// GITOPS BY EXAMPLE – REPOS, PATTERNS, PROMOTION, PROCESSES

Johannes Schnatterer, Cloudogu GmbH

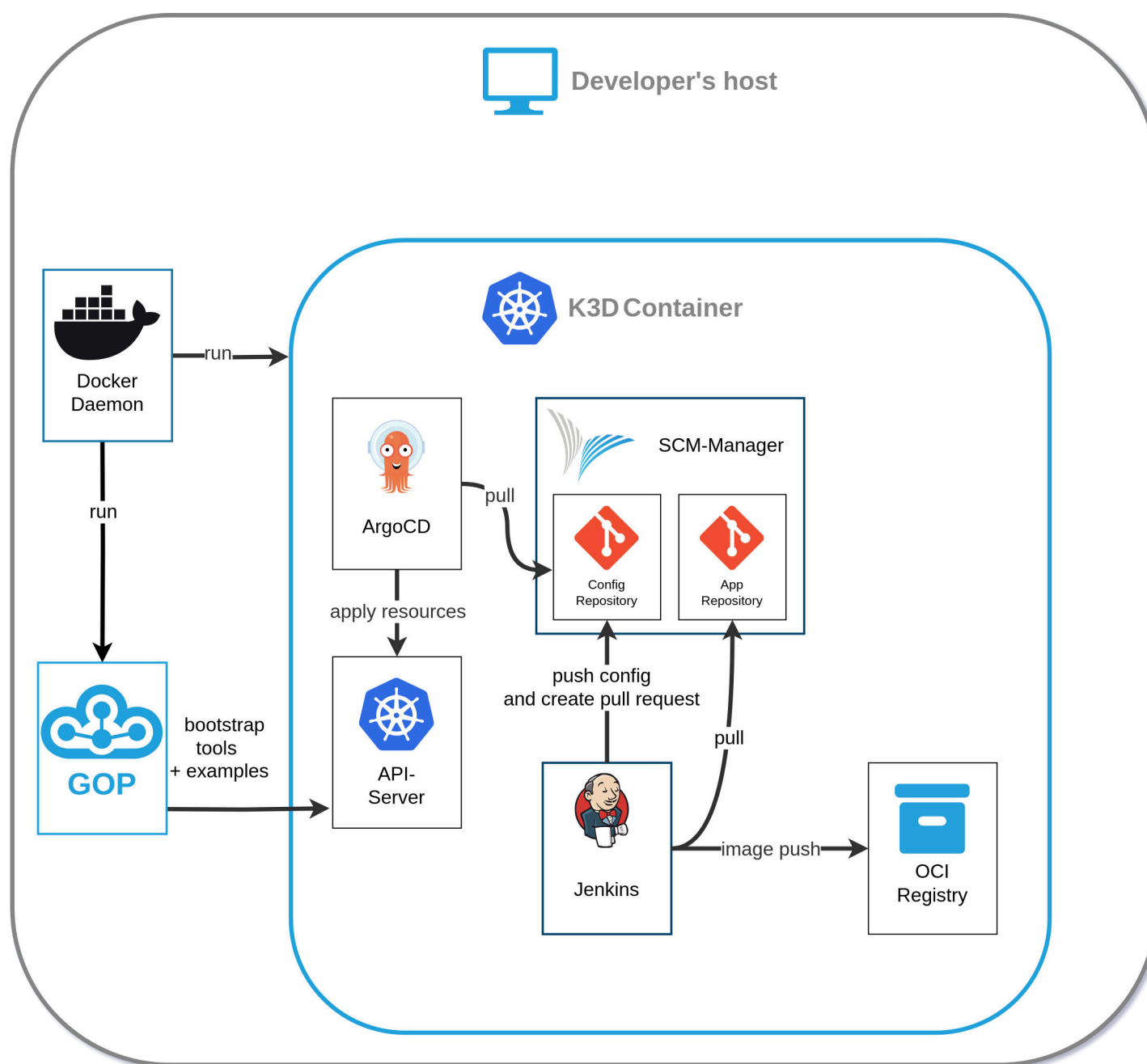**Slides**

@schnatterer@floss.social   in/jschnatterer

Version: 202507011454-a74173c

Live example

Developer's host

K3D Container

Docker Daemon → run →

Docker Daemon → run → GOP

GOP → bootstrap tools + examples → API-Server

ArgoCD → pull → SCM-Manager (Config Repository, App Repository)

