Comment le **GitOps** me permet de **dormir tranquille** la nuit

Déploiement de **microservices** sur **Kubernetes**
avec des outils **GitOps** : **Helm** & **ArgoCD**

Conférence MiNET x IP Paris
Samy Djemaï
13 avril 2023

$ whoami

- Samy Djemaï
- Télécom SudParis, Promo 2022
  - VAP ASR + Polytechnique Montréal
  - Ex-BDA/BPM/INTv/Promo2Tel
- **SRE** @ Padok

Samy Djemaï @SamyDjemai samyd@padok.fr

WTF is **SRE?**

- Site Reliability Engineer

  Ingénieur Cloud

- Croisement entre ingénieur logiciel et sysadmin

  Pas de développement applicatif **métier**

  Docker, **Kubernetes**, Python, Go, Bash...

- Métier ancré dans le Cloud

  Amazon Web Services, Google Cloud Platform, Microsoft Azure

- **Pourquoi ce métier existe ?**

Étude de cas : site d'e-commerce MyGoodies

Site d'e-commerce de vente de goodies personnalisés

Ouvert au public, assez peu de visiteurs (pour le moment)

Conçu en Python à la hâte par trois étudiants

Tourne sur un VPS low-cost chez un hébergeur

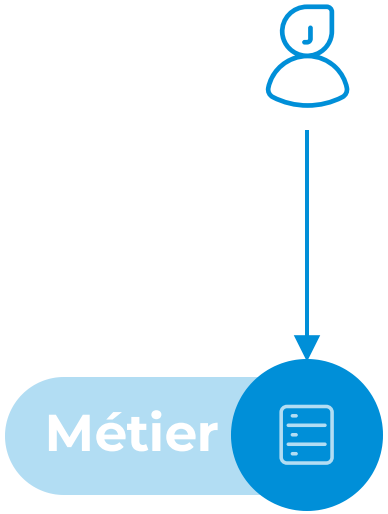
Code déployé avec des outils de CI/CD (déploiement continu)

C'est pas de tout repos...

Le **monolithe** MyGoodies

Monolithe

Application dont l'ensemble du code et des fonctionnalités est implémenté dans un seul programme



Avantages

Temps de développement réduit

Tout est au même endroit

Inconvénients

Les problèmes du **monolithe**

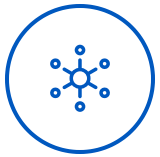
...tout est au même endroit

Traitement de requêtes séquentiel

- **Temps de latence plus grand**

Crashes impactant toute l'application

- **Risques d'indisponibilité globale**



Composants interdépendants

- **Features et fixes complexes**
- **Onboarding difficile**

Surprise ! Les campagnes BDE arrivent

Pic de charge sur MyGoodies, le site est KO.

- Il nous faut **plusieurs serveurs !**
- Redéployer un VPS est **painful,** on va utiliser **Docker**

Outil de **containerisation** lancé en 2013

Permet d'encapsuler des applications dans des images, déployables sur tout ordinateur ayant Docker.

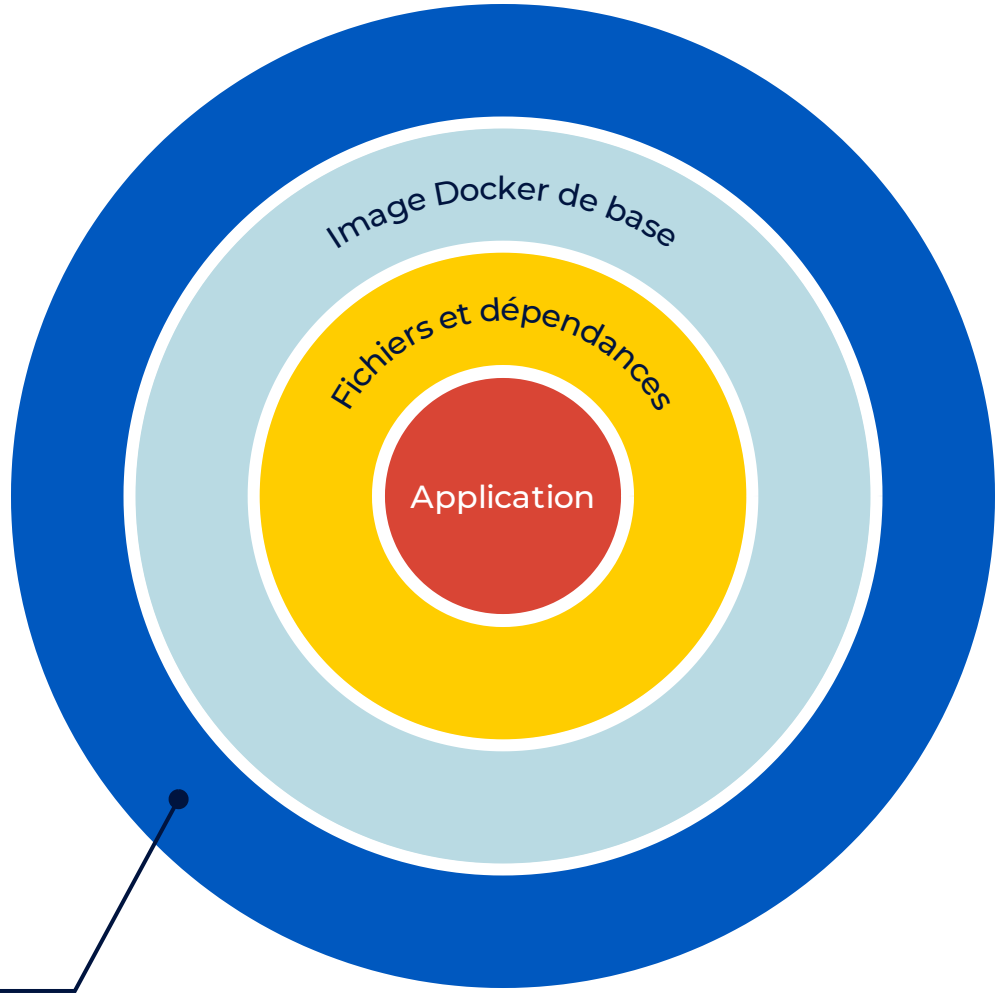


Image Docker

Une instance de cette image est un **container**

Outil de **containerisation** lancé en 2013

Permet d'encapsuler des applications dans des images, déployables sur tout ordinateur ayant Docker.

