



# // GITOPS REPO STRUCTURES AND PATTERNS

Johannes Schnatterer, Cloudogu GmbH

 [@schnatterer@floss.social](mailto:@schnatterer@floss.social)

 [in/jschnatterer](https://www.linkedin.com/in/jschnatterer)

 [@jschnatterer](https://twitter.com/jschnatterer)

Version: 202311141844-4f0e8a0

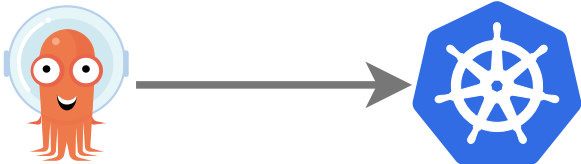
# Categories of patterns

- **Operator deployment:** GitOps operators ↔ Clusters/Namespaces
- **Repository:** How many repos?
- **Promotion:** How to model environments/stages?
- **Wiring:** Bootstrapping operator, linking repos and folders

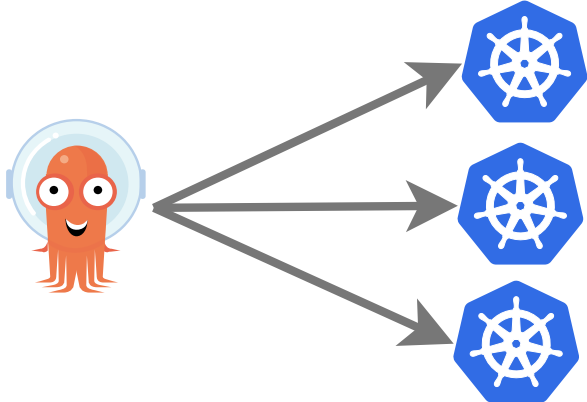
# GitOps Operator deployment patterns

How many GitOps operators per cluster?

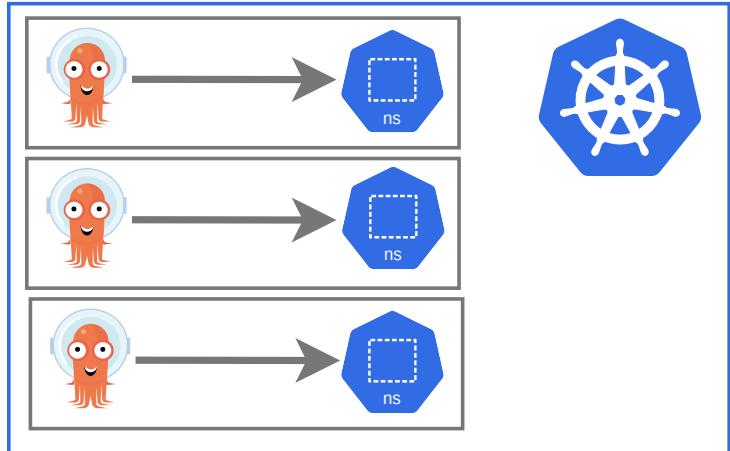
**Instance per Cluster**



**Hub and Spoke**



**Instance per Namespace**



# Repository patterns

How many config repos?

- **Monorepo**
- **Repo per Team / Tenant**
- **Repo per App**
  - Repo Separation
  - Config replication
  - Repo pointer
  - Config Split
- **Repo per environment** 🕒

💡 Can be mixed 🗑️

# Repository types

## Config repo

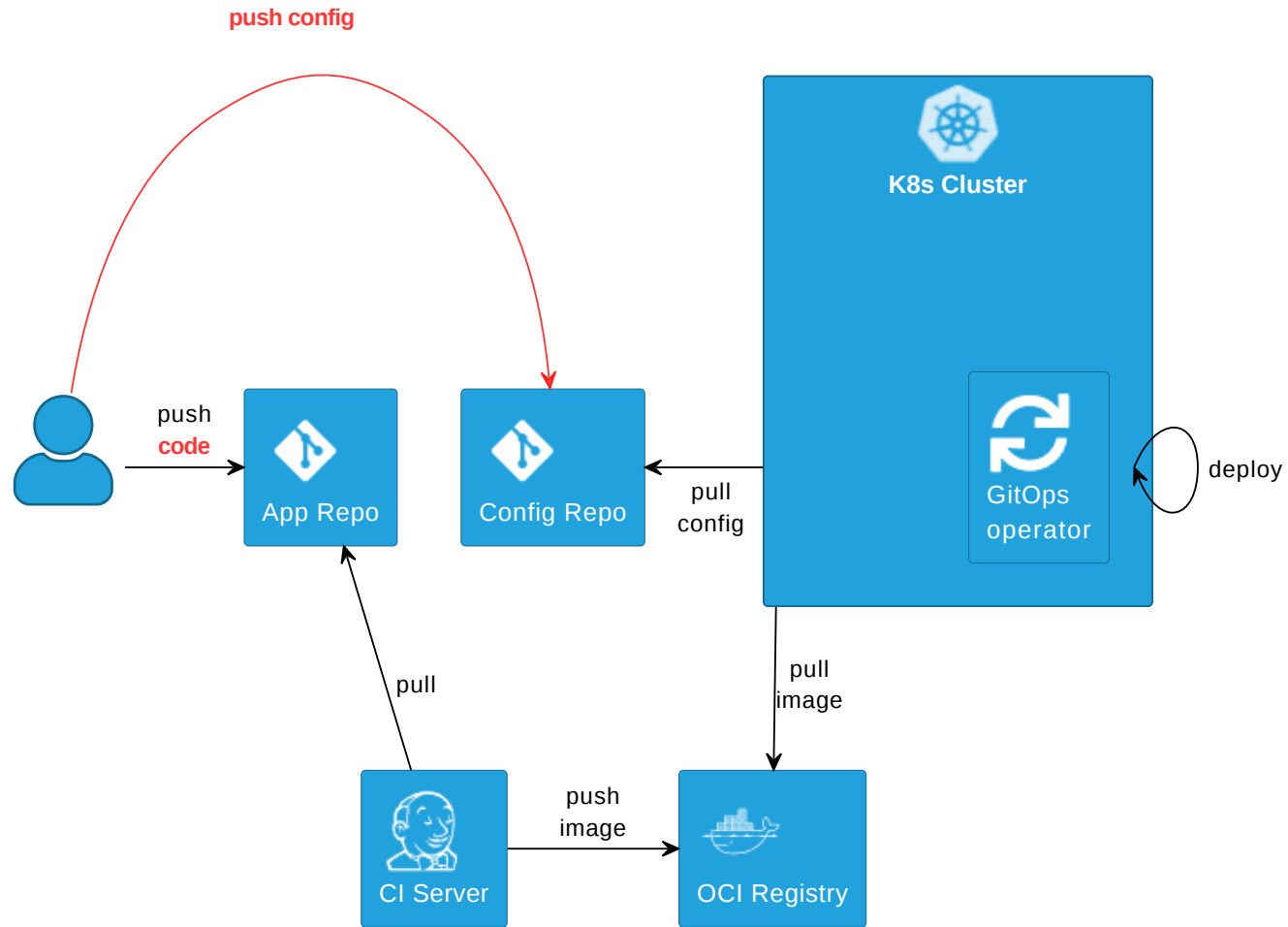
## App repo

Content	Config/Manifests/YAMLs (IaC)	Application source code
Synonyms	<ul style="list-style-type: none"><li>• GitOps repo</li><li>• Infra repo</li><li>• Environment repo</li><li>• Payload repo</li></ul>	<ul style="list-style-type: none"><li>• Source code repo</li><li>• Source repo</li></ul>

