



// INTRODUCTION TO GITOPS – A NEW AGE OF AUTOMATION?

Johannes Schnatterer, Cloudogu GmbH

 @jschnatterer

Version: 202104261633-caa70e0



Agenda

- What is GitOps?
- Where can it be used?
- How can it be used?
- What challenges arise?

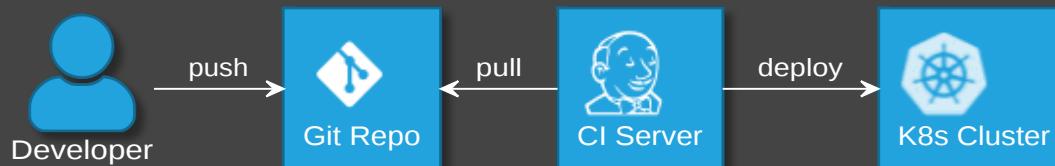
What is GitOps?

- Operating model
- Term (August 2017):

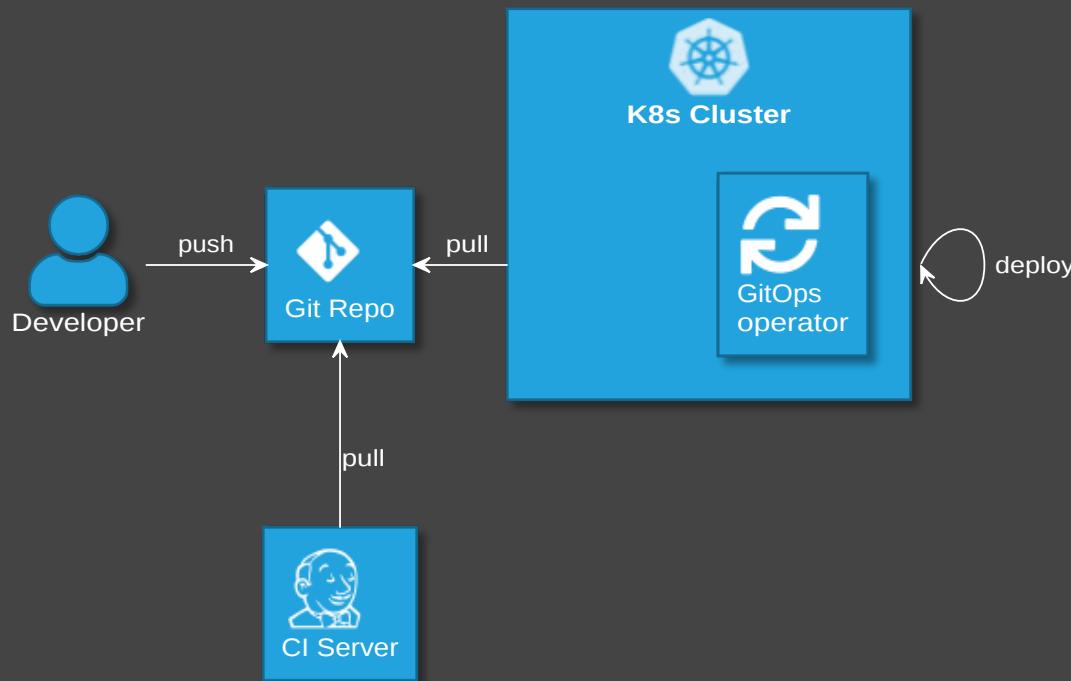
Use developer tooling to drive operations

 weave.works/blog/gitops-operations-by-pull-request

"Classic" Continuous Delivery ("CIOps")



GitOps



GitOps Principles

- 1 The principle of declarative desired state
- 2 The principle of immutable desired state versions
- 3 The principle of state reconciliation
- 4 The principle of operations through declaration

沩 WIP!

GH github.com/gitops-working-group/gitops-working-group/pull/48

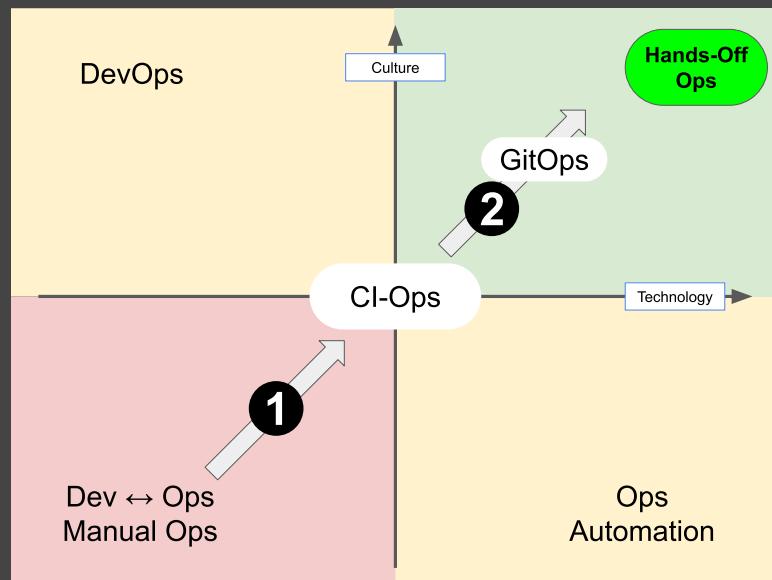
MD hackmd.io/arwvV8NUQX683uBM3HzyNQem



GitOps vs DevOps

- DevOps is about collaboration of formerly separate groups (mindset)
- GitOps focuses on ops (operations model)
- GitOps can be used with or without DevOps