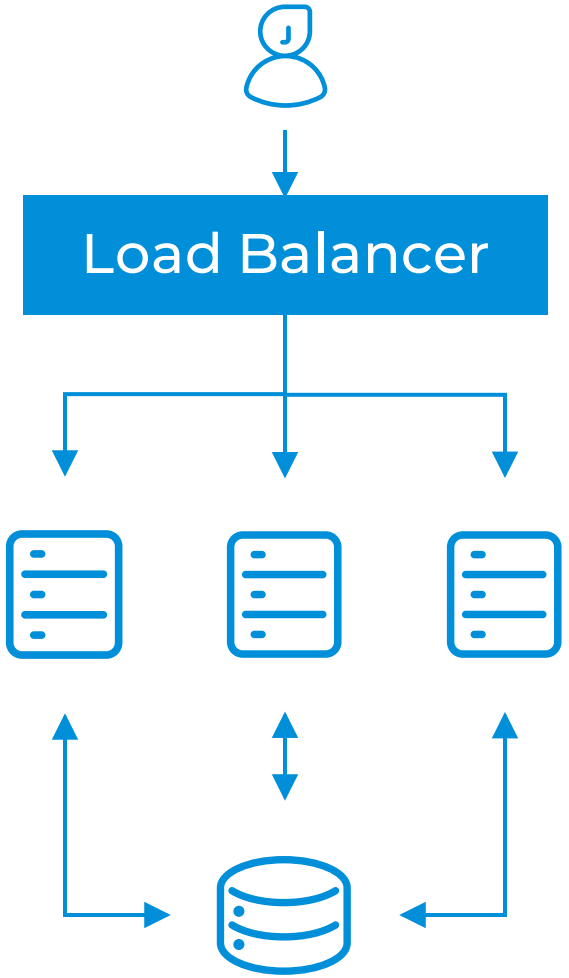
```
# Dockerfile
FROM ubuntu:22.04

WORKDIR /app

RUN apt-get update && apt-get install -y \
    python3 python3-pip

COPY requirements.txt /app/
RUN pip3 install --no-cache-dir -r requirements.txt

COPY . /app/

EXPOSE 8000
CMD ["python3", "manage.py", "runserver",
"0.0.0.0:8000"]
```

MyGoodies sur **plusieurs serveurs**

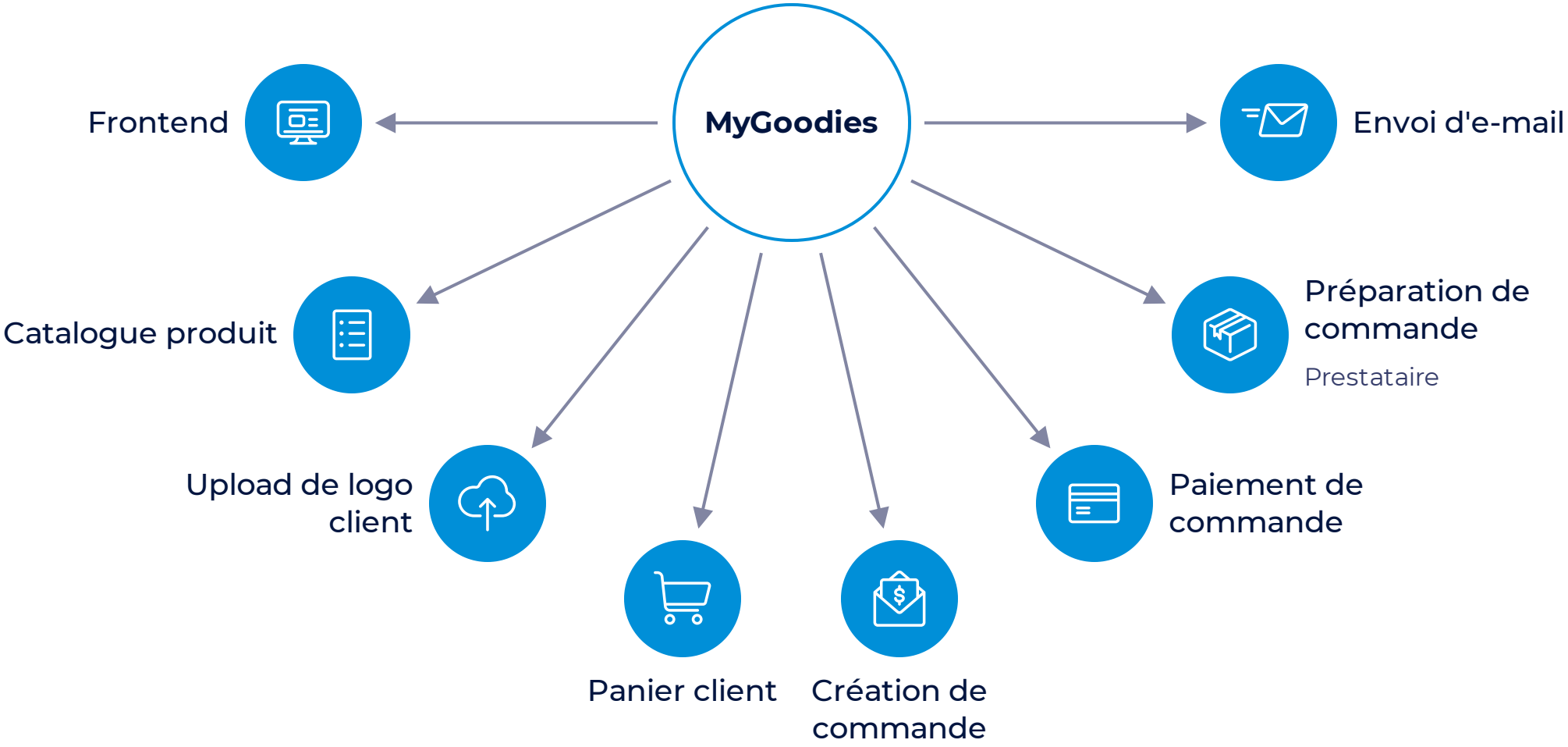
On lance des containers Docker sur chaque serveur

La mise en place d'un serveur est painful

Consistance des données ?
Crash d'un container ?