## Example



# Repo Separation



Recommendation: Keep config separate from code

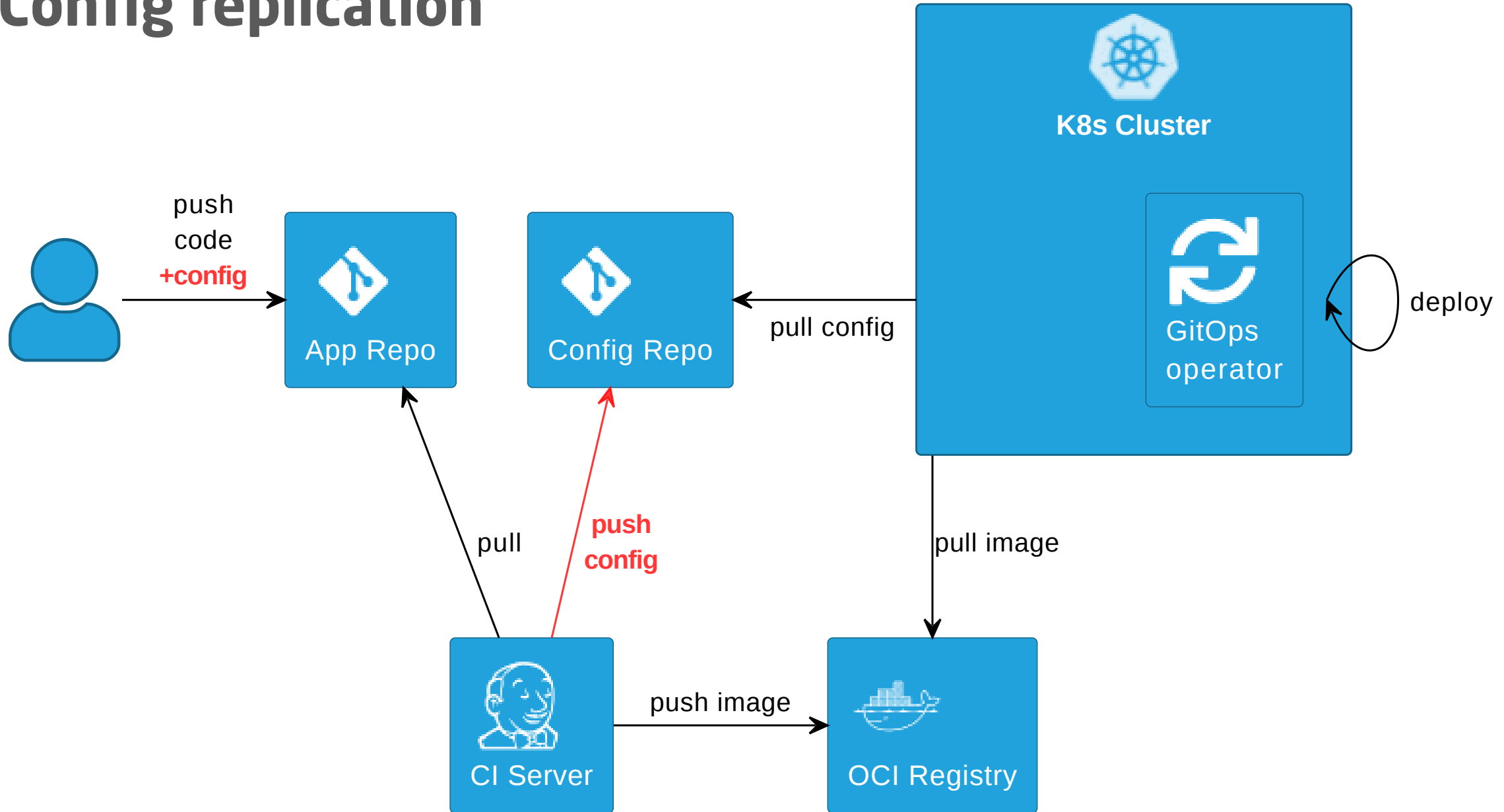
 [argo-cd.readthedocs.io/en/release-2.8/user-guide/best\\_practices](https://argo-cd.readthedocs.io/en/release-2.8/user-guide/best_practices)

## Disadvantages

- Separated maintenance & versioning of app and infra code
- Review spans across multiple repos
- Local dev more difficult
- No static code analysis on config repo

# How to avoid those?

# Config replication





# Advantages

- Single repo for development: higher efficiency
- Shift left: static code analysis + policy check on CI server,
- Automate config update (image tag + PR creation) 🕒
- Simplify review by adding info to PRs

