Octopus Deploy

Scaling up Argo CD for the Enterprise

Shipped 2025

Kostis Kapelonis | December 2025

Kostis Kapelonis

Developer Advocate (Octopus Deploy/Codefresh)

Argo Maintainer (Argo CD, Argo Rollouts)

Co-author GitOps certification

http://learning.octopus.com

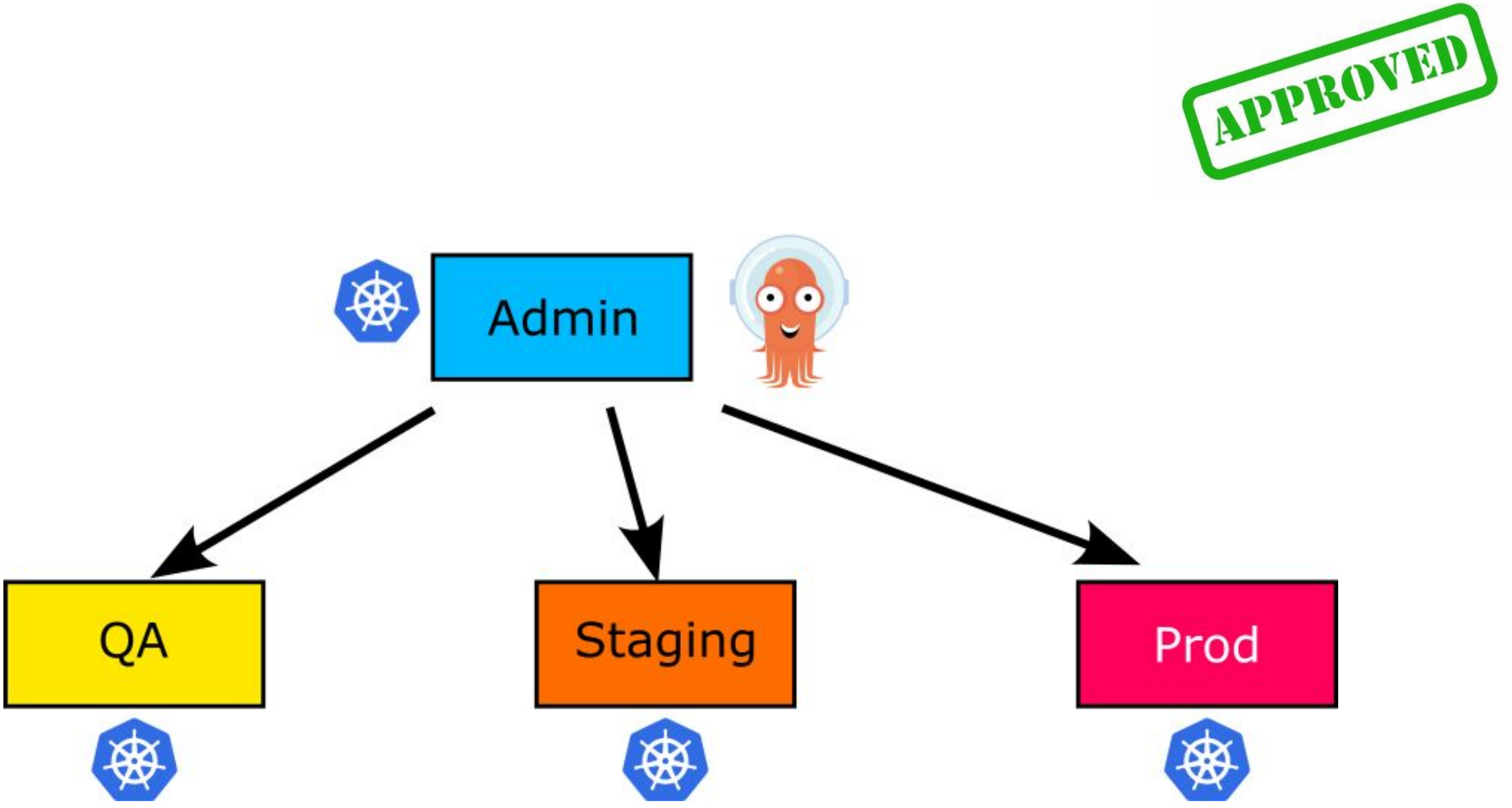
Topics covered

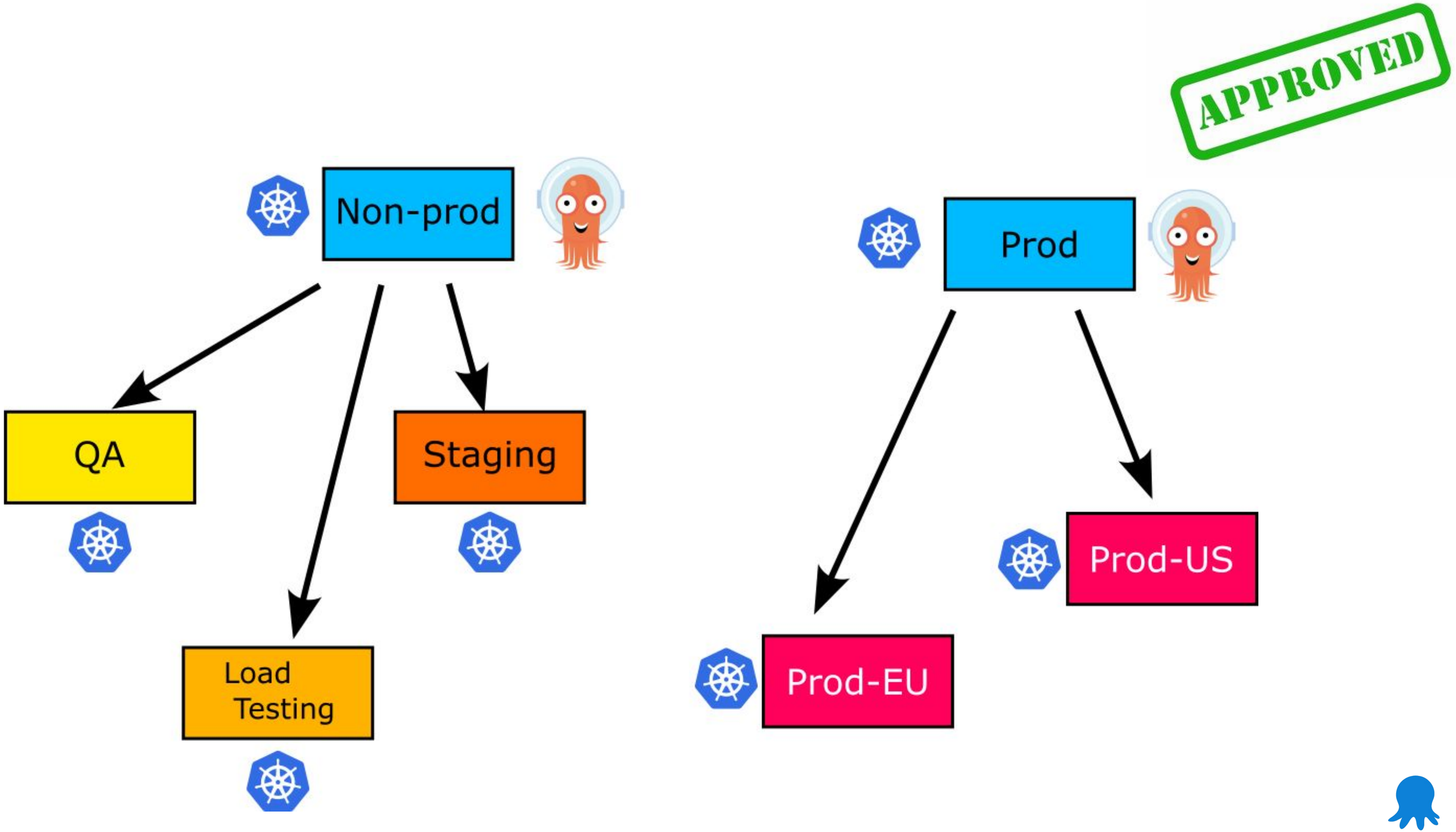
- Central Hub installation
- Git repository organization
- Helm Hierarchies
- Auto-sync/self-heal
- Sync Waves/phases
- Developer Experience
- RBAC model
- Finalizers

ArgoCD per cluster or not?

Which is best

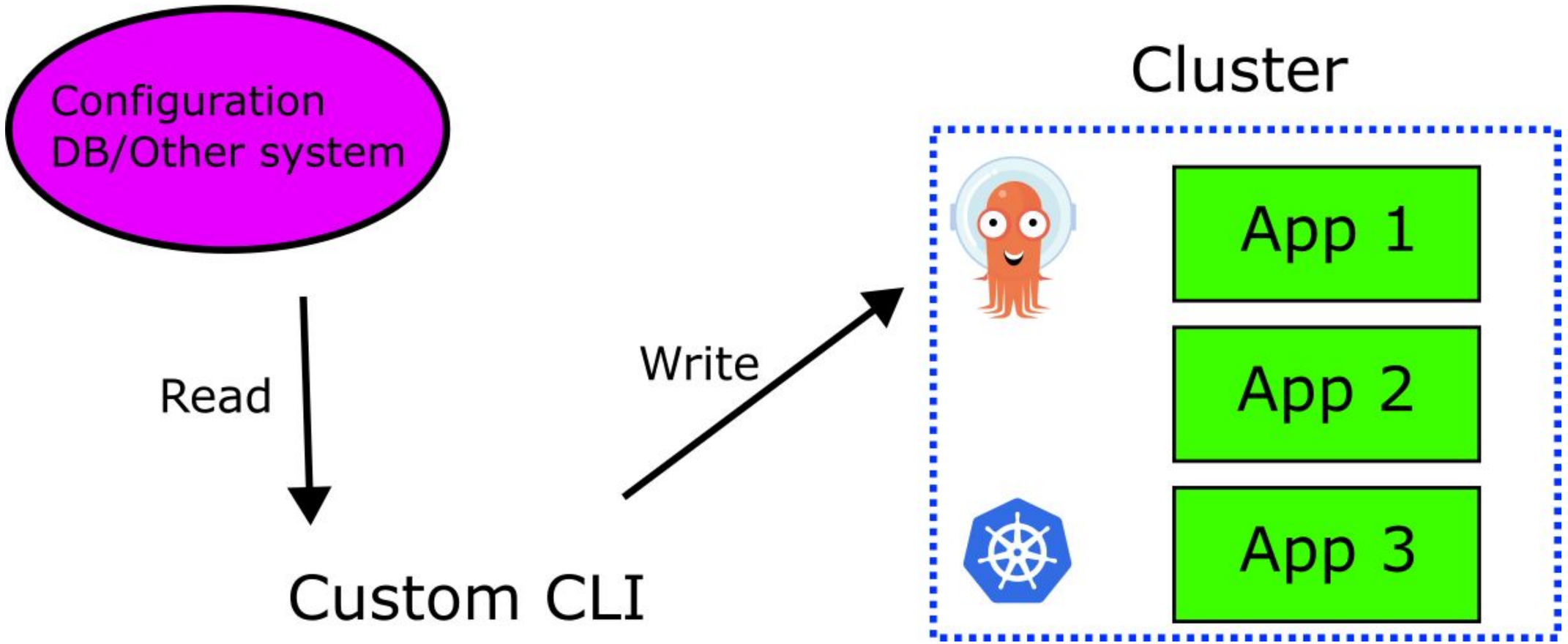
https://codefresh.io/learn/argo-cd/a-comprehensive-overview-of-argo-cd-architectures-2025/


APPROVED
Admin
QA
Staging
Prod


APPROVED
Non-prod
QA
Staging
Load
Testing
Prod
Prod-US
Prod-EU

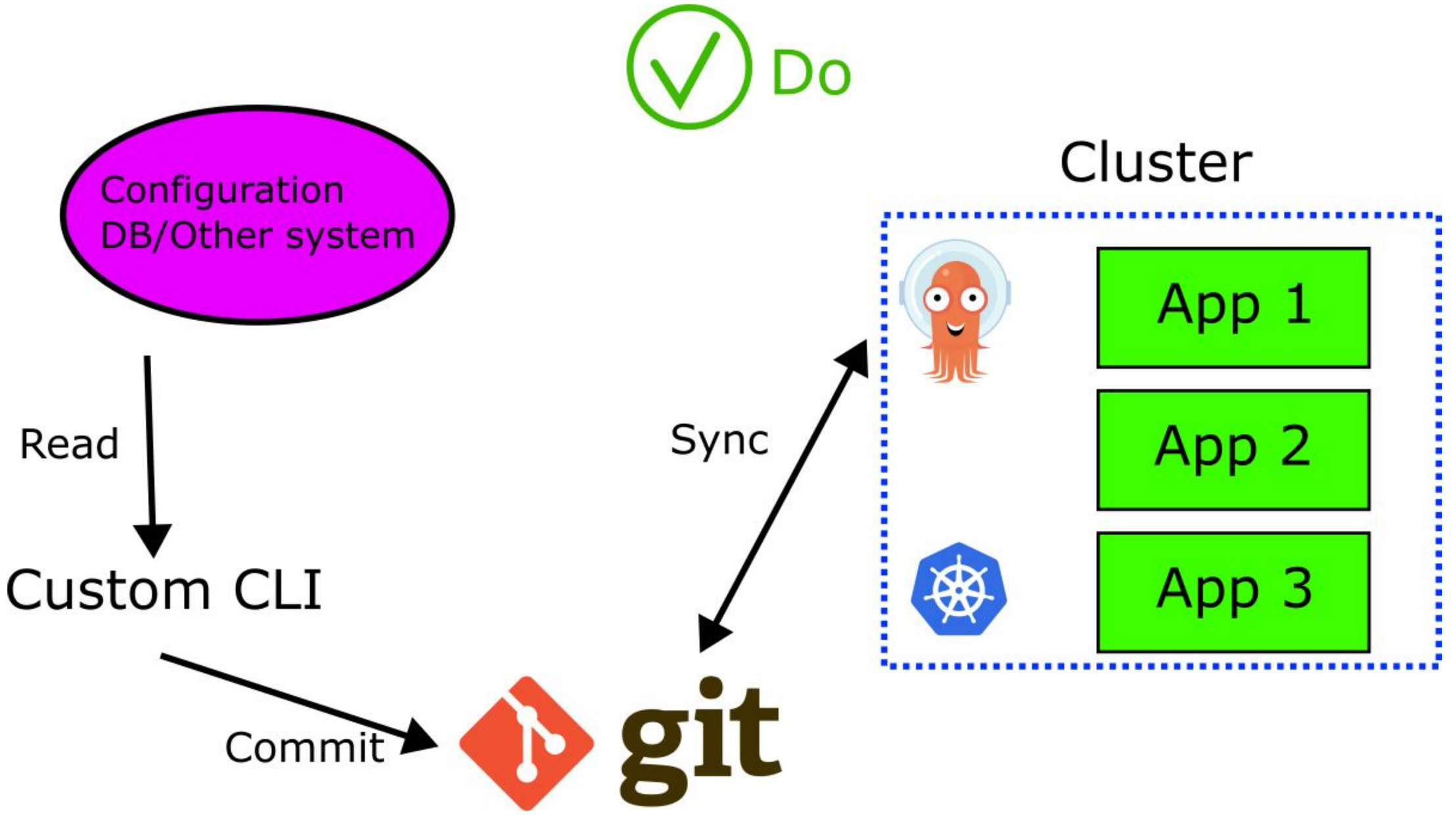
Avoid Dynamic applications

Don't


Configuration
DB/Other system
Read
Write
Custom CLI
Cluster
App 1
App 2
App 3

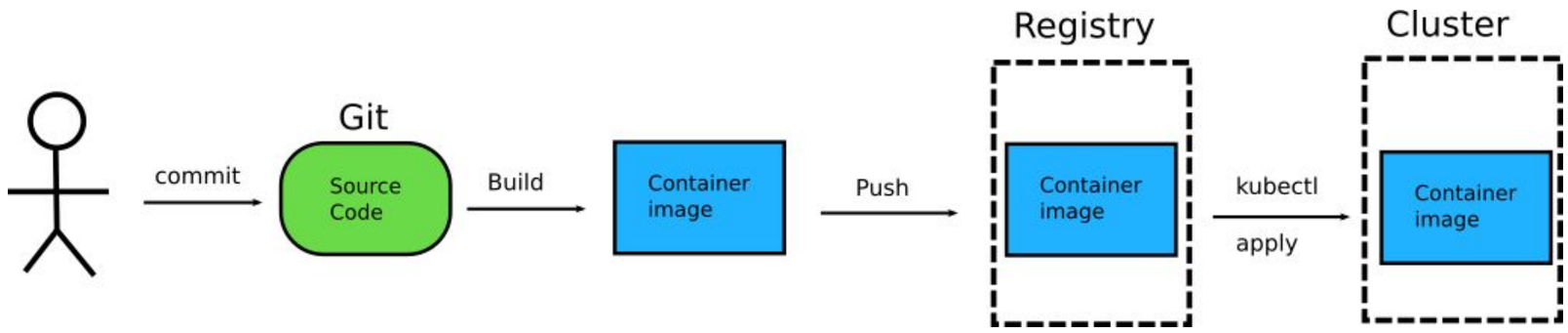

```
my-app-cli new-app-name | argocd create -f -
```

```
envsubst < my-app-template.yaml | kubectl apply -f -n argocd
```