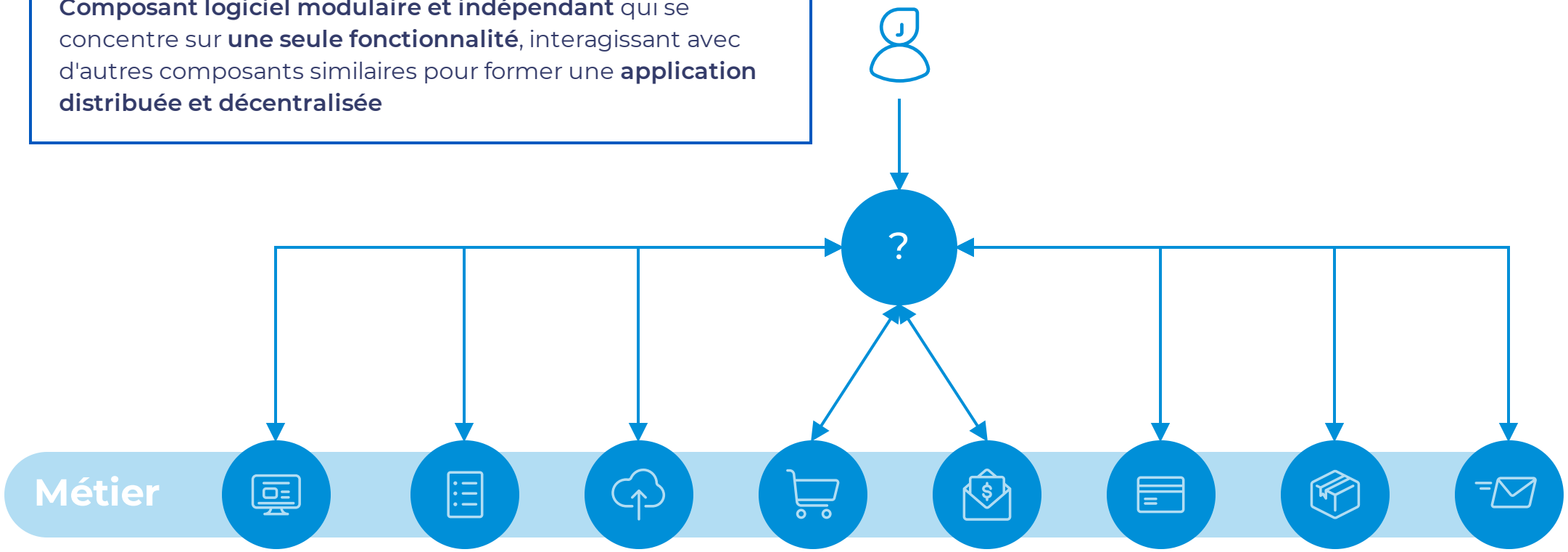
Qu'est-ce que **MyGoodies ?**



MyGoodies en **microservices**

Microservice

Composant logiciel modulaire et indépendant qui se concentre sur **une seule fonctionnalité**, interagissant avec d'autres composants similaires pour former une **application distribuée et décentralisée**