Comments Commits Diff



**[production] #2** [argocd/petclinic-plain@f448c2b](#)

Changeset [c9e3bf1](#) was committed 5 minutes ago



Authored by [Johannes Schnatterer](#) and committed by

↔ Details

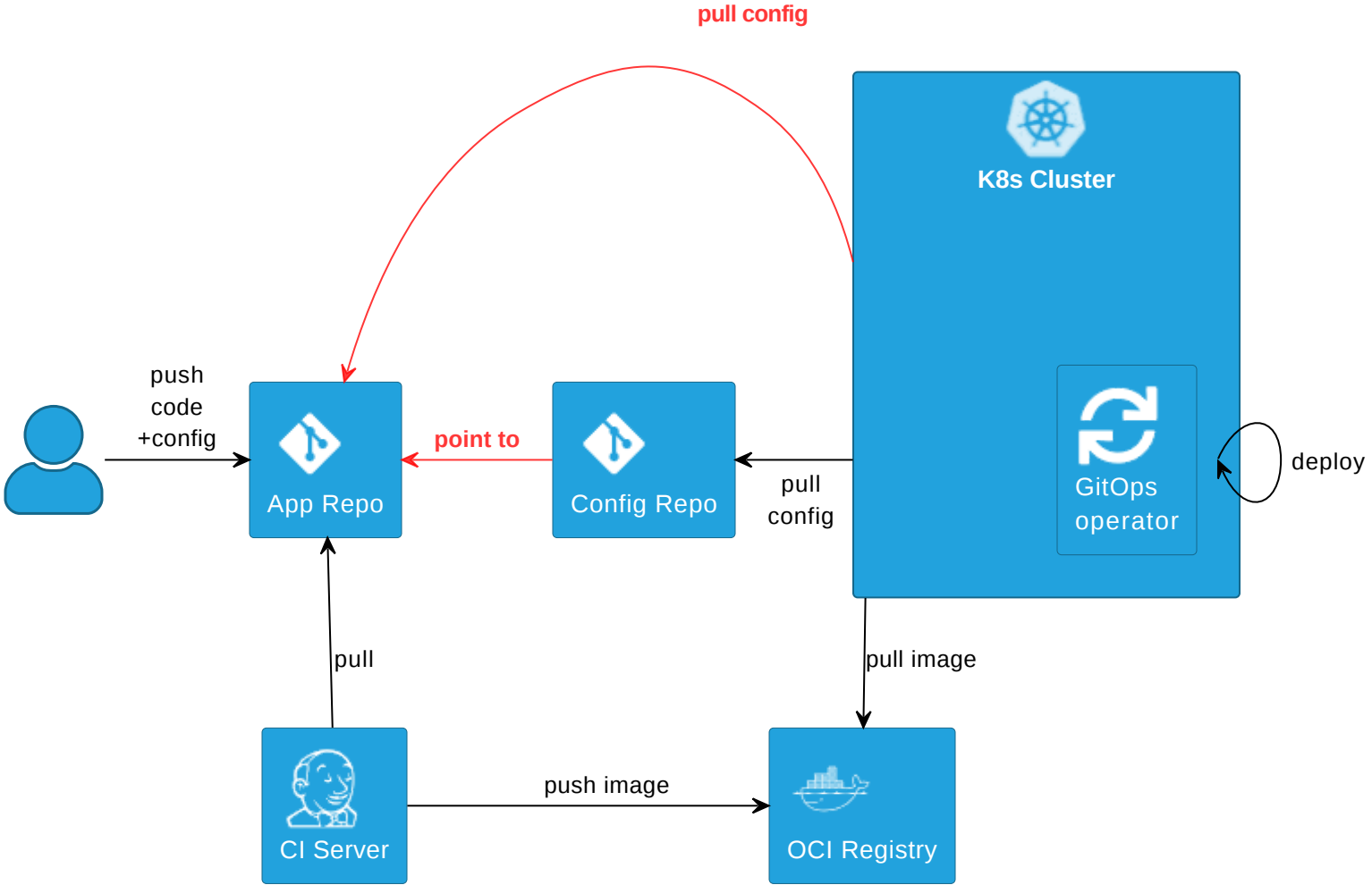
</> Sources



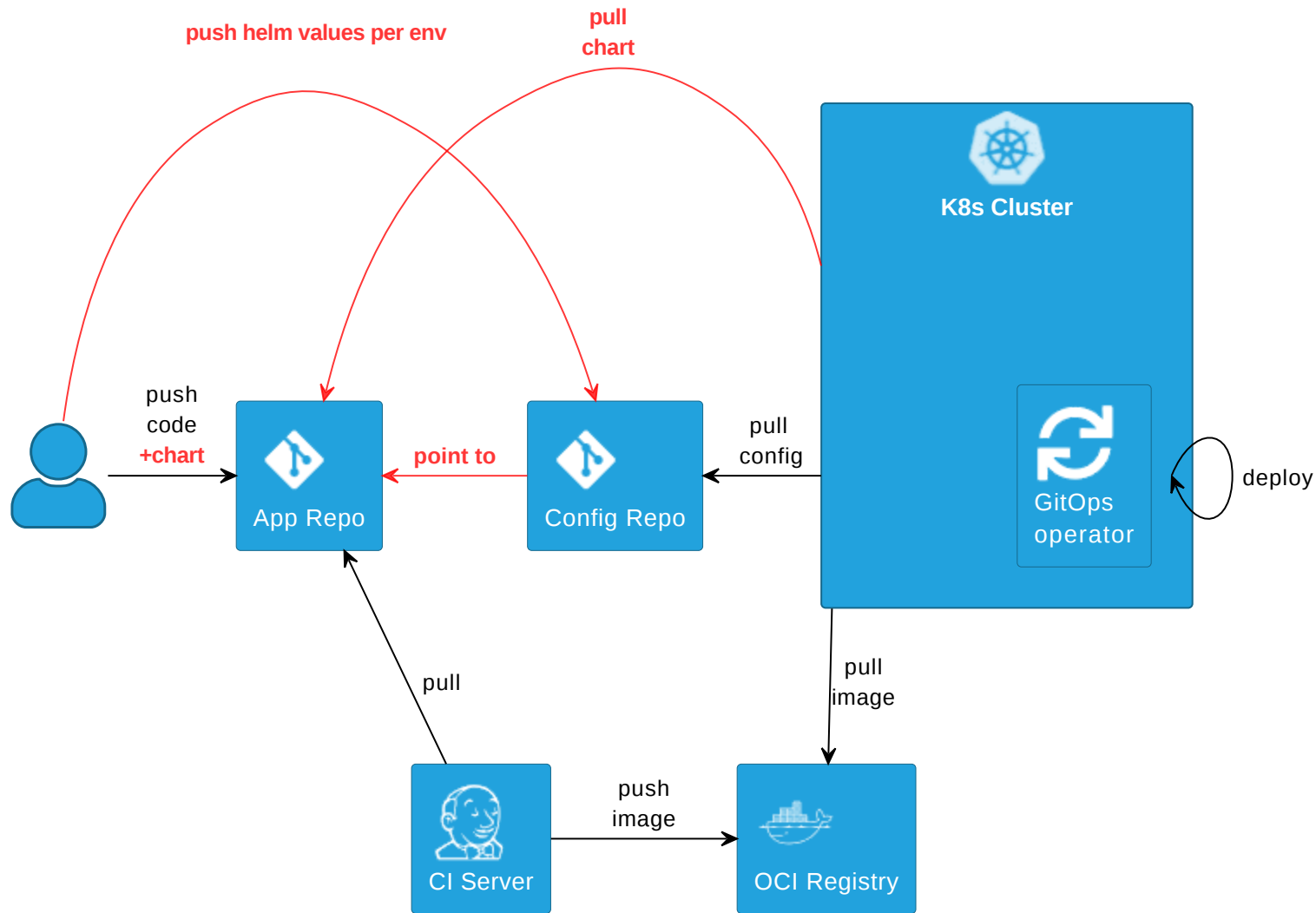
## Disadvantages

- Complexity in CI pipelines
  - ➔ Recommendation: Use a plugin or library, e.g.  
 [cloudogu/gitops-build-lib](https://github.com/cloudogu/gitops-build-lib) 
- Redundant config (app repo + config repo)

