GitOps

“in a nutshell”

Worum gehts hier?

nutshell

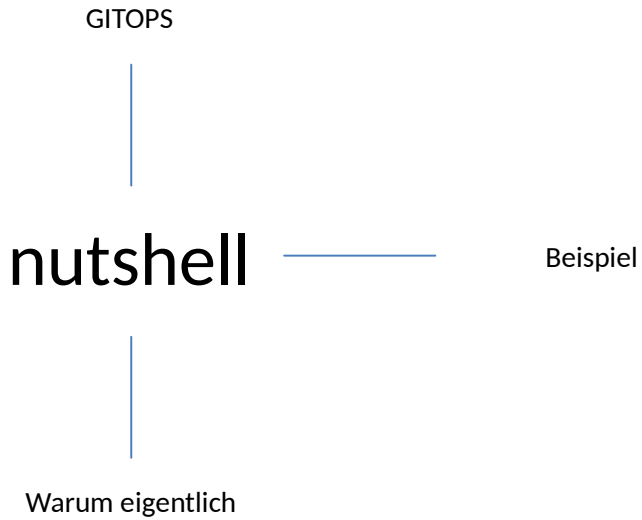
Worum gehts hier?

GITOPS

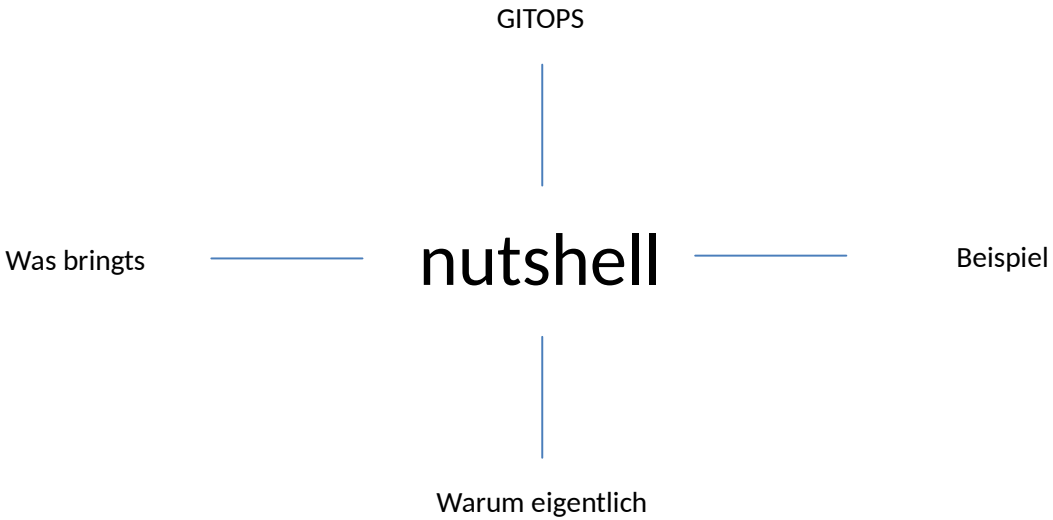
nutshell

Worum gehts hier?

Worum gehts hier?



Worum gehts hier?



Warum darf ich hier stehen?

Willkommen

@cedricboesiger

huma

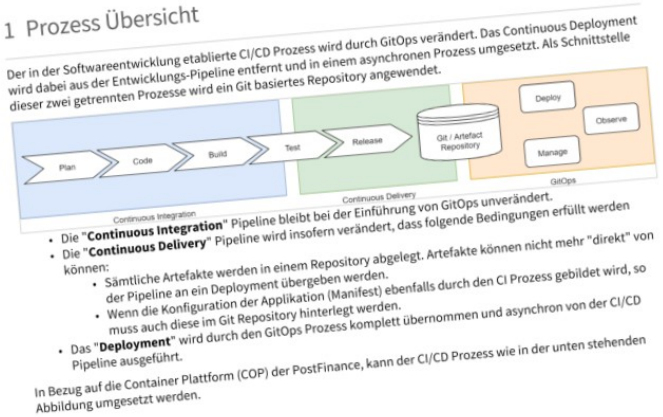
PostFinance

Willkommen

huma

PostFinance

@cedricboesiger



1 Prozess Übersicht

Der in der Softwareentwicklung etablierte CI/CD Prozess wird durch GitOps verändert. Das Continuous Deployment wird dabei aus der Entwicklungs-Pipeline entfernt und in einem asynchronen Prozess umgesetzt. Als Schnittstelle dieser zwei getrennten Prozesse wird ein Git basiertes Repository angewendet.

- Die "**Continuous Integration**" Pipeline bleibt bei der Einführung von GitOps unverändert.
- Die "**Continuous Delivery**" Pipeline wird insofern verändert, dass folgende Bedingungen erfüllt werden können:
  - Sämtliche Artefakte werden in einem Repository abgelegt. Artefakte können nicht mehr "direkt" von der Pipeline an ein Deployment übergeben werden.
  - Wenn die Konfiguration der Applikation (Manifest) ebenfalls durch den CI Prozess gebildet wird, so muss auch diese im Git Repository hinterlegt werden.
- Das "**Deployment**" wird durch den GitOps Prozess komplett übernommen und asynchron von der CI/CD Pipeline ausgeführt.

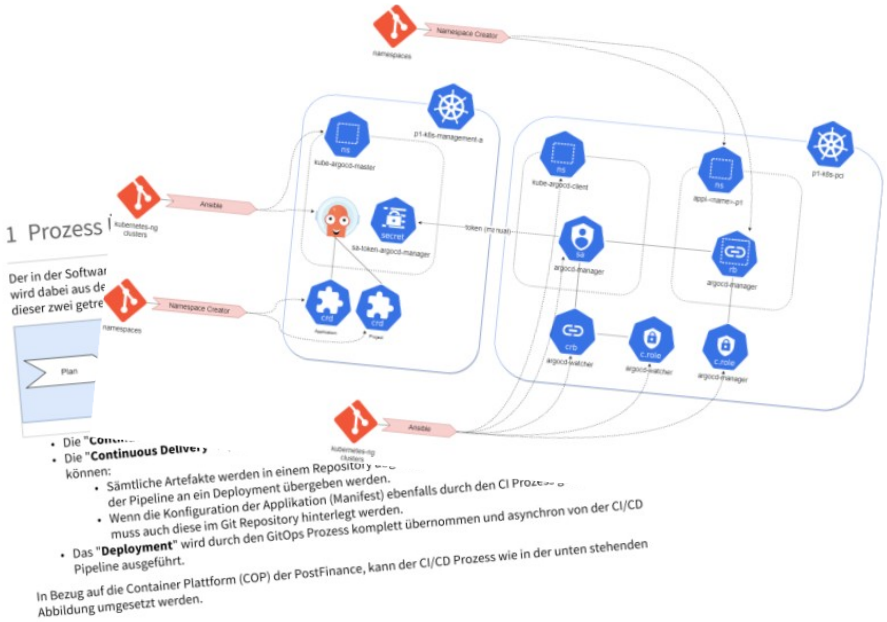
In Bezug auf die Container Plattform (COP) der PostFinance, kann der CI/CD Prozess wie in der unten stehenden Abbildung umgesetzt werden.

