



Laurent Gil - ATOS  
Olivier Jacques - AWS

# Platform Engineering

Lorsque Kubernetes devient la clé du royaume

duck @ojacques2, duck @anegar



**SNOWCAMP**



## Laurent GIL

DevOps Coach,  
AWS Solution Architect,  
Kubernetes evangelist



# Top Strategic Technology Trends **2023**



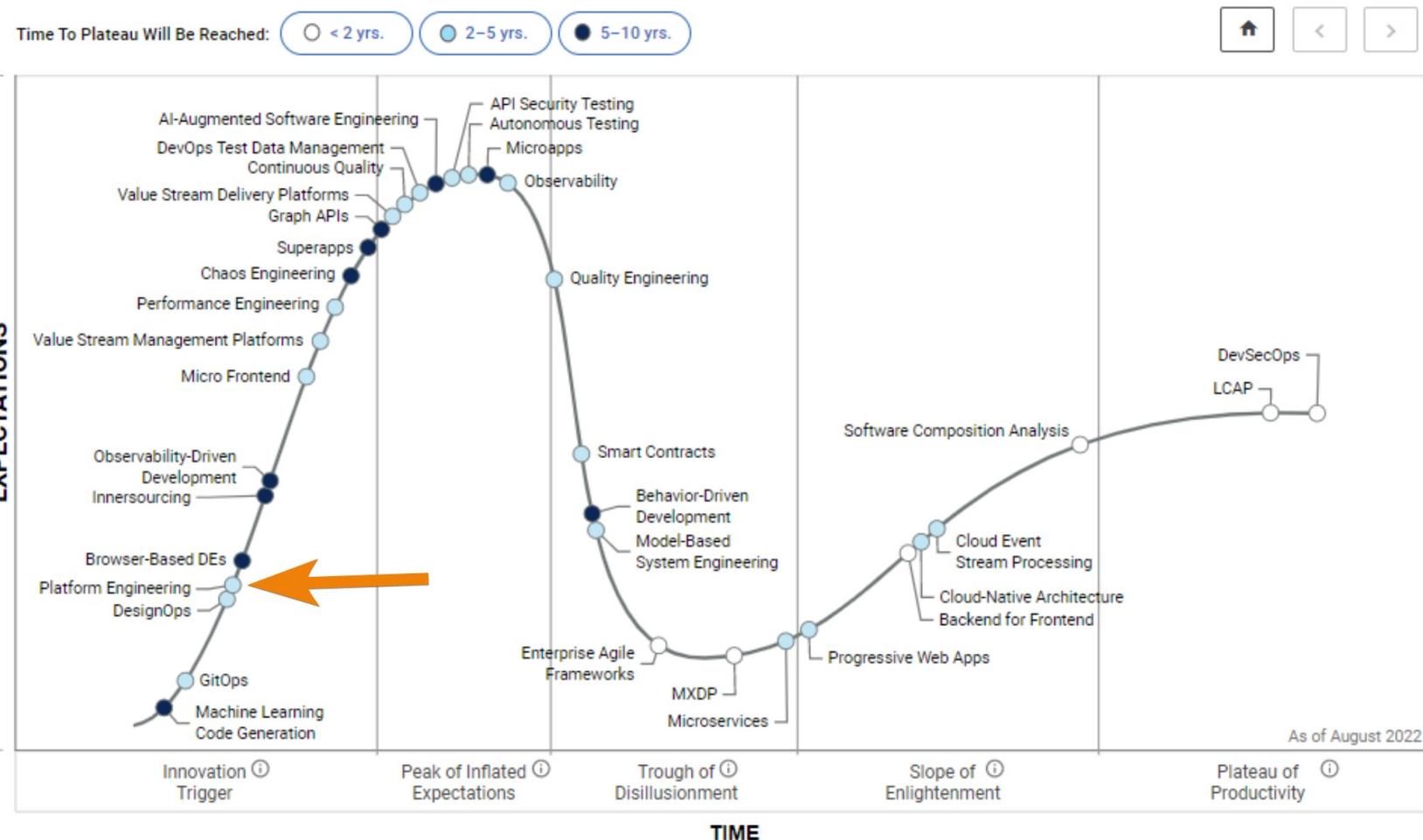
- |           |                          |          |                       |          |                            |
|-----------|--------------------------|----------|-----------------------|----------|----------------------------|
| <b>1</b>  | Digital Immune System    | <b>2</b> | Applied Observability | <b>3</b> | AI TRiSM                   |
| <b>4</b>  | Industry Cloud Platforms | <b>5</b> | Platform Engineering  | <b>6</b> | Wireless-Value Realization |
| <b>7</b>  | Superapps                | <b>8</b> | Adaptive AI           | <b>9</b> | Metaverse                  |
| <hr/>     |                          |          |                       |          |                            |
| <b>10</b> | Sustainable Technology   |          |                       |          |                            |

[gartner.com](https://gartner.com)

Source: Gartner  
© 2022 Gartner, Inc. All rights reserved.

Gartner®

# Cycle de l'effervescence Gartner



# DevOps est 💀 ?



protect kylie at all costs  
@mattstratton

...

👀 #kubeconNA



3:49 PM · Oct 26, 2022

15 Retweets 39 Quote Tweets 222 Likes



# Conclusion

(oui, autant partir sur la conclusion)

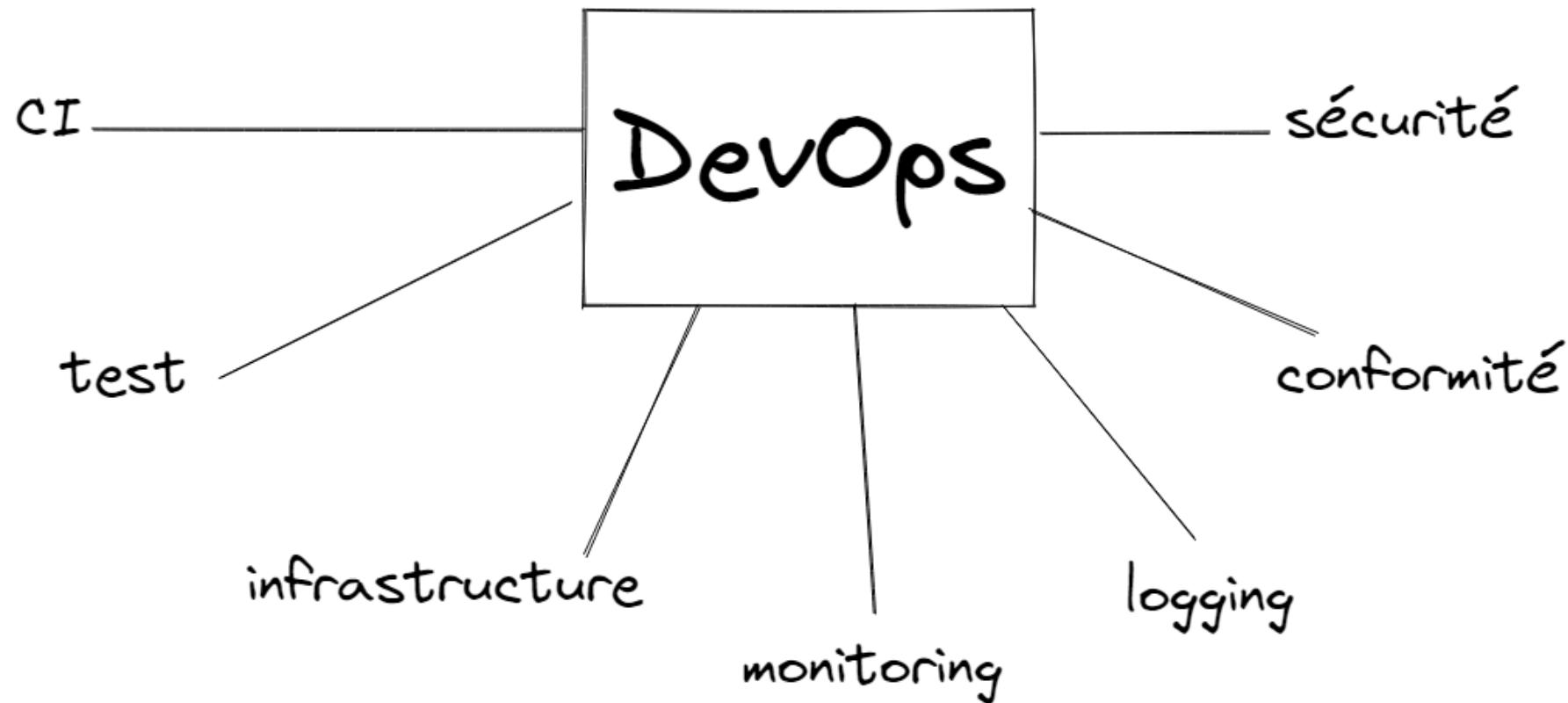
- Non, DevOps n'est pas mort
- Le Platform Engineering permet d'accélérer l'innovation
- Kubernetes comme plateforme de base