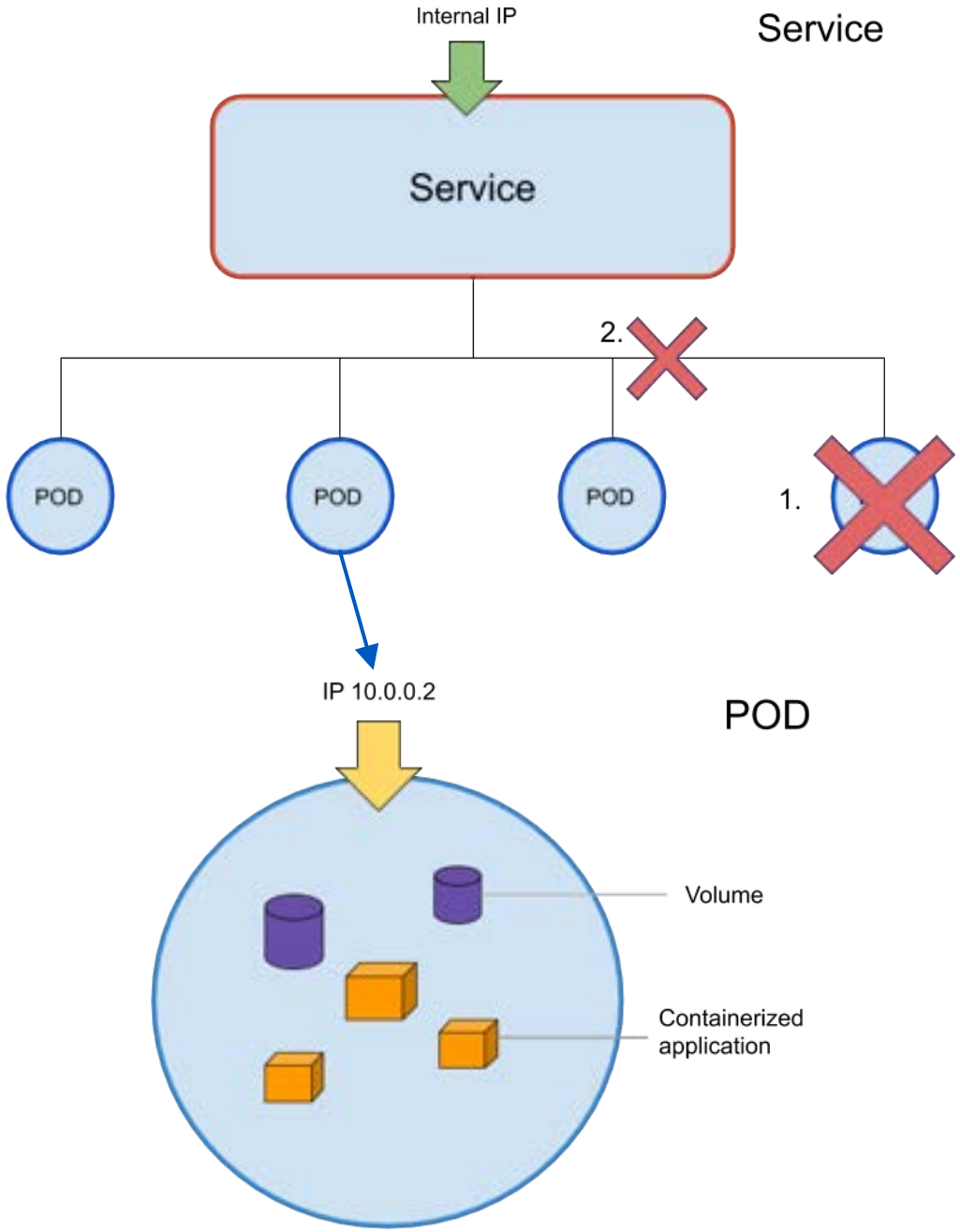
Kubernetes

Système créé par Google en 2014

Système fournissant une API permettant d'**orchestrer** des *pods*, objets contenant des containers, des volumes et des adresses IP, sur plusieurs serveurs ou *nodes*

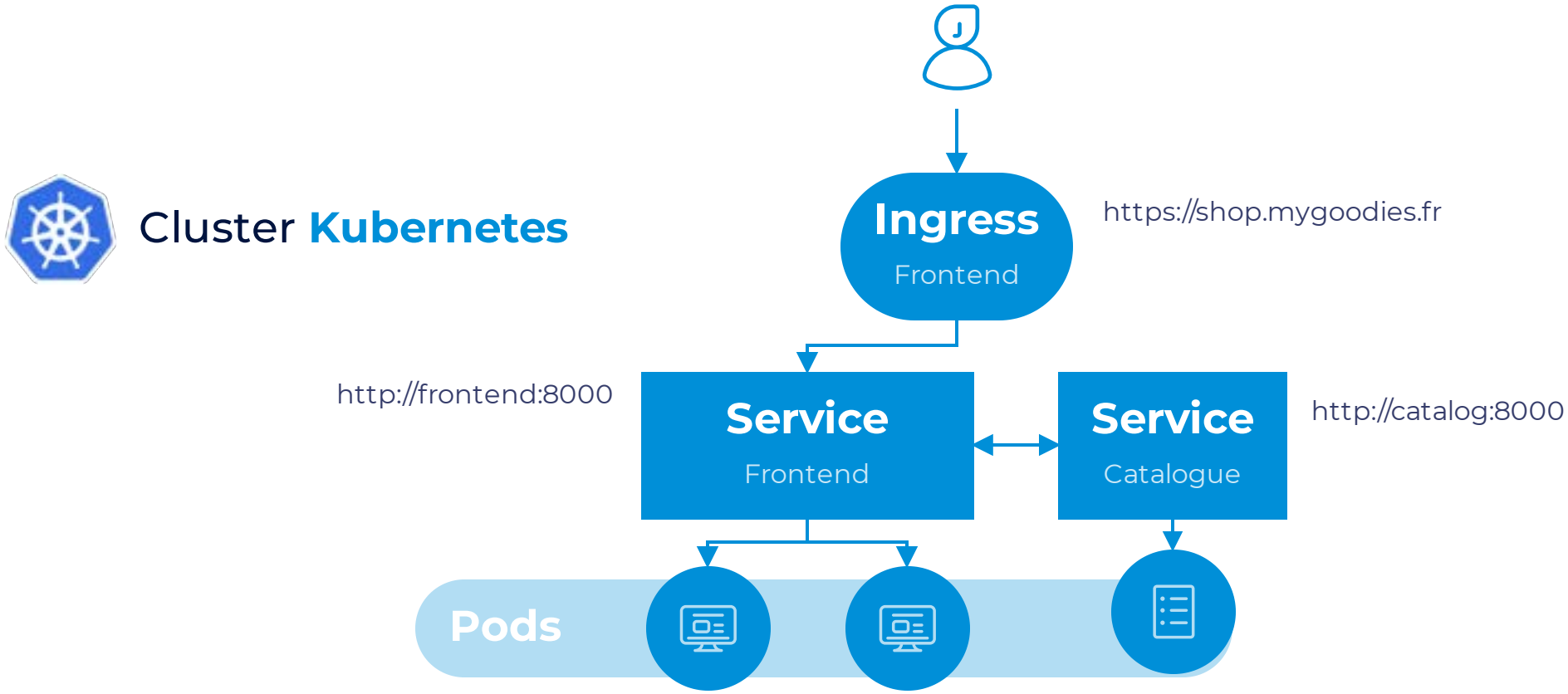
Un *pod* peut être redémarré, tué ou créé à tout moment, en fonction de la sollicitation des ressources ou la charge du trafic

Un *service* permet d'accéder à un ensemble de pods, selon leur disponibilité et leur charge



MyGoodies en **microservices**

Exemple : frontend et catalogue produits



Microservices

Avantages

- Agilité dans les développements
- Codebases plus petites
- Abstraction de logiques extérieures
- **Isolation d'erreurs**
- **Résilience**
- **Scaling**

Inconvénients

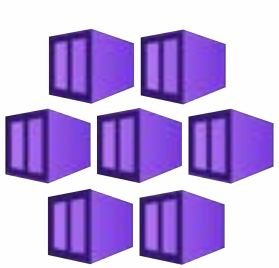
- Plus complexe qu'un monolithe
- Overhead pas forcément justifié
- Du gros rework côté applicatif (communication)
- **Mise en place de nodes Kubernetes complexe** on-premises
  - Importance des **cloud providers**

Cloud providers & KaaS

Kubernetes as a Service, parmi de nombreux services abstraits (exécution de code, de containers, base de données...)