Willkommen

@cedricboesiger

huma

PostFinance

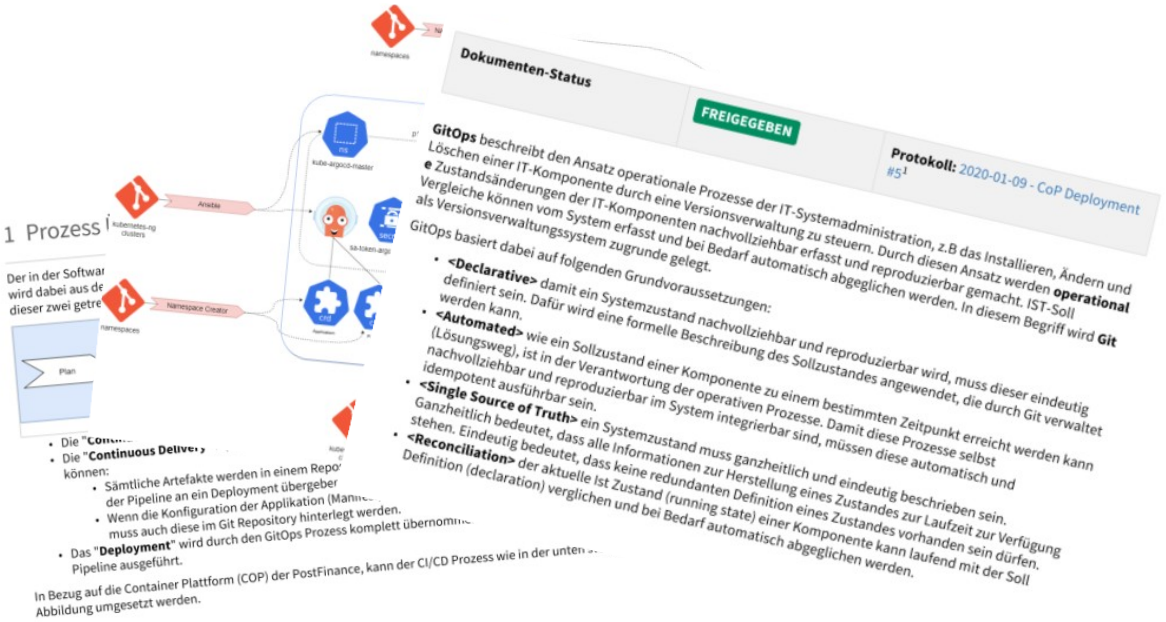


Willkommen

@cedricboesiger

huma

PostFinance



Ok, und was ist GitOps?

GitOps?

Git

Ops

GitOps?

Git

Laufzeitumgebung

GitOps?

Git

SOLL Zustand des Services

Laufzeitumgebung

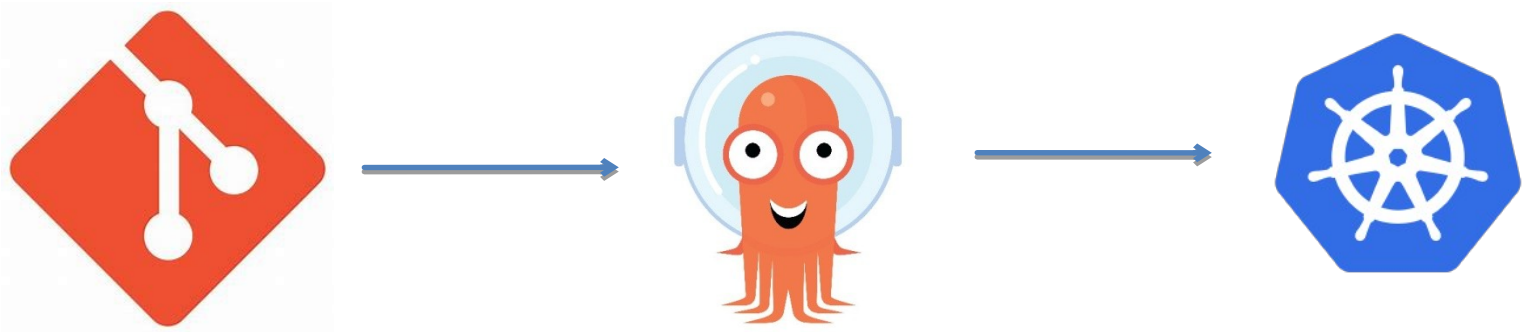
IST Zustand des Services

GitOps?

Git

Agent

Laufzeitumgebung



SOLL Zustand des Services

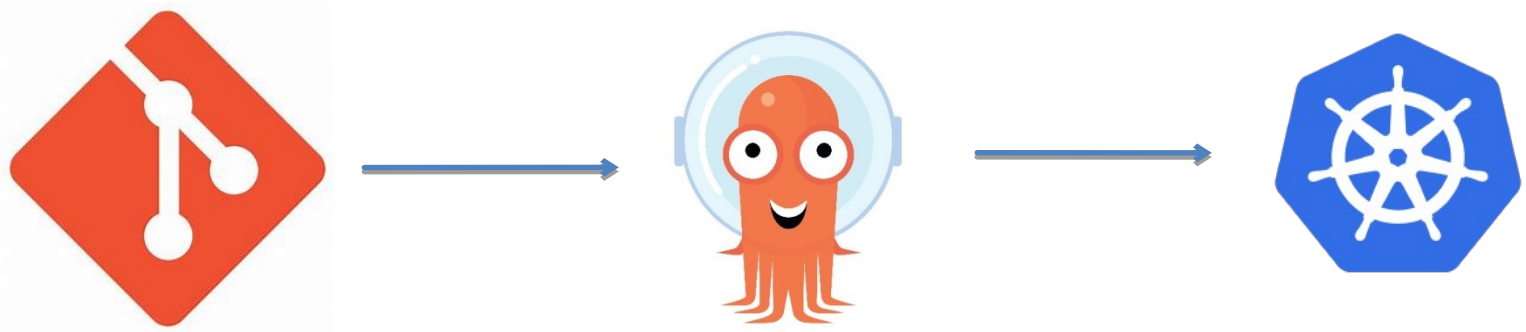
IST Zustand des Services

GitOps?