# Avoid Redundancy: Repo pointer



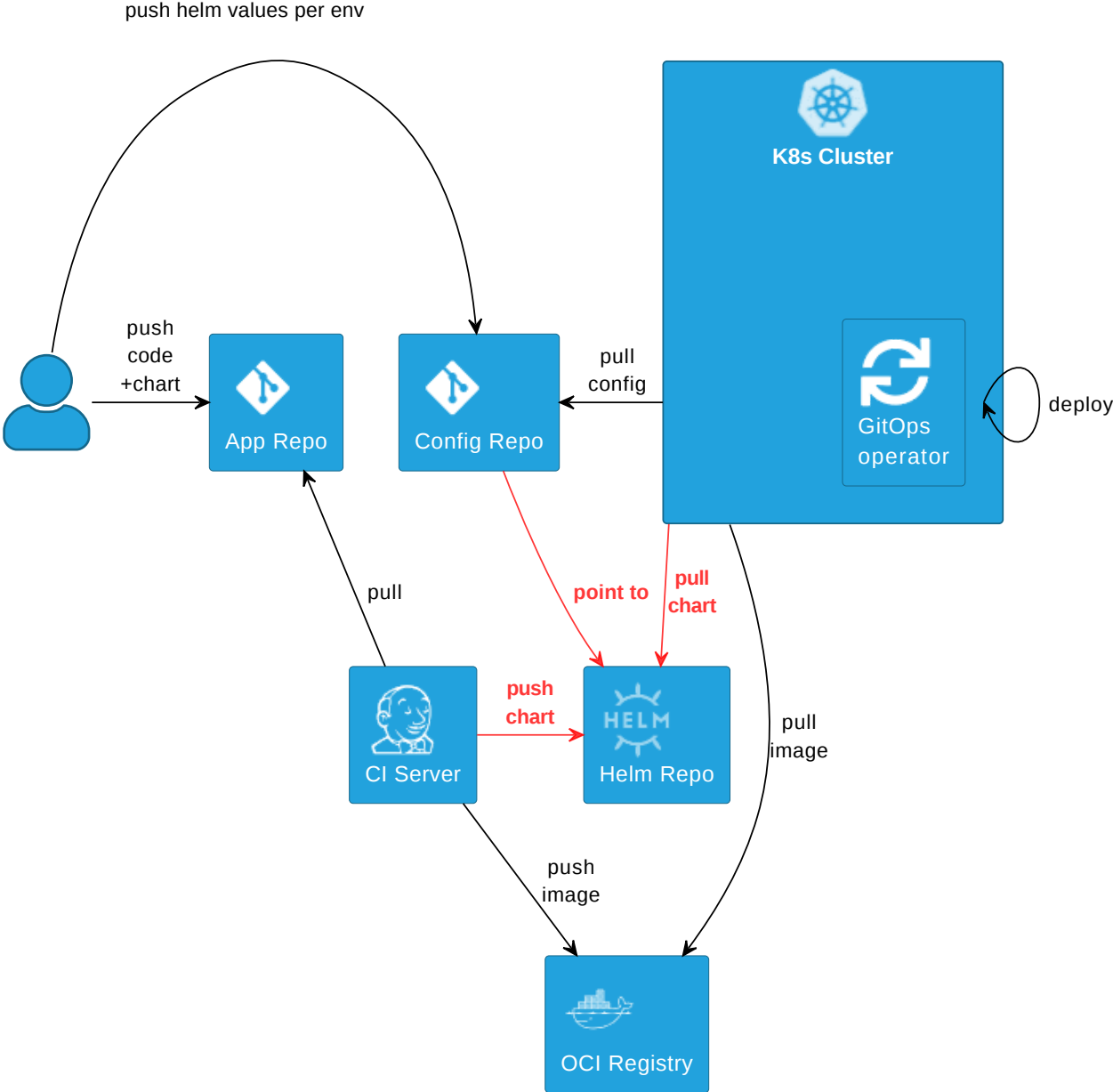
# Middle ground: Config Split

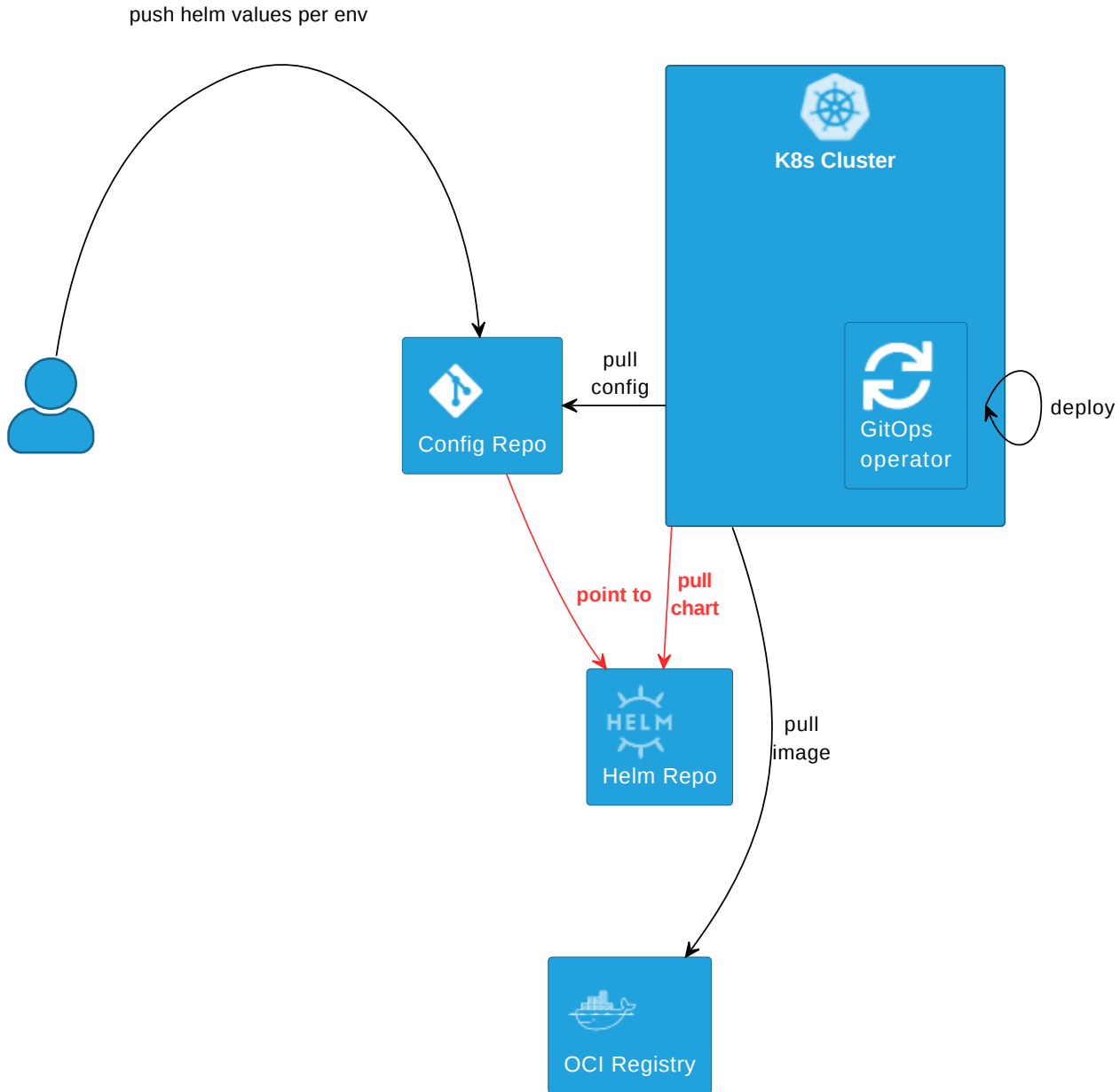


👉 HELM example

💡 Also works with 

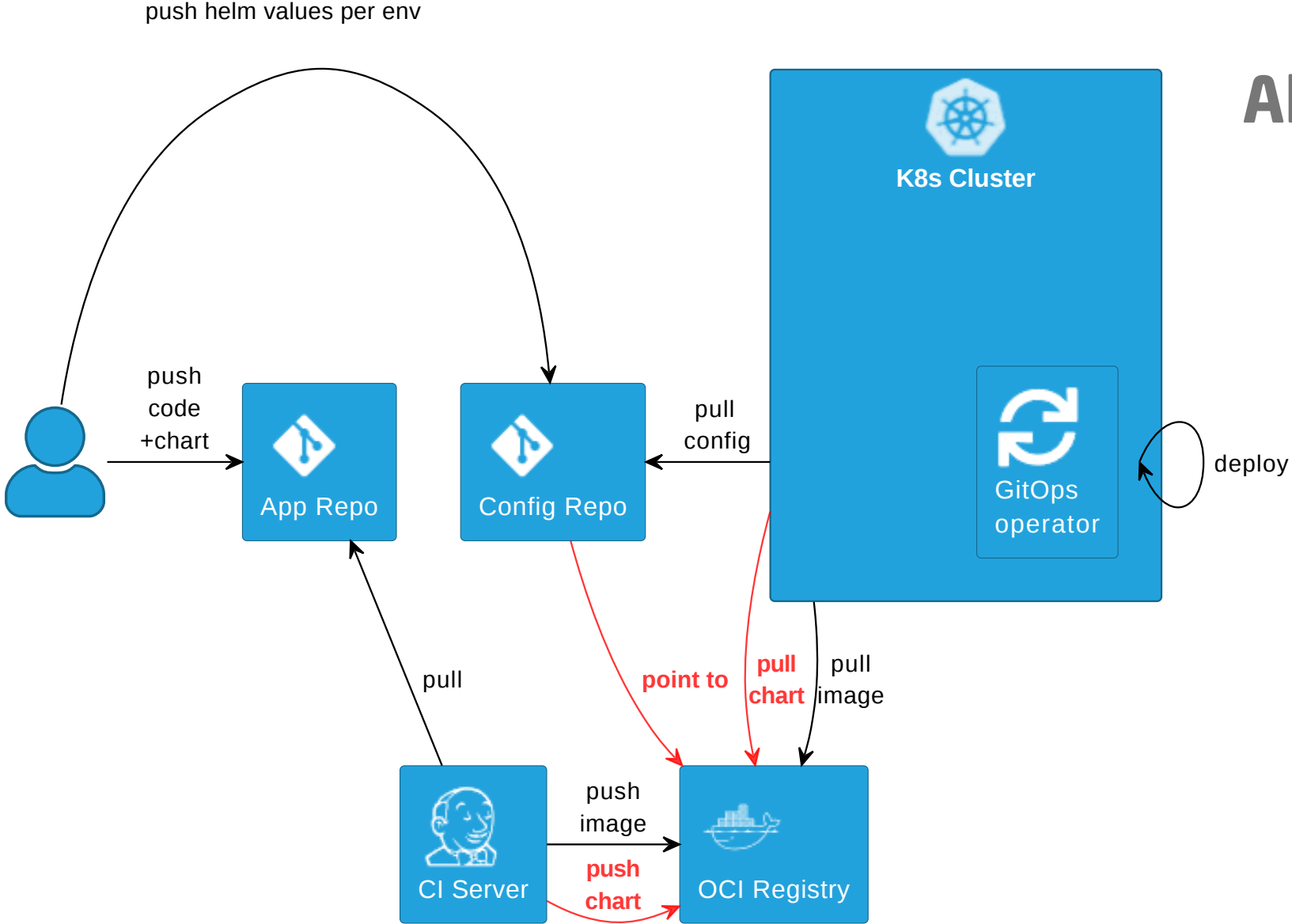
# Alternative: Helm repo



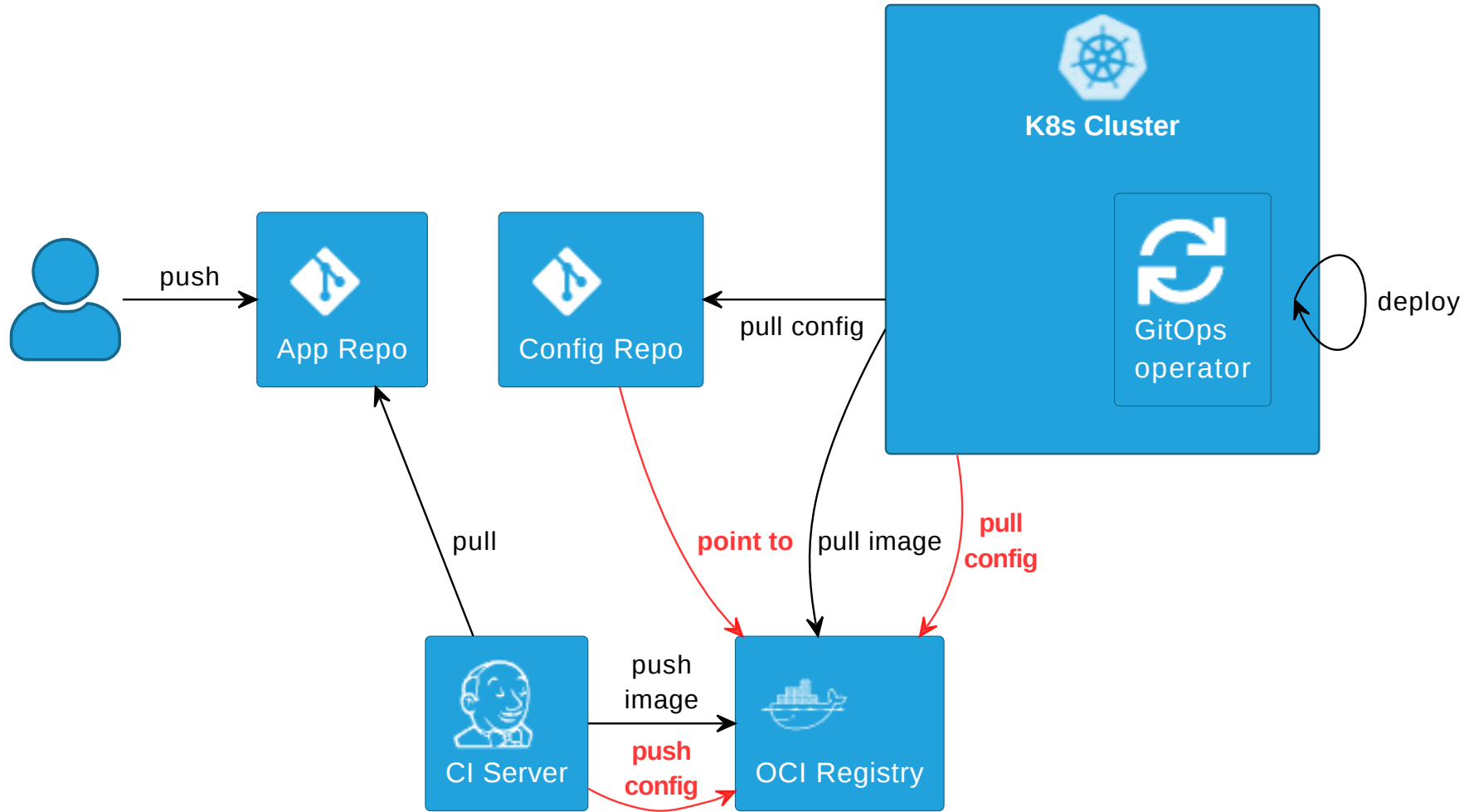


💡 Same pattern for 3rd-party apps

# Alternative 2: Helm in OCI



## Alternative 3: OCI artifacts











 [fluxcd.io/flux/cheatsheets/oci-artifacts](https://fluxcd.io/flux/cheatsheets/oci-artifacts)