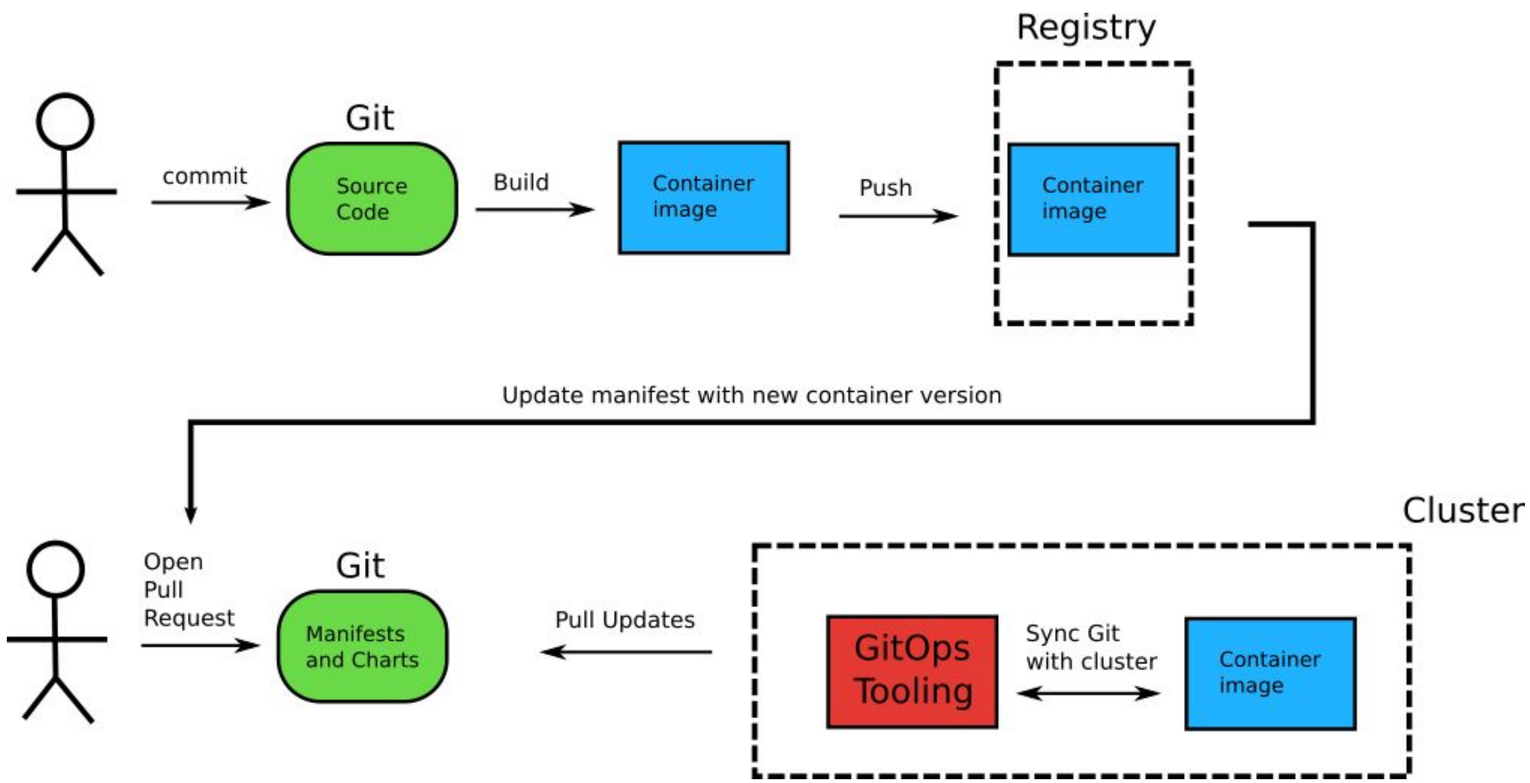

Do
Configuration
DB/Other system
Cluster
App 1
App 2
App 3
Read
Sync
Custom CLI
Commit
git


```
kubectl set image deployment/my-deployment
my-container=my-new-app:4.3.0
```

```
kubectl patch deployment <deployment-name> [.....patch here...]
```



Abusing CI as CD



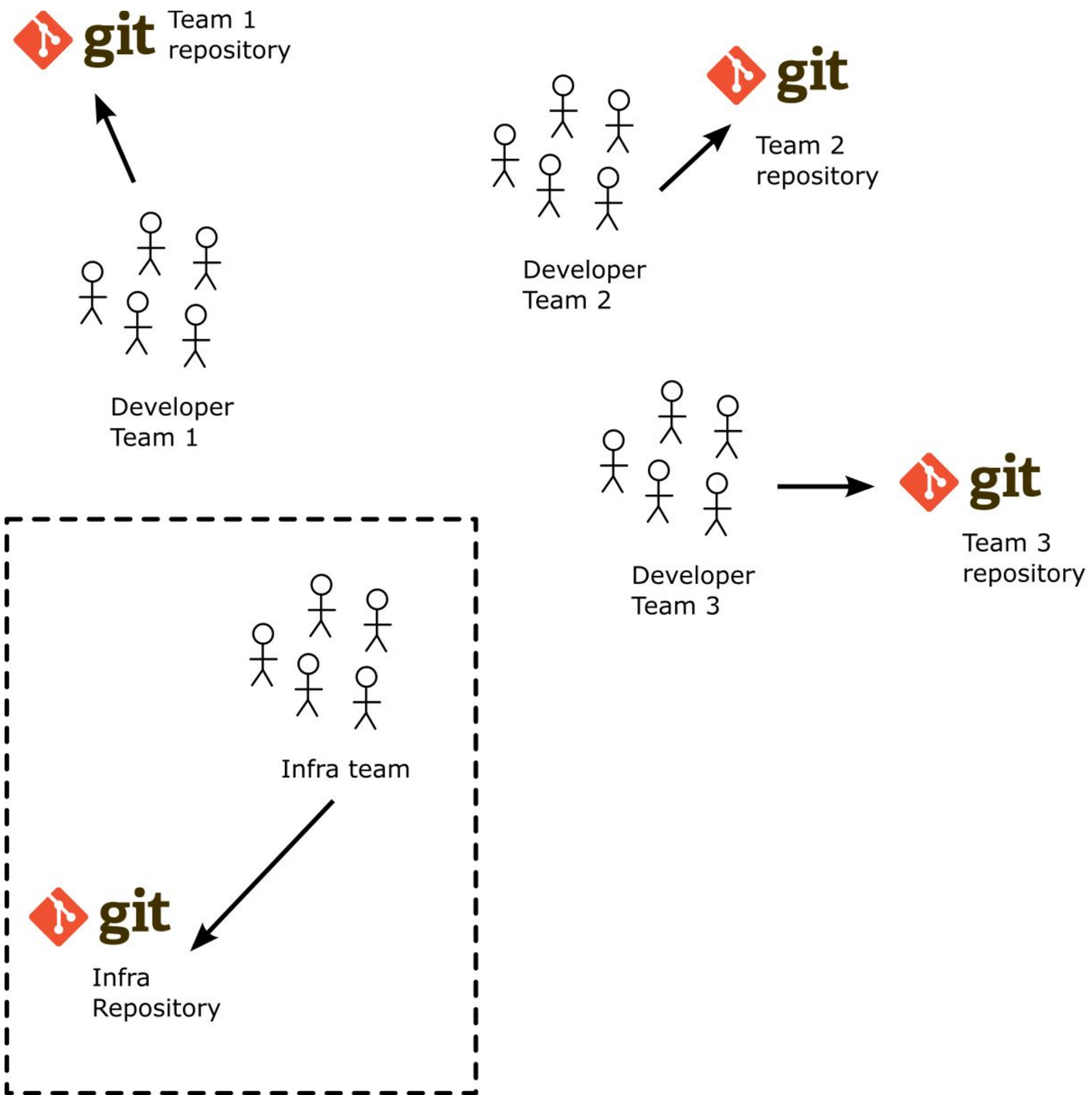
With Argo CD

Organize your Git repos

Different Git repositories

- Git Repository with source code
- Git repository with Kustomize/Helm file
- Git repository with Application Sets
- Source code changes very often. Kustomize/Helm charts change less frequently.

  Applications/Application Sets should be created once
- Understand the lifecycle of each resource


git
Team 1
repository
Developer
Team 1
git
Team 2
repository
Developer
Team 2
Developer
Team 3
git
Team 3
repository
Infra team
git
Infra
Repository

Argo CD and monorepos

- Disable sync polling
- Use webhooks for Applications **AND** Application Sets
- Use the **manifest-generate-paths** annotation
- Enable timeouts for everything
- Monitor and adapt resources
- Check https://argo-cd.readthedocs.io/en/latest/operator-manual/high_availability/

Master Helm before Argo CD

How Helm Hierachies work

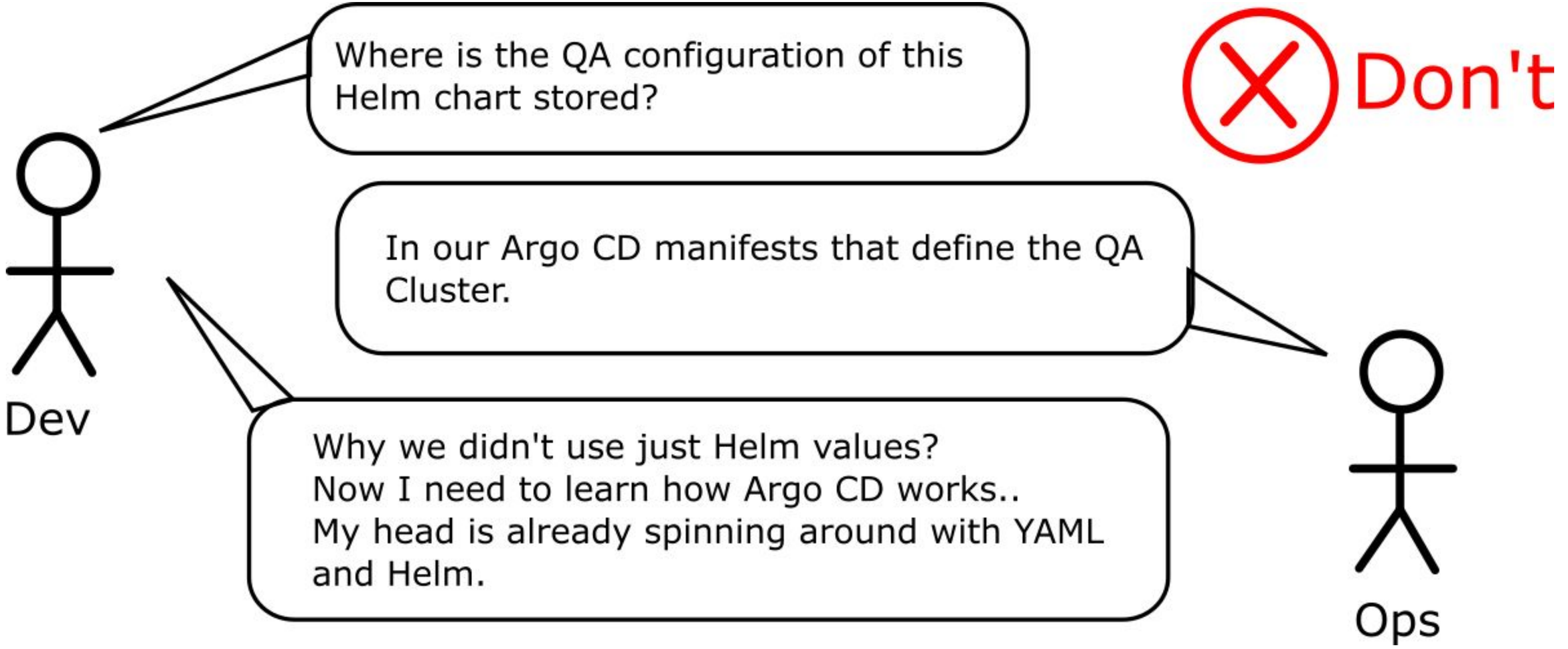
```
common-values.yaml
+-----all-prod-envs.yaml
   +----specific-prod-cluster.yaml
```

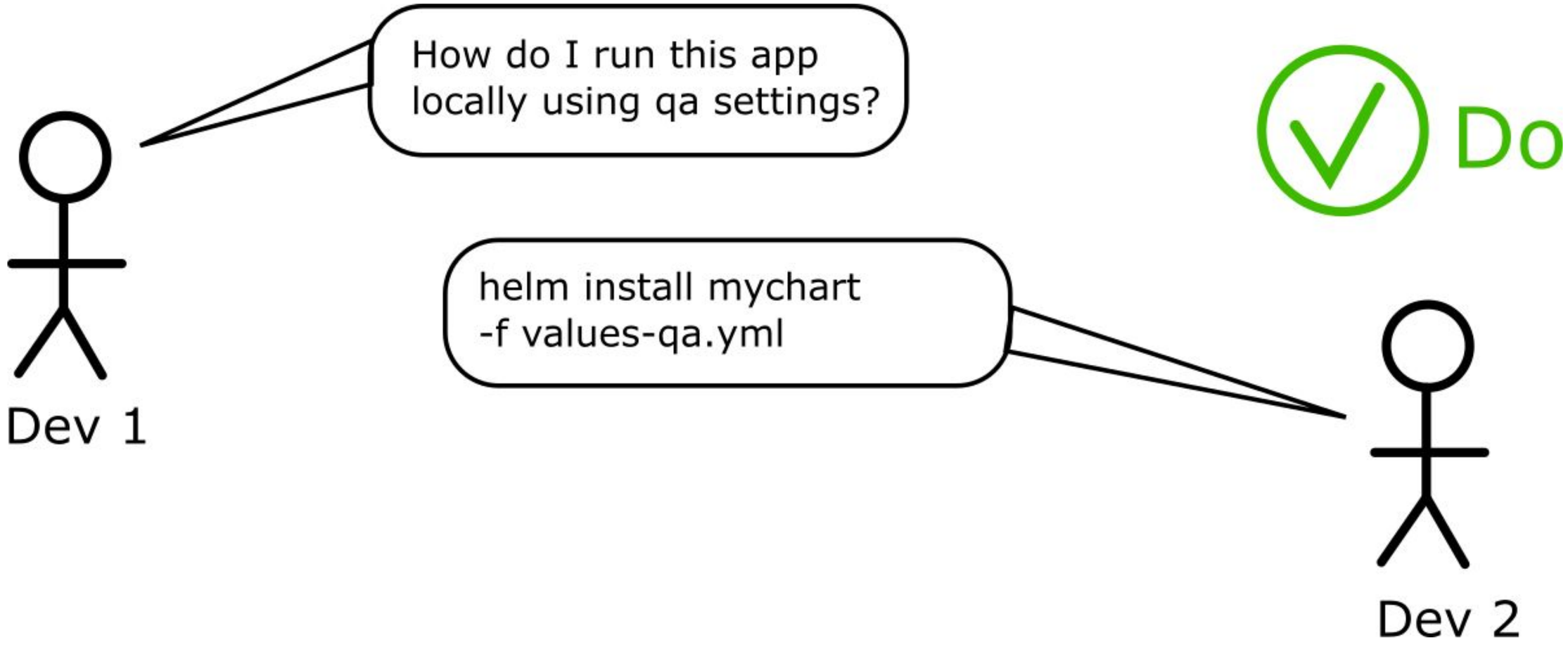
```
helm install ./my-chart/ --generate-name
-f common.yaml -f more.yaml -f some-more.yaml
```