Git

Agent

Laufzeitumgebung



SOLL Zustand des Services

Abgleich Soll/Ist

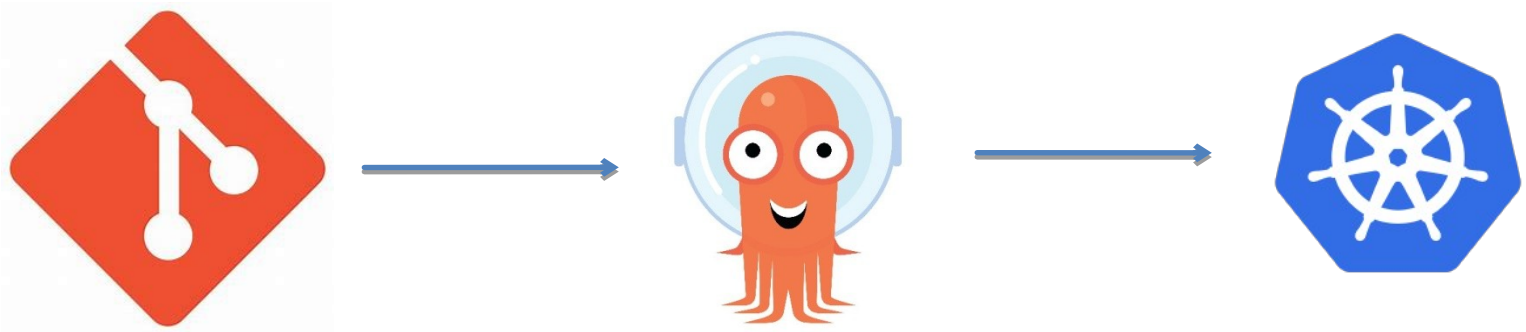
IST Zustand des Services

GitOps?

Git

Agent

Laufzeitumgebung



SOLL Zustand des Services

Abgleich Soll/Ist

IST Zustand des Services

Single Source of Truth

Deploy

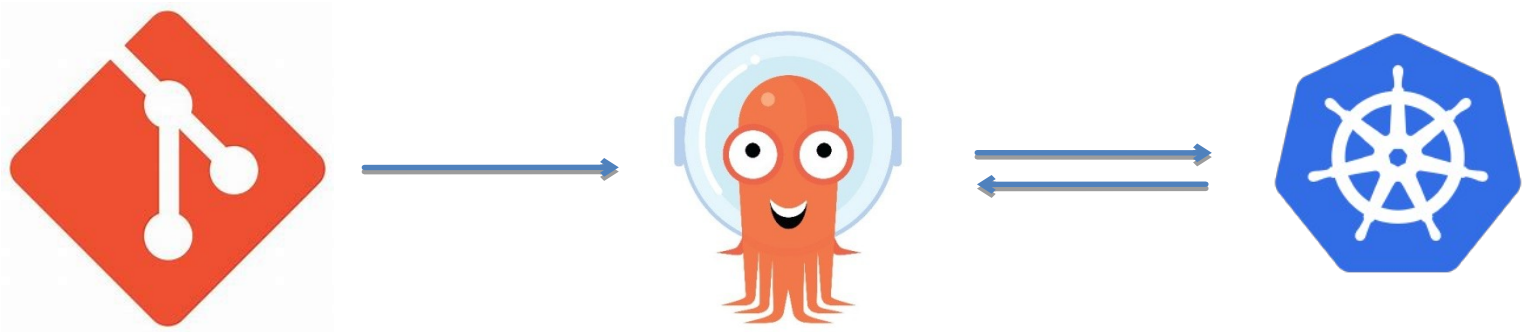
Run

GitOps?

Git

Agent

Laufzeitumgebung



SOLL Zustand des Services

Abgleich Soll/Ist

IST Zustand des Services

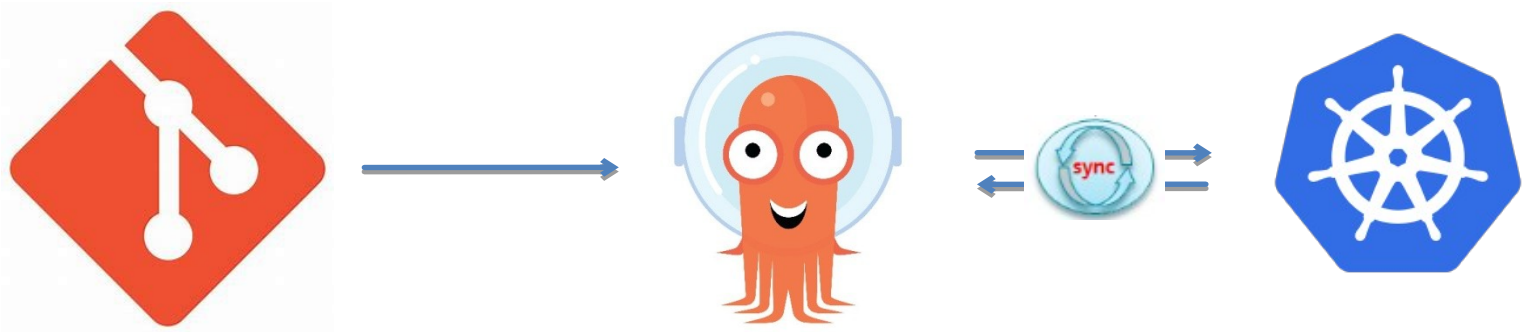
Single Source of Truth
Declarative

Deploy

Run
Observe

GitOps?