# AWS re:Invent

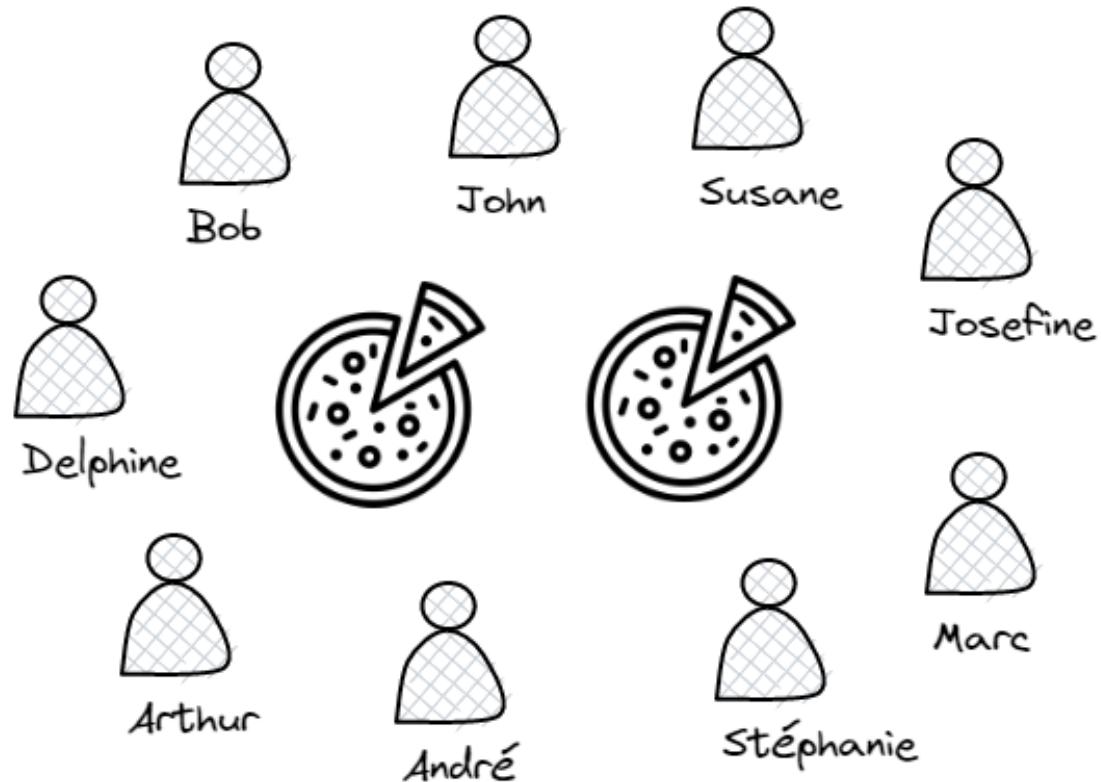
"You build it, you run  
it"



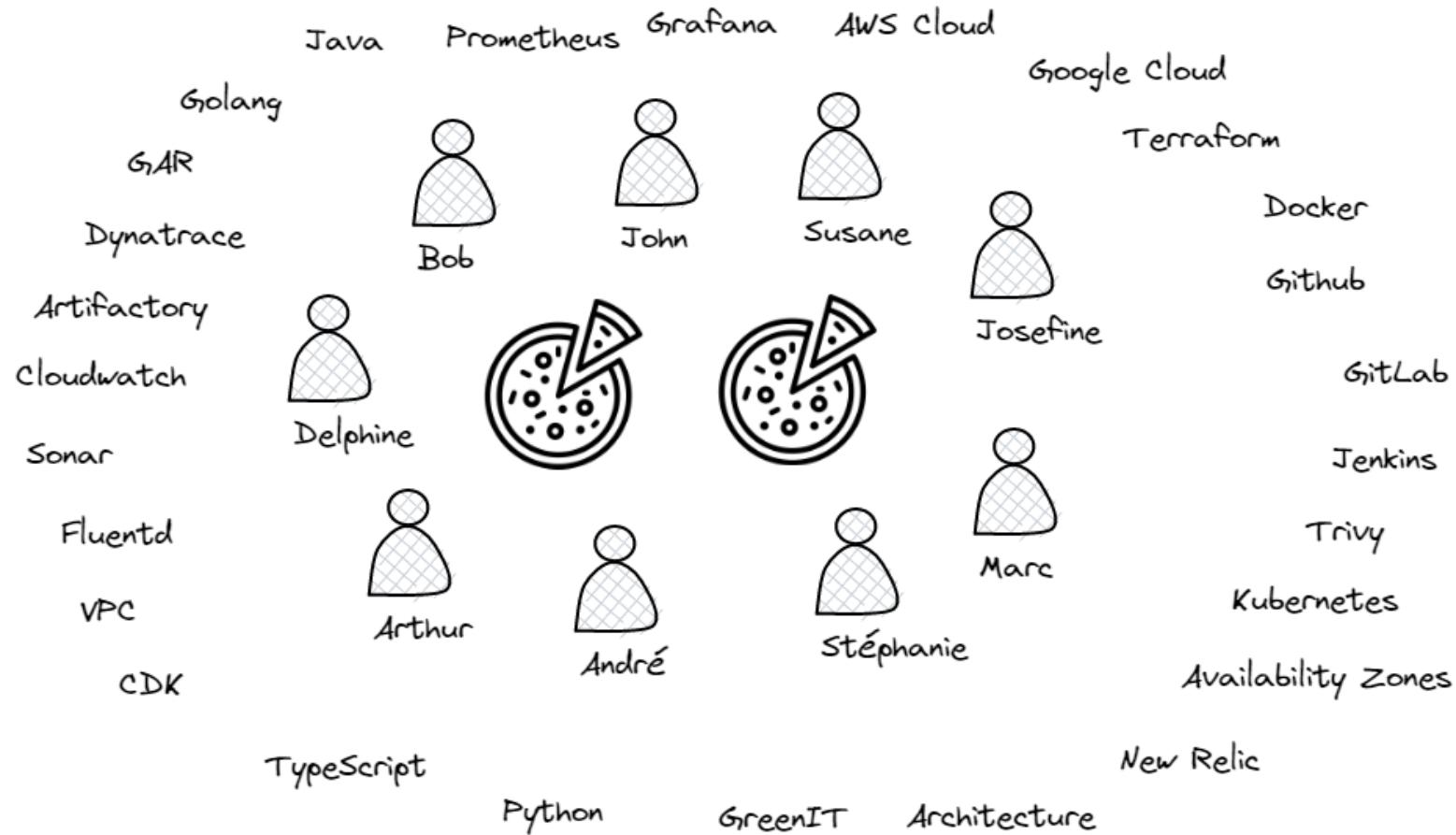
# "You build it, you run it" - cela veut dire quoi ?