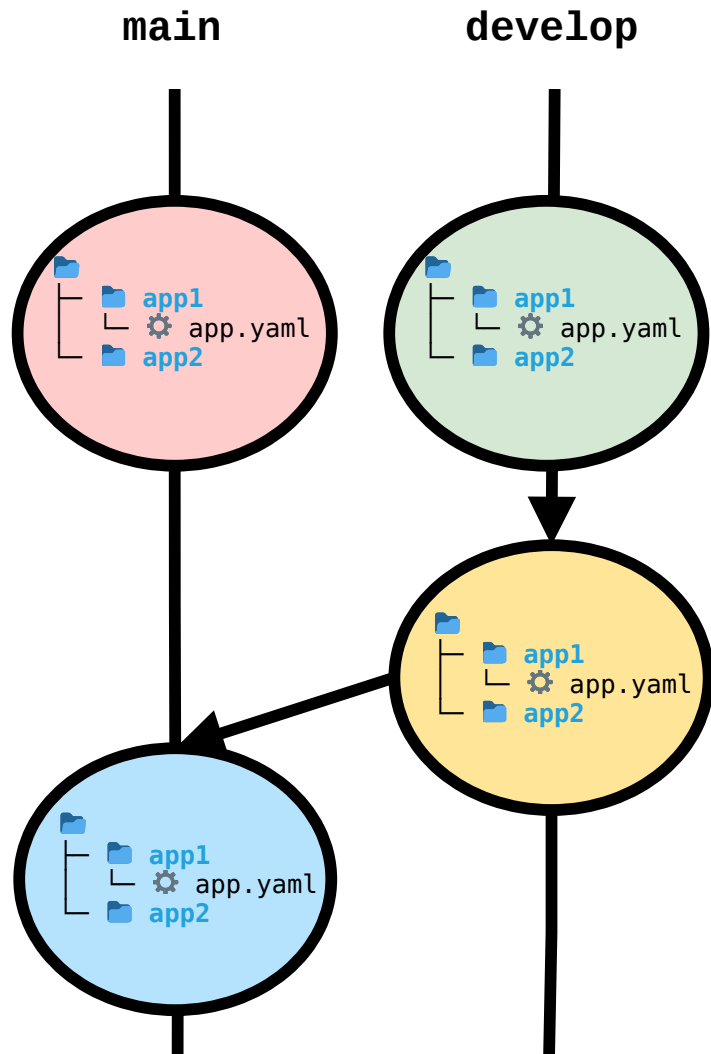
# Promotion patterns

How to model environments/stages?

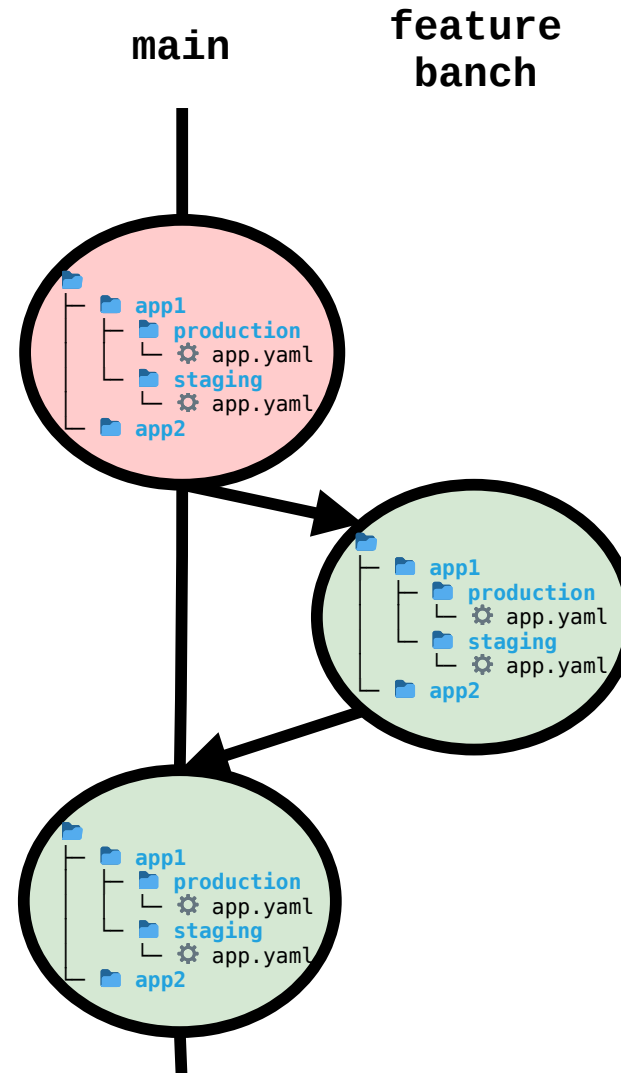
- **Branch per environment**
- **Folder/Directory per environment**
- **Repo per environment** (edge case)
-  **Preview environments**

	<b>Branch per env</b>	<b>Folder per env</b>
envs	permanent branches	trunk-based folders
mapping example	 develop  staging	 staging  Staging
	 main  production	 production  Production
promotion	merge	copy (+merge short-lived branches)

# Branch per env



# Folder per env



## Branch per env

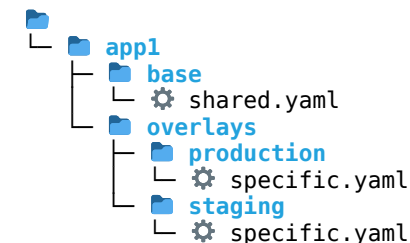
## Folder per env

pros

- Forces PRs
- Feels natural for devs

- Avoids conflicts/drift
- Copy vs cherry pick
- Scales with envs

CM tool support (DRY)



references

1

1, 2, 3, 4, ...

Branches = anti-pattern

# Repo per environment

Why would you want to use one repo per env?

- Access to folders more difficult to constrain than repos
- Organizational constraints, e.g.
  - "devs are not allowed to access prod"
  - security team needs to approve releases

 Repos more complicated than folders. Use only when really necessary.

## Preview environments

- An environment that is created with a pull request
- and deleted on merge/close

 `ApplicationSet`, using the `PullRequest` generator

 `GitOpsSets`  $\approx$  








# Implementing promotion






# Configuration Management tools

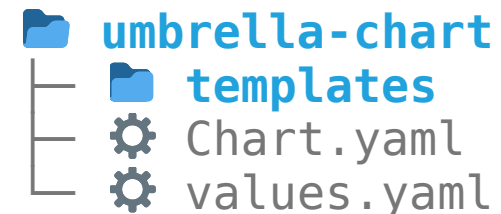
Tools for separating config of envs, keeping them DRY

- Kustomize

- plain  `kustomization.yaml`    "agnostic"
- ≠ Flux CRD  `Kustomization`
- `kustomize build / kubectl kustomize` via CI server 