Git | Agent | Laufzeitumgebung



SOLL Zustand des Services

Abgleich Soll/Ist

IST Zustand des Services

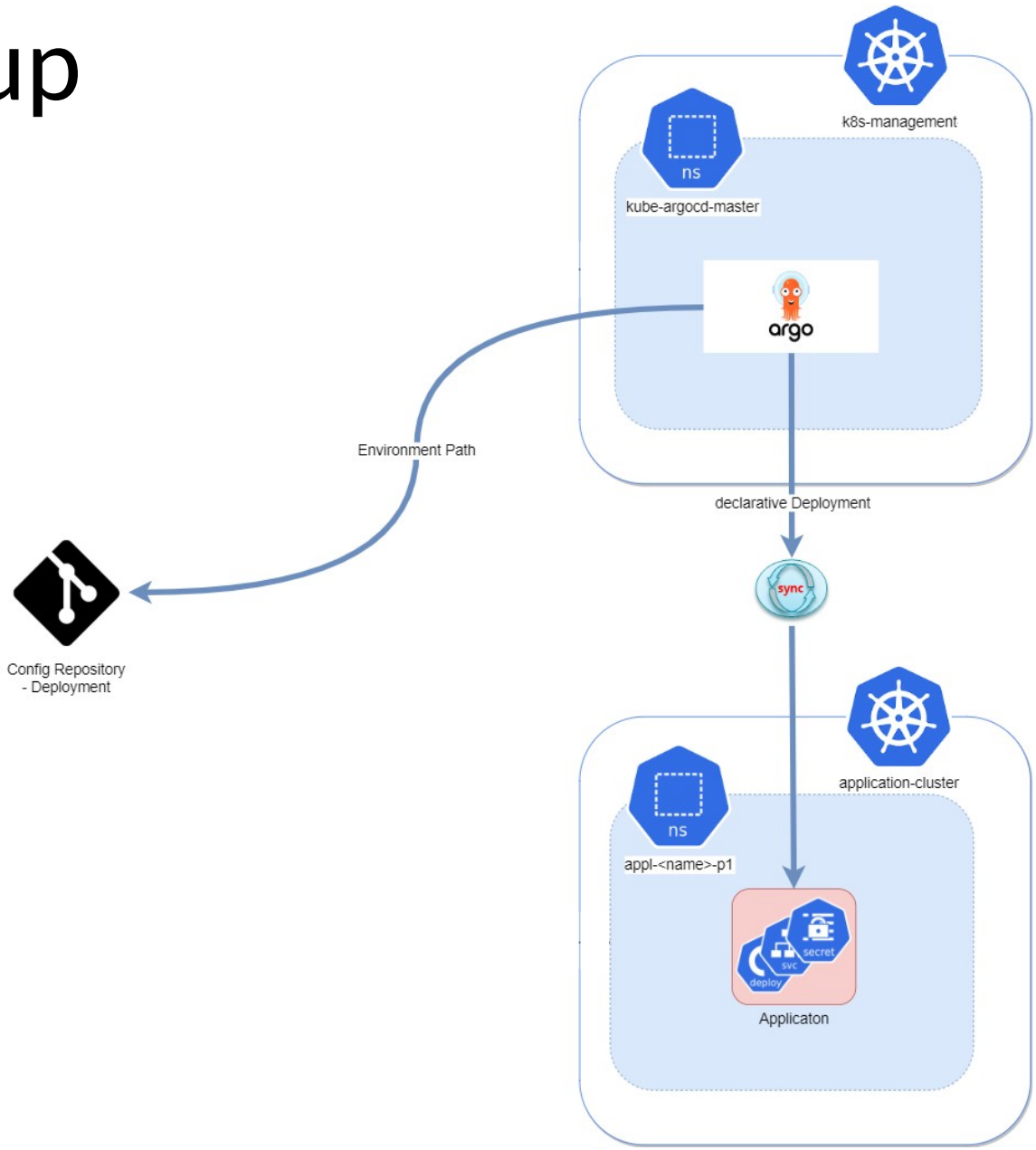
Single Source of Truth
Declarative

Deploy
Converge / Reconcile

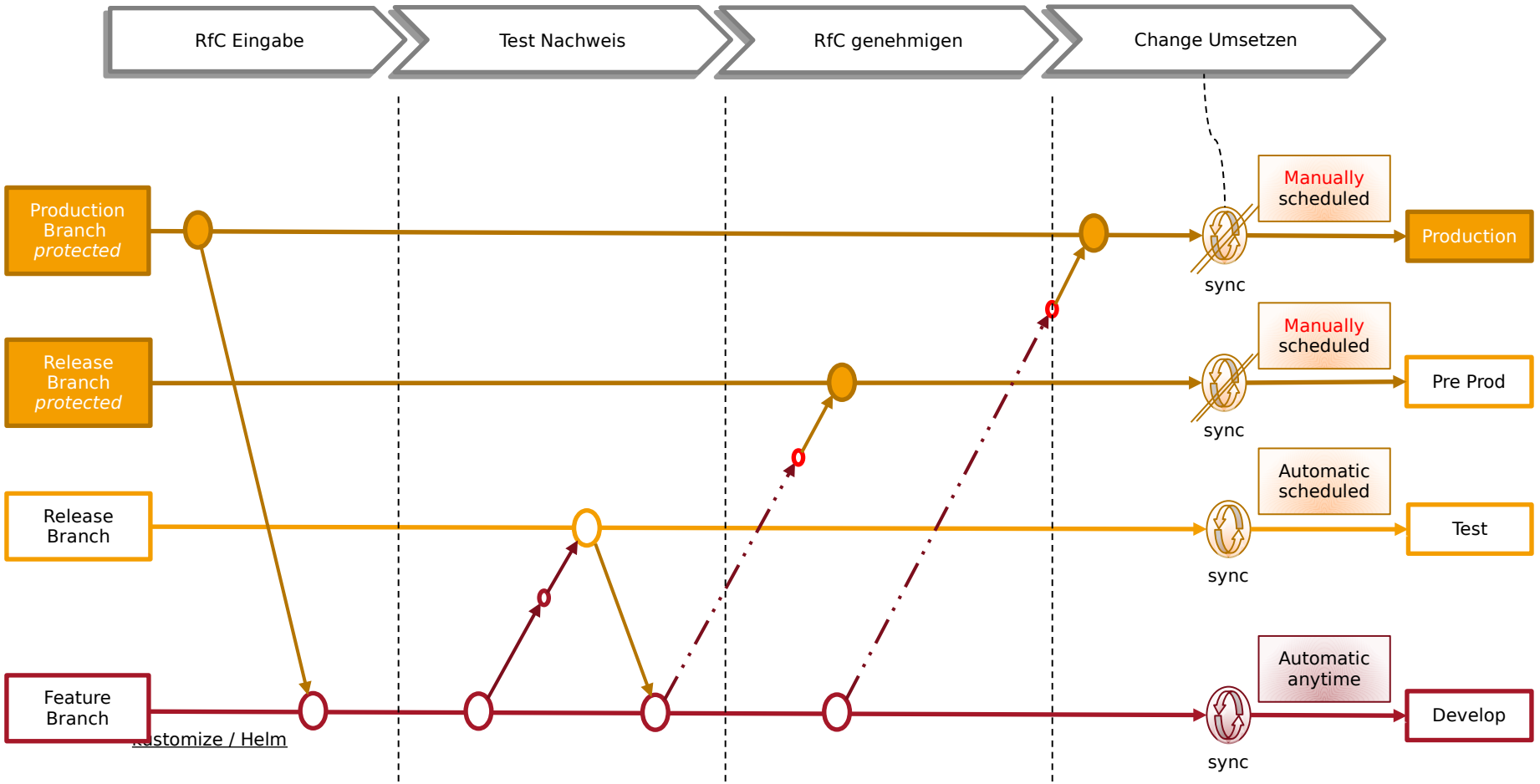
Run
Observe

Und wie könnte ein
echtes Beispiel aussehen ?