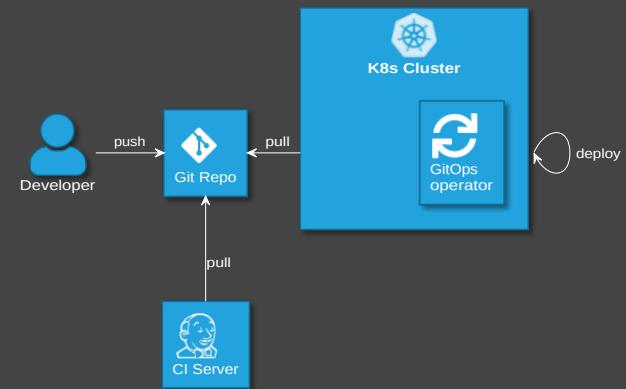
"The right way to do DevOps" (Alexis Richardson)



- youtu.be/lvLqJW0ixDI
- [\(iX 4/2021\)](https://heise.de/select/ix/2021/4/2032116550453239806)
- schlomo.schapiro.org

Advantages of GitOps

- (Almost) no access to cluster from outside
- No credentials on CI server
- Forces 100% declarative description
 - auditable
 - automatic sync of cluster and git
- Enterprise: Accessing git is simpler
(no new firewall rules)



A photograph of a blue claw hammer resting on a light-colored wooden surface. Several metal nails are scattered around the hammer, some partially driven into the wood. The lighting creates strong shadows, emphasizing the texture of the wood and the metallic surfaces.

What can GitOps be used for?



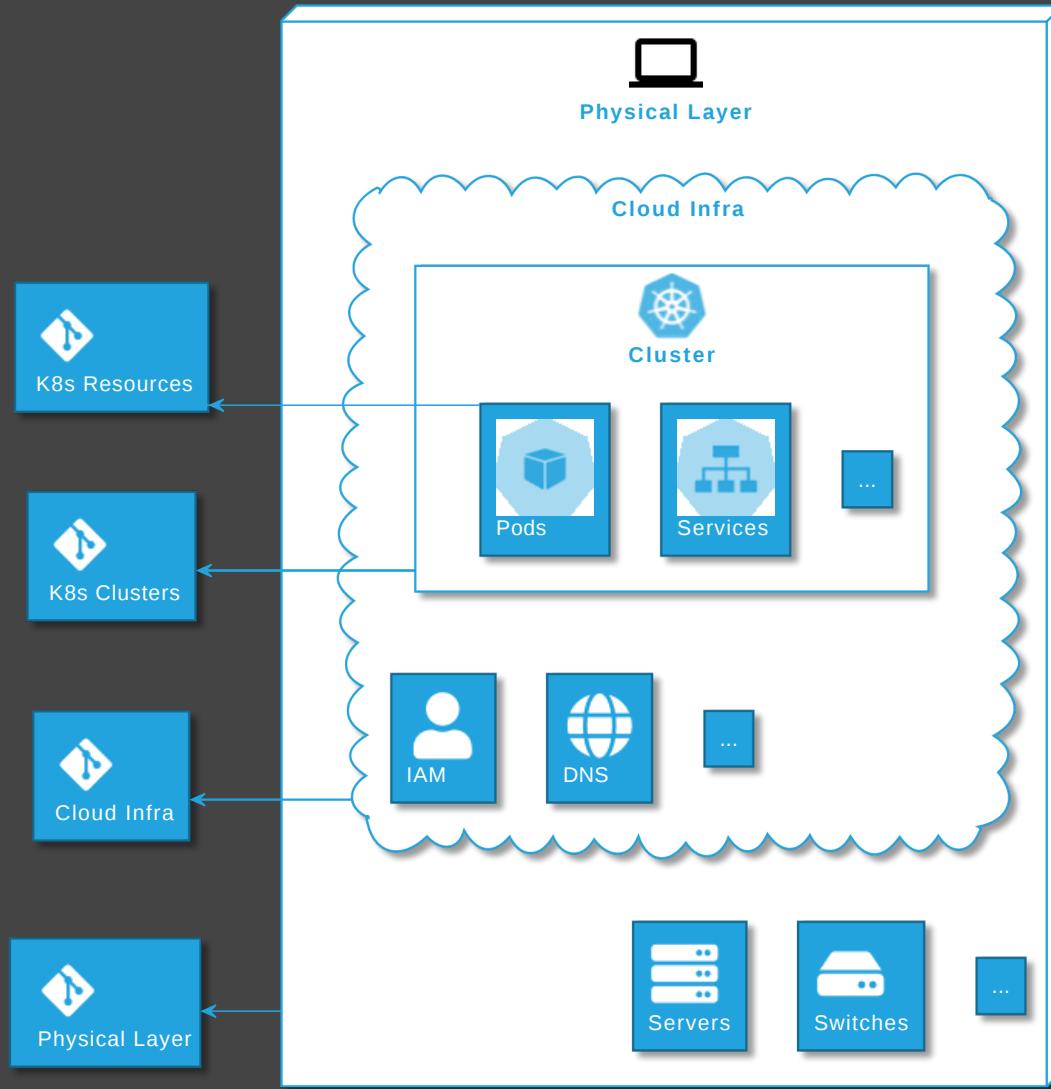
GitOps History in a nutshell

- grew up operating applications on Kubernetes,
- is now rising above it, operating clusters and other (cloud) infrastructure

