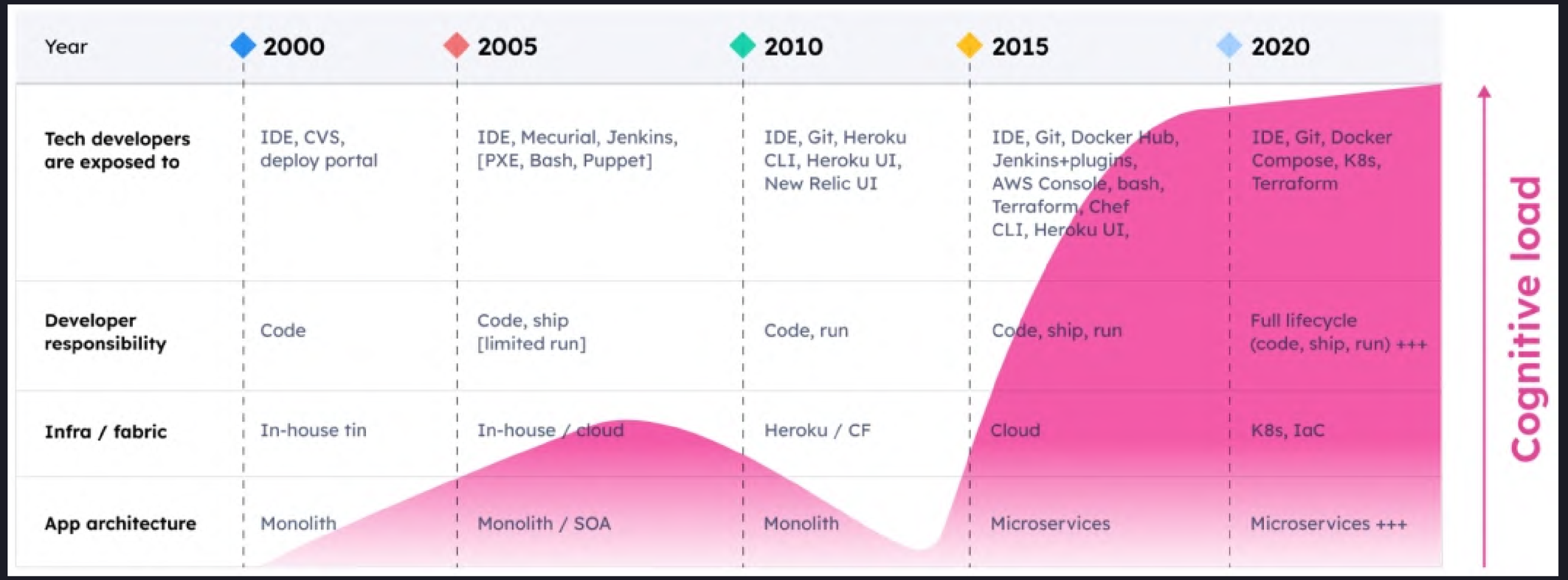
It's evolution!



Modus operanti

- Enterprises working project focused
  - restricted time
  - restricted money
  - expecting only one outcome
- Established teams have often support functions
- Reality writes DevOps like dev**OPS**

Everything got to much



Source: Whitepaper: State of Platform Engineering Report Vol. 1 - graphic inspired by Daniel Bryant

What is Platform Engineering?

"… Some organizations have tackled this challenge by creating **platform engineering product teams**. These teams operate an **internal platform** which enables delivery teams to **self-service deploy** and **operate systems** with **reduced lead time** and stack complexity. The emphasis here is on **API-driven self-service** and supporting tools, with **delivery teams still responsible for supporting what they deploy onto the platform…**"

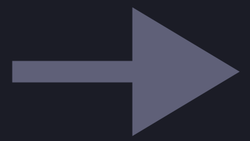
— 2017 Thoughworks Tech Radar

Key aspects of Platform Engineering

1. Design and build toolchains
2. Establish workflows
3. Enable self-service capabilities

Key aspects of Platform Engineering

1. **Design and build toolchains**
2. **Establish workflows**
3. **Enable self-service capabilities**



Provide an integrated product that covers all operational needs for the entire lifecycle of an application.