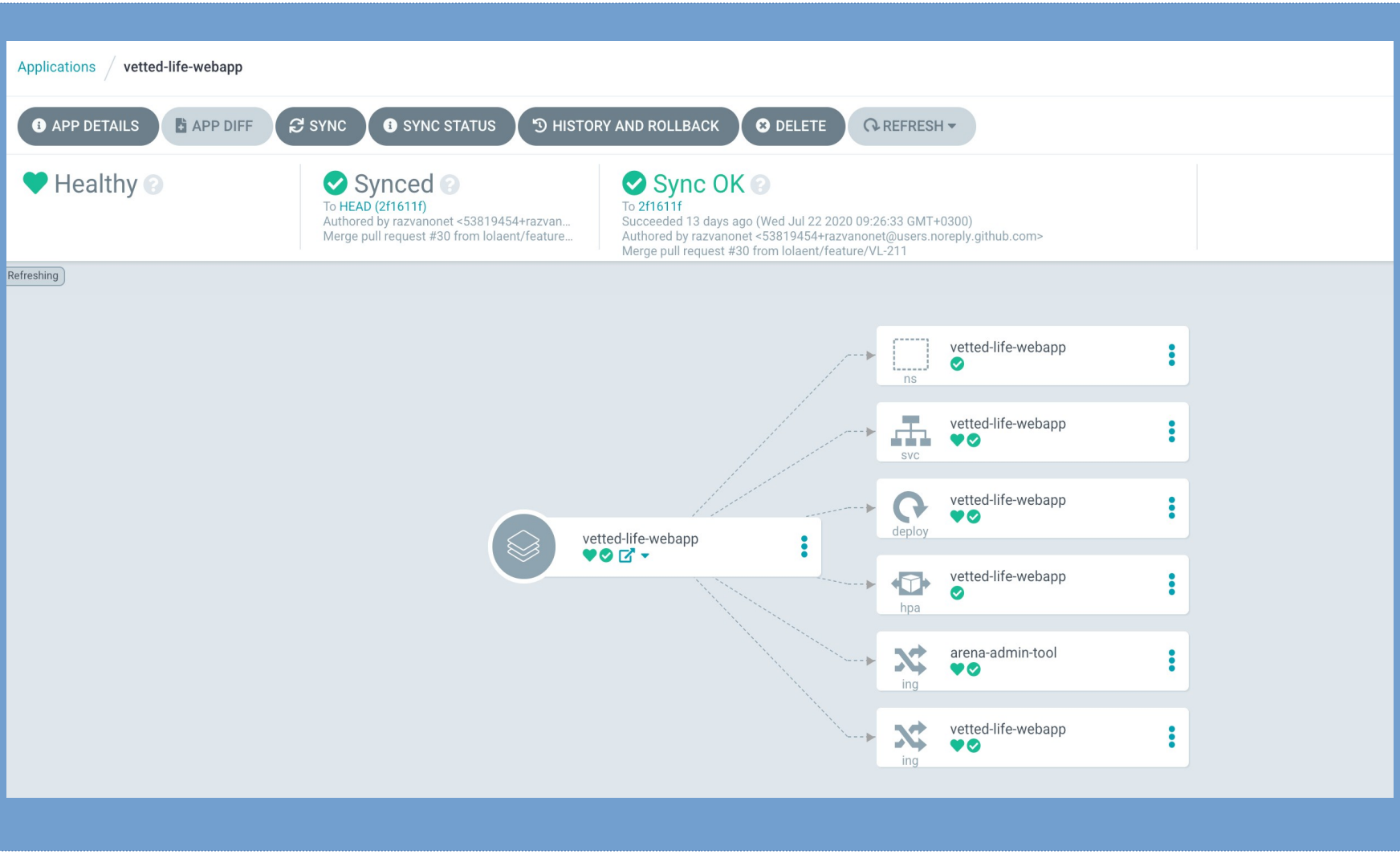
Setup


k8s-management
ns
kube-argocd-master
argo
Environment Path
declarative Deployment
sync
Config Repository
- Deployment
application-cluster
ns
appl-<name>-p1
secret
svc
deploy
Applicaton

Git Flow

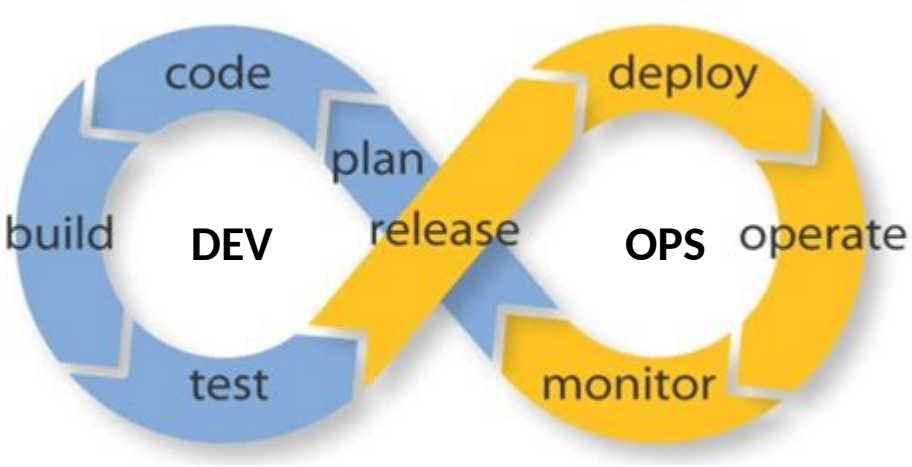

RfC Eingabe
Test Nachweis
RfC genehmigen
Change Umsetzen
Production Branch protected
Release Branch protected
Release Branch
Feature Branch
Kustomize / Helm
Manually scheduled
Production
sync
Manually scheduled
Pre Prod
sync
Automatic scheduled
Test
sync
Automatic anytime
Develop
sync