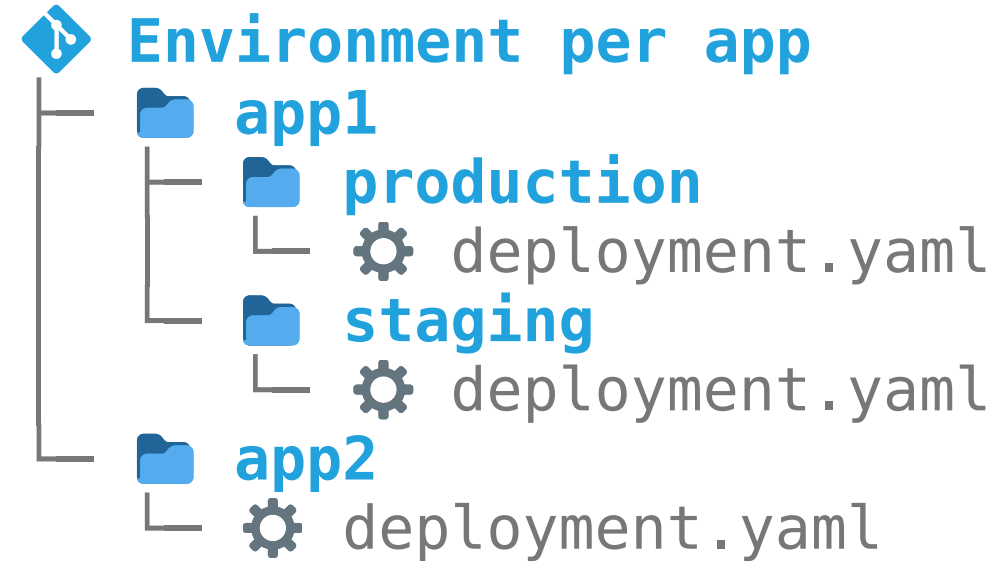
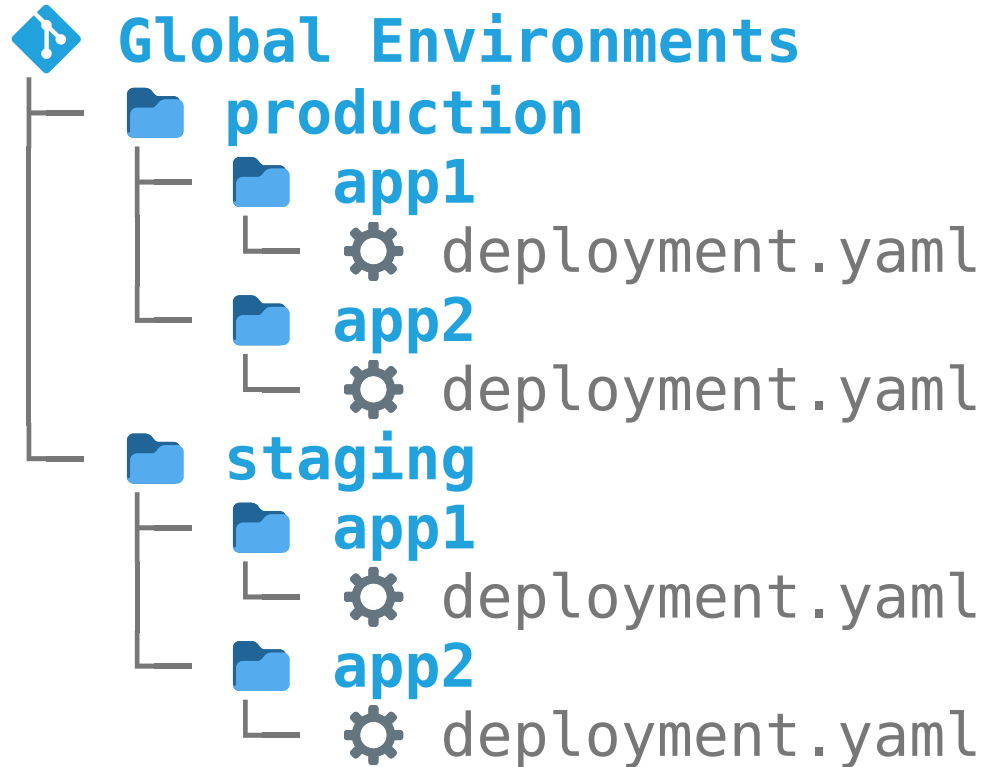
- Helm

- CRD ( `Application`,  `HelmRelease`)
-  Umbrella Chart 
- `helm template` via CI server 





# Global envs vs. env per app



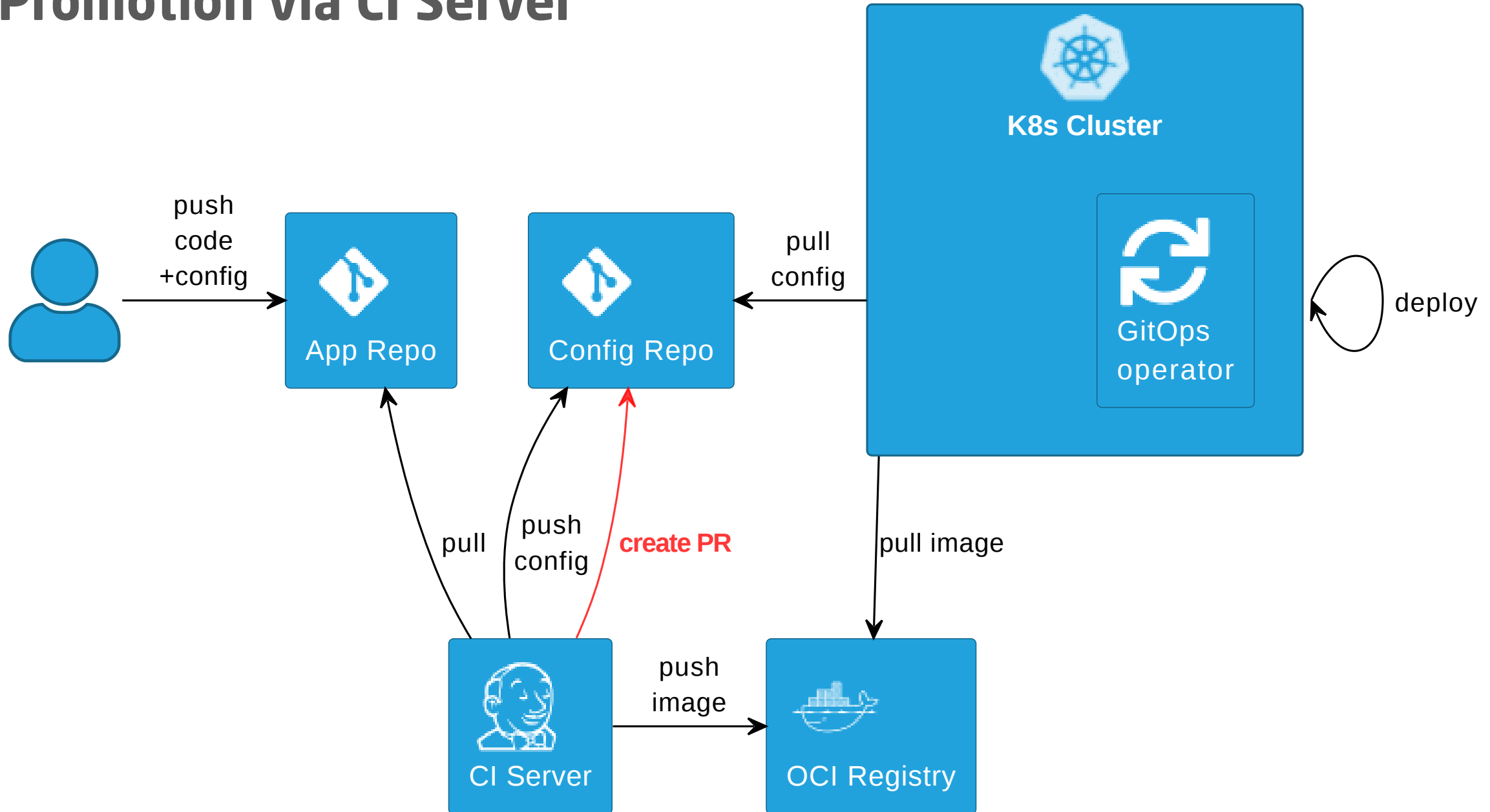
e.g. **Preview Envs**

## Config update

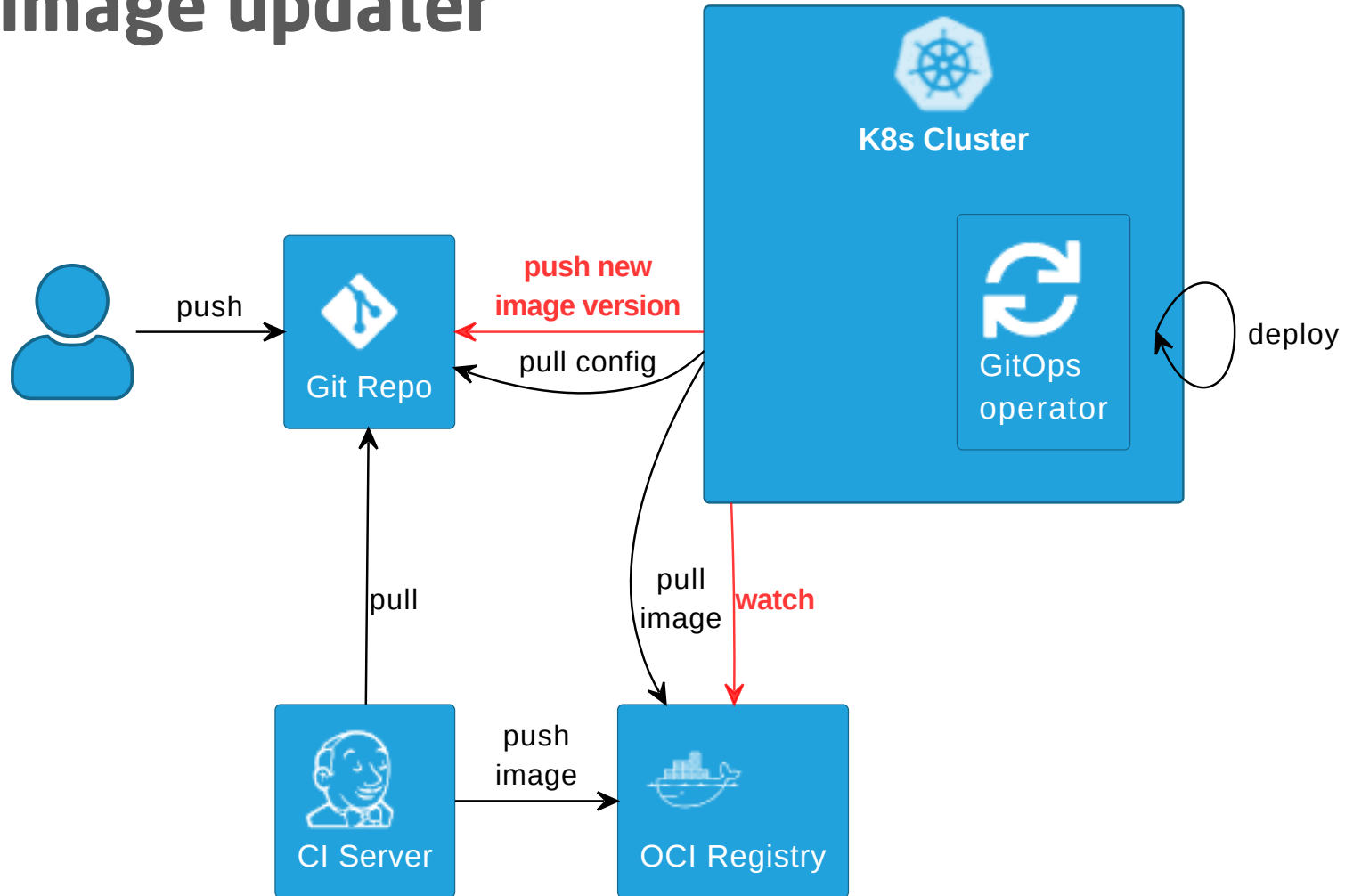
Who updates image version in config repo, creates branch and PR?