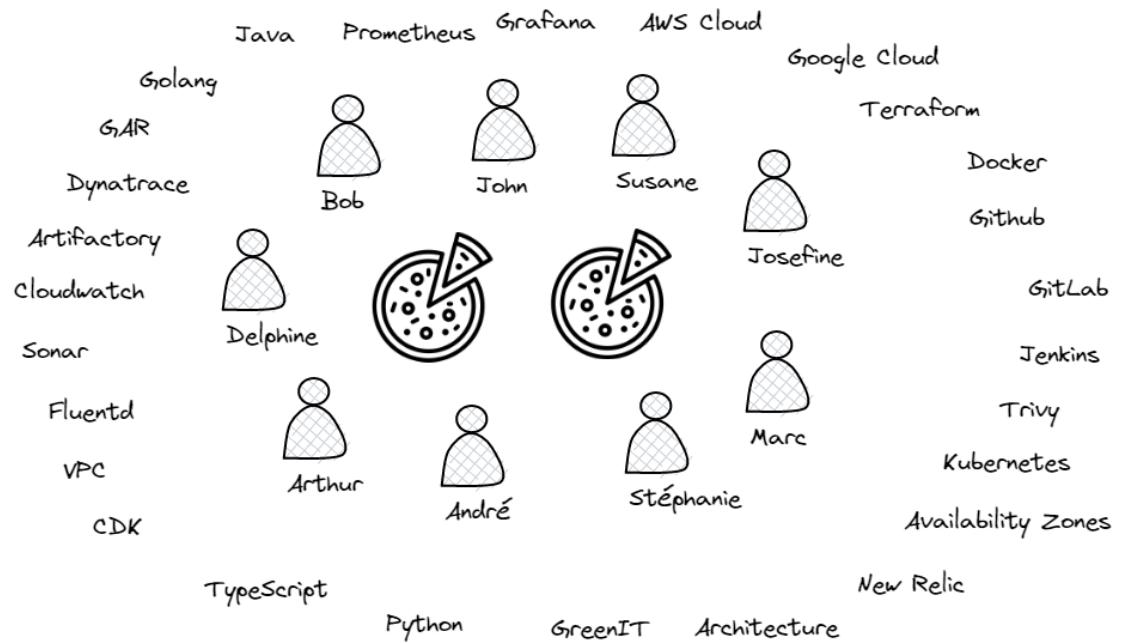
# 2 pizzas pour une équipe



# 2 pizzas pour une équipe

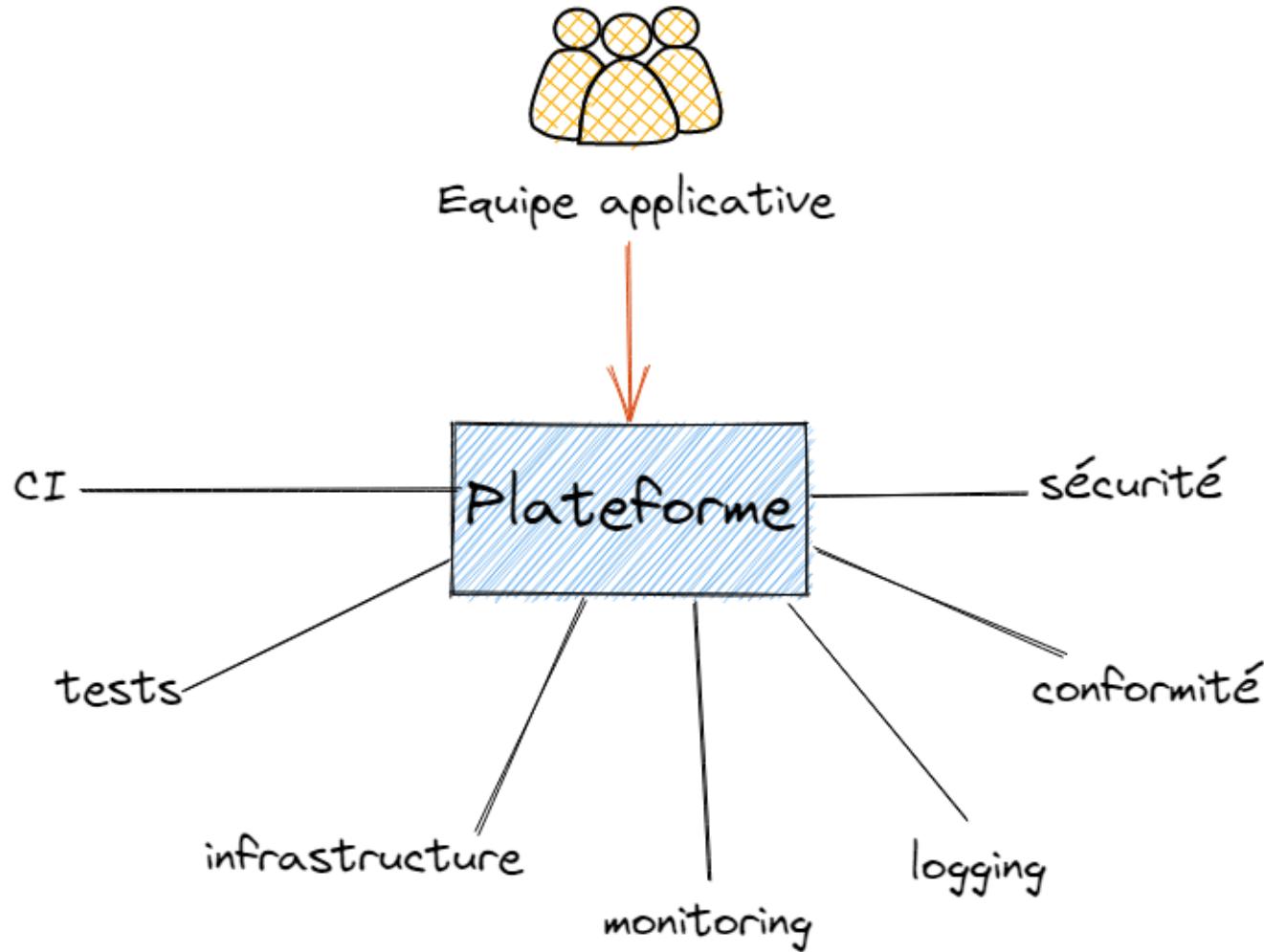


# 2 pizzas pour une équipe

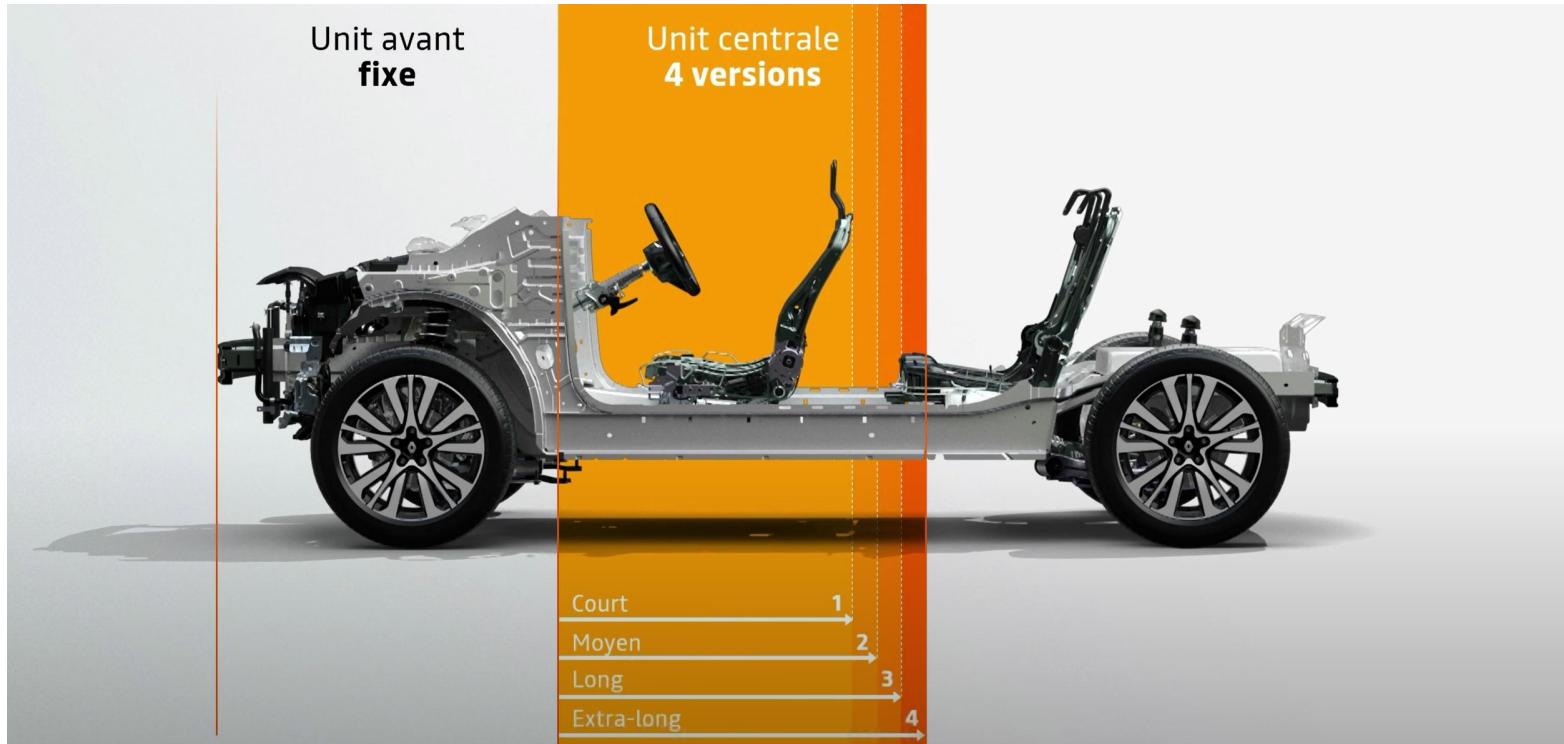


Pas assez de ressources humaines  
Charge cognitive élevée

# Solution ?



# Les plateformes



## Définition - Plateforme

"Les plateformes sont un moyen de centraliser l'expertise, tout en décentralisant l'innovation au client ou l'utilisateur"

Peter Gillard-Moss, ThoughtWorks

# Définition - Platform Engineering

"L'ingénierie des plateformes est la discipline qui consiste à concevoir et à créer des chaînes d'outils et des flux de travail qui permettent aux organisations d'ingénierie logicielle de disposer de capacités en libre-service à l'ère du "cloud-native". Les ingénieurs de plateforme fournissent un produit intégré, souvent appelé "plateforme interne de développement", qui couvre les besoins opérationnels de l'ensemble du cycle de vie d'une application.