ArgoCD → apply resources → API-Server

Jenkins → push config and create pull request → Config Repository

Jenkins → pull → App Repository

Jenkins → image push → OCI Registry

SCM-Manager

Config Repository

App Repository

API-Server

Jenkins

OCI Registry

3

Meet GOP

a GitOps-based operational stack (platform)


```
VERSION='0.8.0'
bash <(curl -s \
   "https://raw.githubusercontent.com/cloudogu/gitops-playground/$VERSION/scripts/init-cluster.sh") \
   && docker run --rm -t -u $(id -u) \
    -v ~/.config/k3d/kubeconfig-gitops-playground.yaml:/home/.kube/config \
    --net=host \
    ghcr.io/cloudogu/gitops-playground:$VERSION --yes --base-url=http://localhost --ingress-nginx --argocd
   # Other interesting options  --monitoring --vault=dev --mailhog --cert-manager
```

 cloudogu/gitops-playground

Example 1: GOP

- **Repo pattern:** Per team 🗂 per app
- **Operator pattern:** Instance per Cluster (Hub and Spoke, Instance per Namespace)
- **Operator:** 🐙
- **Bootstrapping:** GOP (`Helm`, `kubectl`)
- **Linking:** 🐙 `Application`
- **Features:**
  - Operate ArgoCD with GitOps
  - Solution for cluster resources
  - Env per app
  - CI: Config update + replication, rendered Manifest ⎈
  - Multi-tenancy: N tenants on 1 cluster (namespace envs)
- **Source:** 🐙 cloudogu/gitops-playground

Instance per Cluster   Hub and Spoke   Instance per Namespace
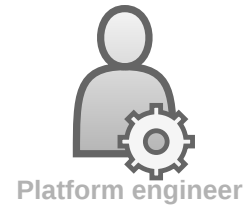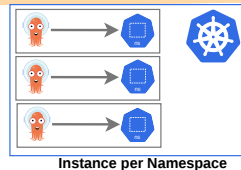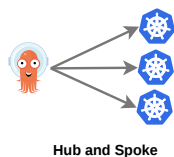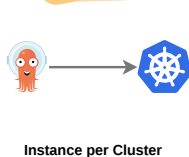
Platform engineer

Software engineer

App of Apps

Umbrella Chart

⚡ **Folder per Env**

```
argocd
└── applications                    1
    ├── ⚙ bootstrap.yaml
 5  ├── ⚙ cluster-resources.yaml
    ├── ⚙ example-tenant.yaml        6
 4  ├── ⚙ projects.yaml
    └── ⚙ argocd.yaml                2
└── argocd
    ├── 📁 templates
    ├── 🔒 Chart.lock
    ├── ⚙ Chart.yaml                 3
    └── ⚙ values.yaml
└── projects
    ├── ⚙ argocd.yaml
    ├── ⚙ cluster-resources.yaml
    ├── ⚙ default.yaml
    └── ⚙ example-tenant.yaml
```

🌐

https://github.com/argoproj/argo-helm/releases/download/argo-cd-5.23.5...

```
cluster-resources
├── 📁 apps
├── 📁 argocd
└── 📁 misc
```

```
example-tenant/gitops
└── 📁 apps              Env per app
    └── 📁 my-app
        ├── 📁 production
        └── 📁 staging
            └── ⚙ deployment.yaml
└── 📁 argocd
    ├── ⚙ my-app-staging.yaml        8
    ├── ⚙ my-app-production.yaml     9
    └── ⚙ misc.yaml         7
└── 📁 misc
    └── ⚙ network-policies.yaml
```

push via PR

push

```
example-tenant/my-app
└── 📁 k8s
    ├── 📁 production
    │   └── ⚙ deployment.yaml
    └── 📁 staging
        └── ⚙ deployment.yaml
└── 📁 src
└── 📄 Jenkinsfile
```