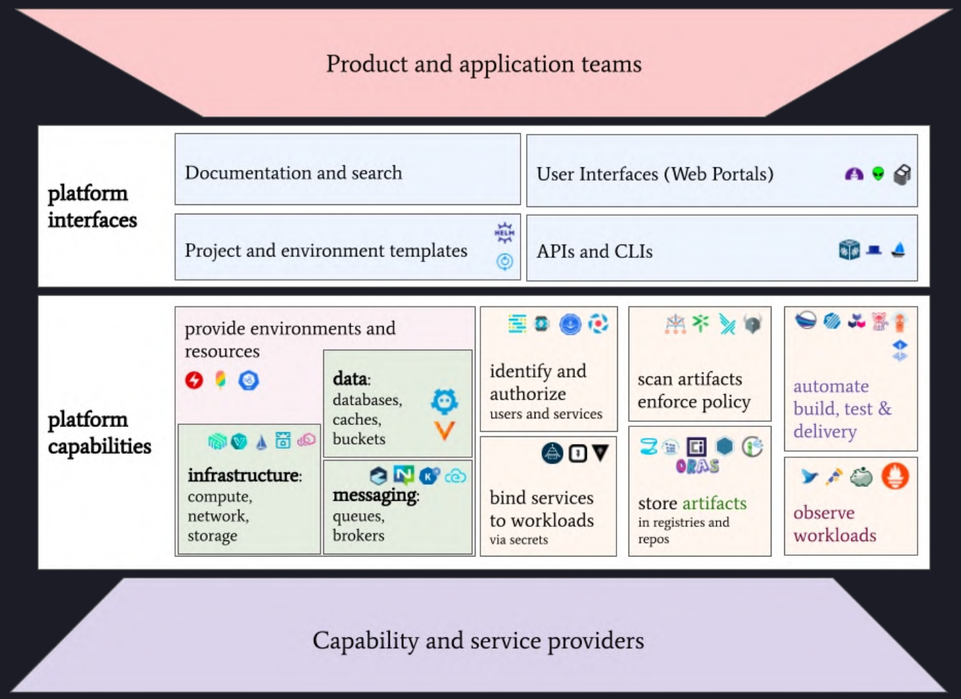
Following Cloud Native Principles

Design for automation

Build for scalability & resiliency

Handle state with smartness & care

Everything is self-containt



Source: https://tag-app-delivery.cncf.io/whitepapers/platforms/

What is a platform?

A digital platform is a foundation of self-service APIs, tools, services, knowledge and support which are arranged as a compelling internal product.

1. Treat the Platform as a Product
2. User experience is relevant
3. Documentation and onboarding must be lean
4. Self-Service capabilities are central focus
5. Reduce cognitive load for users
6. Optional and composable
7. Secure by default

DevOps

DevOps focus often on a certain team, product or domain and their challenges, in its extend doing 24/7 operations and rarely any development.

SRE

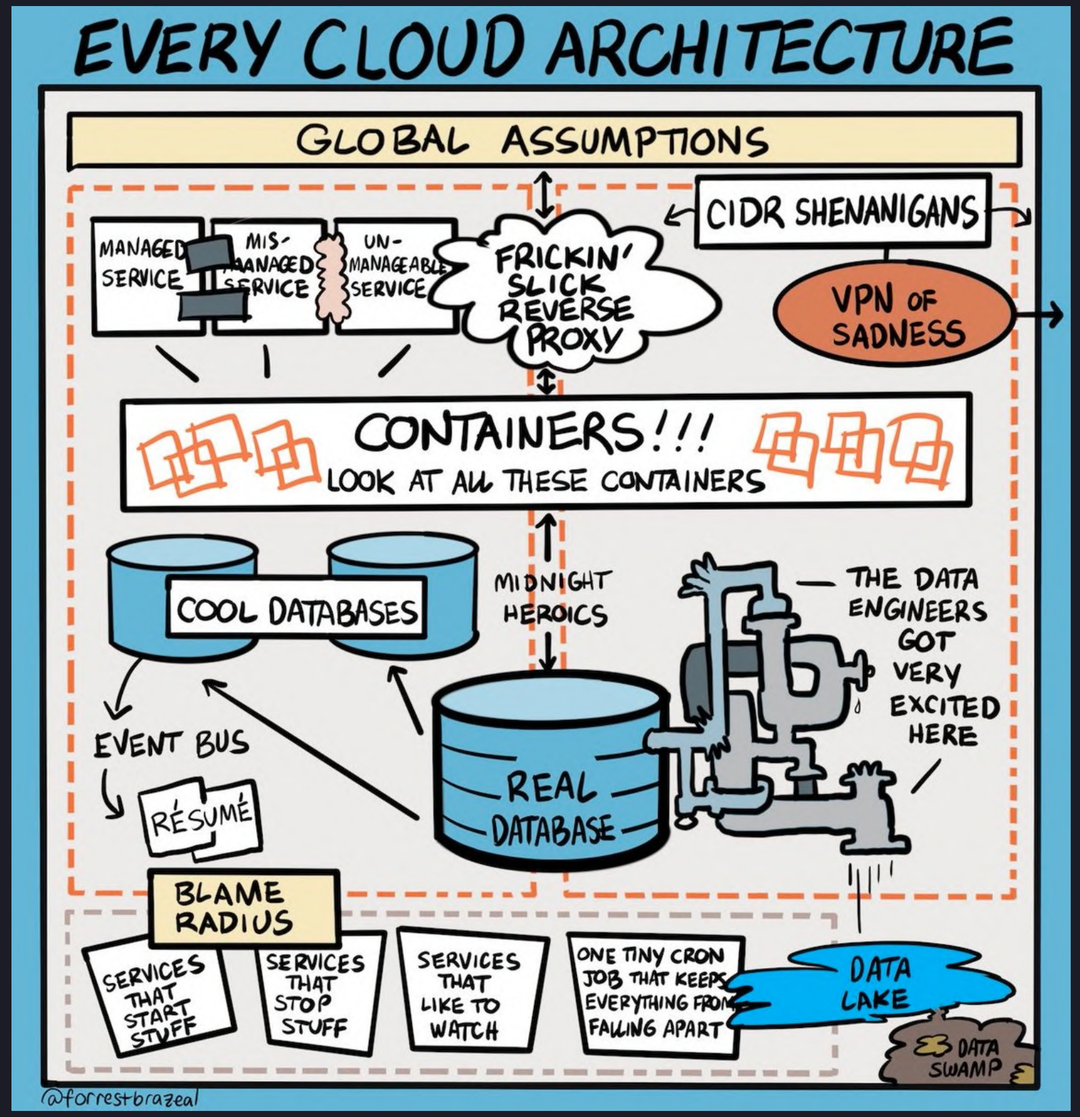
Focusing on the app product reliability, have a close interaction with developers, can fix issues and maintain an app product in a healthy state.