Amazon Web Services

Elastic Kubernetes Service (EKS)



Microsoft Azure

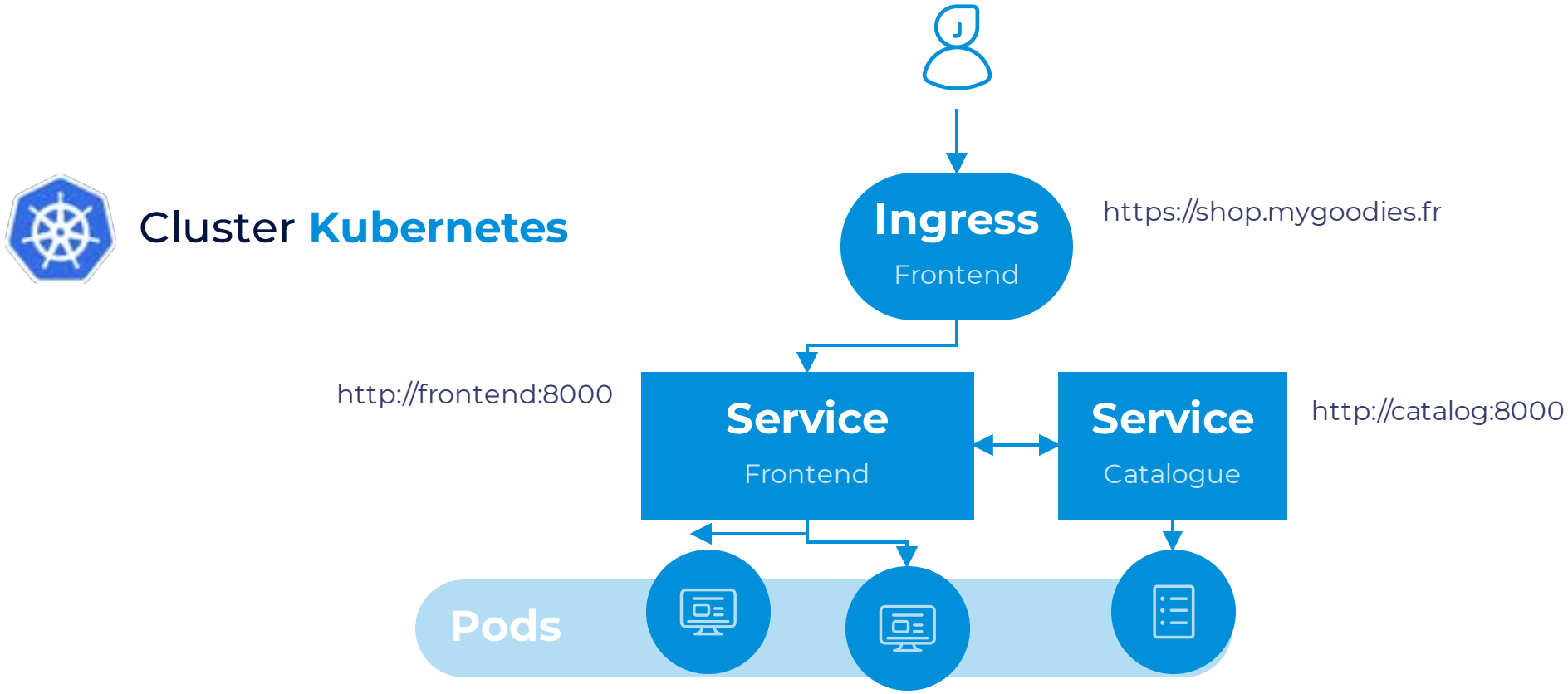
Azure Kubernetes Service (AKS)

Google Cloud

Google Kubernetes Engine (GKE)

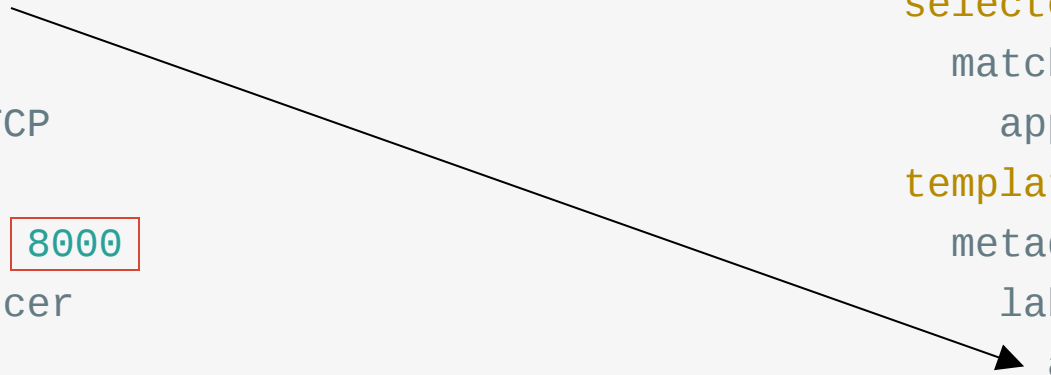
Comment faire du **Kubernetes ?**

Les **manifests** pour déclarer des ressources



```yaml
# service.yaml
apiVersion: v1
kind: Service
metadata:
  name: catalog
spec:
  selector:
    app: catalog
  ports:
    - protocol: TCP
      port: 8000
      targetPort: 8000
  type: LoadBalancer
```

```yaml
# deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: catalog
spec:
  replicas: 3
  selector:
    matchLabels:
      app: catalog
  template:
    metadata:
      labels:
        app: catalog
    spec:
      containers:
      - name: catalog
        image: mygoodies/catalog:1.12.8
        ports:
        - containerPort: 8000
```



```
$ kubectl apply -f service.yaml,deployment.yaml
```