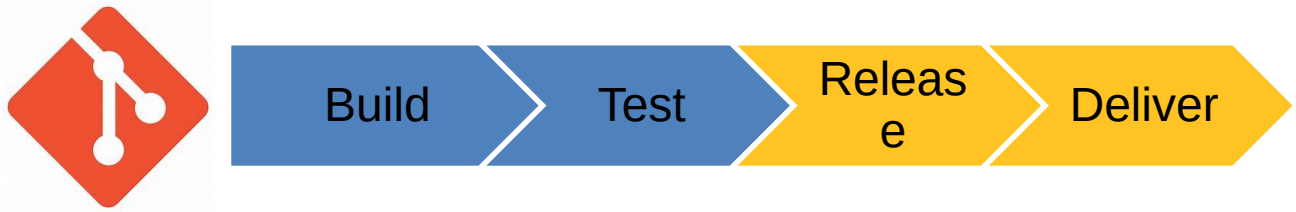
ArgoCD in Action



Warum eigentlich GitOps?

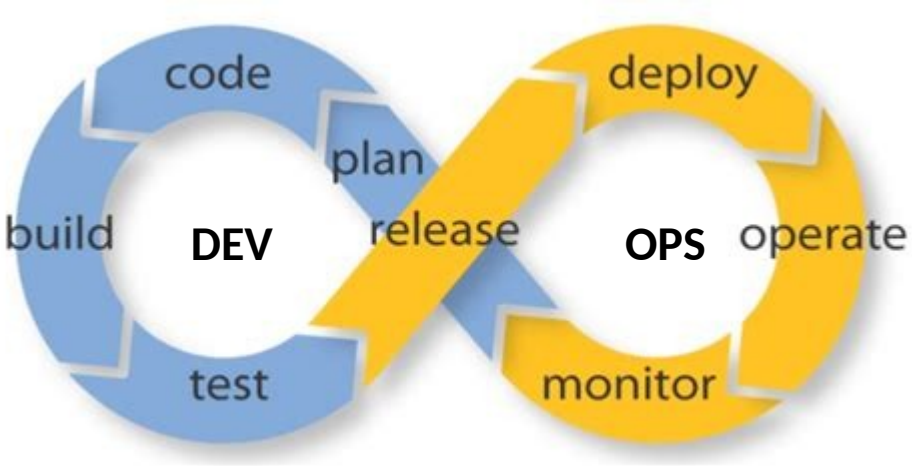
Von DevOps zu CI/CD





Continuous Integration Continuous Delivery

Von DevOps zu CI/CD



Continuous Integration Continuous Delivery Continuous Deployment

Von CI/CD Pipelines zu CI-OPS

(Alexis Richardson)

CI-OPS

Es funktioniert, kann aber schwierig sein

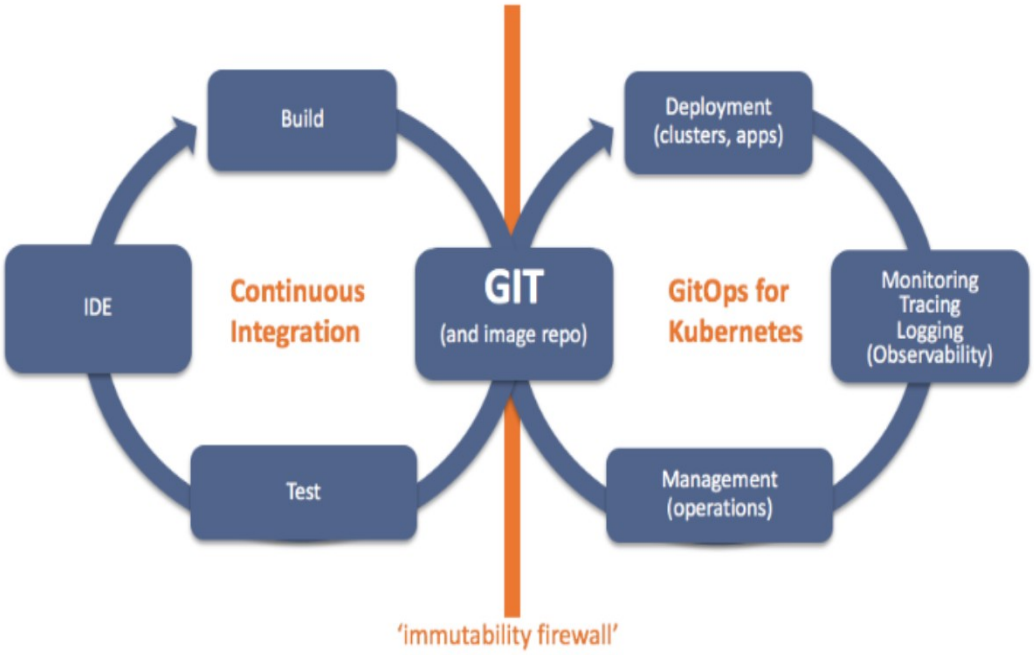
weil

die Hardware häufig am falschen Ort ist
die Verfügbarkeit teils nicht stimmt
die “Segregation of duty” schwierig ist
die Nachvollziehbarkeit teils nicht gegeben ist

und es oft nur scheinbar “Infrastructure as Code” ist

Der GitOps Ansatz

https://www.weave.works/blog/what-is-gitops-really



Git as the single source of truth of a system's desired state

GitOps Diffs compare desired state with observed state (eg Kubediff, Terradiff, Canary..)