- my-values
  - app-version
    - prod-values.yaml
    - qa-values.yaml
    - staging-values.yaml
  - env-type
    - non-prod-values.yaml
    - prod-values.yaml
  - envs
  - regions
    - asia-values.yaml
    - eu-values.yaml
    - us-values.yaml
  - common-values.yaml
  - README.md

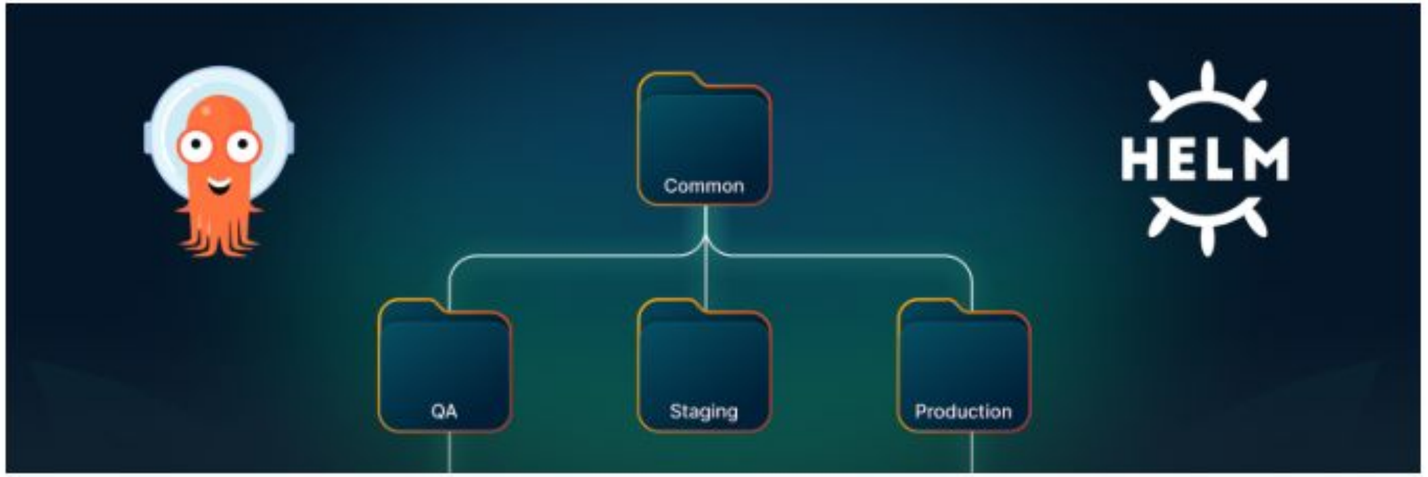
- my-values
  - app-version
  - env-type
  - envs
    - integration-gpu-values.yaml
    - integration-non-gpu-values.ya...
    - prod-eu-values.yaml
    - prod-us-values.yaml
    - qa-values.yaml
    - staging-asia-values.yaml
    - staging-eu-values.yaml
    - staging-us-values.yaml
  - regions
  - common-values.yaml
  - README.md

```yaml
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: my-app
  namespace: argocd
spec:
  project: default
  source:
    chart: my-chart
    repoURL: https://github.com/my-example-app
    targetRevision: 2.43
    helm:
      valueFiles:
      - common-values.yaml
      - all-prod-envs.yaml
      - specific-prod-cluster.yaml
  destination:
    server: "https://kubernetes.default.svc"
    namespace: default
```


Where is the QA configuration of this Helm chart stored?
Don't
In our Argo CD manifests that define the QA Cluster.
Dev
Why we didn't use just Helm values?
Now I need to learn how Argo CD works..
My head is already spinning around with YAML and Helm.
Ops


How do I run this app locally using qa settings?
Do
helm install mychart -f values-qa.yml
Dev 1
Dev 2

HOME > BLOG



BEST PRACTICES

Using Helm Hierarchies in Multi-Source Argo CD Applications for Promoting to Different GitOps Environments

14 min read

https://codefresh.io/blog/helm-values-argocd/

Master Kustomize before Argo CD

Kustomize concepts

- Overlays for different environments
- Reusable component configurations
- Common globals (labels, annotations, namespace prefixes)
- Configuration generators
- Transformers/Replacements
- Remote resources

Kustomize Overlays versus Components

what is the difference between kustomize overlays and kustomize components ?

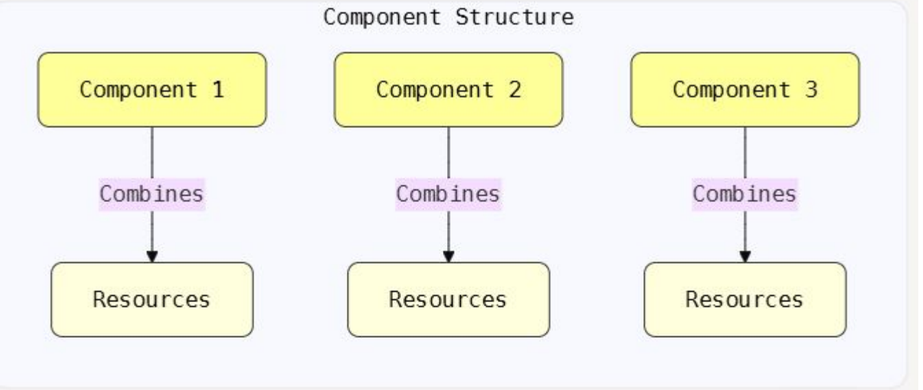
PHIND-70B MODEL

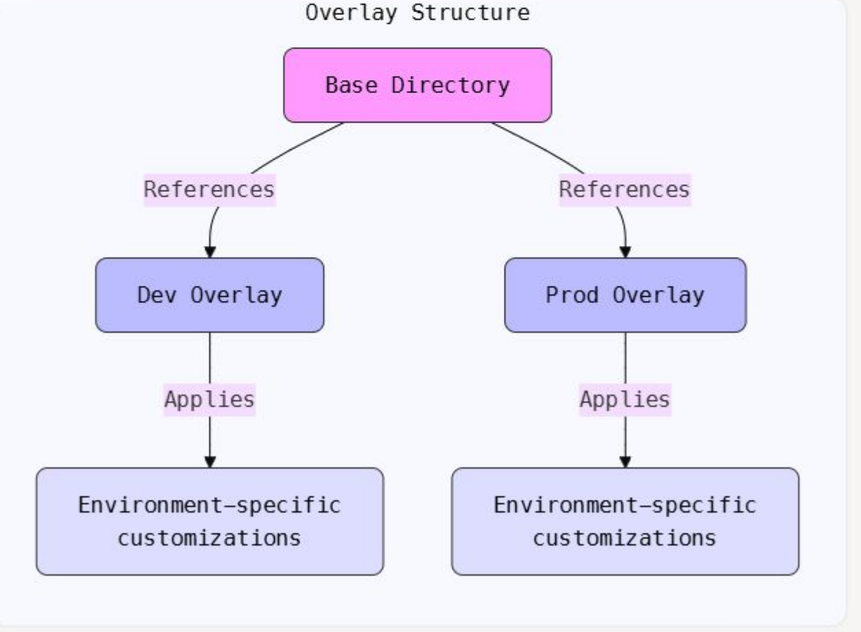
1 Declarative Man... 2 Kustomize in Ku... 3 Components - K... 4 The complete Ku...

Kustomize provides two distinct approaches for organizing and managing Kubernetes configurations: overlays and components. Each serves a specific purpose in managing Kubernetes resources across different environments and deployments.

ASPECT	OVERLAYS	COMPONENTS
Purpose	Environment-specific customizations kubernetes.io	Reusable resource collections kubernetes.io
Structure	Reference-based hierarchy	Flat composition structure
Base Usage	References external bases	Contains self-contained resources
Customization	Applies patches and modifications	Direct resource definition
Reusability	One overlay per environment	Multiple components reusable across projects
Maintenance	Centralized base management	Independent component updates

Let's visualize how overlays and components differ in their structure and relationships:





Learn all this BEFORE using Argo CD

replacements

Substitute field(s) in N target(s) with a field from a source.

Replacements are used to copy fields from one source into any number of specified targets.

The `replacements` field can support a path to a replacement:

`kustomization.yaml`

```yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization

replacements:
  - path: replacement.yaml
```

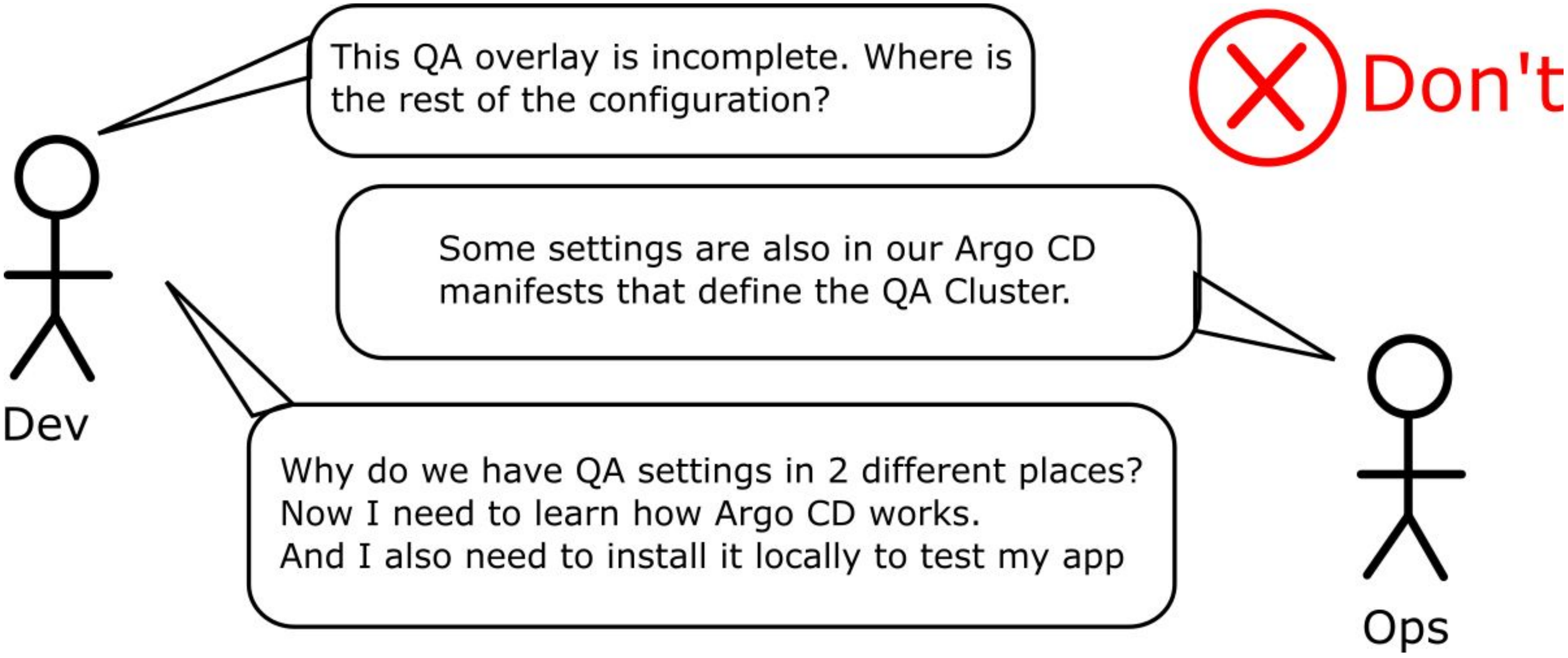
`replacement.yaml`

```yaml
source:
  kind: Deployment
  fieldPath: metadata.name
targets:
  - select:
      name: my-resource
```

Alternatively, `replacements` supports inline replacements:

```yaml
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization

replacements:
- source:
    kind: Deployment
    fieldPath: metadata.name
  targets:
  - select:
      name: my-resource
```


This QA overlay is incomplete. Where is the rest of the configuration?
Don't
Some settings are also in our Argo CD manifests that define the QA Cluster.
Dev
Why do we have QA settings in 2 different places?
Now I need to learn how Argo CD works.
And I also need to install it locally to test my app
Ops

HOME > BLOG

BEST PRACTICES

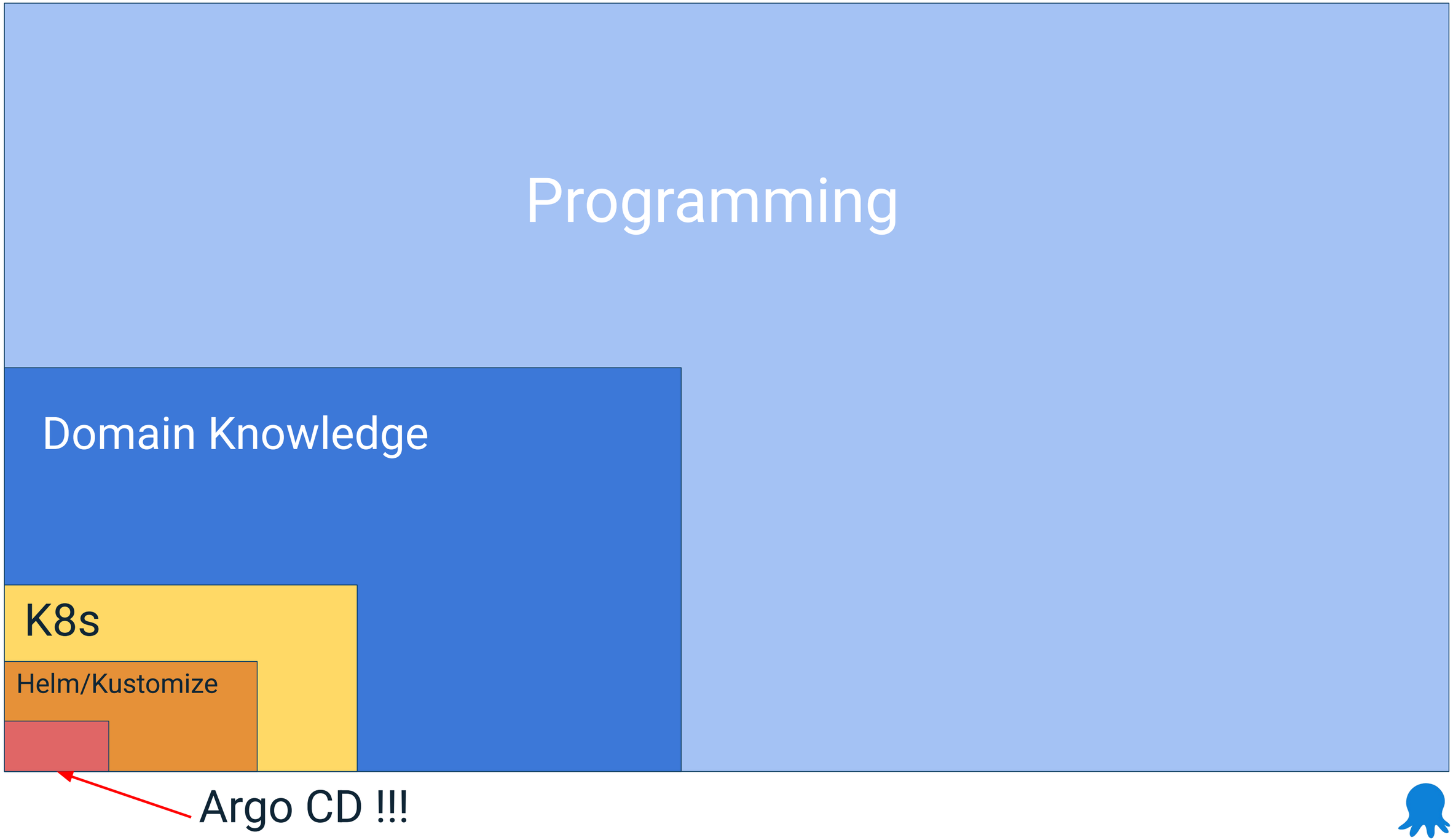
How to Model Your GitOps Environments and Promote Releases between Them