Package manager permettant de gérer et déployer des applications dans des clusters Kubernetes

Chaque chart (package) contient des templates de **manifests**

```yaml
# templates/service.yaml
apiVersion: v1
kind: Service
metadata:
  name: {{ .Values.appName }}
spec:
  selector:
    app: {{ .Values.appName }}
  ports:
    - protocol: TCP
      port: {{ .Values.container.port }}
      targetPort: {{ .Values.container.port }}
  type: LoadBalancer
---
# values.yaml
appName: catalog
replicaCount: 3
container:
  image: mygoodies/catalog
  tag: 1.12.8
  port: 8000
```

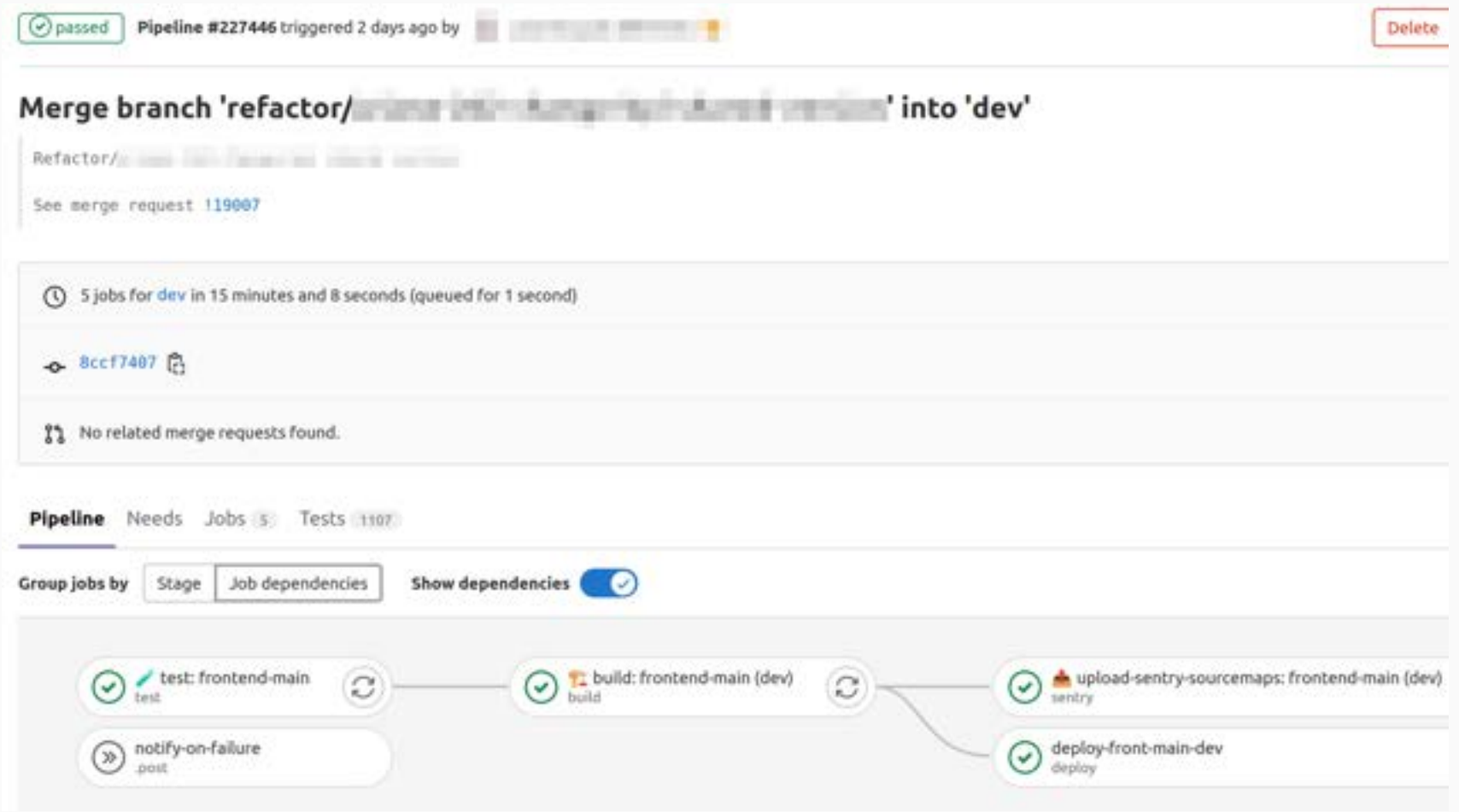
```yaml
# templates/deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ .Values.appName }}
spec:
  replicas: {{ .Values.replicaCount }}
  selector:
    matchLabels:
      app: {{ .Values.appName }}
  template:
    metadata:
      labels:
        app: {{ .Values.appName }}
    spec:
      containers:
      - name: {{ .Values.appName }}
        image: {{ .Values.container.image }}:{{
.Values.container.tag }}
        ports:
        - containerPort: {{ .Values.container.port }}
```

```
$ helm install catalog .
```