- **Manual:** Human pushes branch and create PR 🤔
- **CI Server:** Build job pushes branch, creates PR
- **Image Updater:** Operator pushes branch, create PR manually
- **Dependency Bot:** Bot pushes branch, creates PR

# Promotion via CI Server



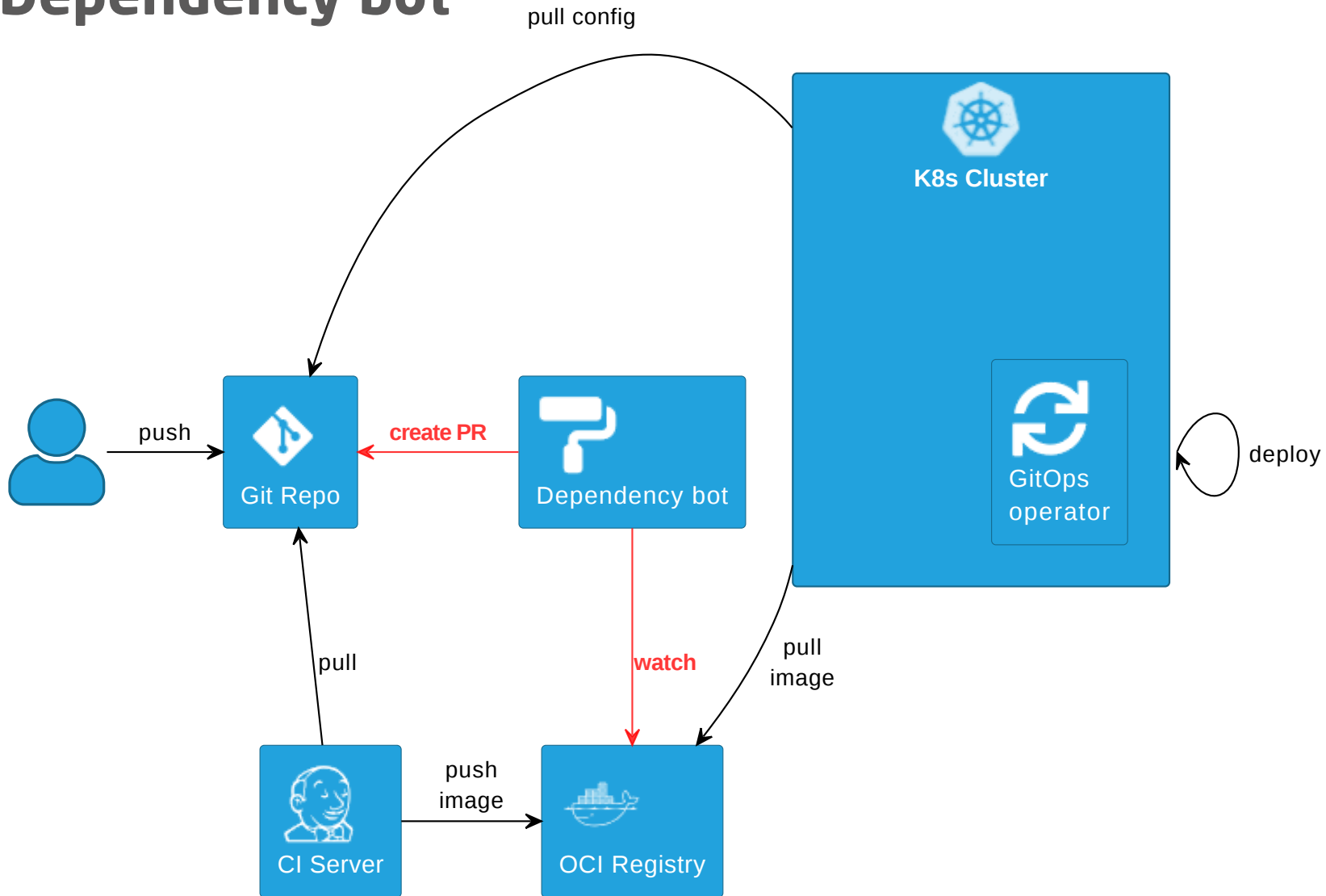
# Image updater



 [github.com/argoproj-labs/argocd-image-updater](https://github.com/argoproj-labs/argocd-image-updater)

 [fluxcd.io/docs/guides/image-update](https://fluxcd.io/docs/guides/image-update)

# Dependency bot



e.g.  [github.com/renovatebot/renovate](https://github.com/renovatebot/renovate)

# Pull Requests

GitOps - Operations by Pull Request

 [weave.works/blog/gitops-operations-by-pull-request](https://weave.works/blog/gitops-operations-by-pull-request)






But: avoid cargo cult

 PRs not mentioned in [principles](#)



# Wiring patterns

Wiring up operator, repos, folders, envs, etc.

- **Bootstrapping:** `kubectl`, operator-specific CLI
- **Linking/Grouping:**
  - Operator-specific CRDs
    -  `Kustomization`
    -  `Application`
  - **Nesting:**  App of Apps  
(same pattern with  `Kustomization`)
  - **Templating:**  `ApplicationSets` - folders, lists, config files, PRs

# Example + Demo



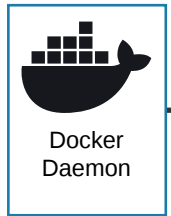


# GitOps playground

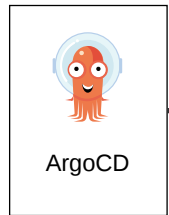
- **Repo pattern:** Per team 🗑️ per app
- **Operator pattern:** Instance per Cluster (Hub and Spoke)
- **Operator:** 🧑
- **Boostrapping:** Helm, kubectl
- **Linking:** 🧑 Application
- **Features:**
  - Operate ArgoCD with GitOps
  - Solution for cluster resources
  - Config update + replication via CI
  - Env per app pattern
  - Mixed repo patterns
- **Source:** 🔄 [cloudogu/gitops-playground](https://github.com/cloudogu/gitops-playground)

Developer's host

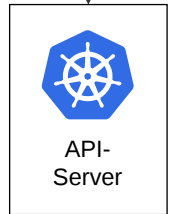
K3D Container



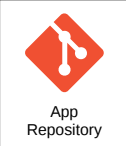
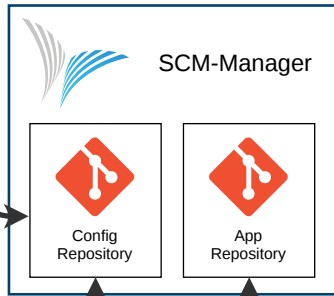
run