18 min read

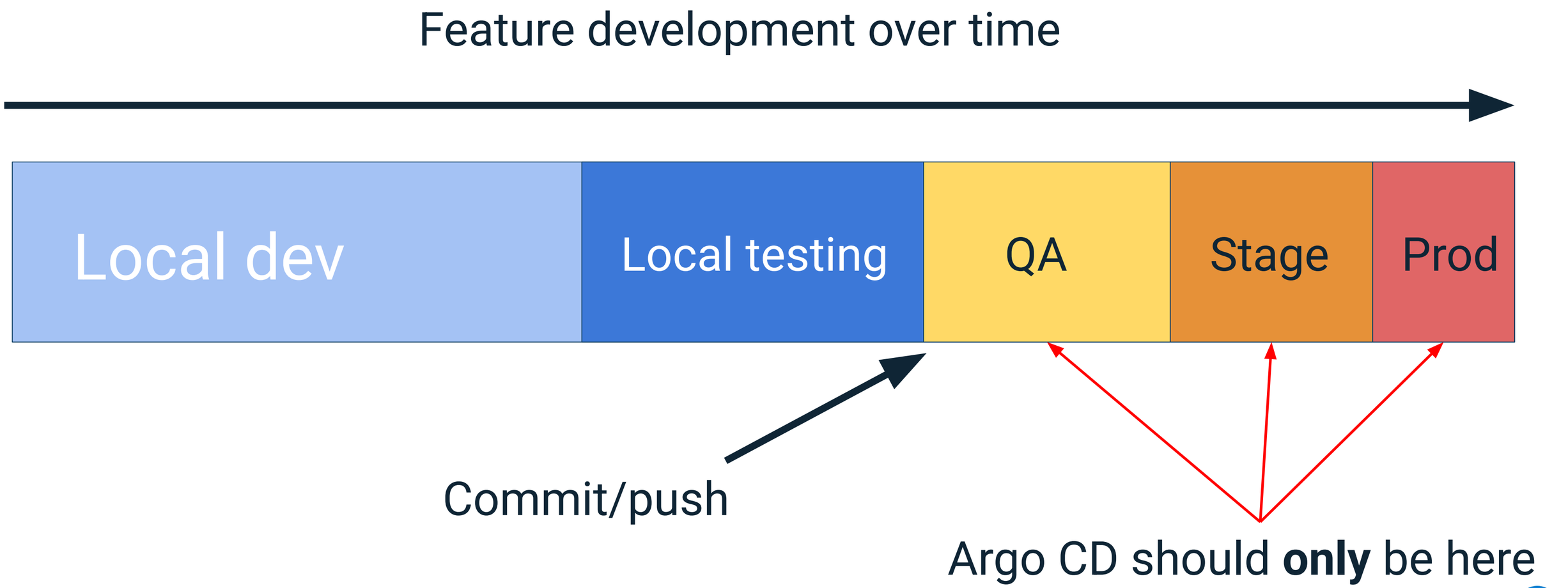
Kostis Kapelonis March 23, 2022

https://codefresh.io/blog/how-to-model-your-gitops-environments-and-promote-releases-between-them/

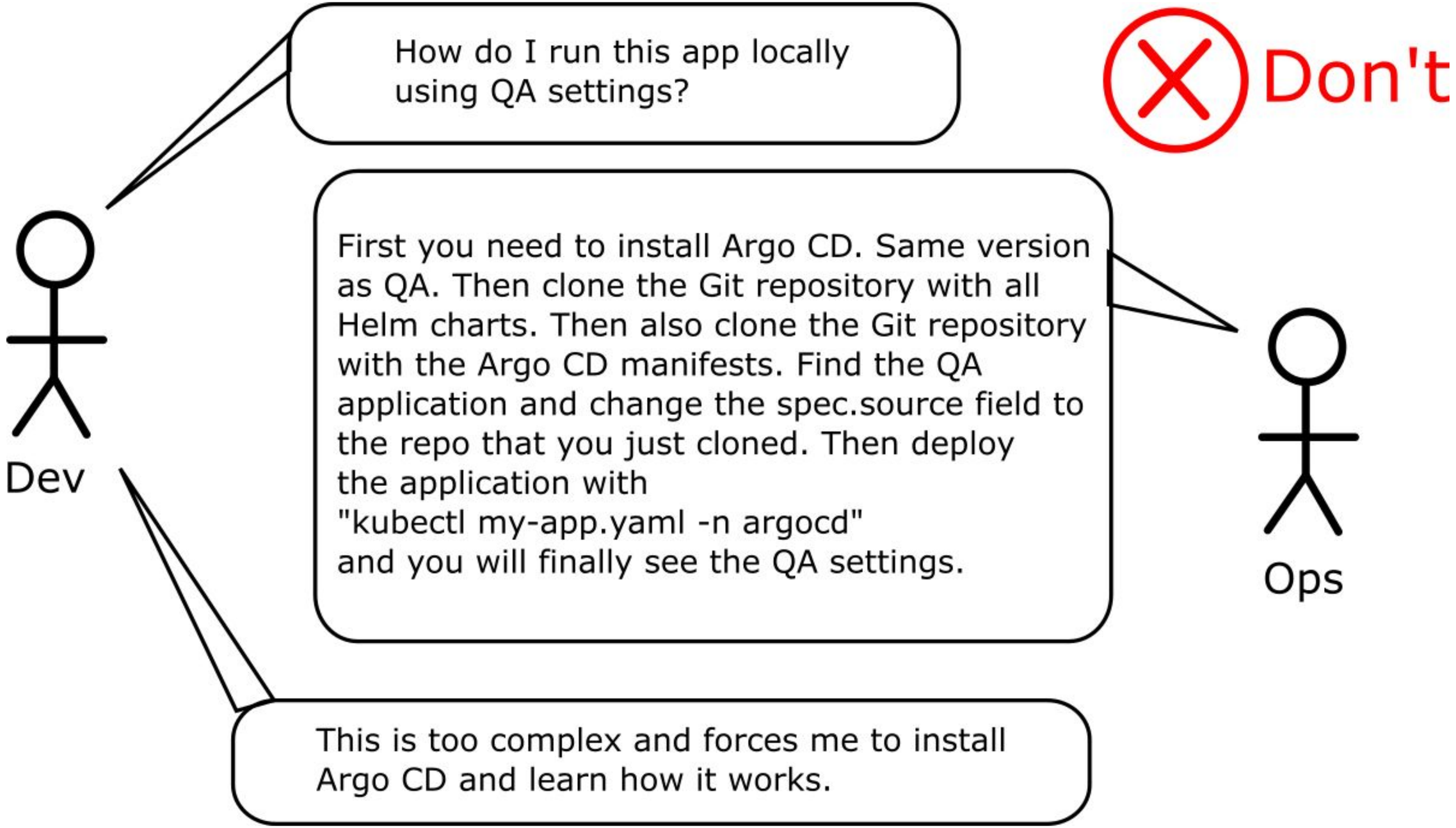
Don’t force developers to use Argo CD

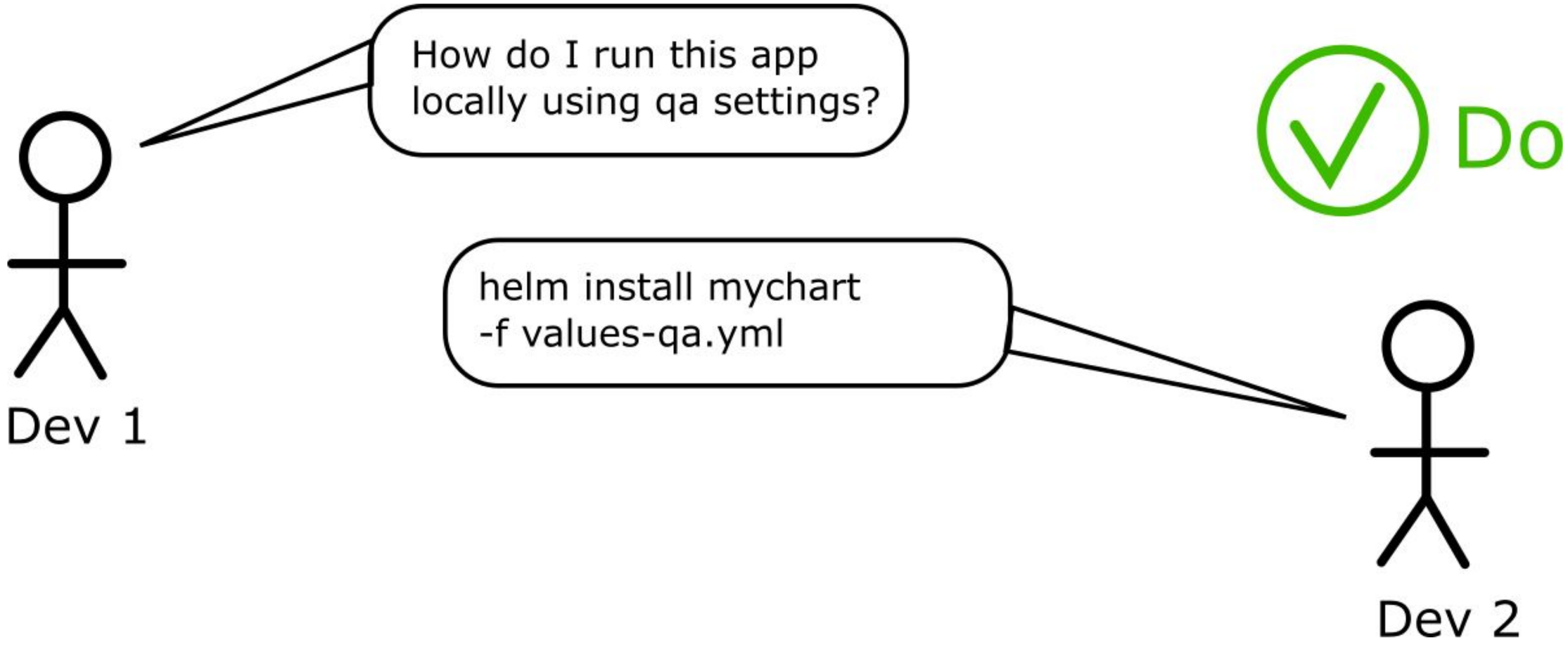

Programming
Domain Knowledge
K8s
Helm/Kustomize
Argo CD !!!

Developers do NOT care about Argo CD manifests


Feature development over time
Local dev
Local testing
QA
Stage
Prod
Commit/push
Argo CD should **only** be here

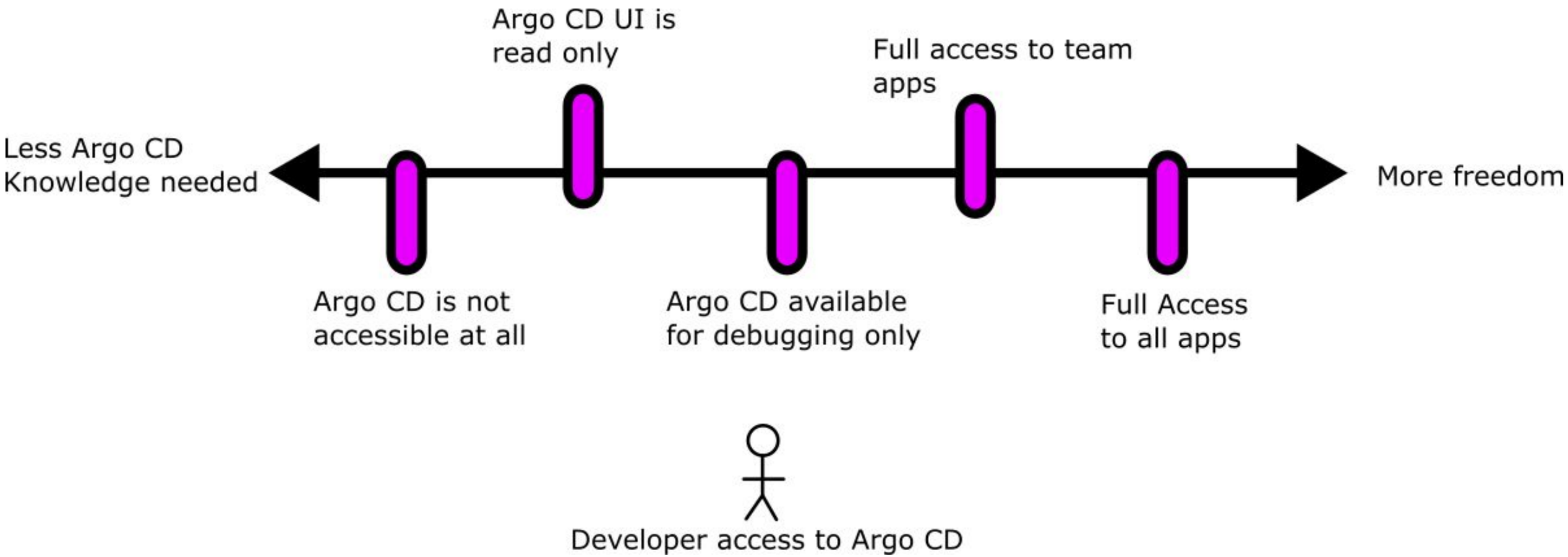
If developers need Argo CD for LOCAL development you make their life miserable


How do I run this app locally using QA settings?
Don't
First you need to install Argo CD. Same version as QA. Then clone the Git repository with all Helm charts. Then also clone the Git repository with the Argo CD manifests. Find the QA application and change the spec.source field to the repo that you just cloned. Then deploy the application with "kubectl my-app.yaml -n argocd" and you will finally see the QA settings.
Dev
Ops
This is too complex and forces me to install Argo CD and learn how it works.