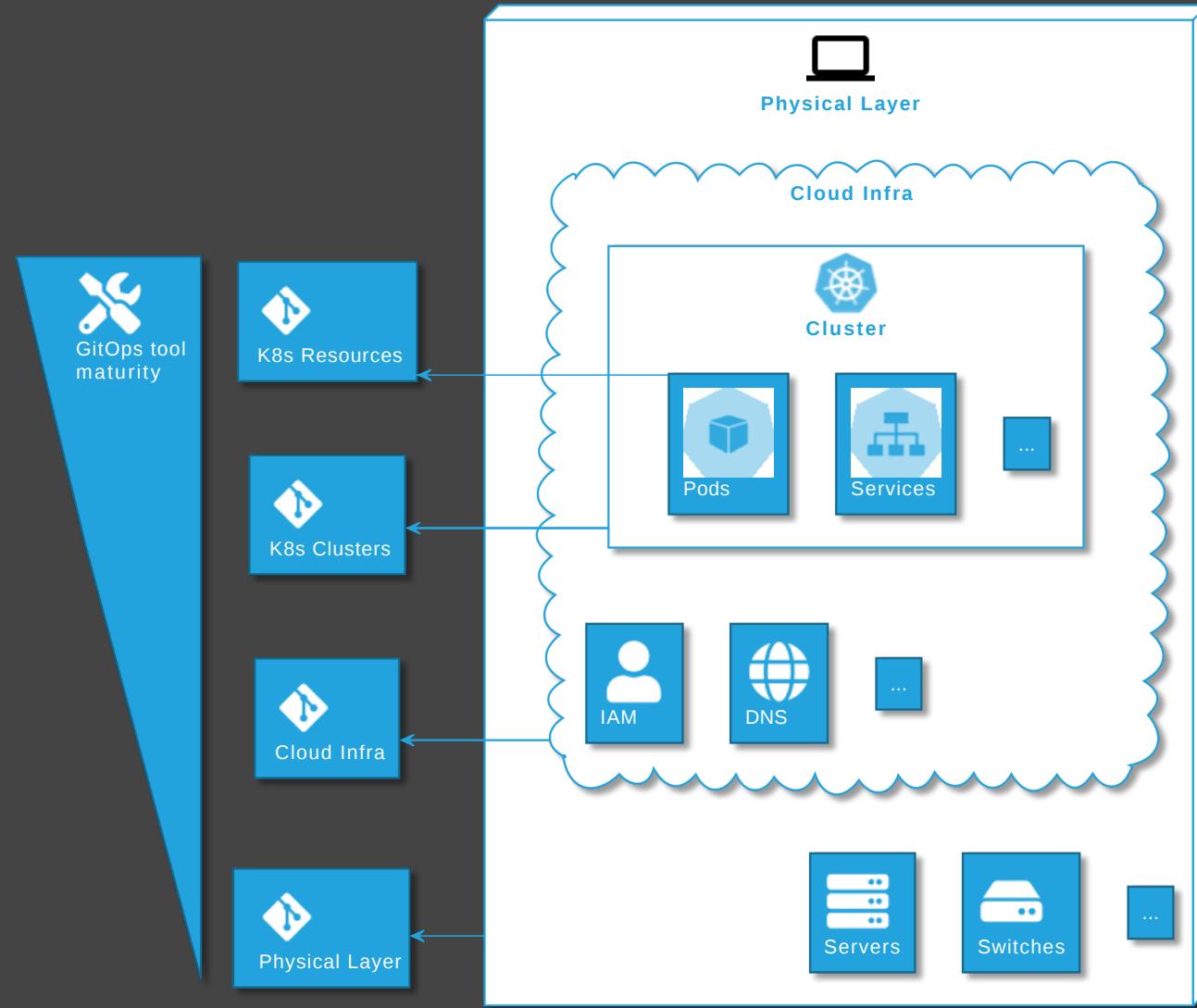
More on the history of GitOps:

🎥 <https://youtu.be/lvLqJWOixDI>

A GitOps Dream



GitOps reality

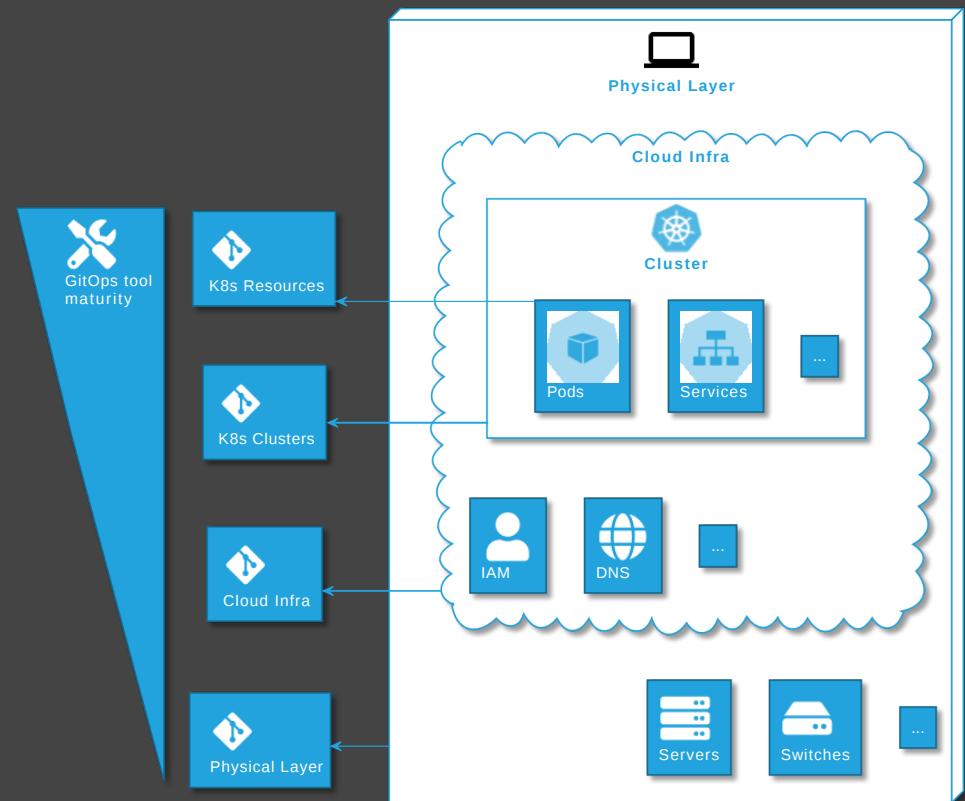




How can GitOps be used? Tools

Categories

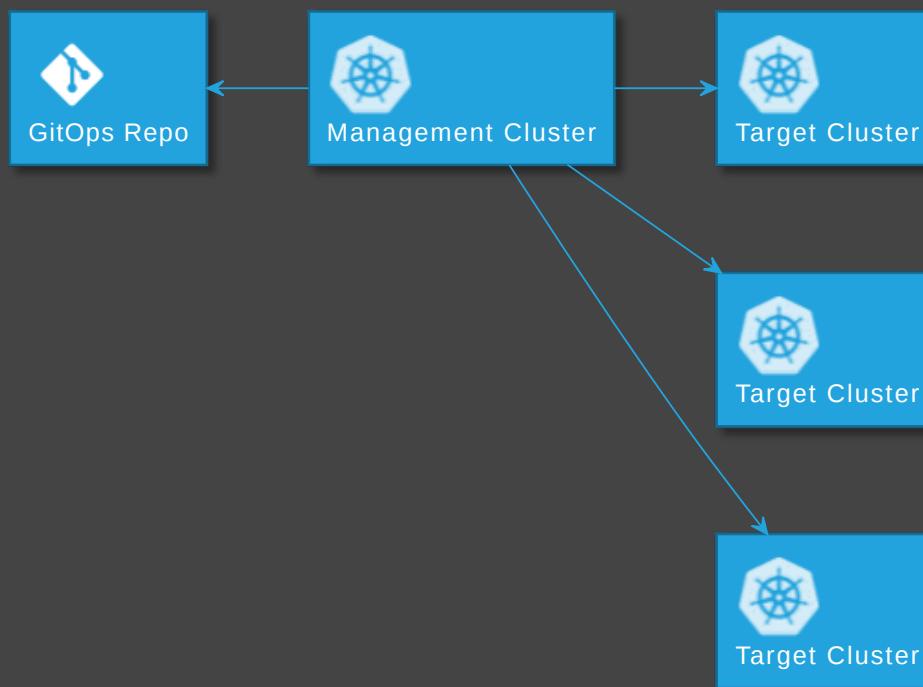
- Tools for Kubernetes AppOps
- Tools for Kubernetes ClusterOps
- Tools Close to Infrastructure
 - with or
 - without Kubernetes
- Supplementary GitOps tools



GitOps Tools for Kubernetes AppOps



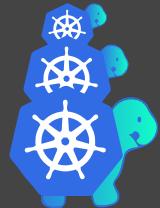
Operate Kubernetes with Kubernetes



GitOps Tools for Kubernetes ClusterOps



+



|



+

Operator

- [hashicorp/terraform-k8s](#)
- [rancher/terraform-controller](#)
-

Tools Close to Infrastructure

- with Kubernetes



|



+ Operator

- without Kubernetes



Supplementary GitOps tools

Secrets

-  [bitnami-labs/sealed-secrets](#)
-  [mozilla/sops](#) + K8s integration
 -  [isindir/sops-secrets-operator](#)
 -  [jkroepke/helm-secrets](#) (plugin)
 - flux v2 (native support)
-  [Soluto/kamus](#)
- Operators for Key Management Systems
 -  [external-secrets/kubernetes-external-secrets](#)
 -  [ContainerSolutions/externalsecret-operator](#)
 -  [ricoberger/vault-secrets-operator](#)