pull

CI server
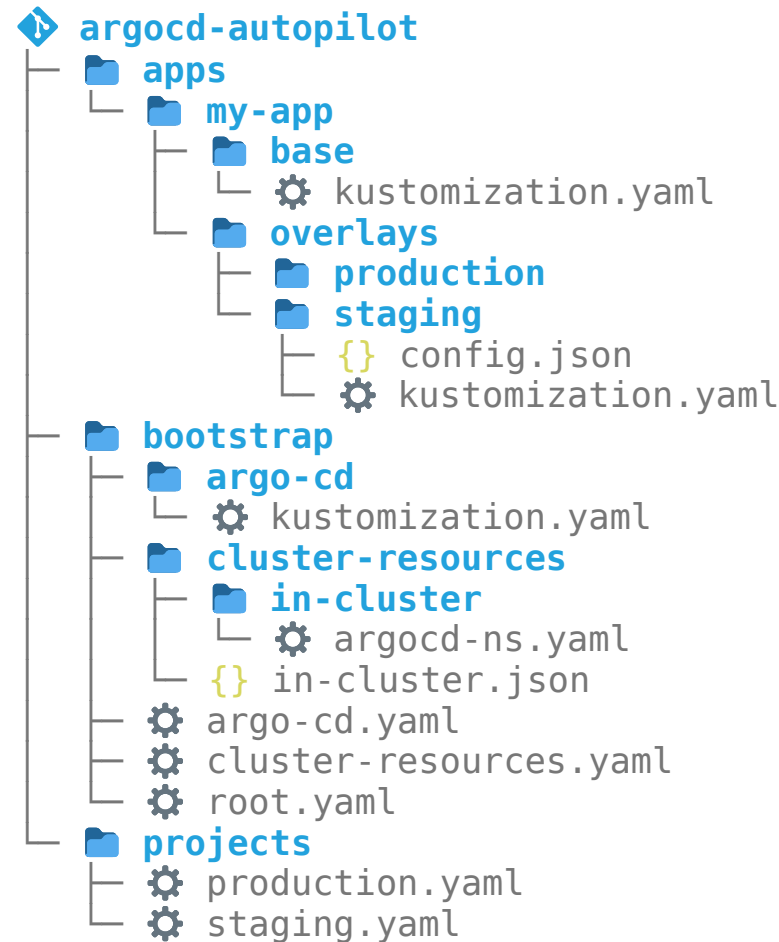
GitOps Operator
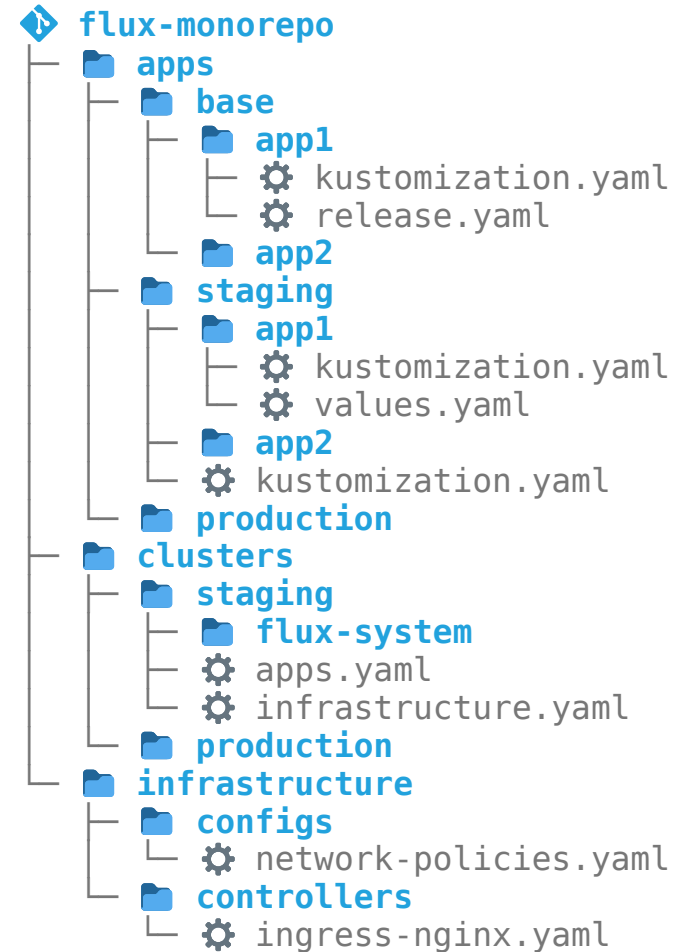
🐙 cloudogu/gitops-build-lib