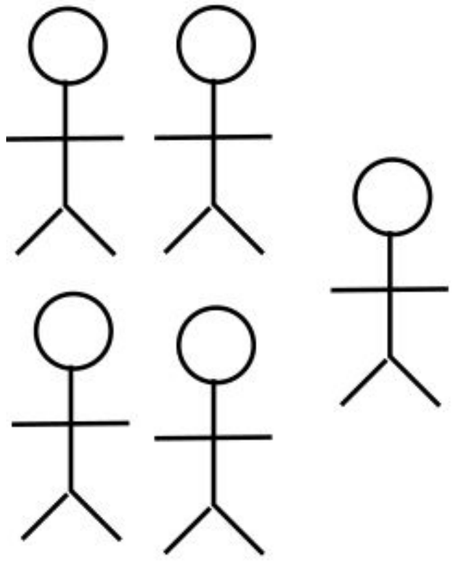

How do I run this app locally using qa settings?
Do
helm install mychart -f values-qa.yml
Dev 1
Dev 2

Developer Experience

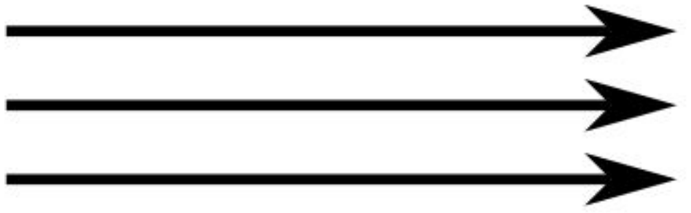
Understand your developers



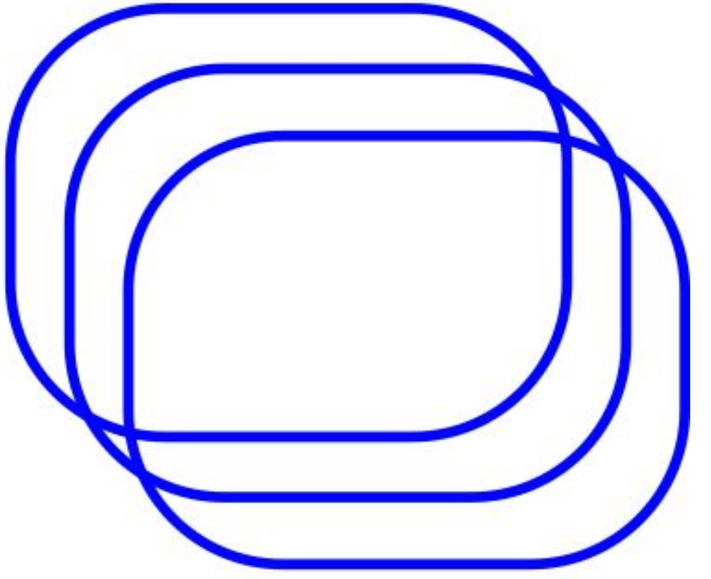
Team A
Devs



Read
Edit
Sync



Applications
of Team A



Who

(User Group)

How

(Policy)

What

(App-Project)

Applications

+ NEW APP | SYNC APPS | REFRESH APPS | Search applications... /

billing

Project:	team-a
Labels:	
Status:	Healthy Synced
Repository:	https://github.com/kostis-codefresh/intro-ar...
Target Rev...	HEAD
Path:	apps/team-a/billing
Destination:	https://kubernetes.default.svc
Namespa...	billing
Created At:	09/30/2024 14:46:58 (a day ago)
Last Sync:	10/01/2024 14:53:53 (4 minutes ago)

SYNC | REFRESH | DELETE

orders

Project:	team-b
Labels:	
Status:	Healthy Synced
Repository:	https://github.com/kostis-code
Target Rev...	HEAD
Path:	apps/team-b/orders
Destination:	https://kubernetes.default.svc
Namespa...	orders
Created At:	09/30/2024 14:46:58 (a day ag
Last Sync:	09/30/2024 14:49:22 (a day ag

SYNC | REFRESH | DELETE

SYNCHRONIZE | CANCEL

Synchronizing application manifests from https://github.com/kostis-codefresh/intro-argocd-rbac.git

Revision
HEAD

- [ ] PRUNE
- [ ] DRY RUN
- [ ] APPLY ONLY
- [ ] FORCE

SYNC OPTIONS

- [ ] SKIP SCHEMA VALIDATION
- [x] AUTO-CREATE NAMESPACE
- [ ] PRUNE LAST
- [ ] APPLY OUT OF SYNC ONLY
- [ ] RESPECT IGNORE DIFFERENCES
- [ ] SERVER-SIDE APPLY

PRUNE PROPAGATION POLICY: foreground

- [ ] REPLACE ⚠
- [ ] RETRY

SYNCHRONIZE RESOURCES: all / out of sync / none

- [x] /SERVICE/ORDERS/SIMPLE-SERVICE
- [x] APPS/DEPLOYMENT/ORDERS/SIMPLE-DEPLOYMENT

Unable to sync: permission denied: applications, sync, team-b/orders, sub: jane, iat: 2024-10-01T11:57:17Z

TECHNICAL GUIDES

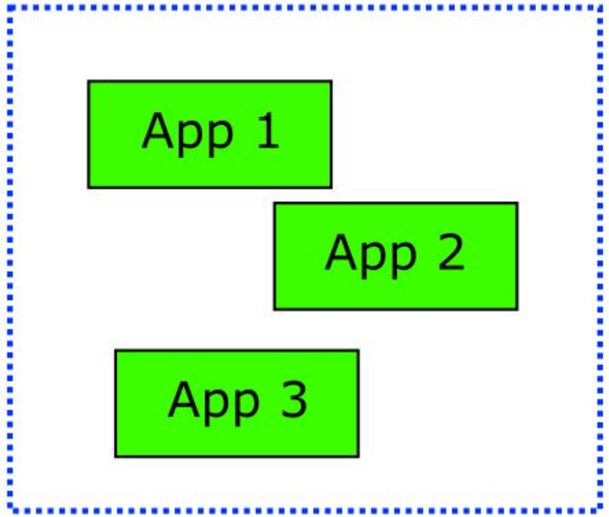
Securing Argo CD in a Multi-Tenant Environment with Application Projects

10 min read

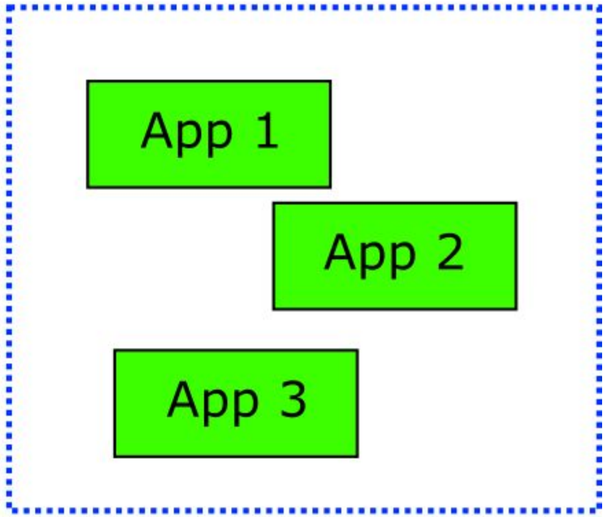
https://codefresh.io/blog/multi-tenant-argocd-with-application-projects/

Application Grouping

Umbrella Helm chart

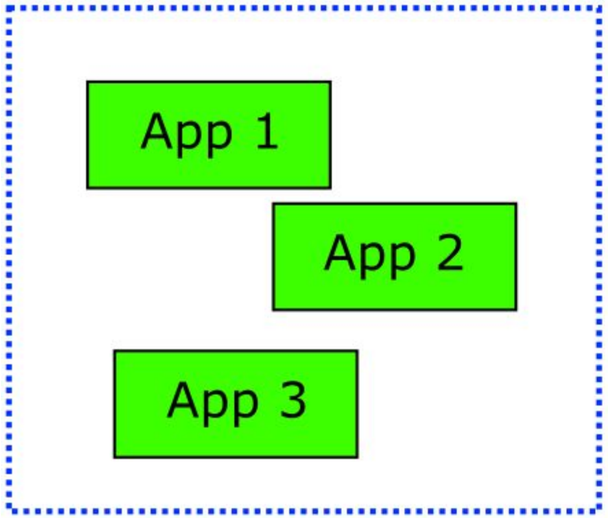


Single Argo CD app

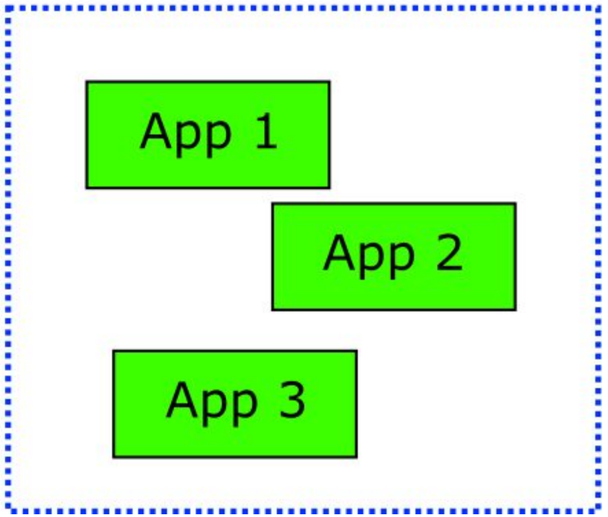


Which choice
is best???

Three Argo CD apps



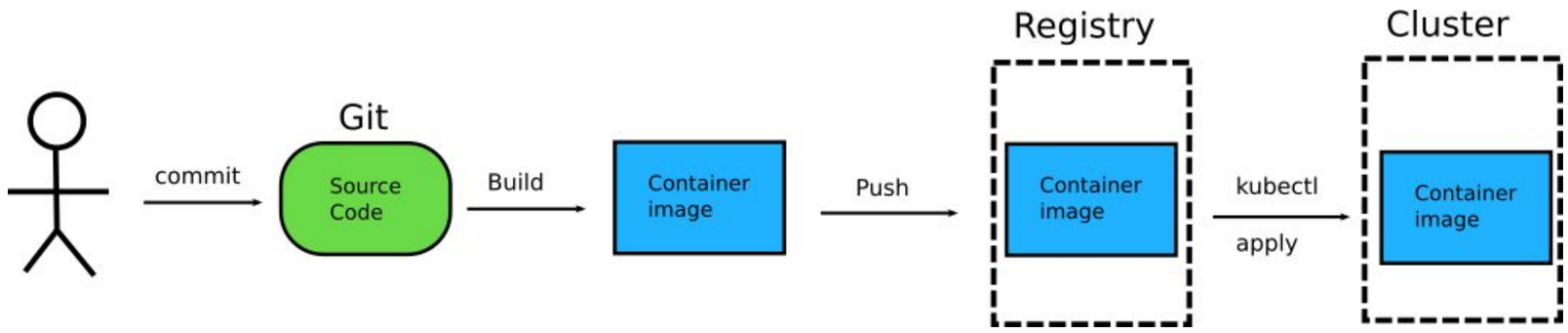
Application Set



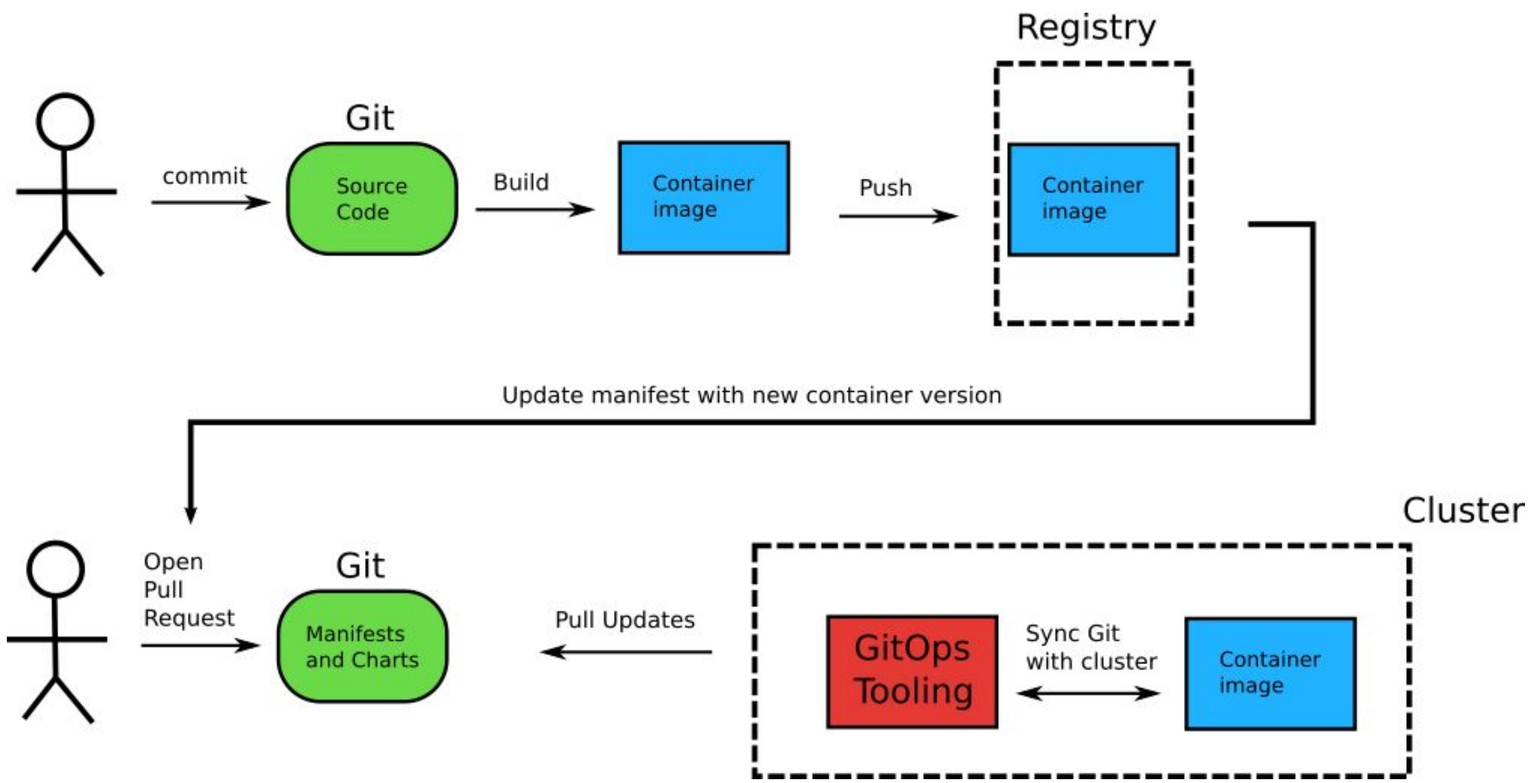
Questions to ask

- Are those applications always deployed and upgraded as a single unit?
- Are those applications related in a business or technical manner?
- Do you want to use different configurations for different clusters for these applications?
- Is this combination of applications always the same? Do you sometimes wish to deploy a subset of them or a superset?
- Are these applications managed by a single team or multiple teams?