Others

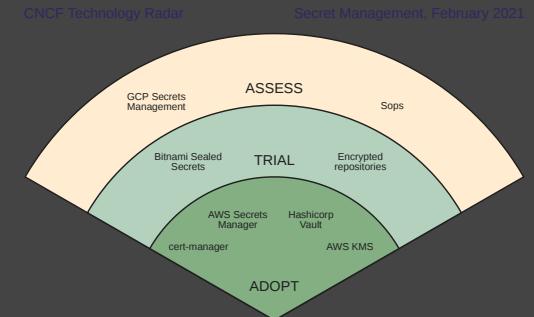
- Deployment Strategies - Progressive Delivery



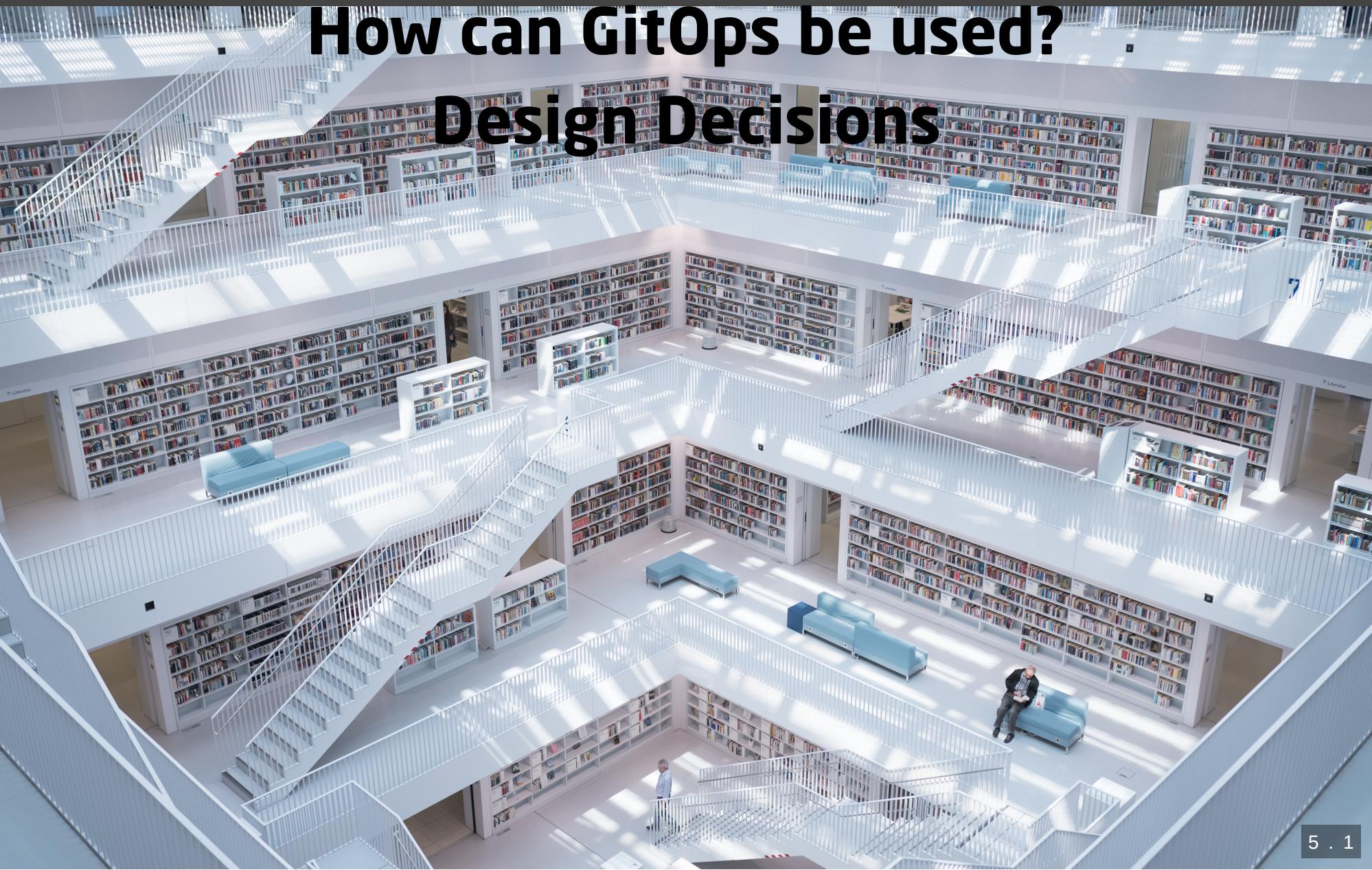
- Backups
- Horizontal Pod Autoscaler
- ...

See also

-  cloudogu.com/blog/gitops-tools (iX 4/2021)
 - General tool comparison,
 - tips on criteria for tool selection,
 - comparison of ArgoCD v1 and Flux v2
-  radar.cncf.io/2021-02-secrets-management
-  weaveworks/awesome-gitops
-  gitops.tech



How can GitOps be used? Design Decisions



- Implementing stages
- Role of CI server
- Number of Repos
- ...