Luca Galante

# Pourquoi construire une plateforme ?

- Réduire la charge cognitive
- Augmenter la productivité
- Forcer la standardisation

Grandir les équipes, tout en préservant ce qui permet d'être productif: l'autonomie, avec un minimum de coordination et de l'infrastructure en self-service.

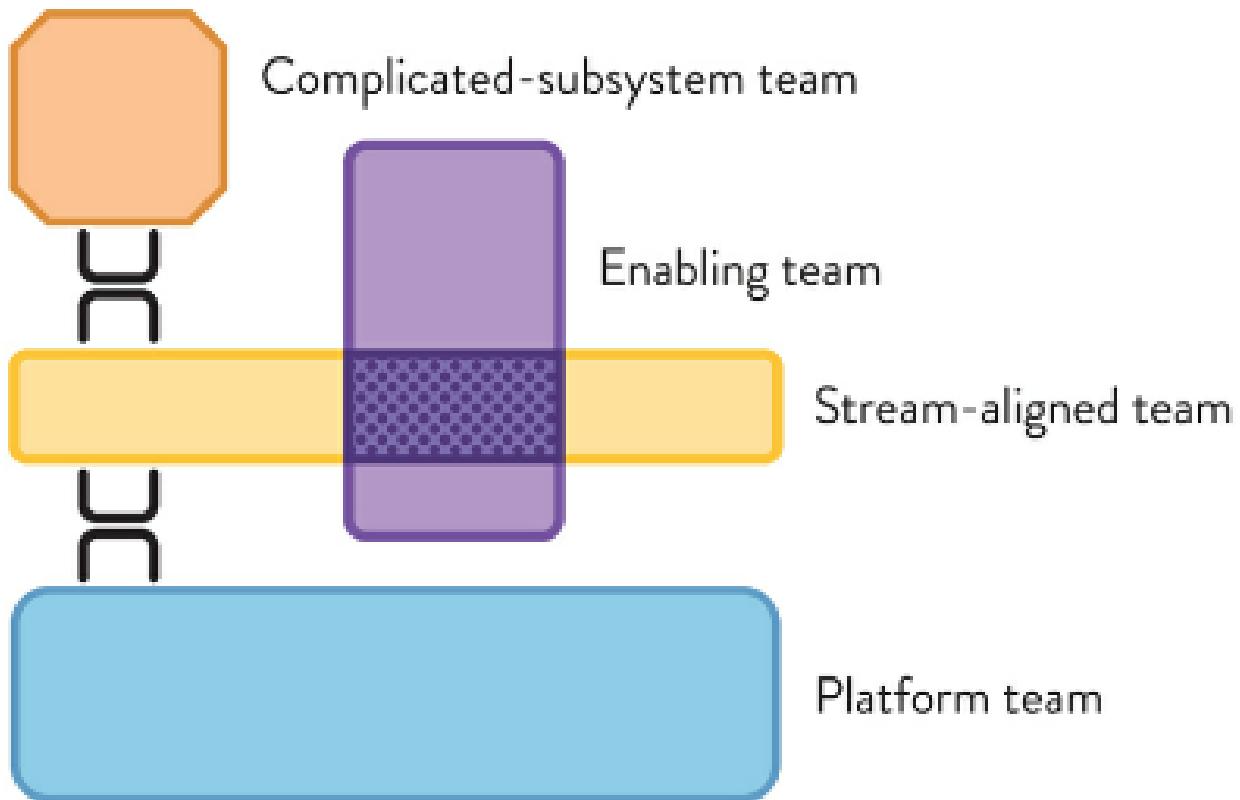
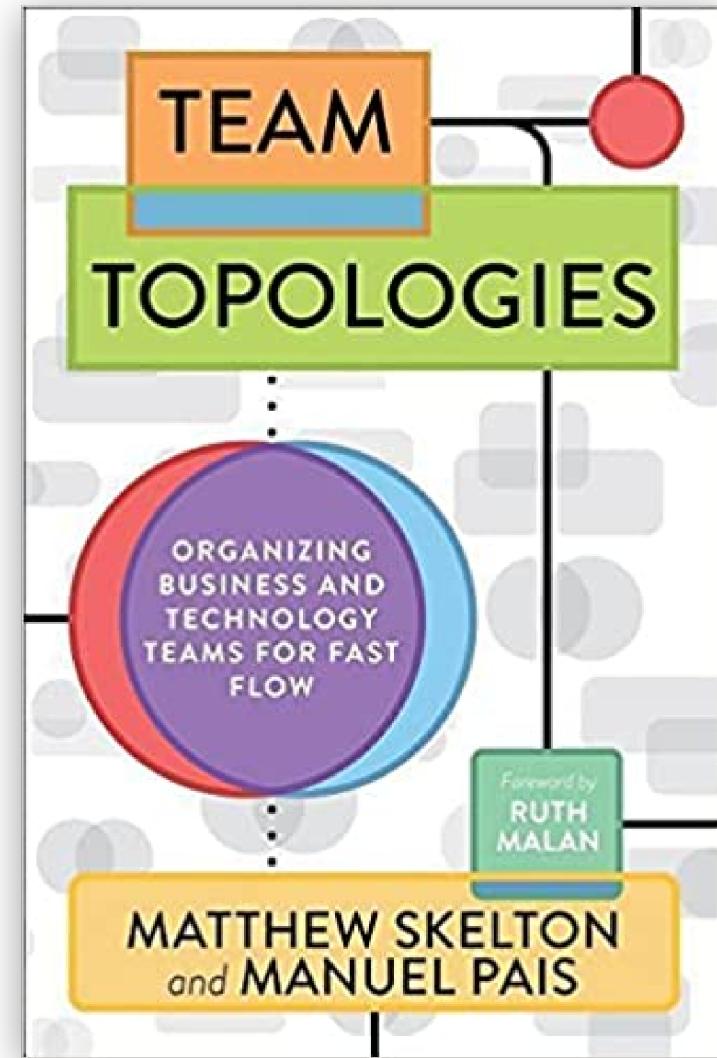


Figure 5.1: The Four Fundamental Team Topologies



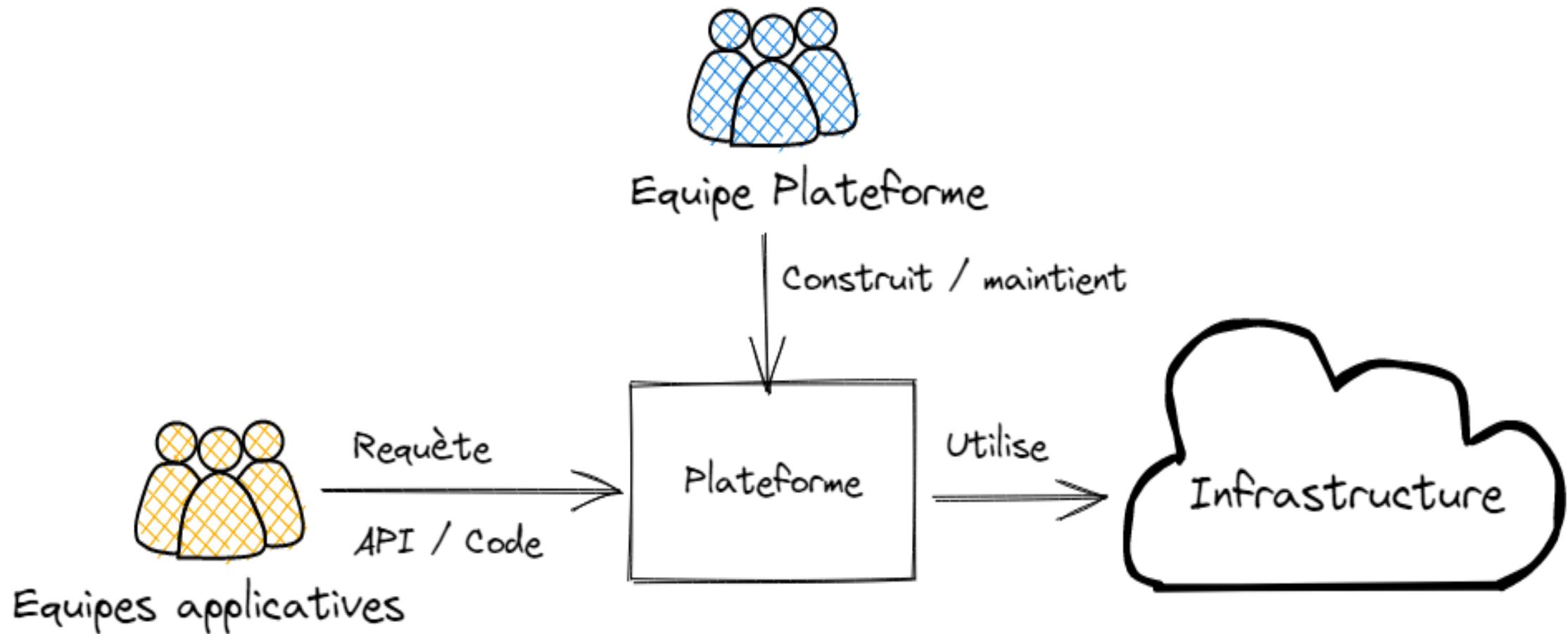
# Réussir une plateforme



# Caractéristiques d'une bonne plateforme

- Facile à adopter
- Transparente dans sa gouvernance, son fonctionnement
- Responsabilité partagée (Inner Source)
- Flexible et extensible