ALL intended operations are committed by pull request, for all environments

ALL diffs between GIT and observed state lead to (auto) convergence using tools like K8s

ALL changes are observable, verifiable and audited indisputably, with rollback & D/R

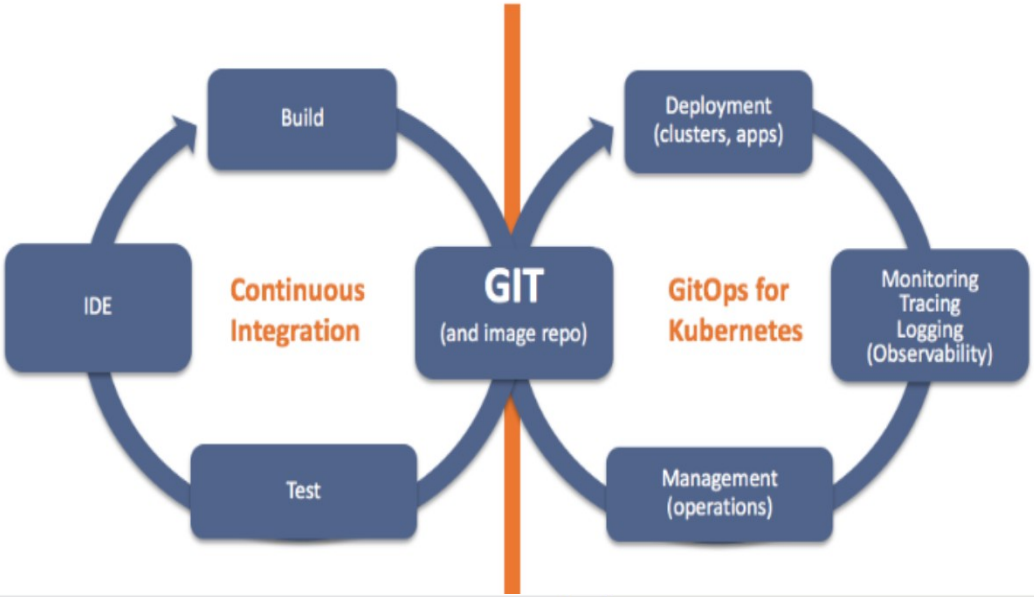
Git-Push
DEV

merge →

Git-Pull
OPS

Der GitOps Ansatz

https://www.weave.works/blog/what-is-gitops-really



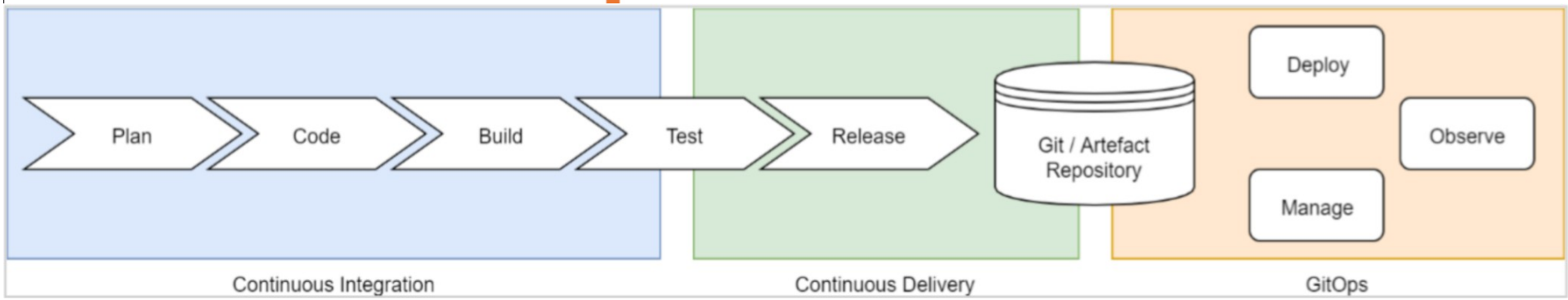
Git as the single source of truth of a system's desired state

GitOps Diffs compare desired state with observed state (eg Kubediff, Terradiff, Canary..)

ALL intended operations are committed by pull request, for all environments

ALL diffs between GIT and observed state lead to (auto) convergence using tools like K8s

ALL changes are observable, verifiable and audited indisputably, with rollback & D/R



Und noch ein paar Statements

Ein paar Statements zu GitOps

GitOps baut auf DEVOPS auf

Ein paar Statements zu GitOps

Git ist “single source of truth” und deklarativ

Ein paar Statements zu GitOps

Der “running State” ist “observable” und kann “reconciled” werden

Ein paar Statements zu GitOps

Alles ist unter Versionskontrolle,

Nachvollziehbar,

“Rollback”-bar

und “Restore”-bar

Ein paar Statements zu GitOps

Das Modell ist “nice and easy”