5

More examples

Example 2: ArgoCD autopilot

- **Repo pattern:** Monorepo
- **Operator pattern:**
  Instance per Cluster / Hub and Spoke
- **Operator:** 🐙
- **Bootstrapping:** `argocd-autopilot`
- **Linking:** 🐙 `Application`, `ApplicationSet`, K
- **Features:**
  - Operate ArgoCD with GitOps
  - Solution for cluster resources
  - Env per app Pattern
  - Create structure and YAML via CLI
- **Source:**
  argoproj-labs/argocd-autopilot
  gitops-book/argocd-autopilot-example

```
argocd-autopilot
├── apps
│   └── my-app
│       ├── base
│       │   └── kustomization.yaml
│       └── overlays
│           ├── production
│           └── staging
│               ├── config.json
│               └── kustomization.yaml
├── bootstrap
│   ├── argo-cd
│   │   └── kustomization.yaml
│   ├── cluster-resources
│   │   ├── in-cluster
│   │   │   └── argocd-ns.yaml
│   │   └── in-cluster.json
│   ├── argo-cd.yaml
│   ├── cluster-resources.yaml
│   └── root.yaml
└── projects
    ├── production.yaml
    └── staging.yaml
```

Example 3: Flux Monorepo

- **Repo pattern:** Monorepo
- **Operator pattern:** Instance per Cluster
- **Operator:**
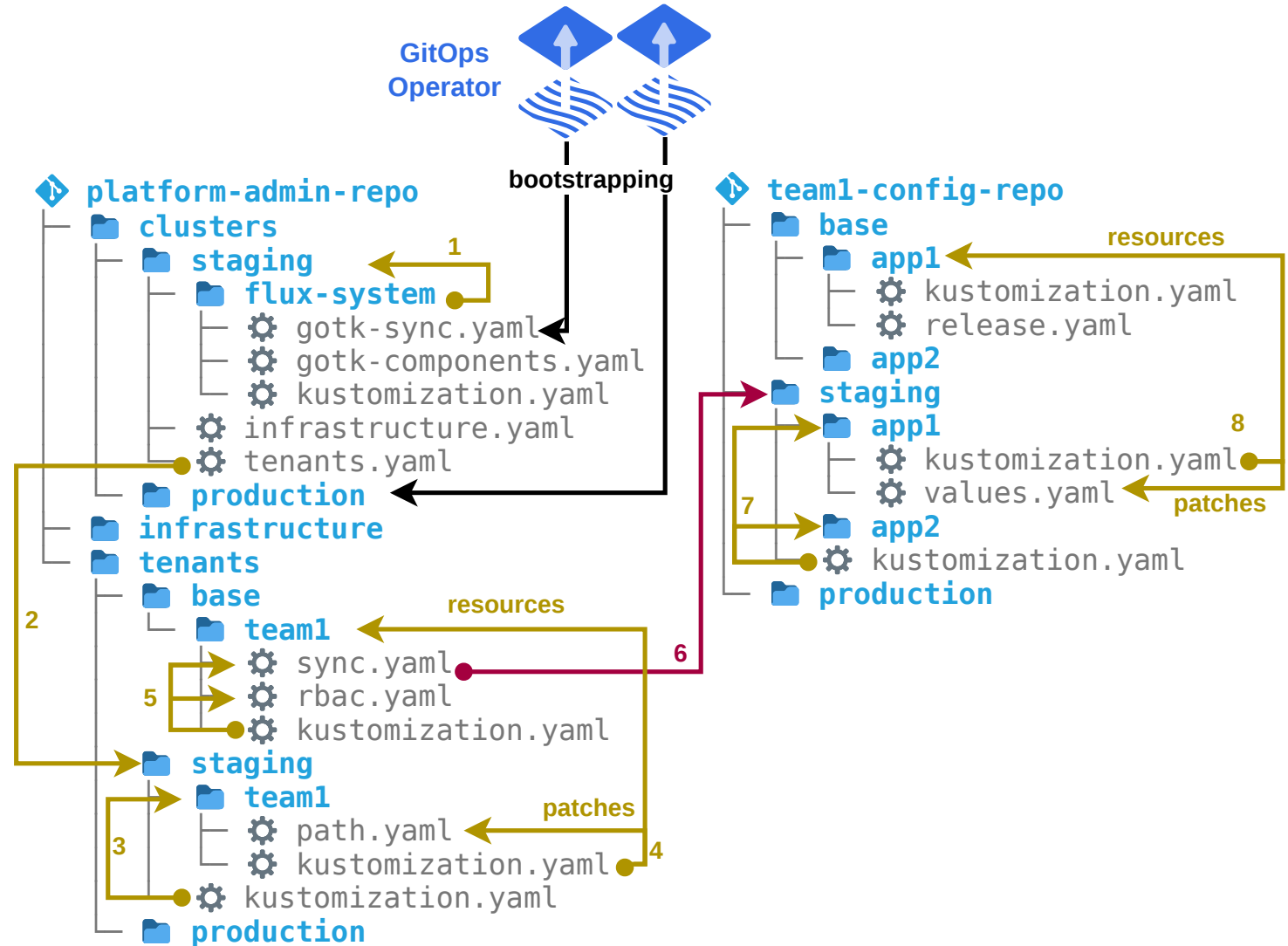- **Bootstrapping:** `flux`
- **Linking:** `Kustomization`,
- **Features:**
  - Operate Flux with GitOps
  - Solution for cluster resources
- **Source:**