# Une plateforme



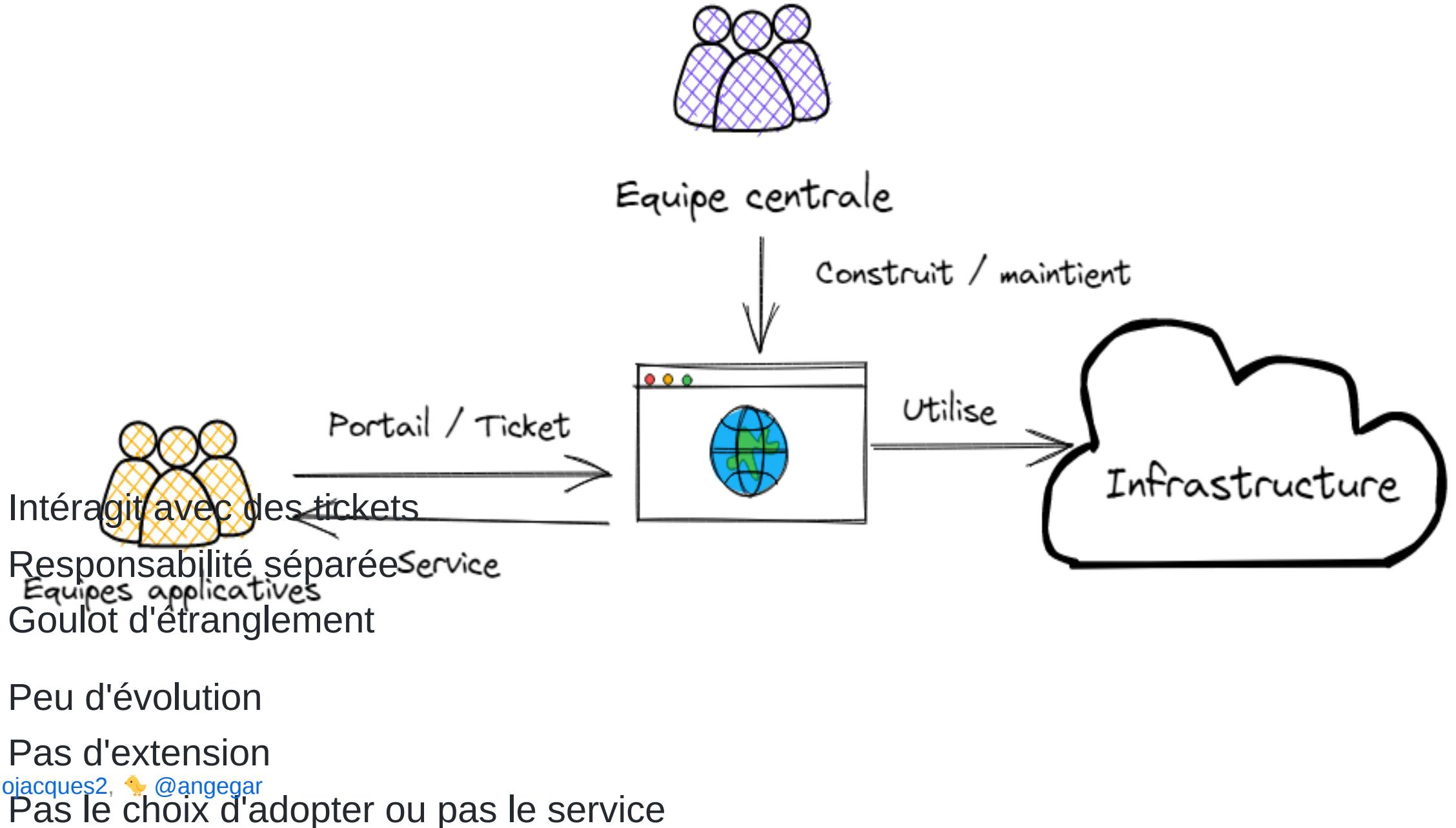
⚠ Une plateforme n'est pas un service

Fournir une base de donnée "clé en main", n'est pas fournir une plateforme. C'est

ducky @ojacques2, ducky @angegar

fournir un service.