Implementing stages

Idea 1: Staging Branches

- Develop → Staging
- Main → Production



Logic for branching complicated and error prone (merges)

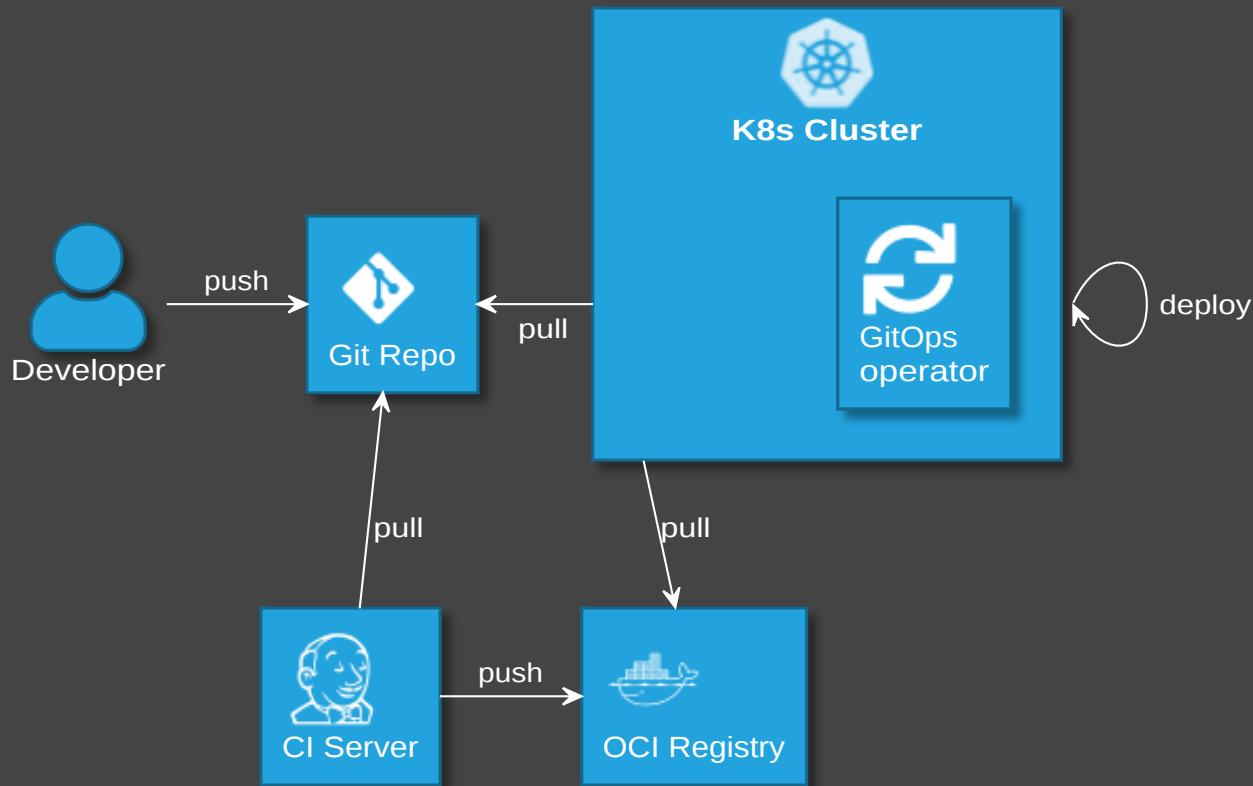
Idea 2: Staging folders

- On the same branch: One folder per stage
- Process: Just commit to staging folder, create PRs for prod
- Risky, but can be automated