  fluxcd/flux2-kustomize-helm-example#16

  fluxcd.io/flux/guides/repository-structure

  gitops-book/flux2-kustomize-helm-example

```
flux-monorepo
├── apps
│   ├── base
│   │   ├── app1
│   │   │   ├── kustomization.yaml
│   │   │   └── release.yaml
│   │   └── app2
│   ├── staging
│   │   ├── app1
│   │   │   ├── kustomization.yaml
│   │   │   └── values.yaml
│   │   ├── app2
│   │   └── kustomization.yaml
│   └── production
├── clusters
│   ├── staging
│   │   ├── flux-system
│   │   ├── apps.yaml
│   │   └── infrastructure.yaml
│   └── production
└── infrastructure
    ├── configs
    │   └── network-policies.yaml
    └── controllers
        └── ingress-nginx.yaml
```

Example 4: Flux repo per team

- **Repo pattern:** Repo per team
- **Operator pattern:** Instance per Cluster
- **Operator:** ◆
- **Bootstrapping:** `flux`
- **Linking:** ◆`Kustomization`, K
- **Features:**
  - Operate Flux with GitOps
  - Solution for cluster resources
  - Multi-tenancy: N tenants on 1 cluster per env (cluster envs)
- **Source:**
  - ⬡ fluxcd/flux2-multi-tenancy
  - ⬡ fluxcd.io/flux/guides/repository-structure
  - ⬡ gitops-book/flux2-multi-tenancy

Example 5: ArgoCD and Flux alternative

- **Repo pattern:** Monorepo
- **Operator pattern:** Instance per Cluster
- **Operator:** 🐙 🔷
- **Bootstrapping:** `kubectl`
- **Linking:** 🐙 `Application`, `ApplicationSet` / 🔷 `Kustomization`, 🇰
- **Features:**
  - Solution for cluster resources
  - ArgoCD **and** Flux examples
- **Source:**

  ◯ christianh814/example-kubernetes-go-repo

  📘 C. Hernandez - The Path to GitOps

```
⬦ monorepo
 └─ 📁 cluster-XXXX
     ├─ 📁 apps
     │   └─ 📁 my-app
     │       ├─ 📁 base
     │       │   ├─ ⚙ deployment.yaml
     │       │   └─ ⚙ kustomization.yaml
     │       └─ 📁 overlays
     │           ├─ 📁 production
     │           ├─ 📁 staging
     │           └─ ⚙ kustomization.yaml
     ├─ 📁 bootstrap
     │   ├─ 📁 base
     │   │   ├─ ⚙ argocd-ns.yaml
     │   │   └─ ⚙ kustomization.yaml
     │   └─ 📁 overlays
     │       └─ 📁 default
     │           └─ ⚙ kustomization.yaml
     ├─ 📁 cluster-config
     │   ├─ 📁 gitops-controller
     │   │   └─ ⚙ kustomization.yaml
     │   └─ 📁 sample-admin-workload
     │       ├─ ⚙ kustomization.yaml
     │       └─ ⚙ sample-admin-config.yaml
     └─ 📁 components
         ├─ 📁 applicationsets
         │   ├─ ⚙ apps-appset.yaml
         │   ├─ ⚙ cluster-config-appset.yaml
         │   └─ ⚙ kustomization.yaml
         └─ 📁 argocdproj
             ├─ ⚙ kustomization.yaml
             └─ ⚙ test-project.yaml
```