Enable auto-sync and self-heal everywhere

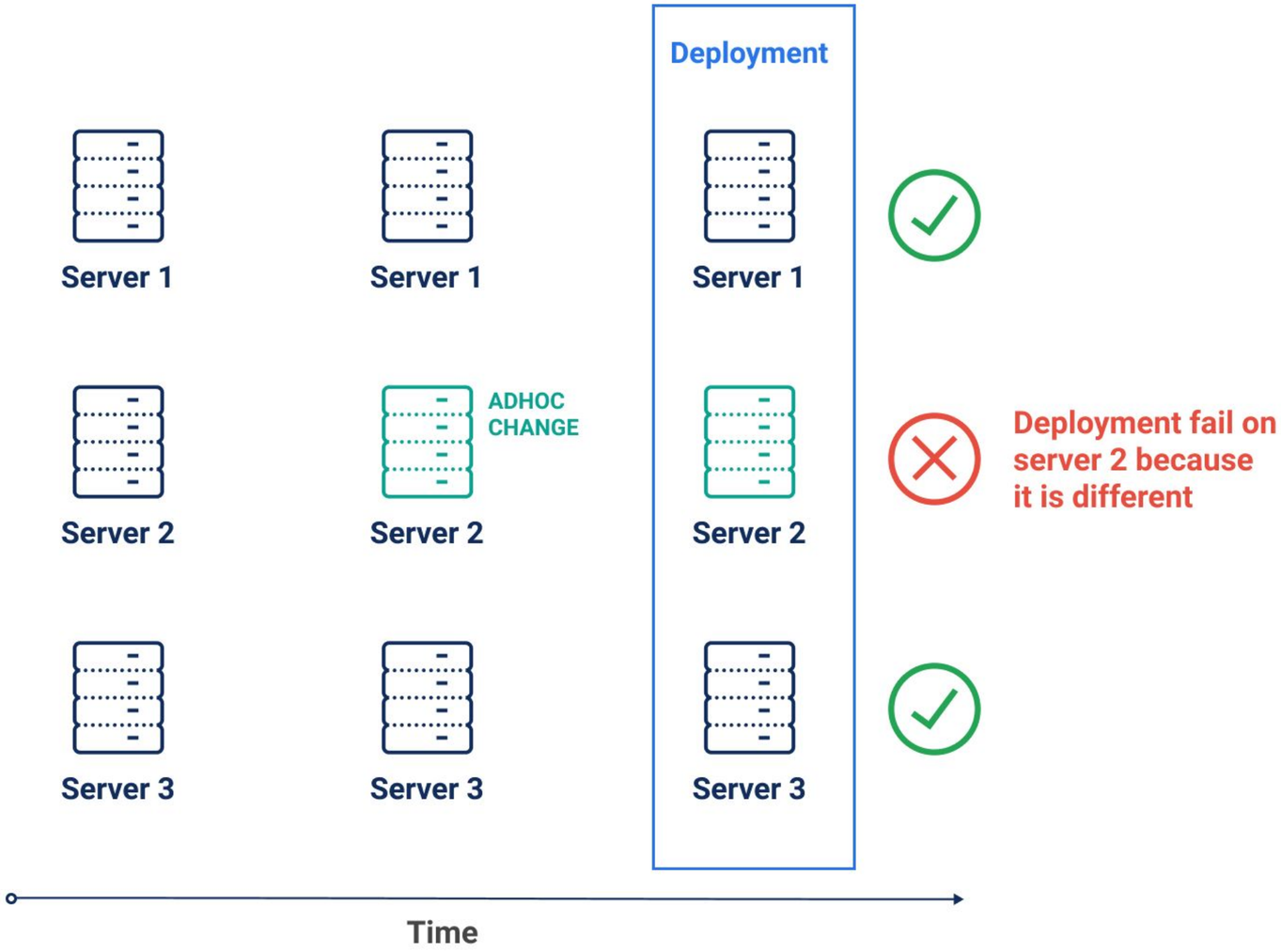


Abusing CI as CD

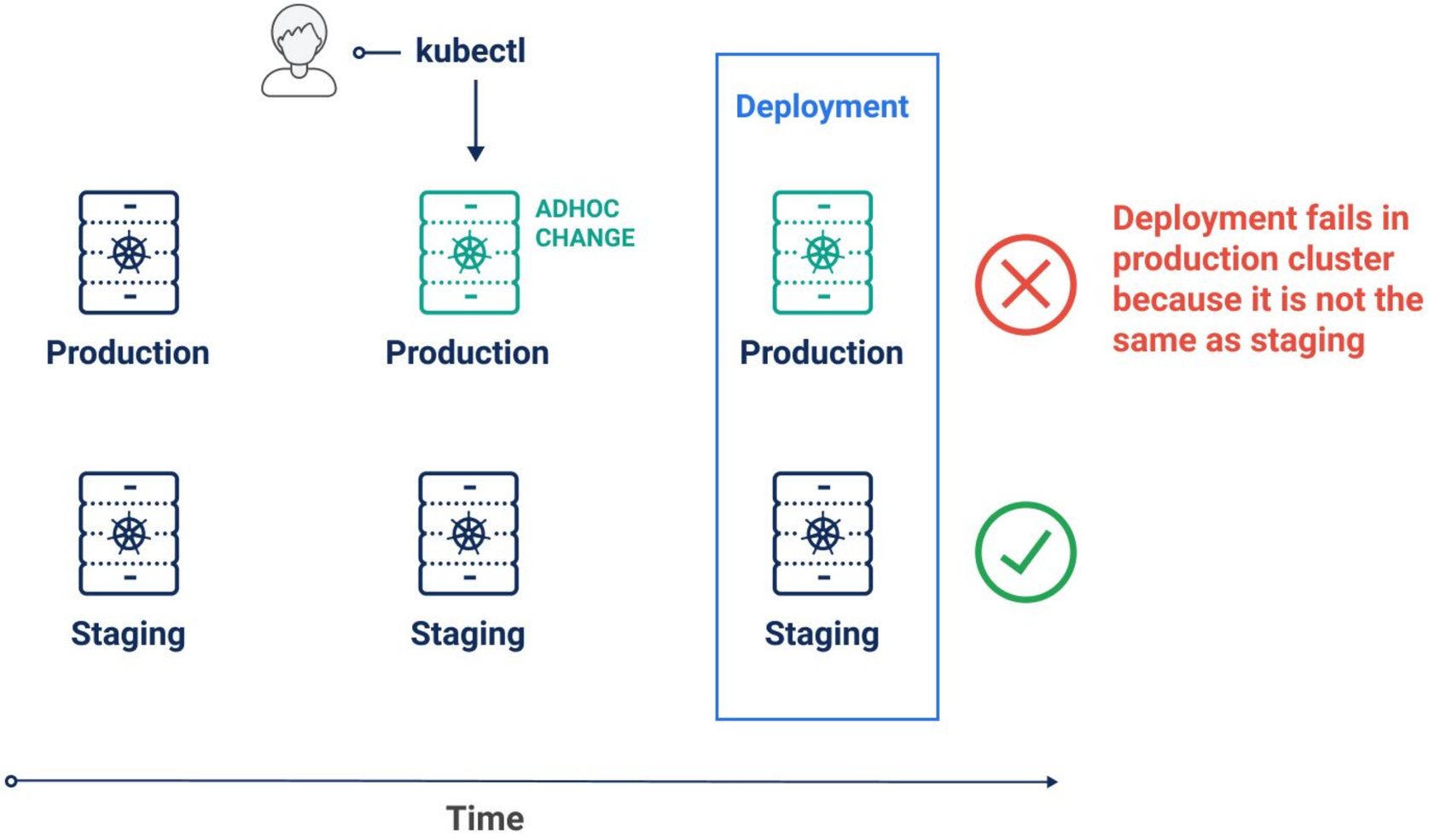


With Argo CD

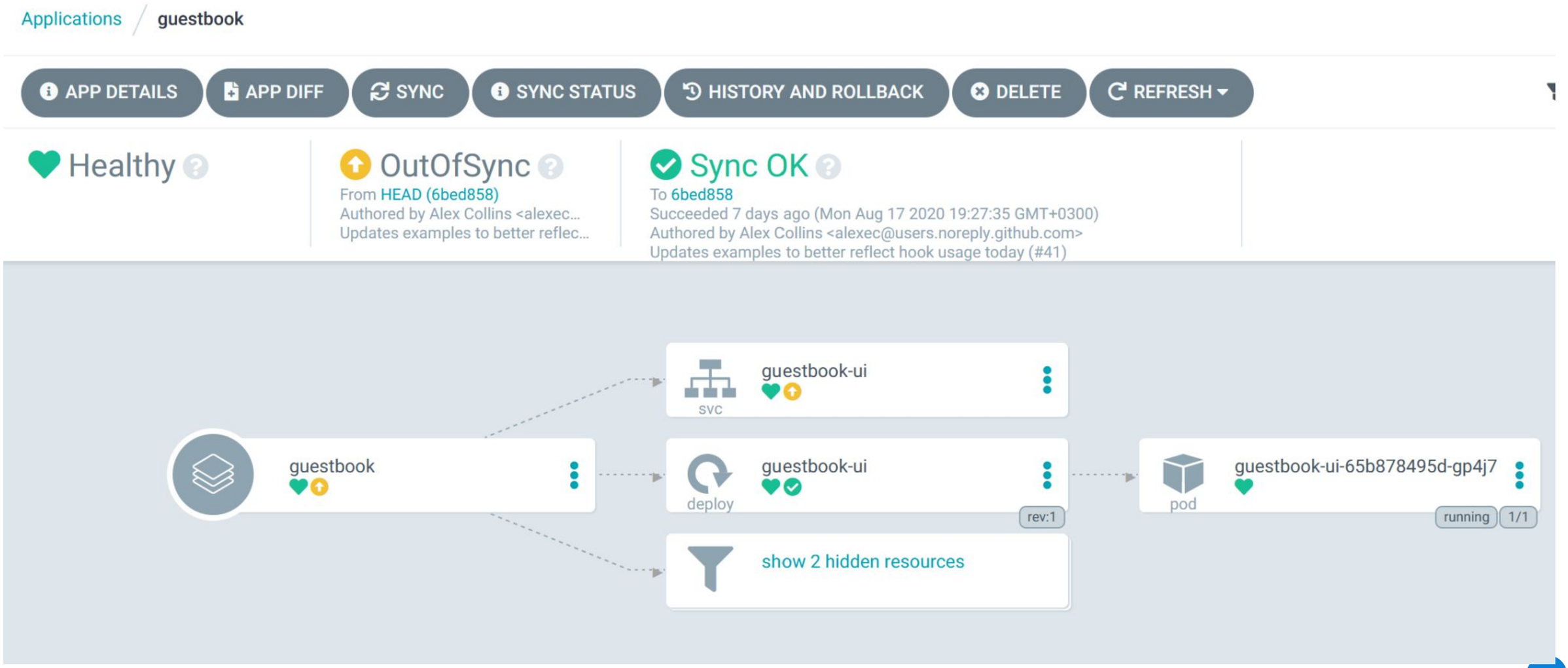
Configuration drift



Configuration drift in Kubernetes



Argo CD detects out-of-process changes



Argo CD shows out-of-process changes

SUMMARY PARAMETERS MANIFEST DIFF EVENTS

Compact diff Inline Diff

/Service/default/guestbook-ui

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    labels:
5      app.kubernetes.io/instance: guestbook
6    name: guestbook-ui
7  spec:
8    ports:
9      - port: 8080
10       targetPort: 80
11   selector:
12     app: guestbook-ui
```

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    labels:
5      app.kubernetes.io/instance: guestbook
6    name: guestbook-ui
7  spec:
8    ports:
9      - port: 80
10       targetPort: 80
11     - port: 8080
12       targetPort: 80
13   selector:
14     app: guestbook-ui
```

Every time you disable auto-sync/self-heal you lose the biggest advantage of Argo CD

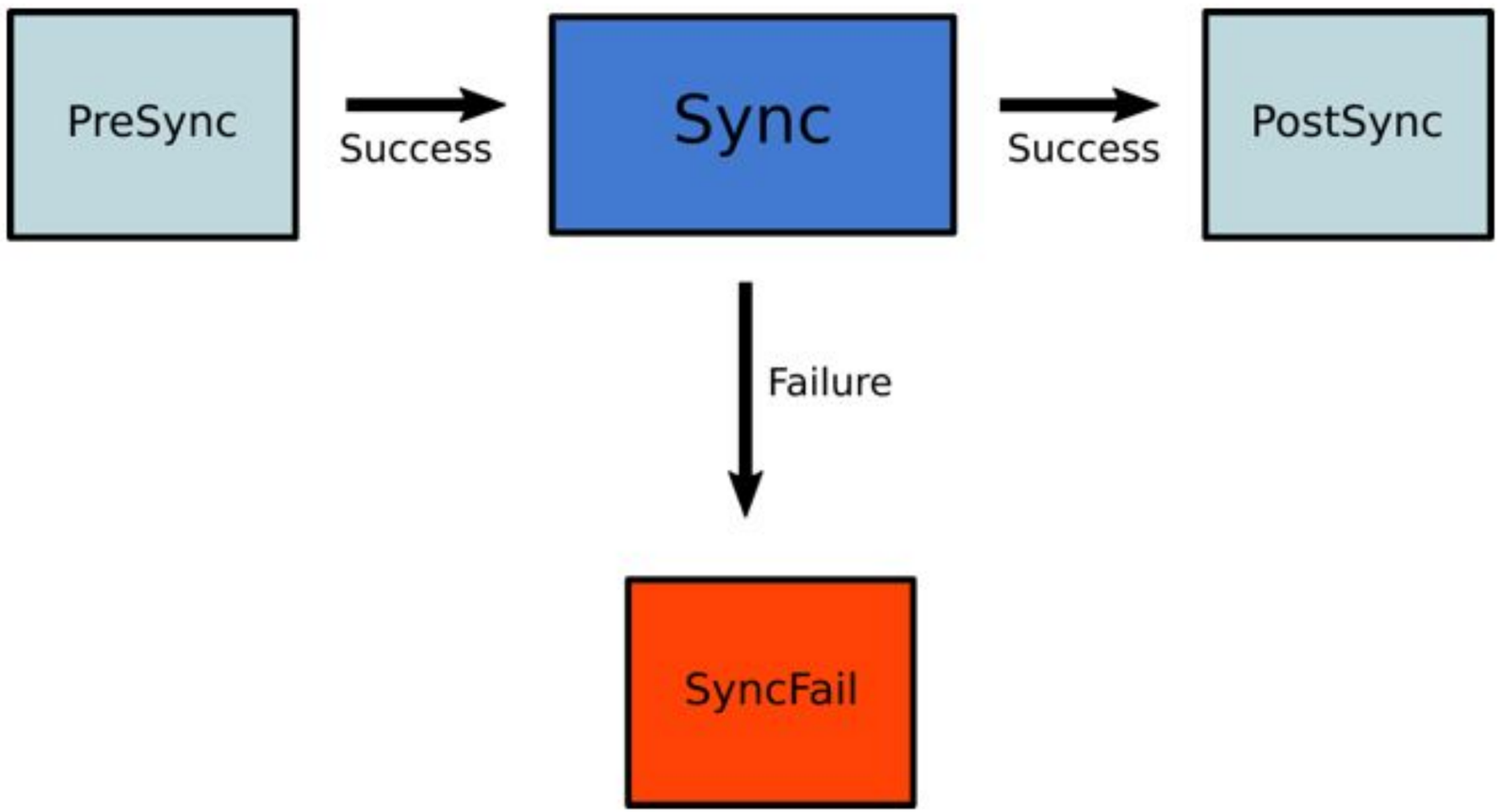
Enable Self heal -> solve configuration drift completely