- Logic for branching simpler
- Supports arbitrary number of stages

Role of CI server

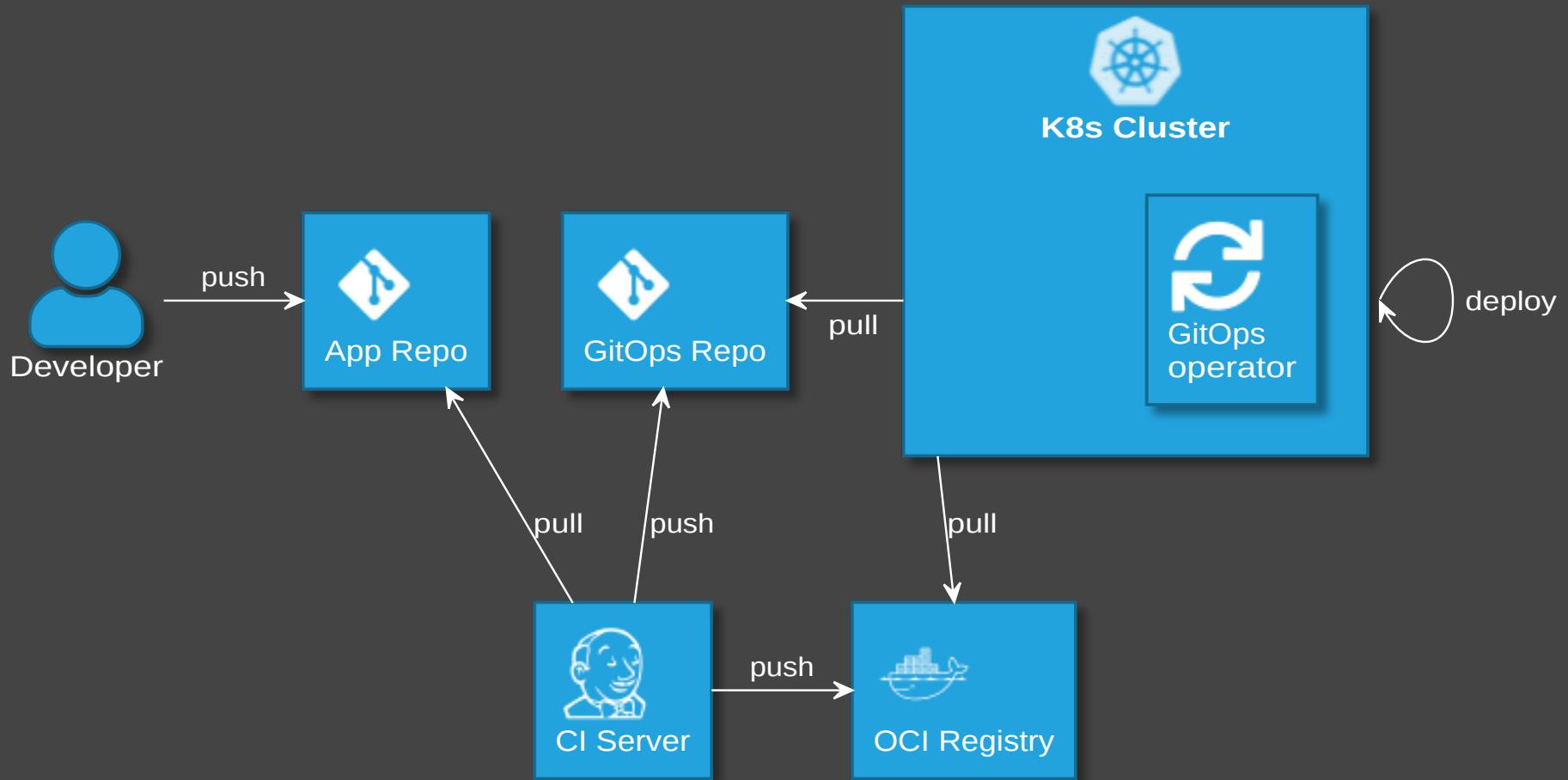


Application repo vs GitOps repo

- Good practice: Keeping everything in app repo (code, docs, infra)
- GitOps: Put infra in separate repo!
 - Advantage: All cluster infra in one repo
 - Disadvantages:
 - Separated maintenance & versioning off app and infra code
 - Review spans across multiple repos
 - Local dev more difficult

Can't we have both?

Yes, we can! Using a CI-Server



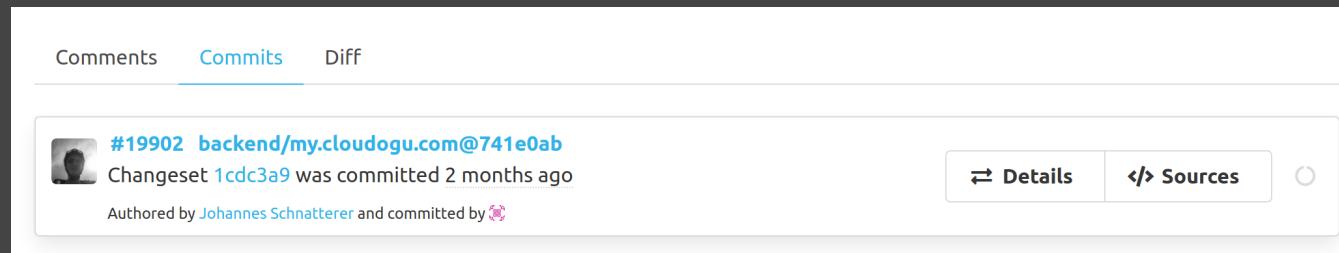
Disadvantages

- Complexity in CI pipelines → efforts for development
 - A lot can go wrong. Examples
 - Git Conflicts caused by concurrency
 - Danger of inconsistencies
- Recommendation: Use a plugin or library

Example:  [clodogu/gitops-build-lib](#) 