# Un service





# Construire une plateforme avec Kubernetes

duckie @ojacques2, duckie @anegar



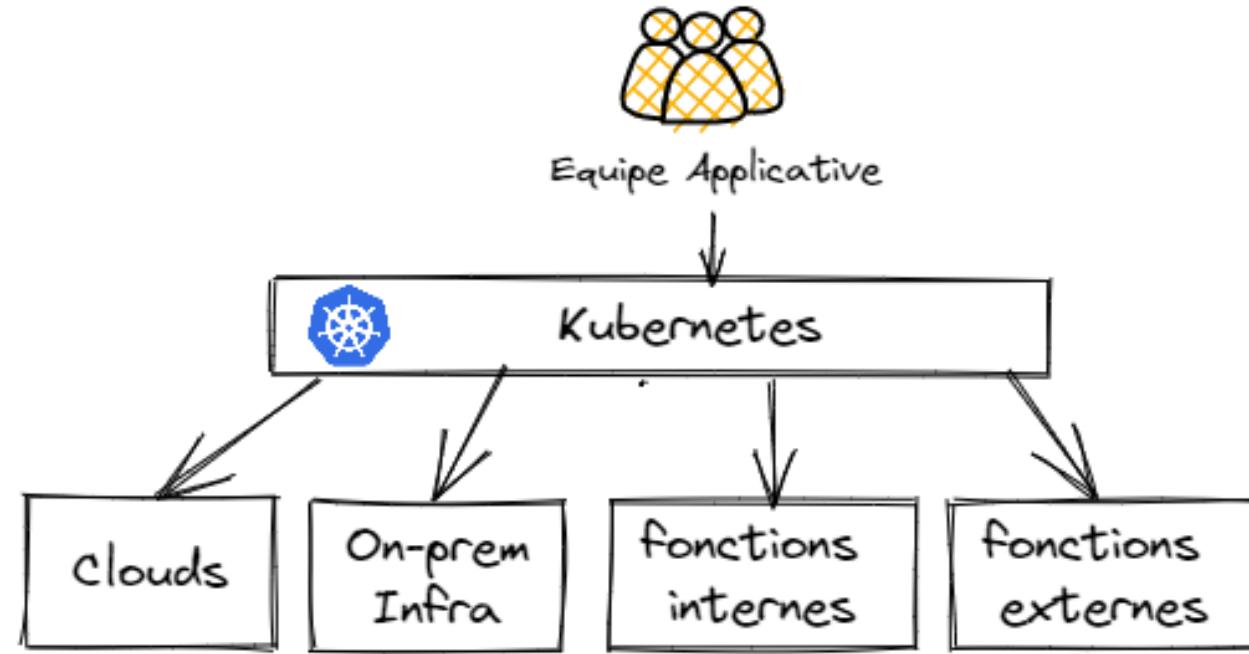
# A propos de Kubernetes

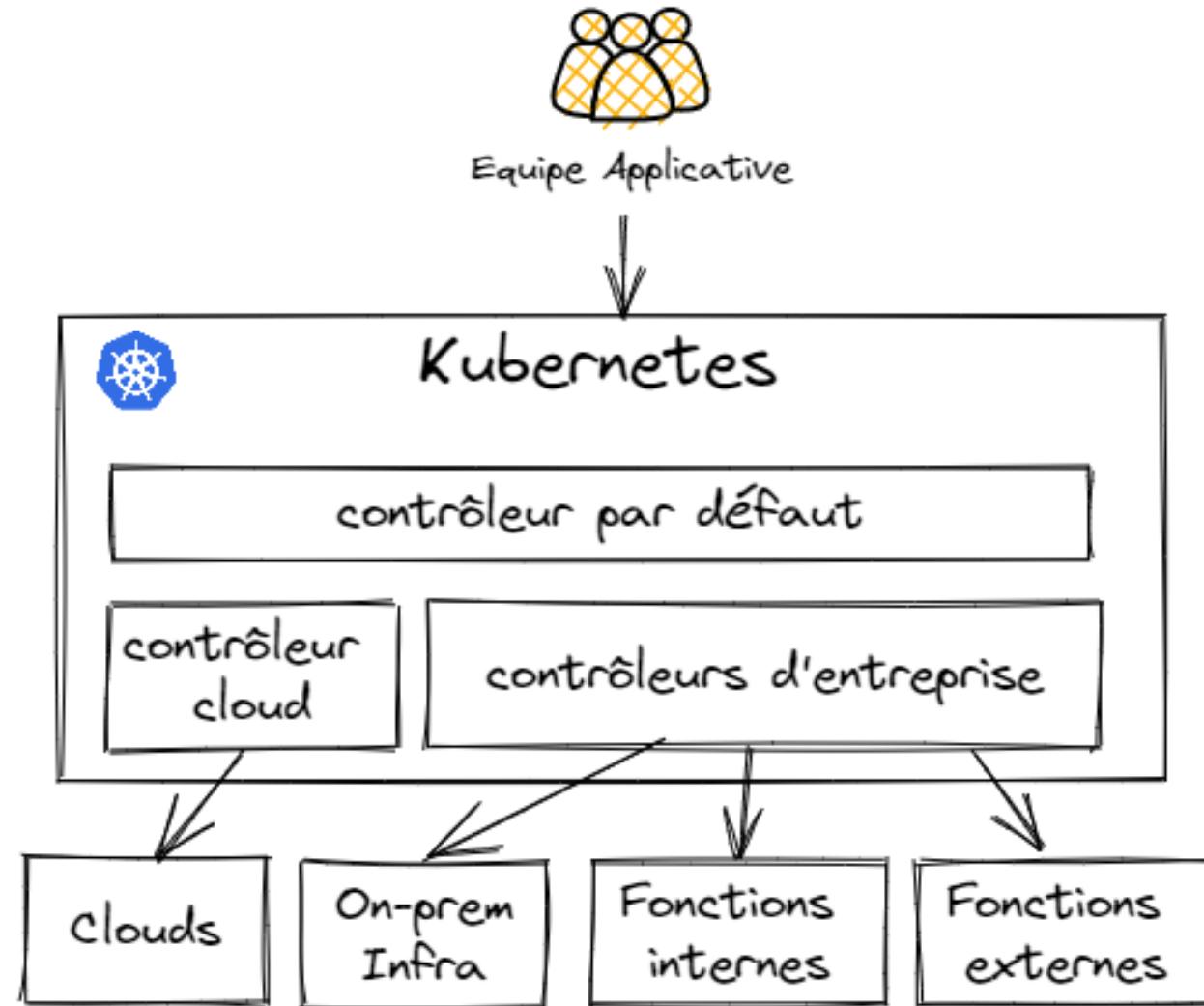
- 📘 Kubernetes est une plateforme open source extensible et portable pour la gestion de charges de travail (workloads) et de services conteneurisés 📙
- 📘 Kubernetes a également été conçu pour servir de plateforme et favoriser la construction d'un écosystème de composants et d'outils facilitant le déploiement, la mise à l'échelle et la gestion des applications. 📙

[source](#)

# Kubernetes comme plateforme framework

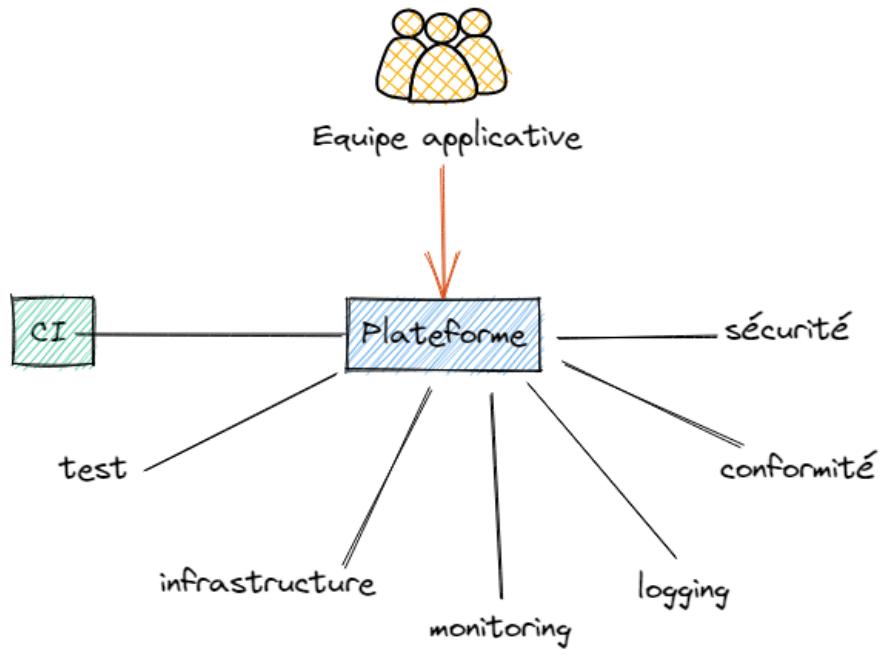
- Possède des qualités intrinsèques tel que
  - self service avec ses APIs
  - self-healing
  - robustness
- Simple d'utilisation grace à une approche déclarative
- Extensible par nature avec les contrôleurs et les définitions de ressource personnalisé





## Qu'est ce qu'un contrôleur Kubernetes

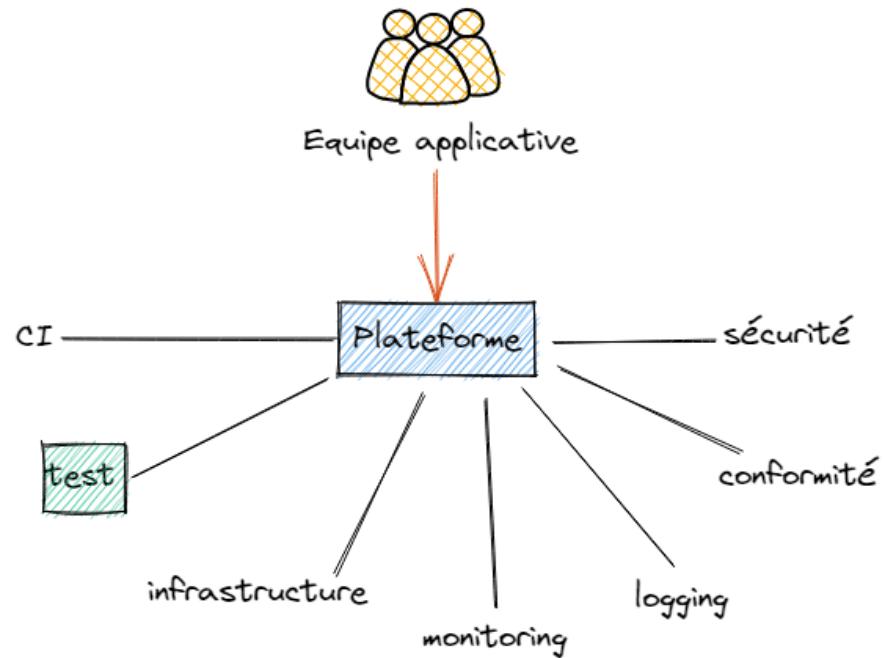
Un contrôleur traque un type de ressource définissant un état souhaité afin de faire converger la plateform vers cet état.