Understand sync phases and waves

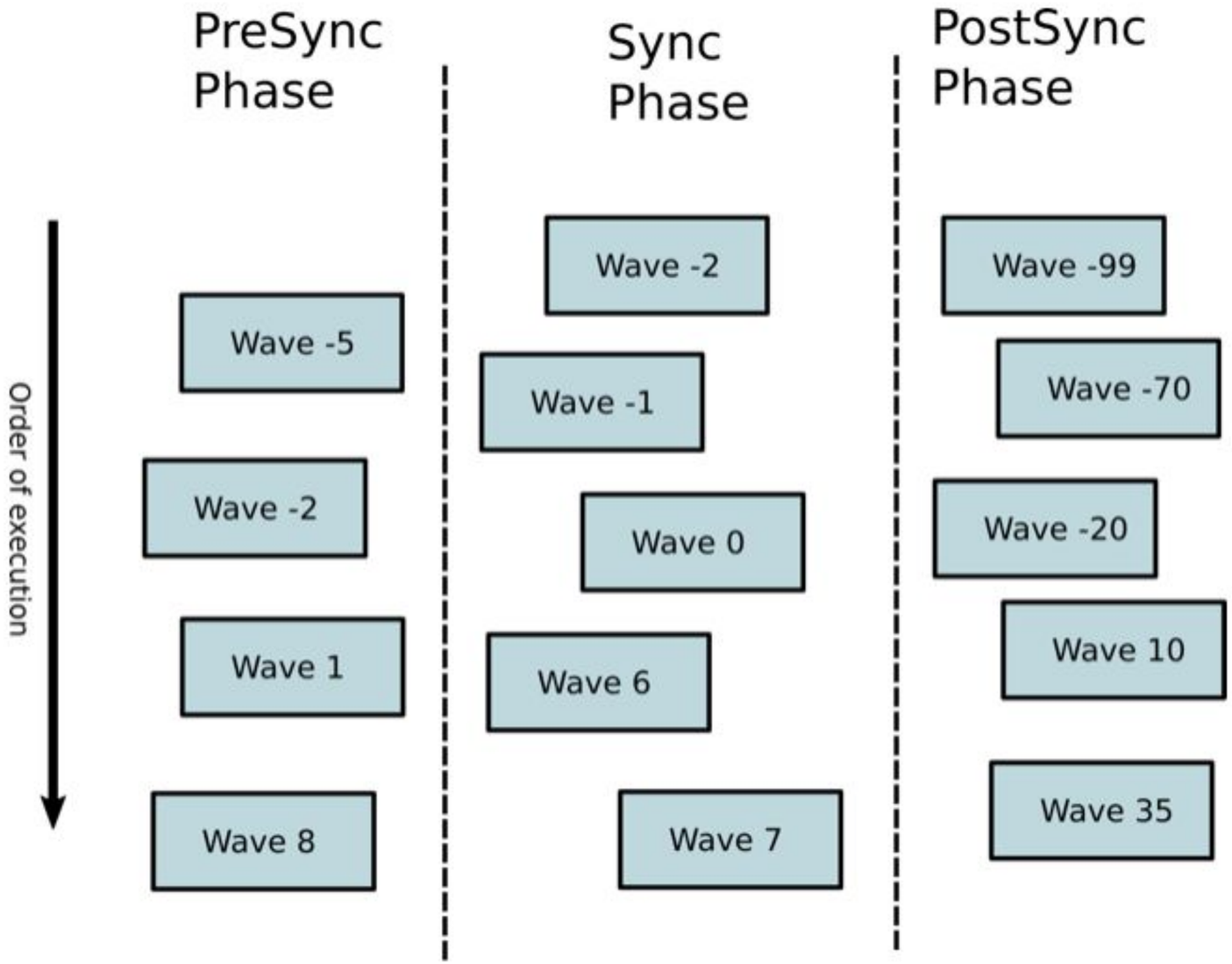
Resource ordering

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: grafana-notify
  annotations:
    argocd.argoproj.io/hook: PreSync
    argocd.argoproj.io/hook-delete-policy: HookSucceeded
    argocd.argoproj.io/sync-wave: '-1'
```

Understand phase lifecycle

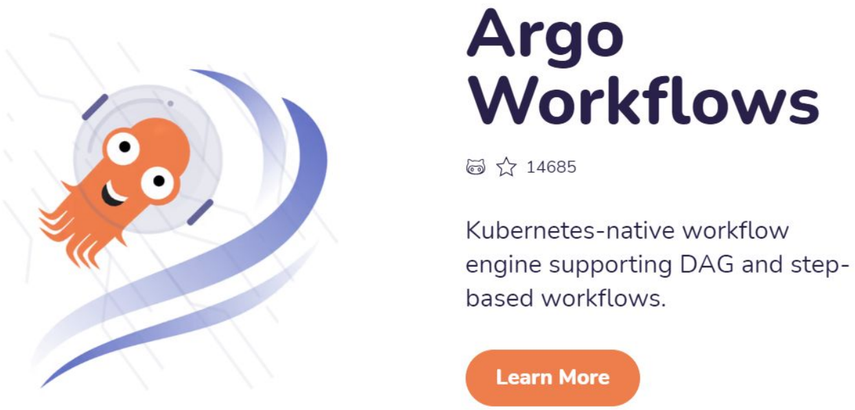


Waves are scoped inside a phase

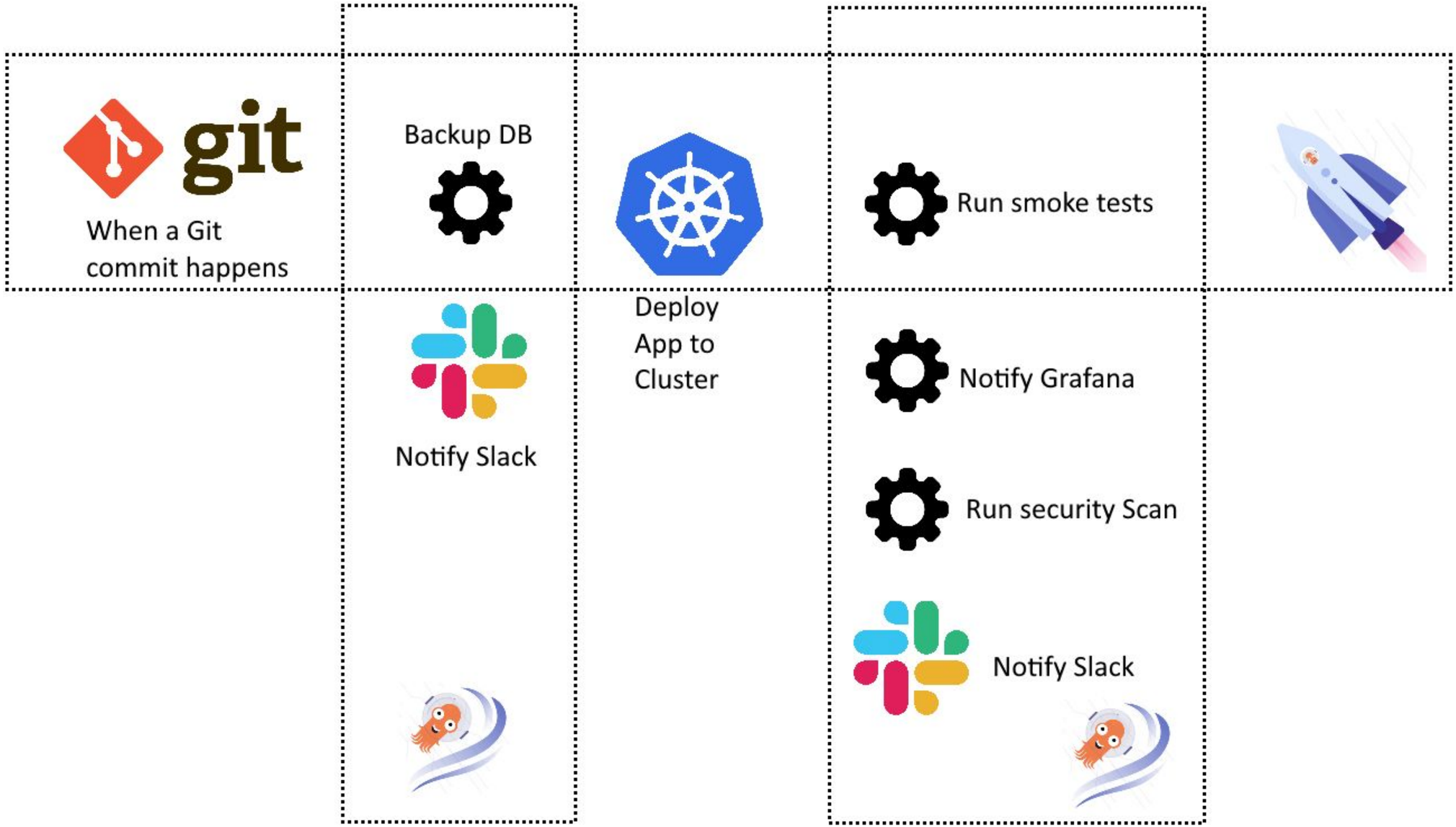


Recommendation

1. Start with **ONLY** sync waves
2. For more control use waves AND phases
3. For more advanced scenarios use Argo Workflows



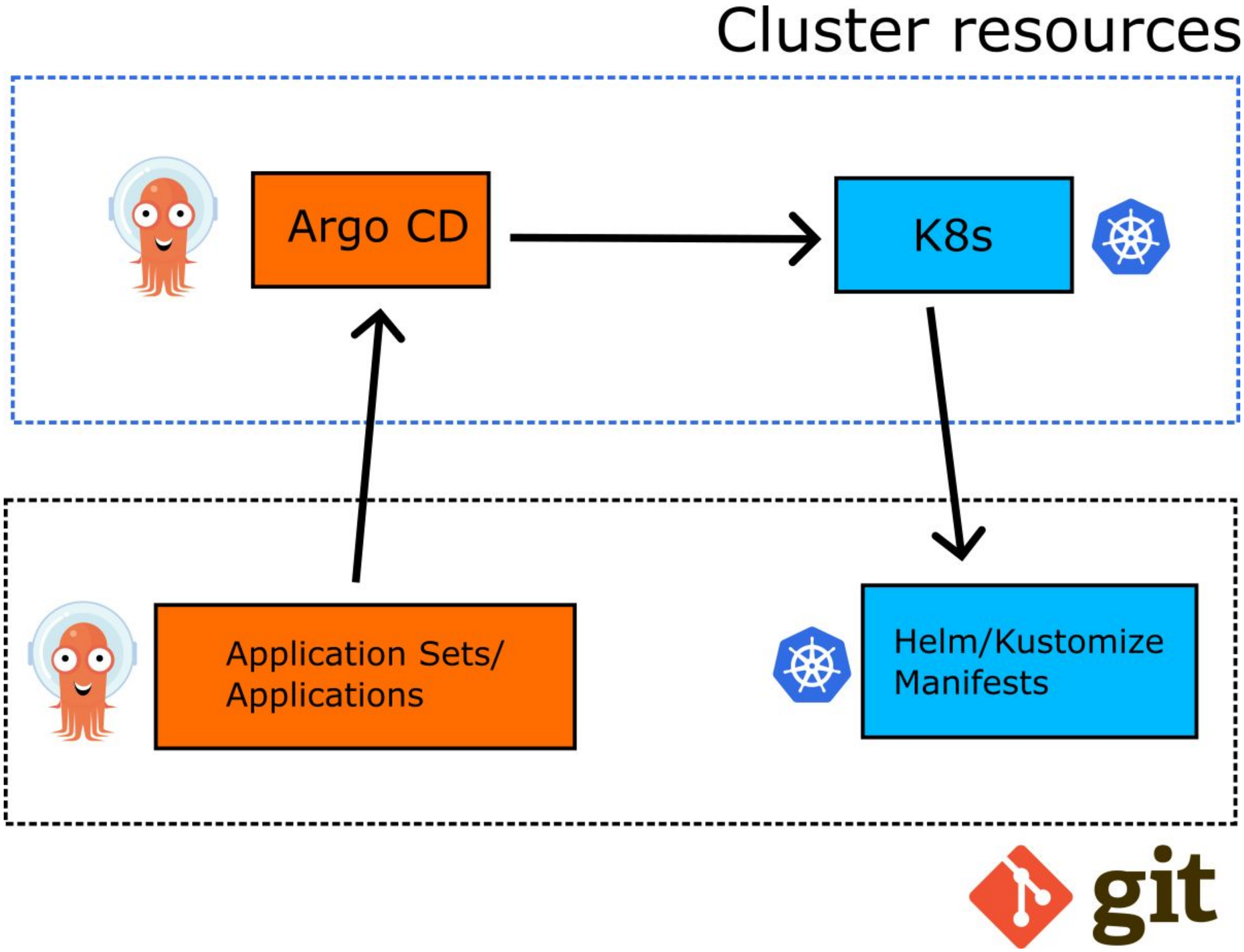
Argo CD and Argo Workflows

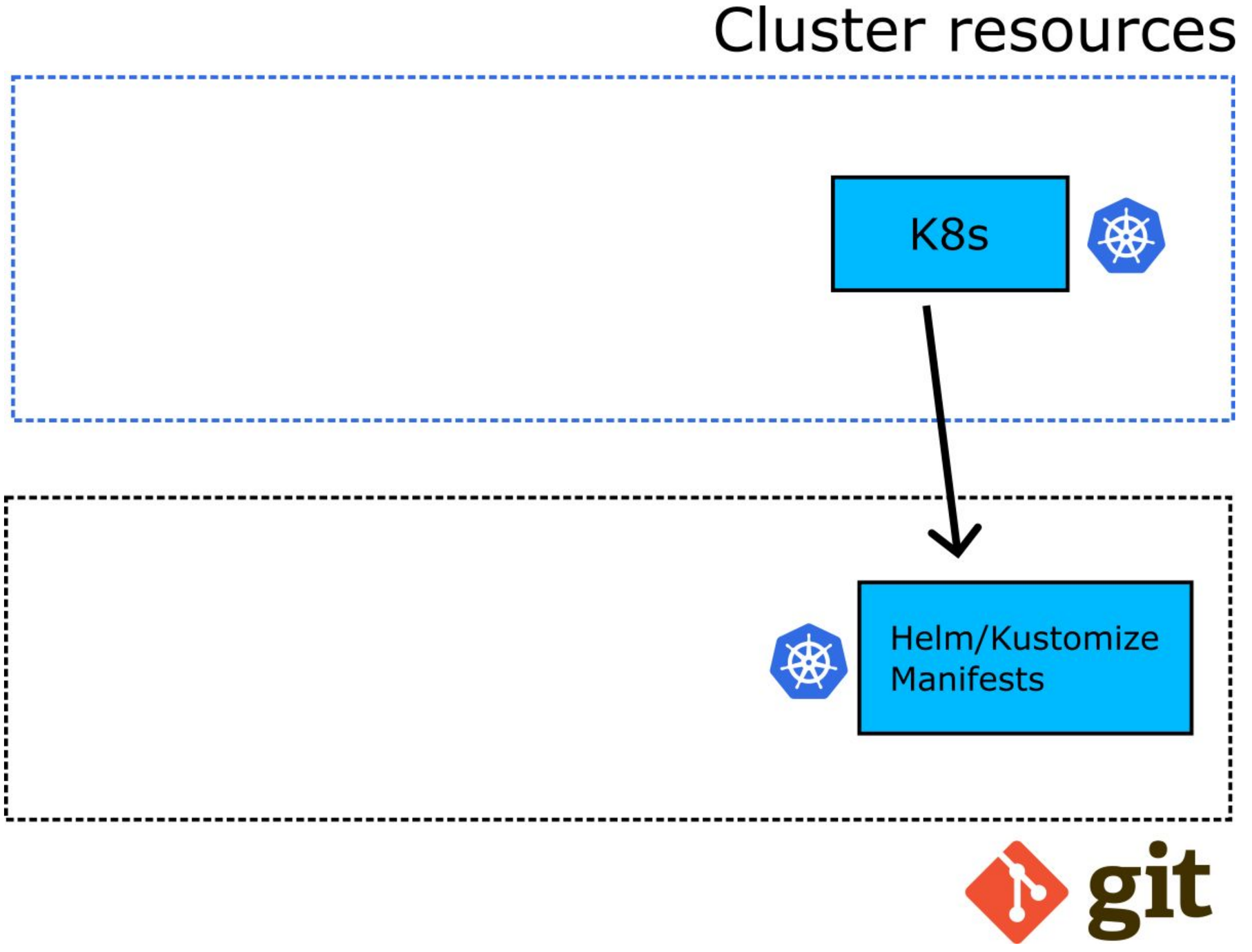


Common pitfalls

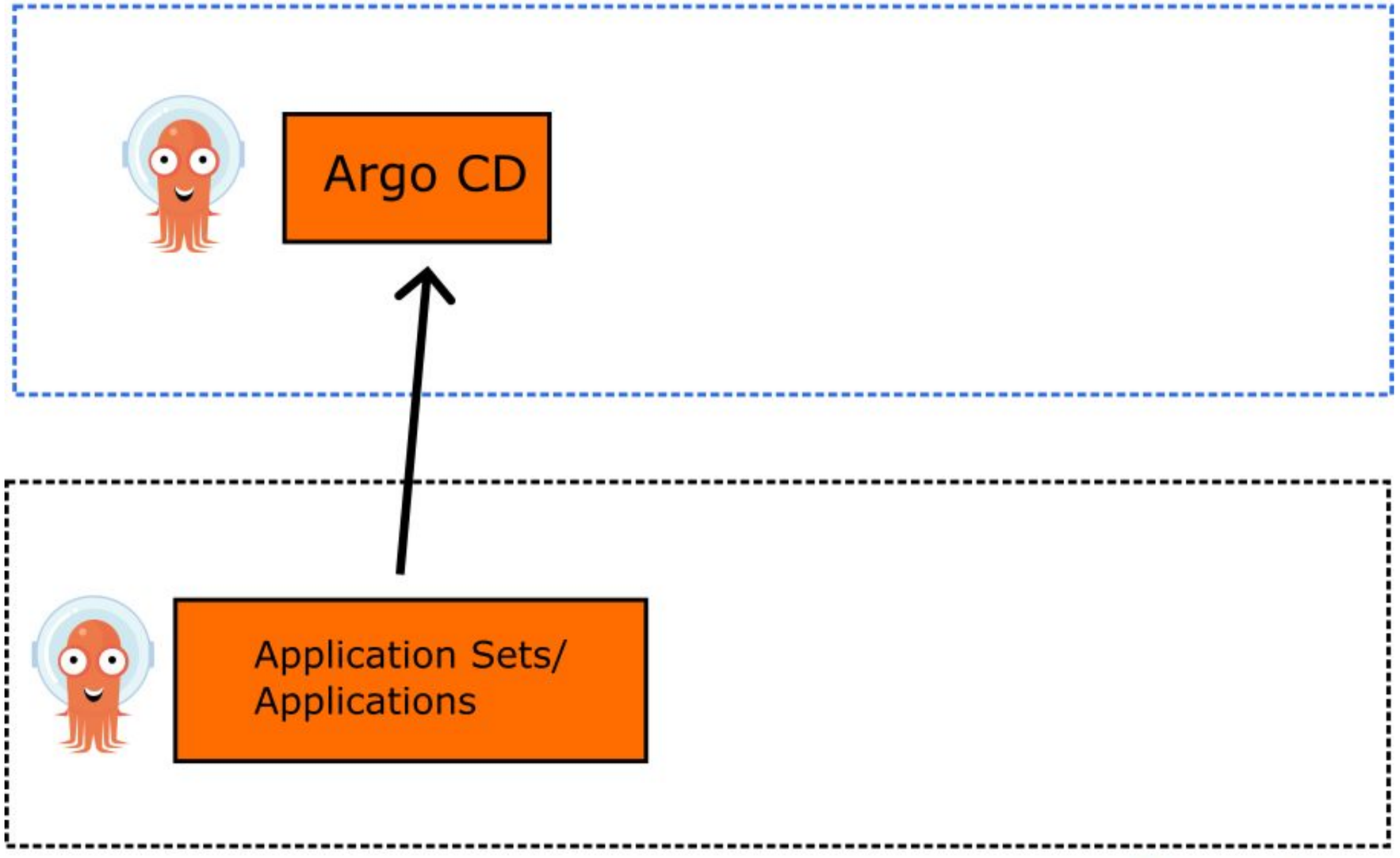