Platform Engineers

They understand the whole required infrastructure, processes and integrations needed to build, run and manage an application as a product. The dev teams are their customers.

In short:

PEs try to offload Developers from all the grown complexity of the past years



The benefits of Platform Teams

Improved Application Lifecycle

x1,5 - x18

Reduced dev cycle error rate /week/developer

~2h

Cost optimization/reduction

20%-30%

Grown reliability

No downtime*

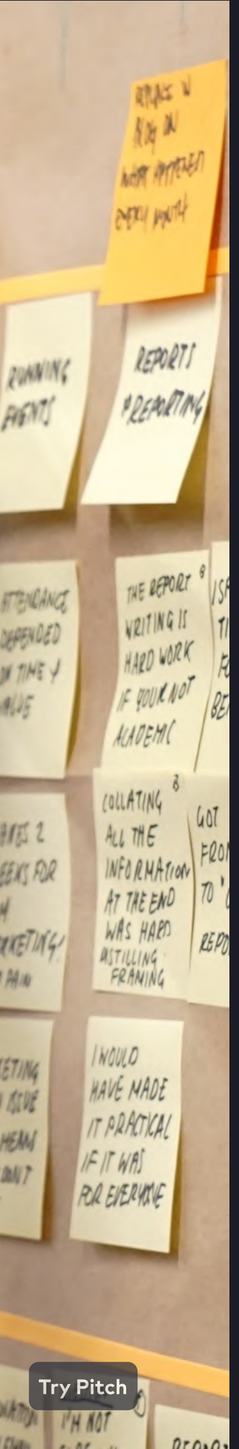
*in our control

Benefits

Lessons learned from enabling PE teams

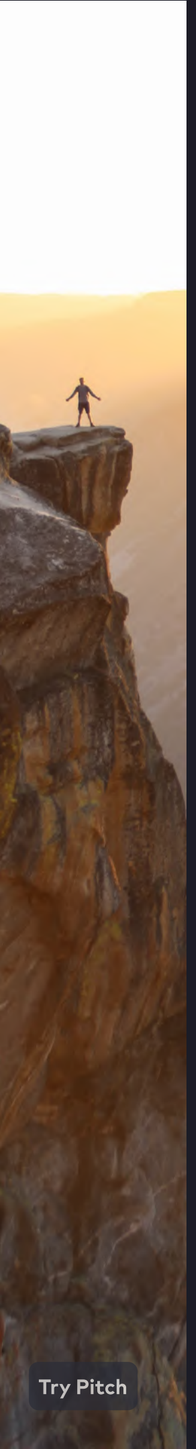
1. Don't build everything by yourself aka don't reinvent the wheel

- Far too often platform teams want to build a solution for their unique case
- A "we can do it better anyhow" mentality is more harmful than productive
- For most projects we migrated custom operator, controller, and shell scripts to given open source projects



2. Platform Teams are not a project

- To have an effective platform engineering they need to be an established team
- We often see budget and resource cuts after some certain functionalities are achieved, this is dump, as the whole IT world is moving 10x faster than the rest of most busnisses
- Find the right way of work - SCRUM or AGILE isn't the answer!



3. You need the commitment, to give the freedom

- Usually highly effective and efficient PE teams are very dedicated to their job
- To achieve this, you have to give the freedom to test things out and trust in them
- From this freedom of finding better solutions, you will get a higher commitment to built an even better platform

4. Things are changing, delete the word of "throw away"

- Forget about the mentality that a once-implemented solution has to stay for ever
- Our cloud native landscape is continuously in the move, reshaping solutions as the community overall faces similar problems and provides common tools to solve them
- "Throwing away" makes space for something new - when did you tidy up your cellar the last time?

— 2017 Thoughworks Tech Radar

"Organizations that consider establishing such a platform team should be very cautious not to accidentally create a separate DevOps team, nor should they simply relabel their existing hosting and operations structure as a platform."

If you haven't a PE team, you should grow one!

Try Pitch

Checklist for your Platform Engineering Team:

- ☐ Provide an environment outside of your organizational bull s**t
- ☐ Have a clear vision & target for the team
- ☐ Give space to grow (budget, time)
- ☐ Give up your project mentallity

Thank you

LinkedIn: maxkoerbaecher

Twitter: mkoerbi

GitHub: mkorbi