Ein paar Statements zu GitOps

“Compliance” kommt in vielen Bereichen einfach so mit

Ein paar Statements zu GitOps

“Collaboration” ist jetzt auch für OPS

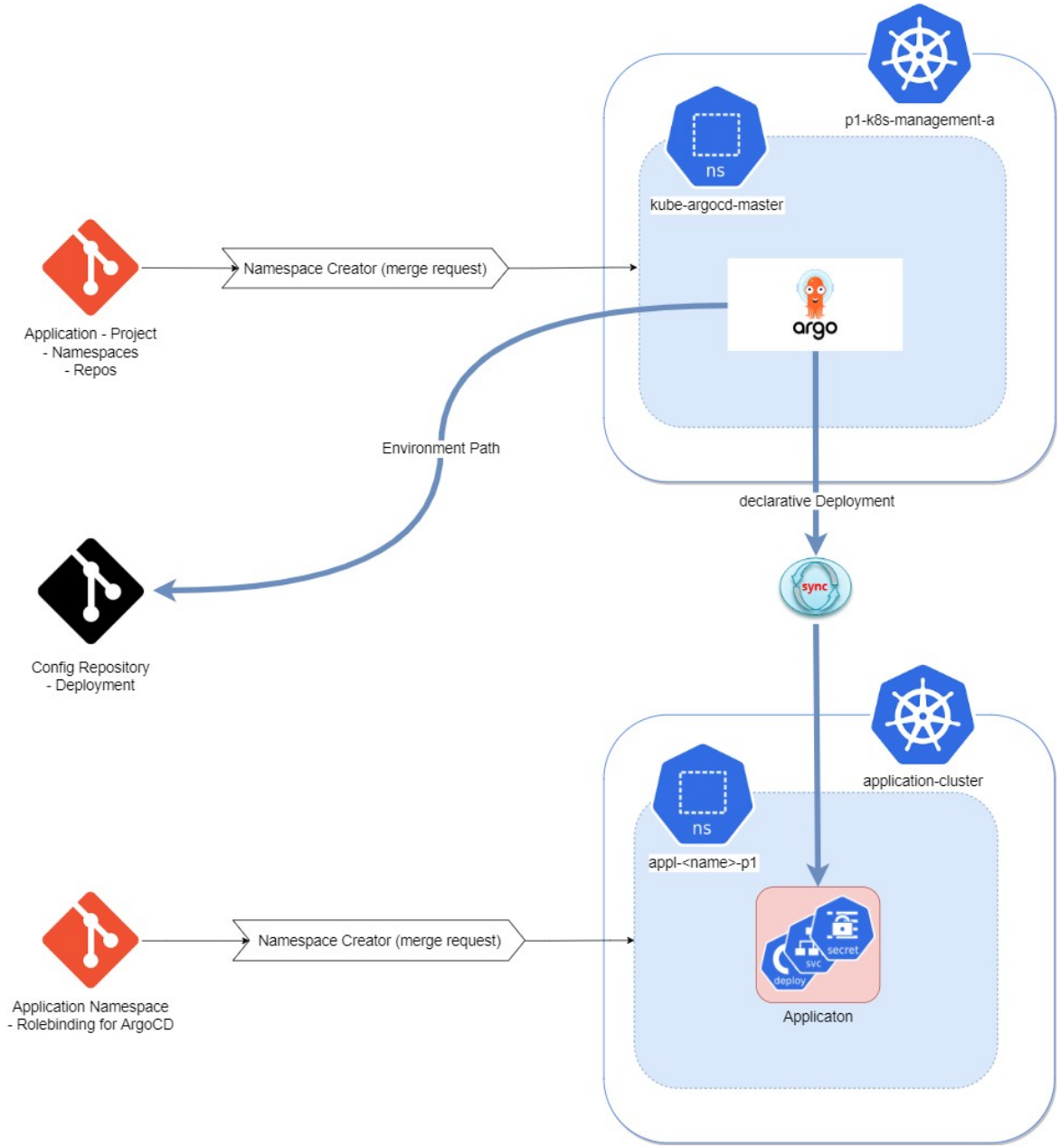
zum Schluss…

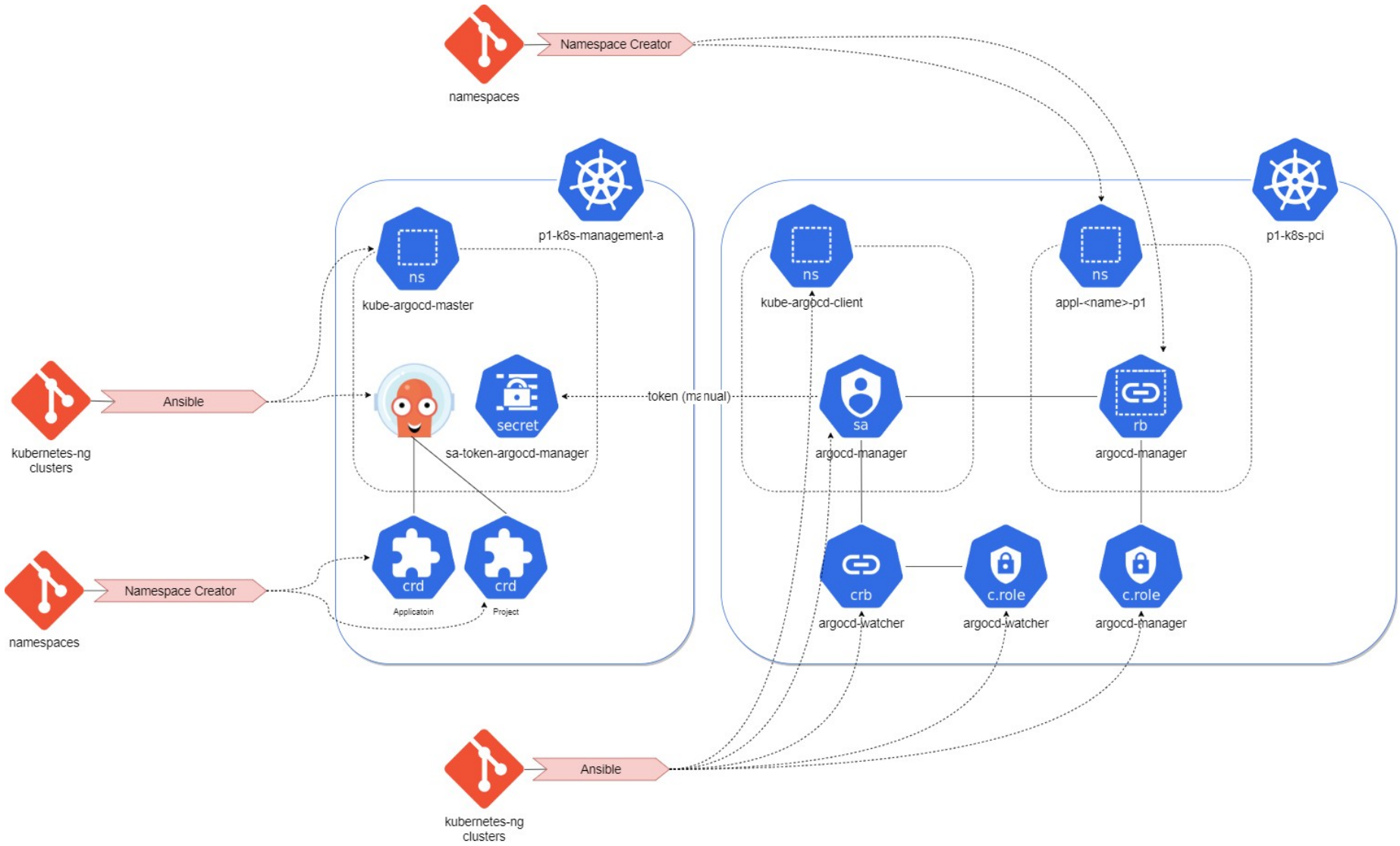
Es hilft die “Security” und “Velocity” zu erhöhen

zum Schluss…

Es hilft die “Security” und “Velocity” zu erhöhen

…vielen Dank!


p1-k8s-management-a
ns
kube-argocd-master
Application - Project
- Namespaces
- Repos
Namespace Creator (merge request)
argo
Environment Path
declarative Deployment
Config Repository
- Deployment
sync
application-cluster
ns
appl-<name>-p1
Application Namespace
- Rolebinding for ArgoCD
Namespace Creator (merge request)
secret
svc
deploy
Applicaton


Namespace Creator
namespaces
p1-k8s-management-a
p1-k8s-pci
ns
kube-argocd-master
ns
kube-argocd-client
ns
appl-<name>-p1
Ansible
kubernetes-ng clusters
secret
sa-token-argocd-manager
token (manual)
sa
argocd-manager
rb
argocd-manager
crd
Applicatoin
crd
Project
Namespace Creator
namespaces
crb
argocd-watcher
c.role
argocd-watcher
c.role
argocd-manager
Ansible
kubernetes-ng clusters