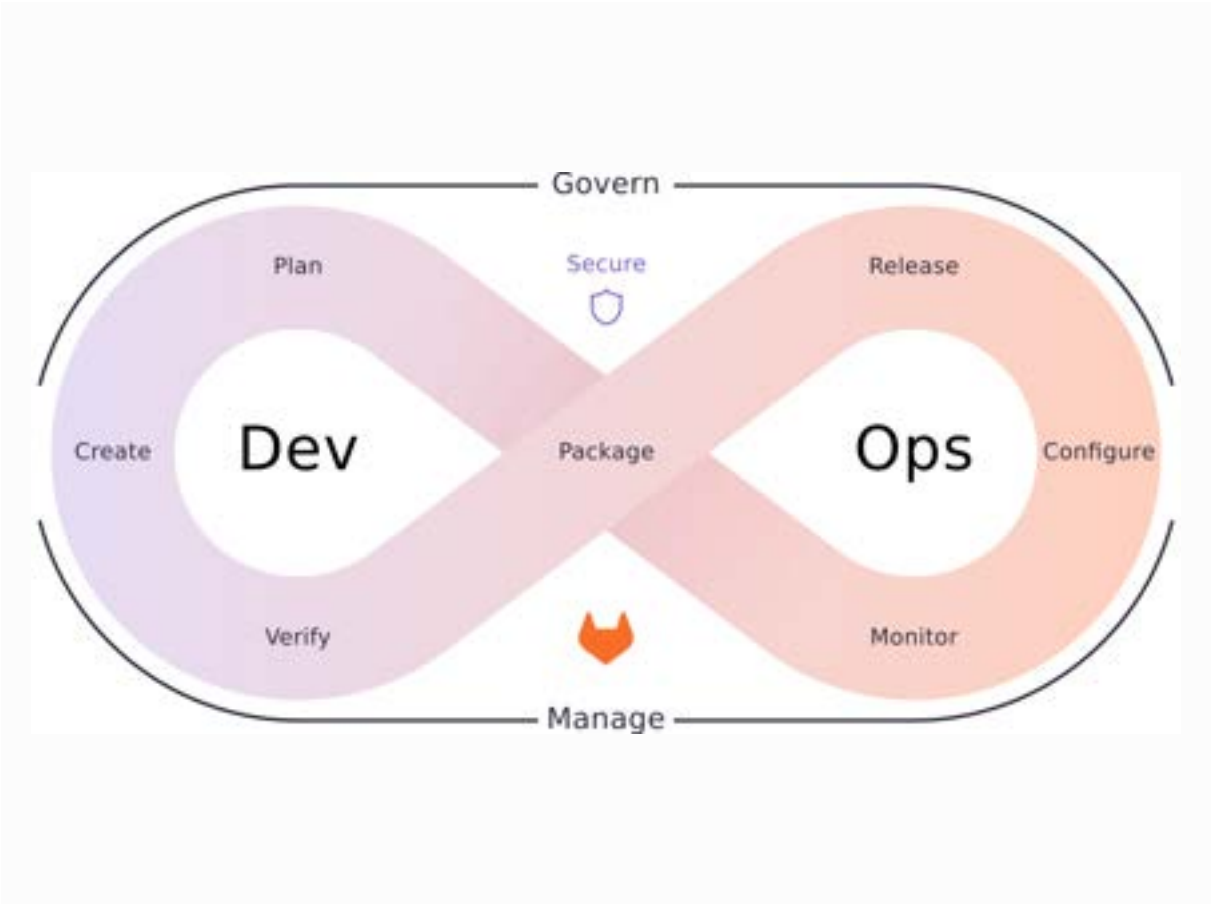
GitLab CI

La plateforme GitLab permet la mise en place de *pipelines*, des séries de commandes permettant d'automatiser des tâches.

C'est un outil **DevOps**, fusionnant développement et déploiement.



Le DevOps



- **Cycles de développement logiciel**

  Passage de développement à opération à développement...

- **Une seule et même boucle**

  On peut donner aux dév la main sur leurs déploiements

Déploiement avec **Helm**

service.yaml

charts > generic-api > templates > service.yaml > {map} > spec > {map}

```yaml
{{ if (list nil true | has .Values.service.enabled) }}
apiVersion: v1
kind: Service
metadata:
  name: {{ include "chart.fullname" . }}
  labels:
{{ include "chart.labels" . | indent 4 }}
spec:
  selector:
    app.kubernetes.io/name: {{ include "chart.name" . }}
    app.kubernetes.io/instance: {{ .Release.Name }}
  type: {{ .Values.service.type }}
  ports:
    - name: http
      port: {{ .Values.service.port }}
      targetPort: {{ .Values.container.port }}
      protocol: TCP
    - name: https
      port: {{ .Values.service.httpsPort }}
      targetPort: {{ .Values.container.port }}
      protocol: TCP
{{ end }}
```

Chart Helm

Exécution d'une commande

HELM

Apply des manifests templatisés

Amazon EKS

Cluster Kubernetes

- Helm doit être lancé dans une pipeline
- Helm ne peut effectuer deux opérations à la fois
- On n'est pas notifié si le cluster est modifié manuellement

Déploiement avec ArgoCD

Outil déclaratif de **déploiement continu GitOps** pour **Kubernetes**

service.yaml

charts > generic-api > templates > service.yaml > {map} > spec > {map}

```yaml
{{ if (list nil true | has .Values.service.enabled) }}
apiVersion: v1
kind: Service
metadata:
  name: {{ include "chart.fullname" . }}
  labels:
{{ include "chart.labels" . | indent 4 }}
spec:
  selector:
    app.kubernetes.io/name: {{ include "chart.name" . }}
    app.kubernetes.io/instance: {{ .Release.Name }}
  type: {{ .Values.service.type }}
  ports:
    - name: http
      port: {{ .Values.service.port }}
      targetPort: {{ .Values.container.port }}
      protocol: TCP
    - name: https
      port: {{ .Values.service.httpsPort }}
      targetPort: {{ .Values.container.port }}
      protocol: TCP
{{ end }}
```