apply resources



pull



push config and create pull request



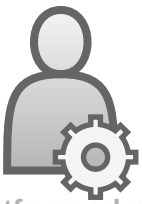
pull

image push

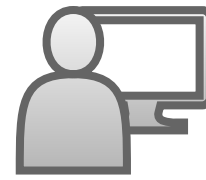




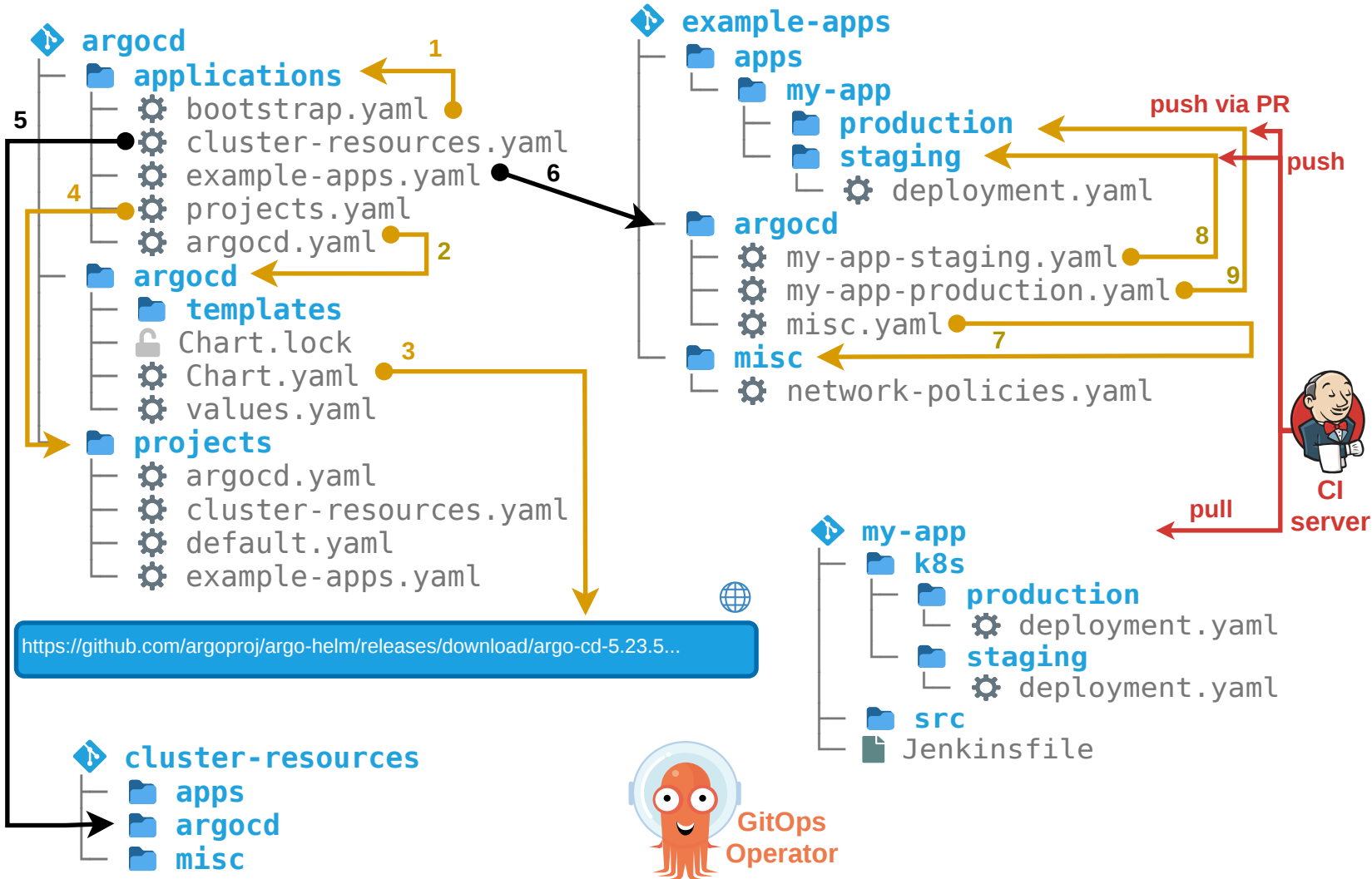
```
COMMIT='8e21bd4'  
bash <(curl -s \  
  "https://raw.githubusercontent.com/cloudogu/gitops-playground/$COMMIT/scripts/init-cluster.sh") \  
  --bind-ingress-port=80 \  
&& sleep 2 && docker run --rm -it --pull=always -u $(id -u) \  
  -v ~/.config/k3d/kubeconfig-gitops-playground.yaml:/home/.kube/config \  
  --net=host \  
  ghcr.io/cloudogu/gitops-playground:$COMMIT --yes --argocd --base-url=http://local.gd  
# Other interesting options --monitoring --vault=dev
```



Platform admin

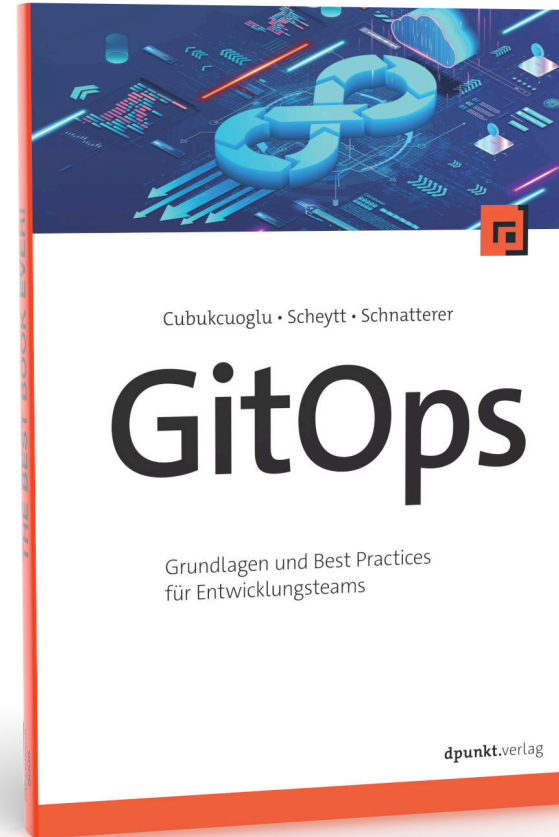
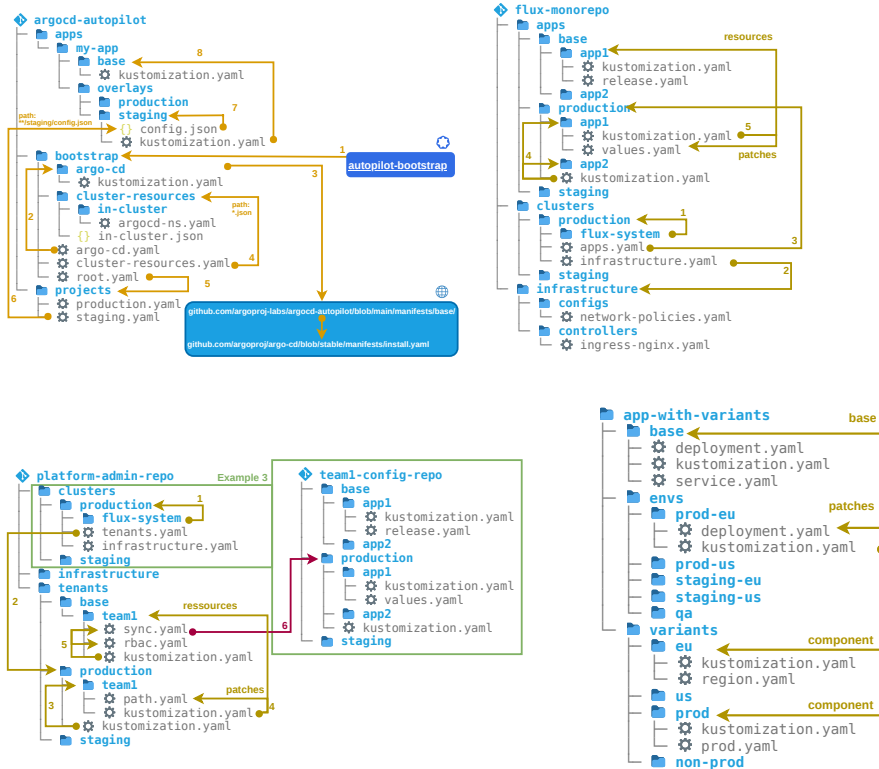


Developer



# More examples + further reading

 [cloudogu/gitops-patterns](https://github.com/cloudogu/gitops-patterns)



Johannes Schnatterer, Cludogu GmbH

Slides



💪 Join my team: [cloudogu.com/join/cloud-engineer](https://cloudogu.com/join/cloud-engineer)

📧 @schnatterer@floss.social

📄 in/jschnatterer

🐦 @jschnatterer

Wir entwickeln einen open source GitOps-Stack für K8s

**Sag uns wie wir GitOps für dich leichter machen können**

Survey



# Image sources

- Implementation

<https://unsplash.com/photos/selective-focus-photography-blue-and-black-makita-power-drill-Klby0nxseY8>

- Demo

<https://unsplash.com/photos/assorted-color-hot-air-balloons-during-daytime-DuBNA1QMpPA>

- coloured-parchment-paper background by brgfx on Freepik

[https://www.freepik.com/free-vector/coloured-parchment-paper-designs\\_1078492.htm](https://www.freepik.com/free-vector/coloured-parchment-paper-designs_1078492.htm)

- Question mark

<https://pixabay.com/illustrations/question-mark-question-response-1020165/>