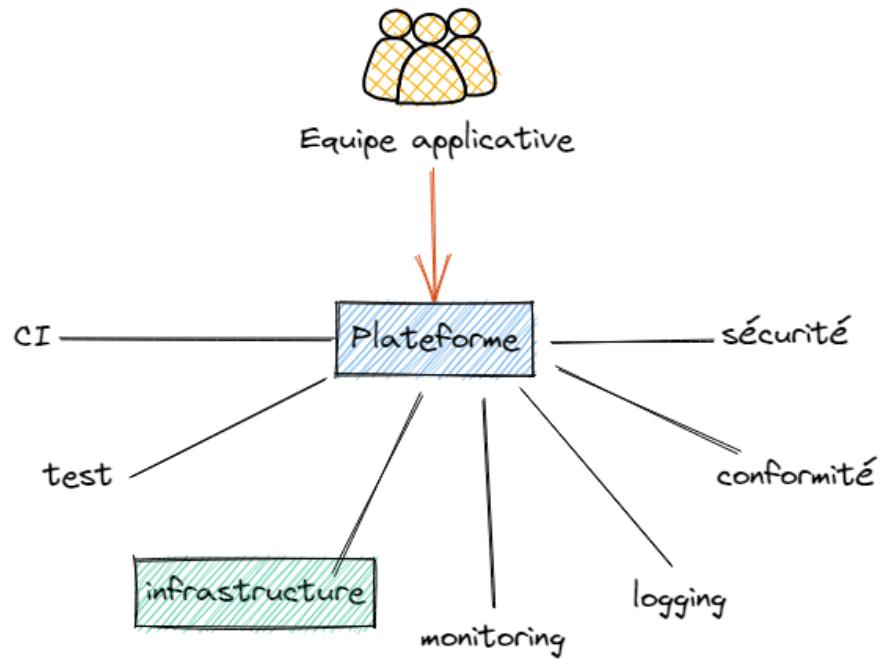
## Pour la CI / CD

- CI
  - JenkinsX
  - Tekton
- CD
  - ArgoCD
  - Flux

# Comme plateforme de test

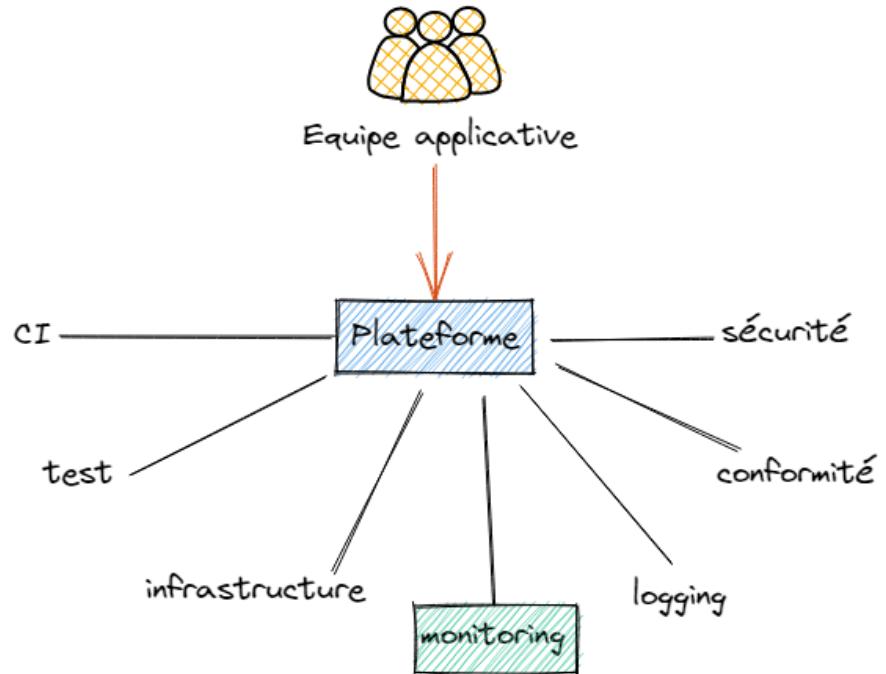


- kubernetes cluster virtuel (nodes et network partagés entre cluster physique et virtuel)
- Créer et détruire des environnements de teste à la volée



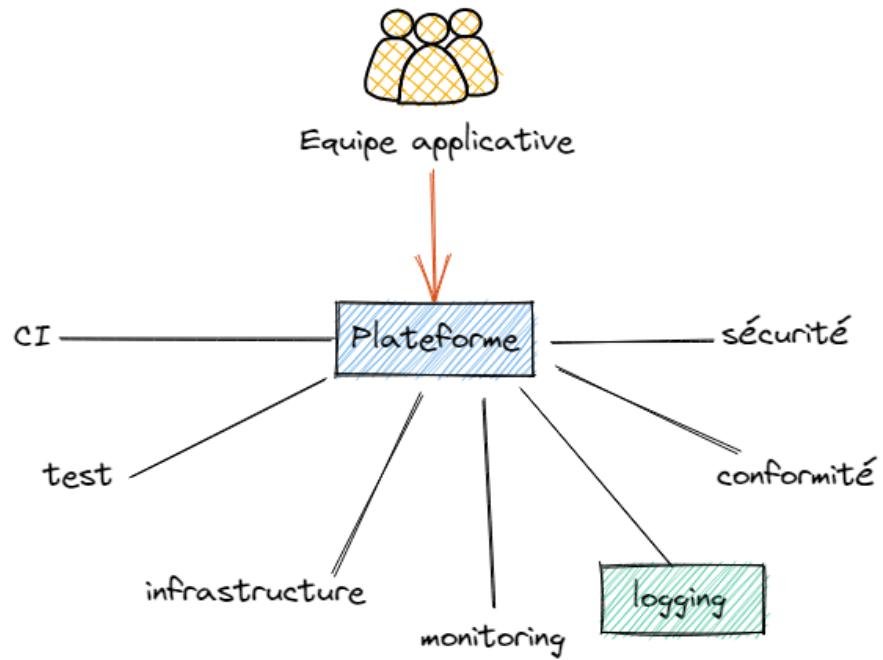
# Pour gérer l'infrastructure

- Crossplane
- AWS ACK Controller
- GCP Config Connector



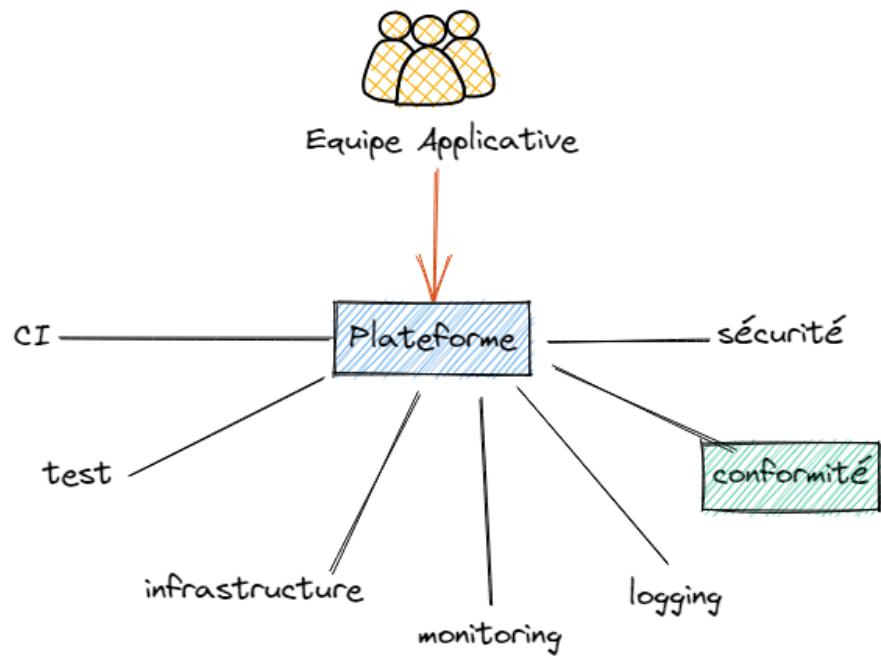
# Monitoring des applications

- [Grafana](#)
- [Dynatrace](#)
- [Datadog](#)



# Collecter les log de manière centrale

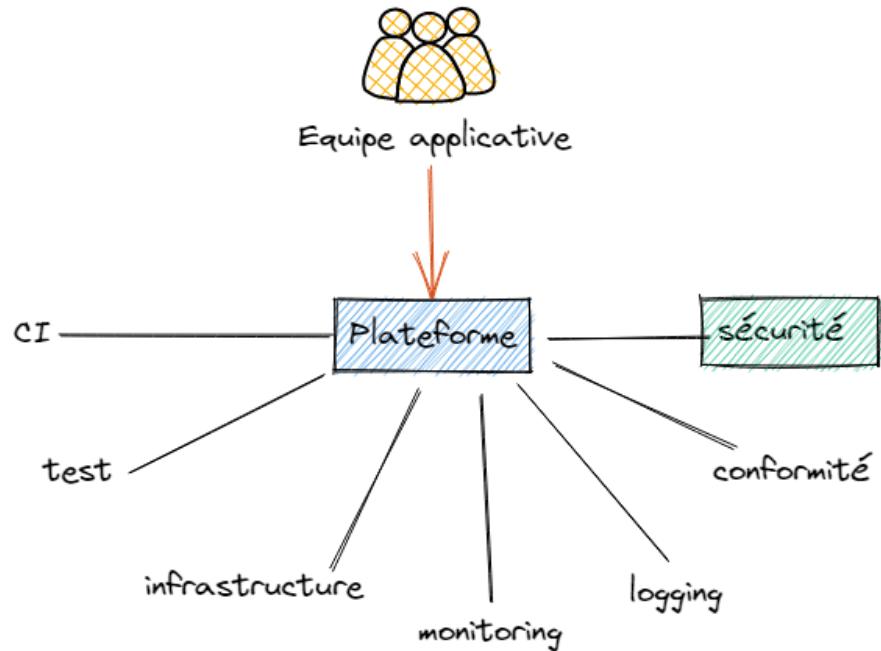
- Fluentd
- Loggie



# Gestion centralisée de la conformité

Instrumenter la stack Kubernetes pour forcer la conformité :