Example 6: Env variations single app
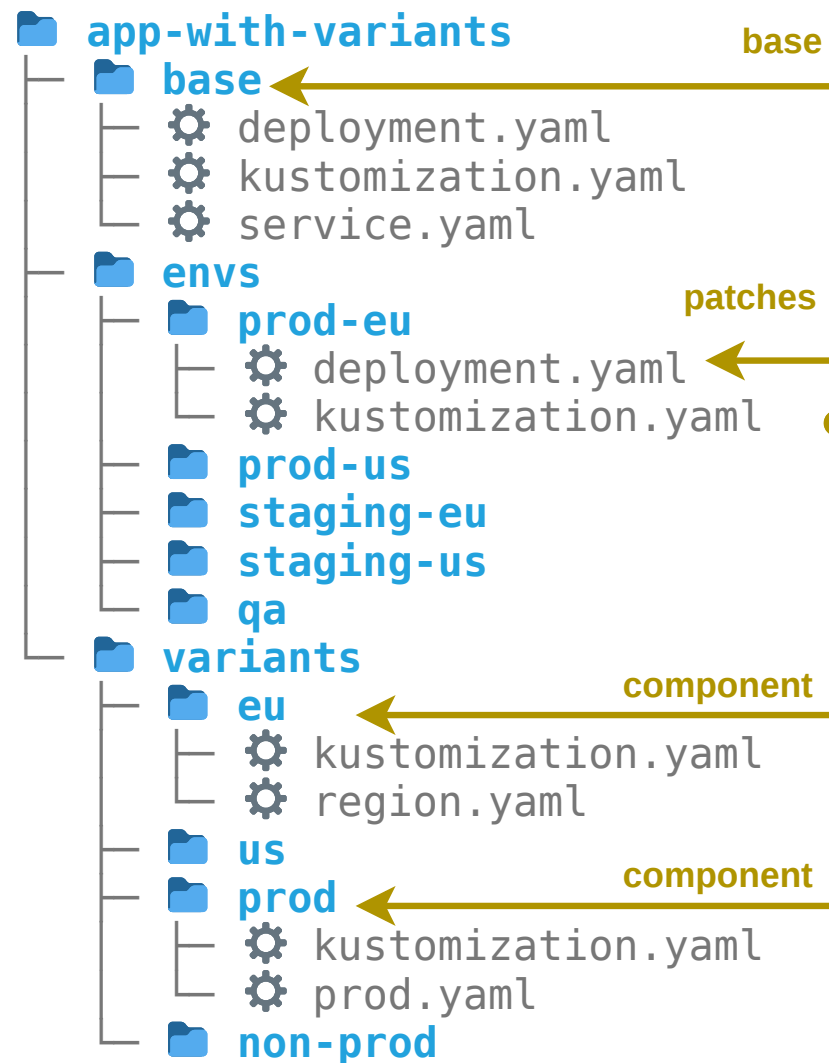
- **Operator:** 🐙 (🔷)
- **Linking:** 🇰
- **Features:**
  - Env variants for a single app
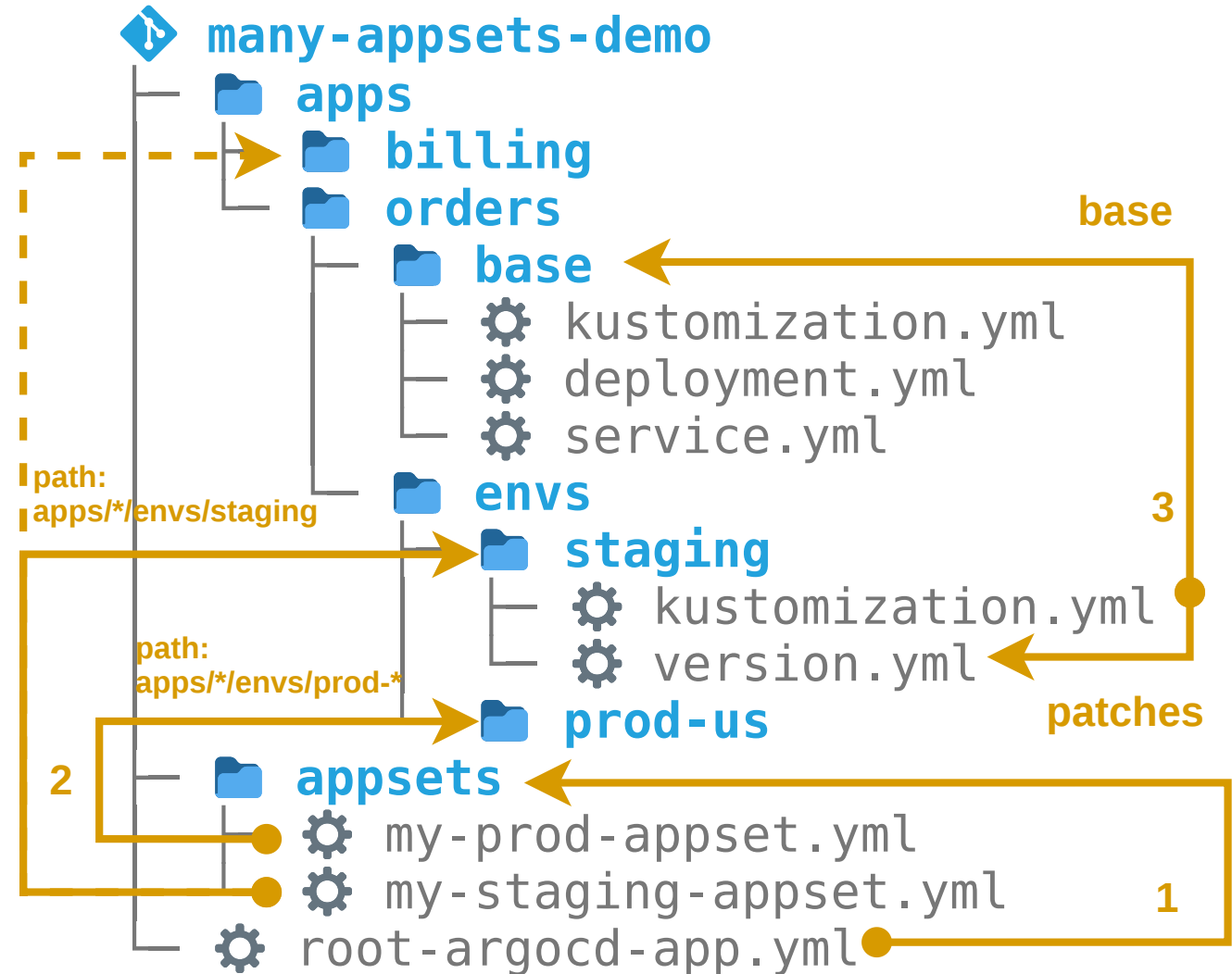 - Promotion "via `cp`"
  - Env per app Pattern
- **Source:**