Chart Helm

dans un repo dédié

Notification lors d'une mise à jour

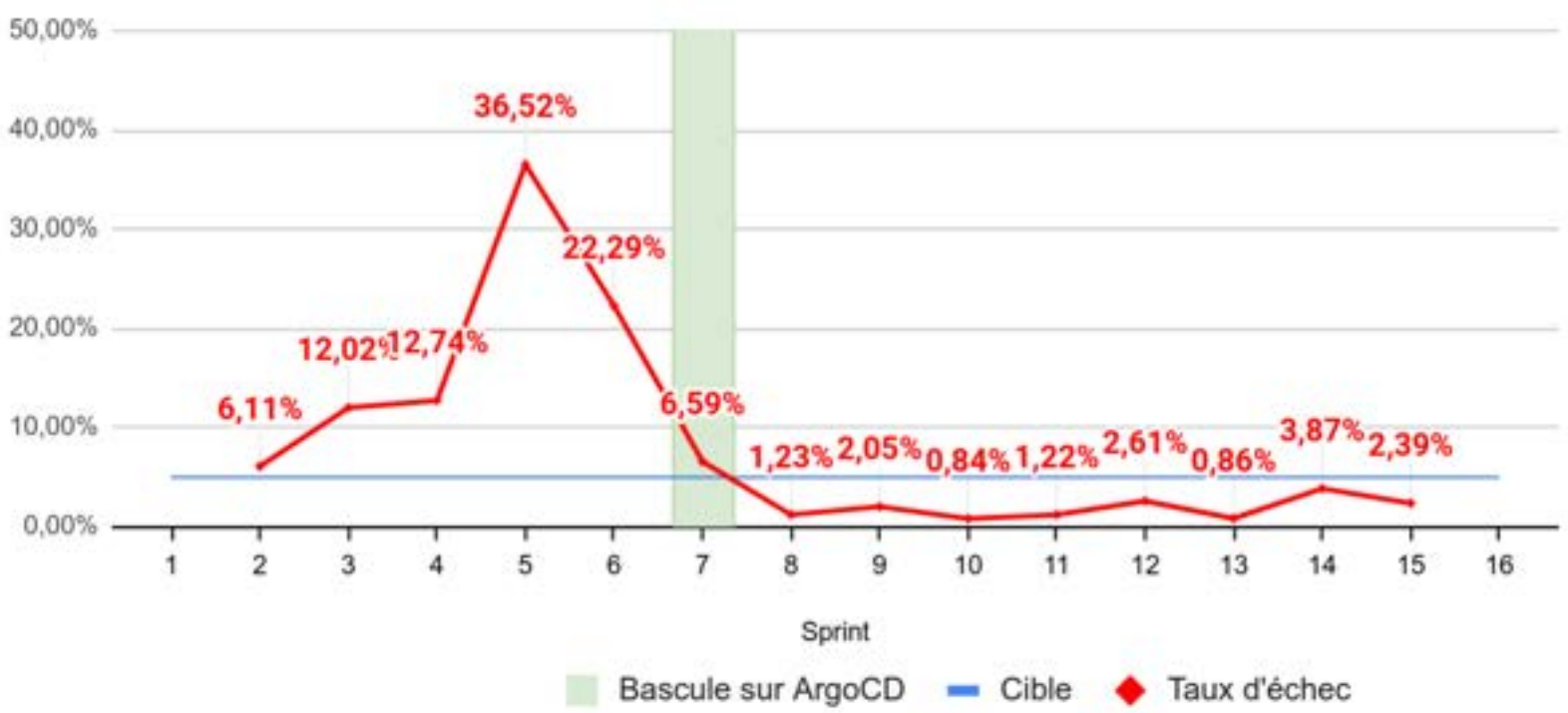
Apply des manifests templatisés

Cluster Kubernetes

GitOps

Approche de gestion et d'automatisation des déploiements d'applications et d'infrastructures, **basée sur l'utilisation de repos Git**

Réduction des erreurs de **déploiement**



Graphe issu d'un vrai projet

- Plus de 100 dévs
- 8-10 ops
- 20 micro-services

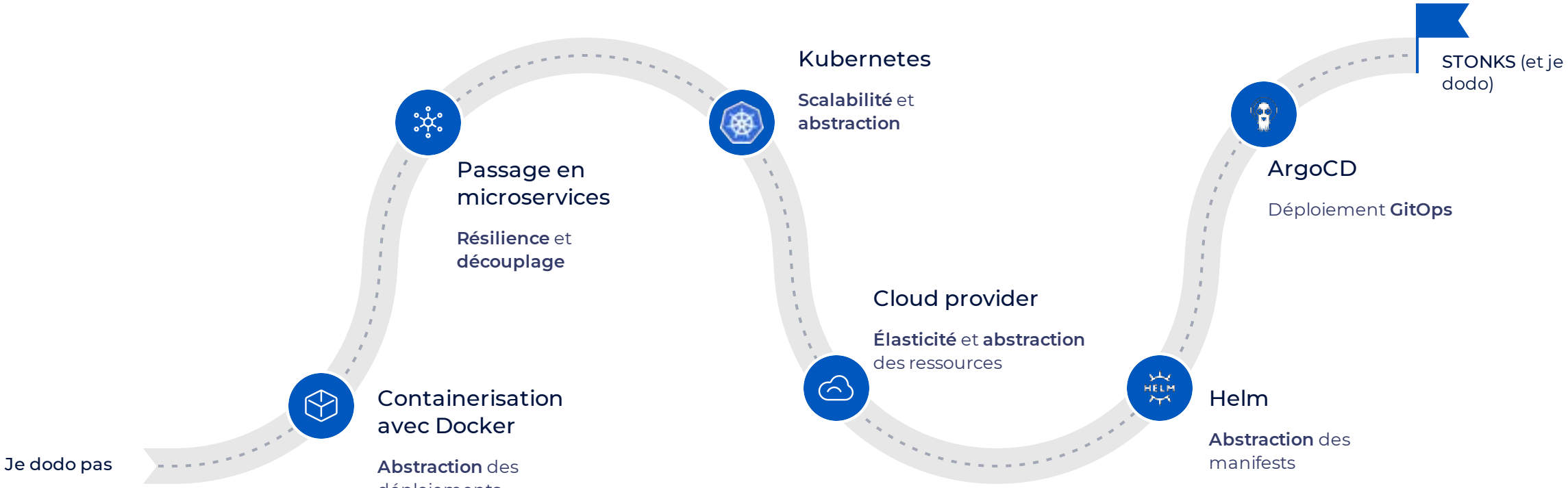
L'interface graphique d'ArgoCD

Les **développeurs** ont de la **visibilité** sur le **lifecycle** de leurs **applications**

La **morale** de l'histoire

GitOps = ce qui est déployé sur le cluster Kubernetes correspond **exactement** à ce qui est déclaré dans un repo Git =

La vraie **morale** de l'histoire



Le métier de SRE : assurer la fiabilité des infrastructures et automatiser les processus

Merci !

Samy Djemaï

@SamyDjemai

samyd@padok.fr