- [OPA Gatekeeper \(policy library\)](#)
- [Kyverno \(policy library\)](#)



# Gestion centralisée de la sécurité

- [KubeArmor](#) : at the system level
- [Trivy-Operator](#)



# Bénéfices d'une plateforme

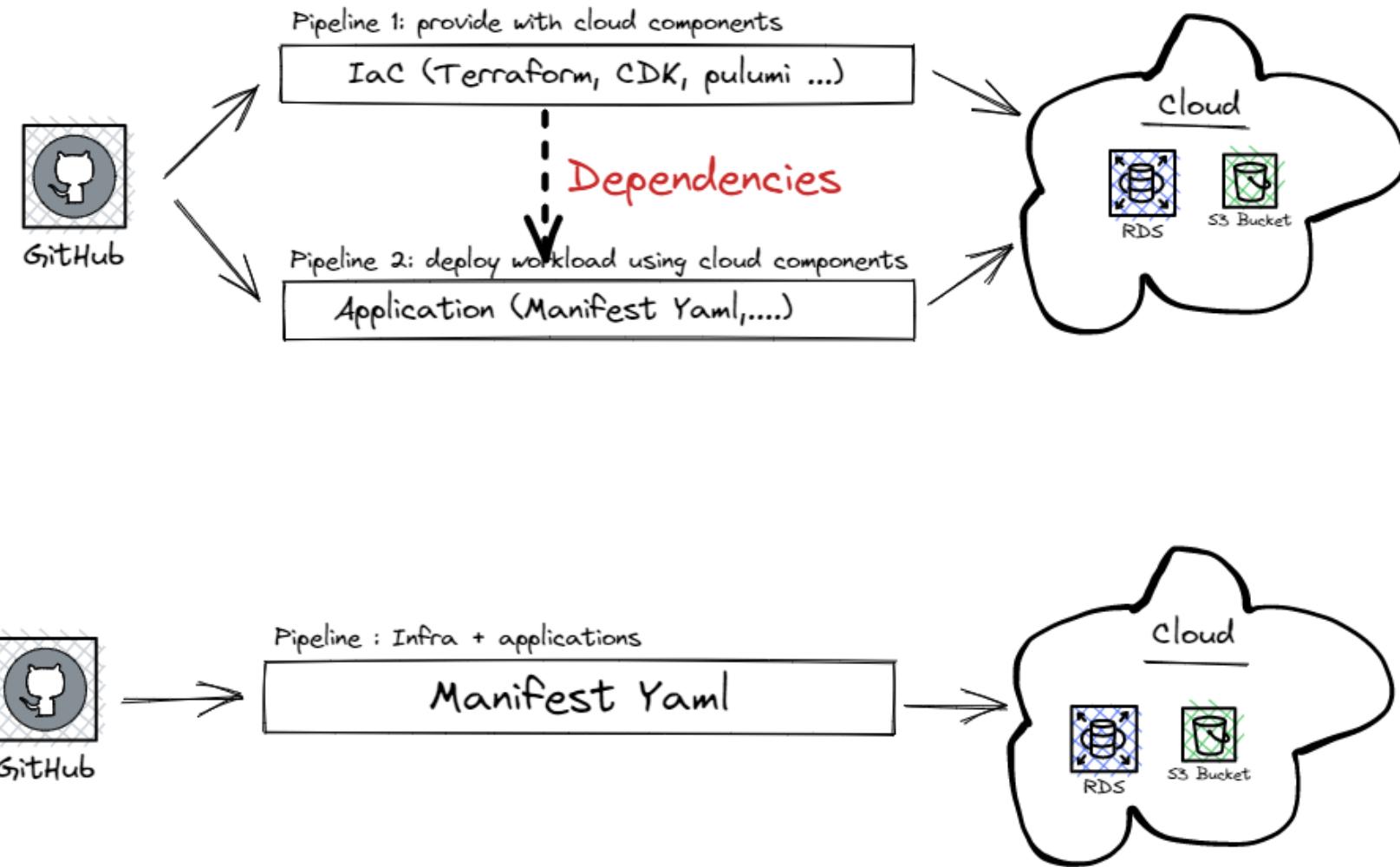
## Facilité d'intégration avec des outils internes

- Créer des définitions de ressource personnalisées
- Créer des contrôleurs personnalisées permettant de piloter des outils internes

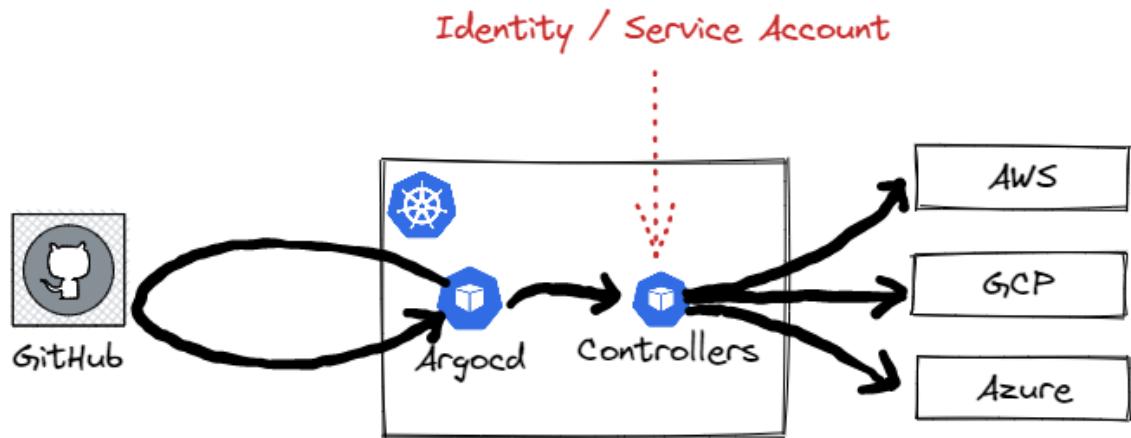
# Opérabilité

- Une seul language pour gérer une multitude de problèmes (infrastructure, application, monitoring ...)
- Une CLI commune à toutes les applications pour la recherche de problèmes
- Des fonctionnalités standards utilisés par toutes les équipes
- Portables à travers les clouds providers

# Améliorer le temps de reprise après sinistre



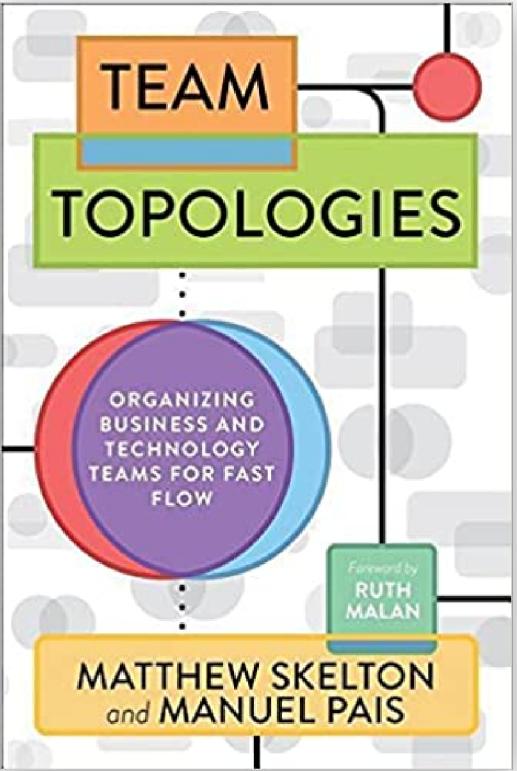
# Passer des pipelines push au pull



- Amélioration de la sécurité  
(gestion des permissions)
- Scalabilité des chaînes de déploiement
- Utilisation d'**outils GitOps** (Flux, Rancher Fleet, ArgoCD)

Les développeurs devraient être capable de déployer et d'opérer leurs applications et services de bout en bout. "Tu le crées, tu l'opères". C'est le vrai DevOps.

# Littérature





# Merci à nos sponsors

« Etoile »



« Flocon »



👉 @ojacques2, 👉 @angegar



chu @ojacques2, chu @angegar



chu @angegar



**SNOWCAMP**