Advantages

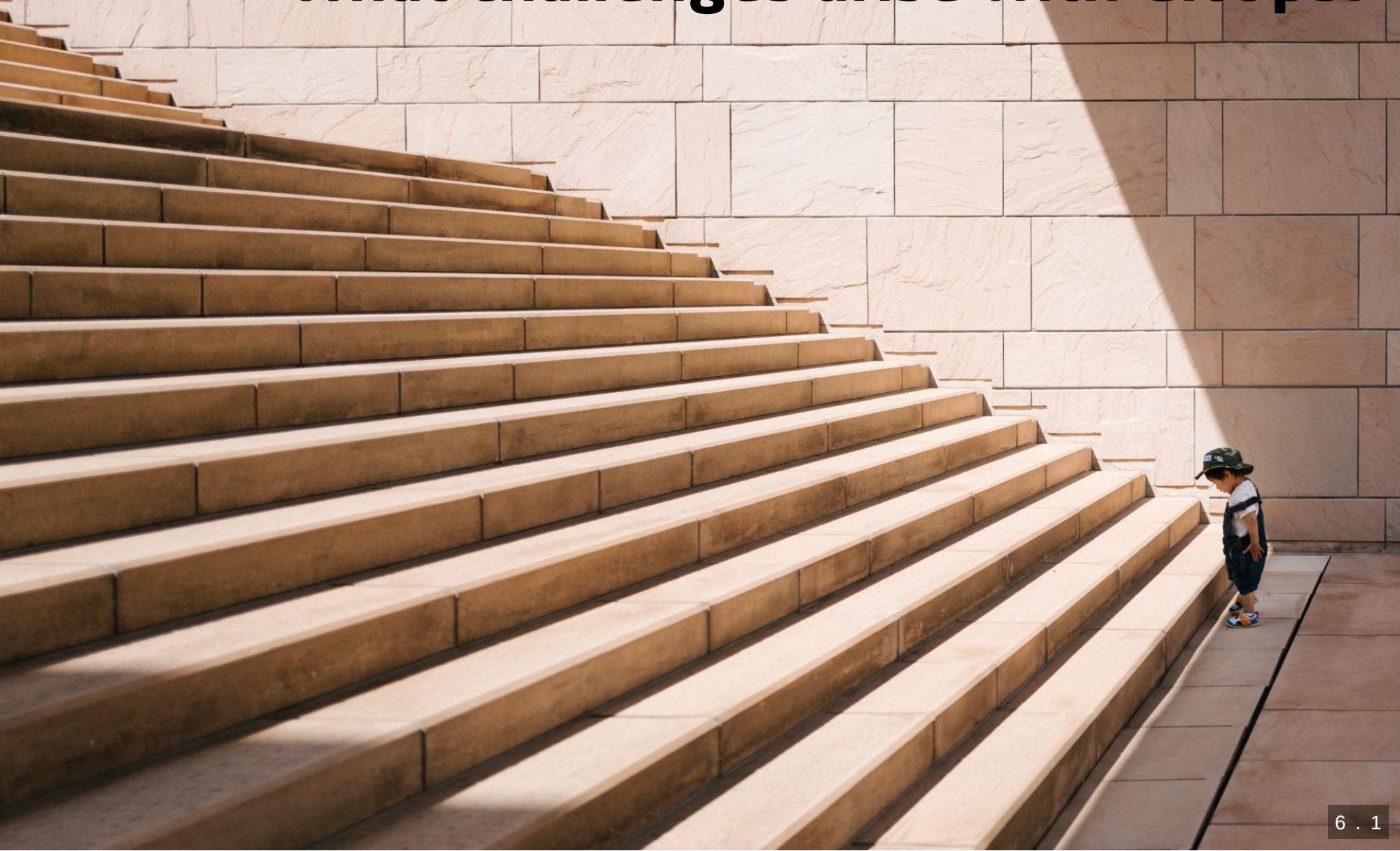
- Fail early: static YAML analysis on CI server,
e.g. yamlint, kubeval, helm lint
- Automated staging (e.g. PR creation, namespaces)
- Use IaC for local dev
- Write config files not inline YAML
 - ➔ Automatically converted to configMap
- Simplify review by adding info to PRs



Demo

 [cloudogu/k8s-gitops-playground](#)

What challenges arise with GitOps?



More Infra ...

- GitOps Operator: One or more custom controllers
- Helm, Kustomize Controllers
- Operators for Supplementary tools (secrets, etc.)
- Monitoring/Alerting systems
- ...

... higher cost

- Maintenance/patching (vendor dependency)
- Resource consumption
- Error handling
 - failing late and silently
 - monitoring/alerting required
 - reason might be difficult to pinpoint
 - operators cause alerts (OOM errors, on Git/API server down, etc.)

Day two questions

- POC is simple
- Operations in prod has its challenges
 - How to structure repos?
 - How to realize staging?
 - How to delete resources?
 - How to realize local dev env?
 - ...

How to delete resources?

- “garbage collection” (Flux) / “resource pruning” (ArgoCD)
disabled by default
-  Enable from the start → avoid manual interaction

Local development

- Option 1: Deploy GitOps operator and Git server on local cluster
 - ➡ complicated
- Option 2: Just carry on without GitOps. Possible when IaC remains in app repo

CONCLUSION



Personal Conclusion

After migrating to and operating with GitOps in production for > 1 year

- Smoother CI/CD,
 - *everything* declarative
 - faster deployment
 - force sync desired state  actual state
- But: security advantages only when finished migration
- A new age of automation? Not yet, but lots of innovation ahead!

GitOps experience distilled

- + Has advantages, once established
- Mileage for getting there may vary

Adopt?

- Greenfield
 - Kubernetes AppOps: Definitely
 - Cloud Infra: Depends
- Brownfield: Depends

TECHNOLOGY RADAR

Download Subscribe Search Build your Radar About



Techniques

GitOps

Published: Apr 13, 2021

APR
2021

HOLD ?

We suggest approaching **GitOps** with a degree of care, especially with regard to branching strategies. GitOps can be seen as a way of implementing **infrastructure as code** that involves continuously synchronizing and applying infrastructure code from **Git** into various environments. When used with a "branch per environment" infrastructure, changes are promoted from one environment to the next by merging code. While treating code as the single source of truth is clearly a sound approach, **we're seeing branch per environment lead to environmental drift and eventually environment-specific configs as code merges become problematic or even stop entirely**. This is very similar to what we've seen in the past with **long-lived branches with GitFlow**.



thoughtworks.com/radar/techniques/gitops

Johannes Schnatterer, Cloudogu GmbH

 cloudogu.com/gitops

-  GitOps Resources (intro, tool comparison, etc.)
-  Links to GitOps Playground and Build Lib
-  Discussions
-  Training



Slides