  kostis-codefresh/gitops-environment-promotion

```
📁 app-with-variants                    base
├── 📁 base ◄──────────────────────────┐
│   ├── ⚙ deployment.yaml              │
│   ├── ⚙ kustomization.yaml           │
│   └── ⚙ service.yaml                 │
├── 📁 envs                          patches
│   ├── 📁 prod-eu                      │
│   │   ├── ⚙ deployment.yaml ◄────────┤
│   │   └── ⚙ kustomization.yaml       │
│   ├── 📁 prod-us                      │
│   ├── 📁 staging-eu                   ●
│   ├── 📁 staging-us                   │
│   └── 📁 qa                           │
└── 📁 variants                     component
    ├── 📁 eu ◄────────────────────────┤
    │   ├── ⚙ kustomization.yaml       │
    │   └── ⚙ region.yaml              │
    ├── 📁 us                       component
    ├── 📁 prod ◄──────────────────────┘
    │   ├── ⚙ kustomization.yaml
    │   └── ⚙ prod.yaml
    └── 📁 non-prod
```

11

Example 7: Env variations multiple apps

- **Operator:** 🐙
- **Linking:** K , 🐙 `ApplicationSet`, (`Application`)
- **Features:**
  - Env variants for multiple apps with few appSets
  - Promotion "via `cp`"
 - Env per app Pattern
- **Source:**
  - kostis-codefresh/many-appsets-demo


```
many-appsets-demo
├── apps
│   ├── billing
│   ├── orders
│   │   ├── base
│   │   │   ├── kustomization.yml
│   │   │   ├── deployment.yml
│   │   │   └── service.yml
│   │   └── envs
│   │       ├── staging
│   │       │   ├── kustomization.yml
│   │       │   └── version.yml
│   │       └── prod-us
├── appsets
│   ├── my-prod-appset.yml
│   ├── my-staging-appset.yml
│   └── root-argocd-app.yml
```

path: apps/*/envs/staging

path: apps/*/envs/prod-*

base

patches

3

2

1

Key Takeaways

- Repo: Monorepo or polyrepo?
- Bootstrapping: The only imperative step before GitOps
- Linking: K
  - `Kustomization`
  - `Application`, `ApplicationSet`
- Multi-tenancy: Repos, operators, clusters, namespaces

➡ Examples + patterns provide guidance, templates, ideas
➡ Find the structure that matches your use case

More patterns + further reading

 cloudogu/gitops-patterns
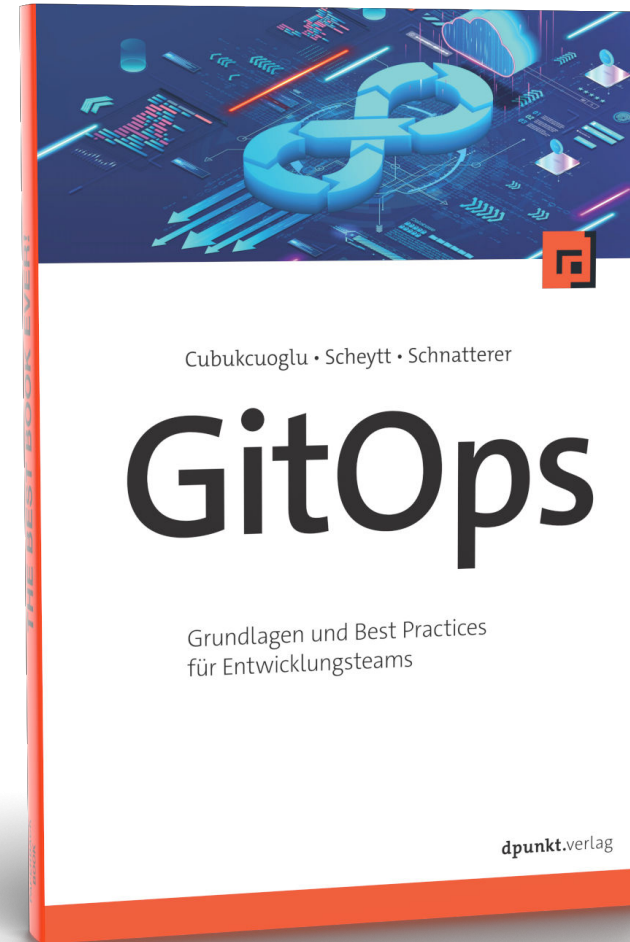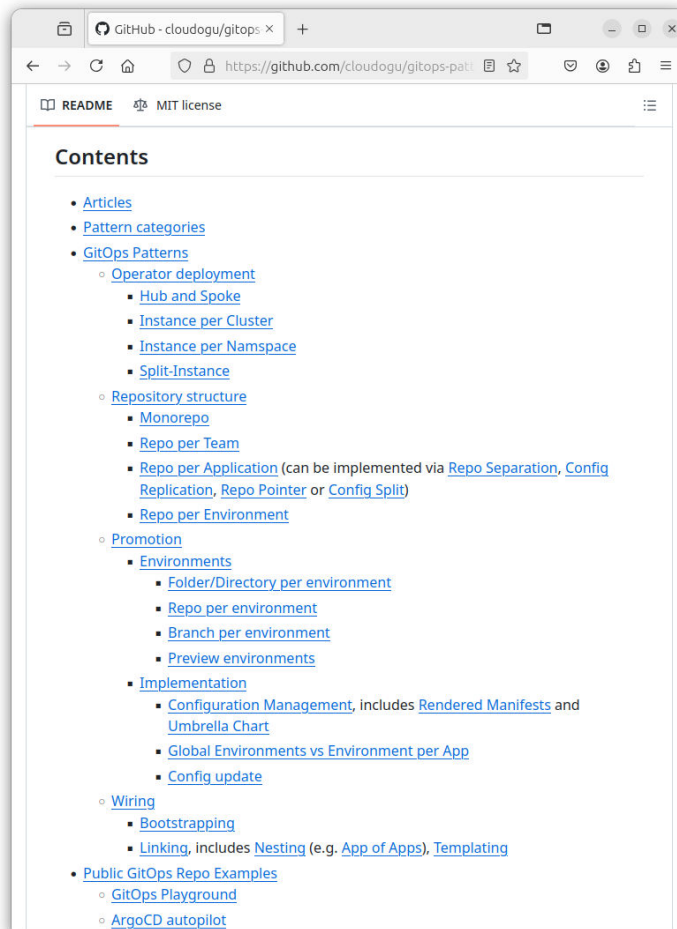
Johannes Schnatterer, Cloudogu GmbH

🐘 @schnatterer@floss.social　　💼 in/jschnatterer

Slides

💪 Join my team: cloudogu.com/join/cloud-engineer

Image sources

- Live Example:

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

- More examples:

  https://unsplash.com/photos/XZc4f2XZc84

- coloured-parchment-paper background by brgfx on Freepik:

  https://www.freepik.com/free-vector/coloured-parchment-paper-designs_1078492.htm