1. “I want Argo CD to run this only the first time it deploys an application”
2. “I want X to happen when an application is deployed and Y to happen when an application is upgraded”.
3. “I want to use pre-sync hooks for DB migrations that run only once”
4. “I want to skip this sync hook when a developer clicks the sync button”

Finalizers


Cluster resources
Argo CD
K8s
Application Sets/
Applications
Helm/Kustomize
Manifests
git

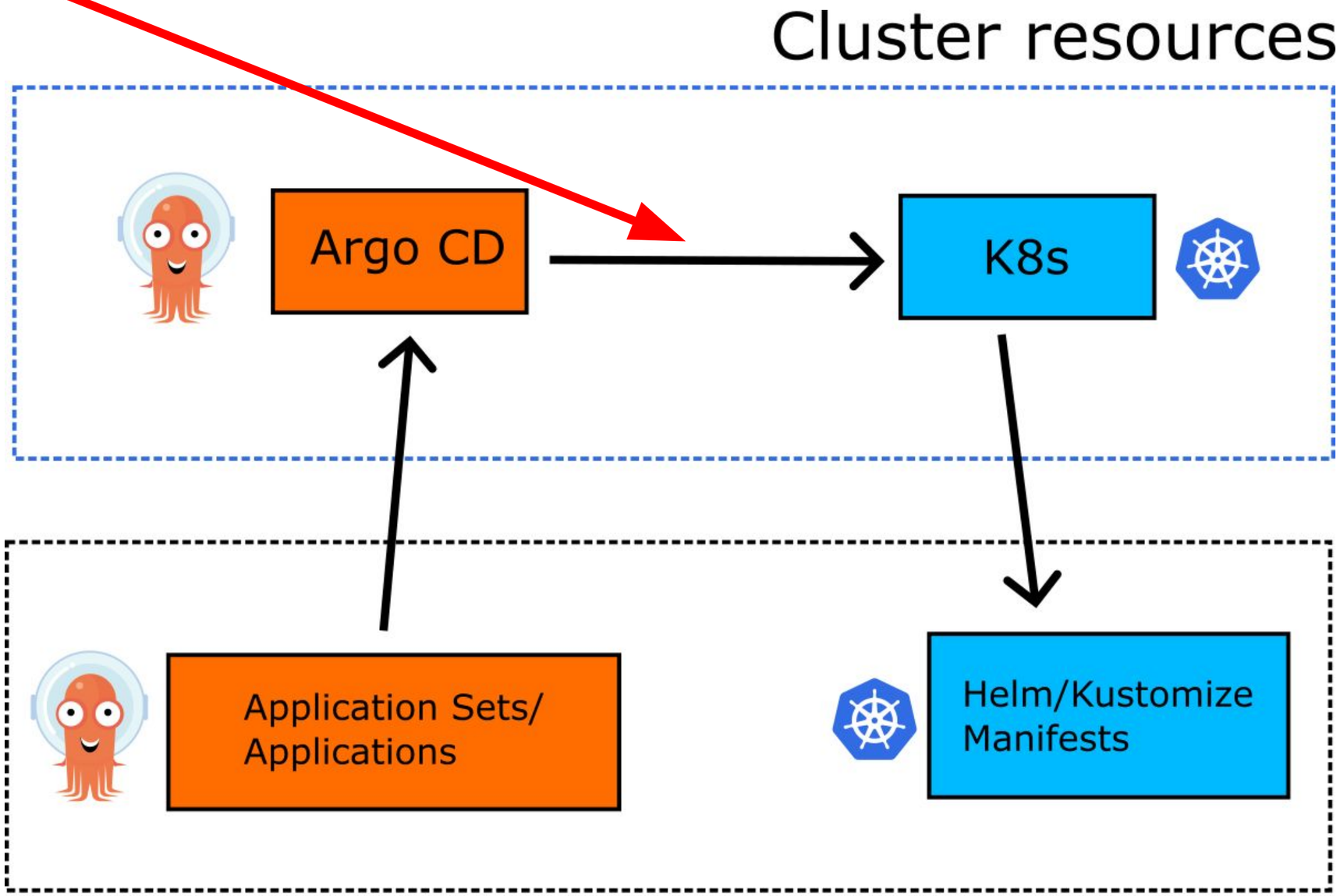

Cluster resources
K8s
Helm/Kustomize
Manifests
git

Cluster resources



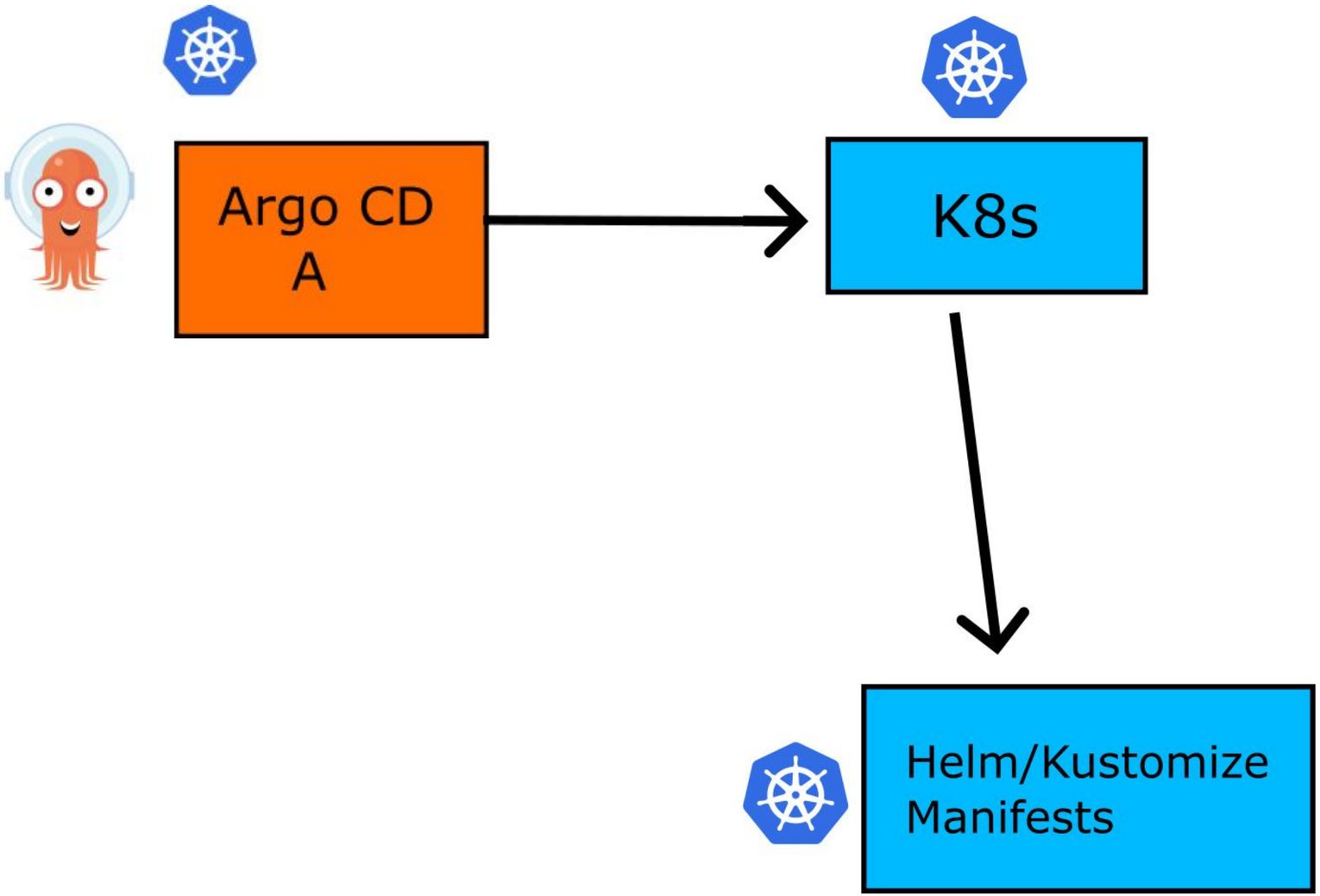
git

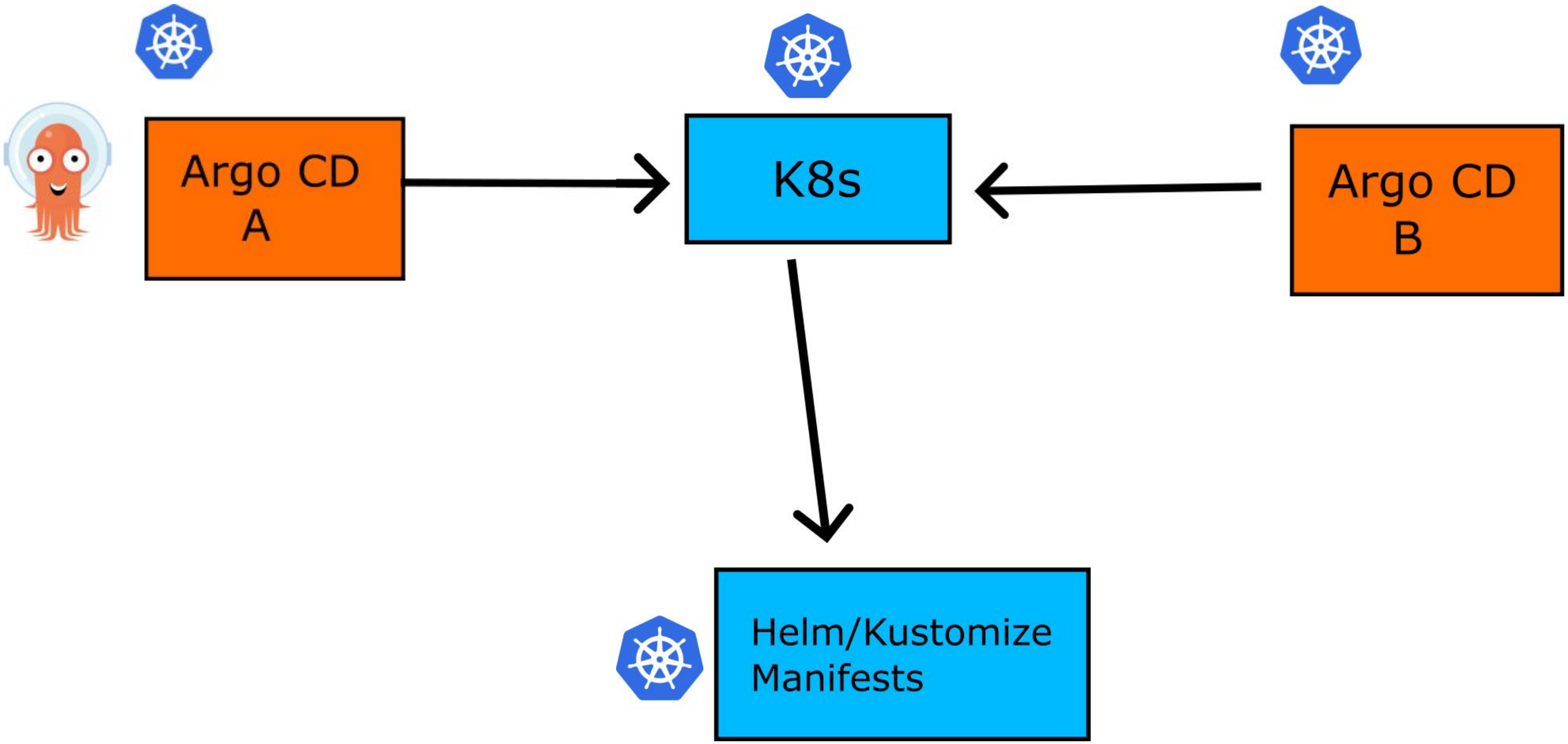
Finalizer

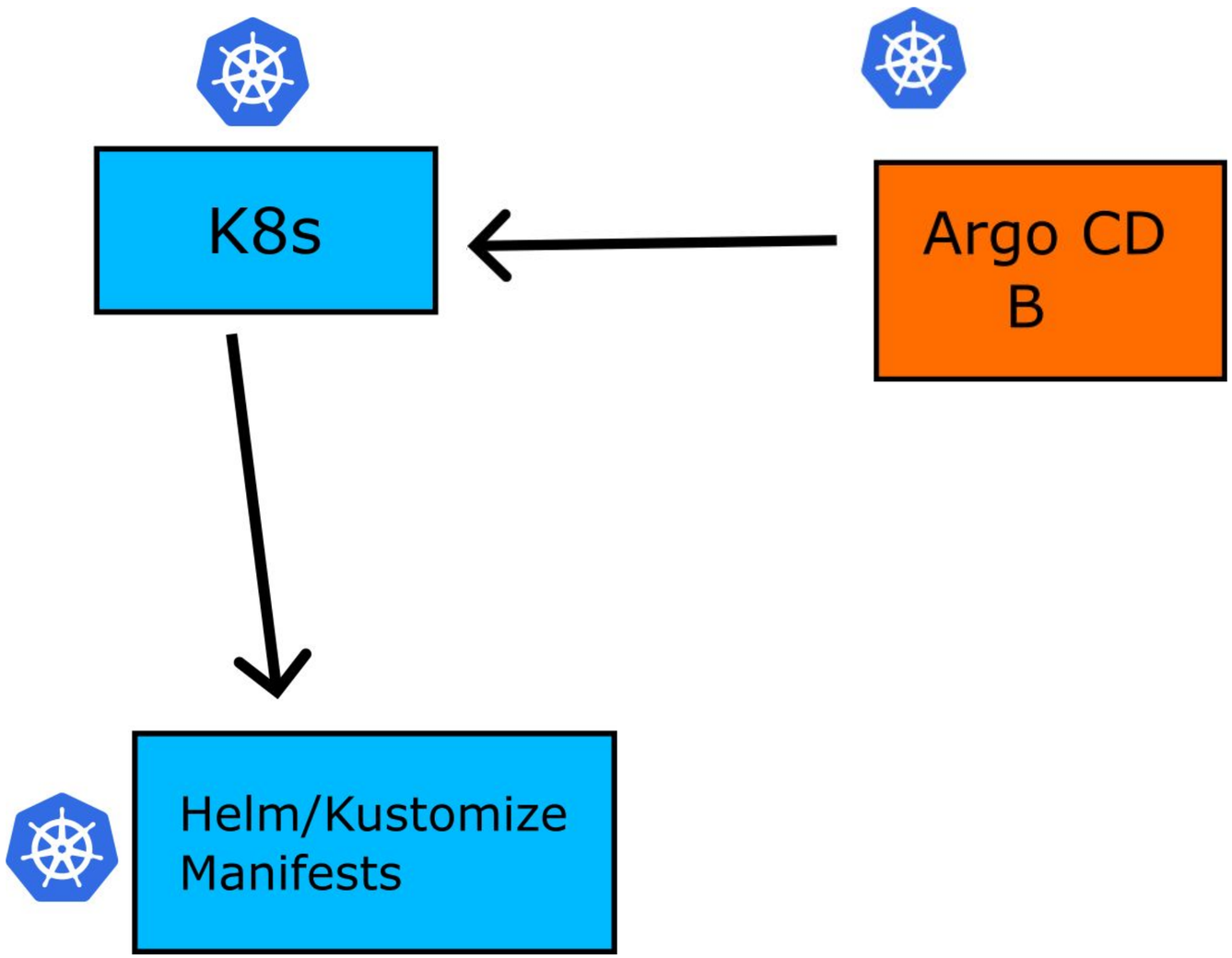

Cluster resources
Argo CD
K8s
Application Sets/
Applications
Helm/Kustomize
Manifests

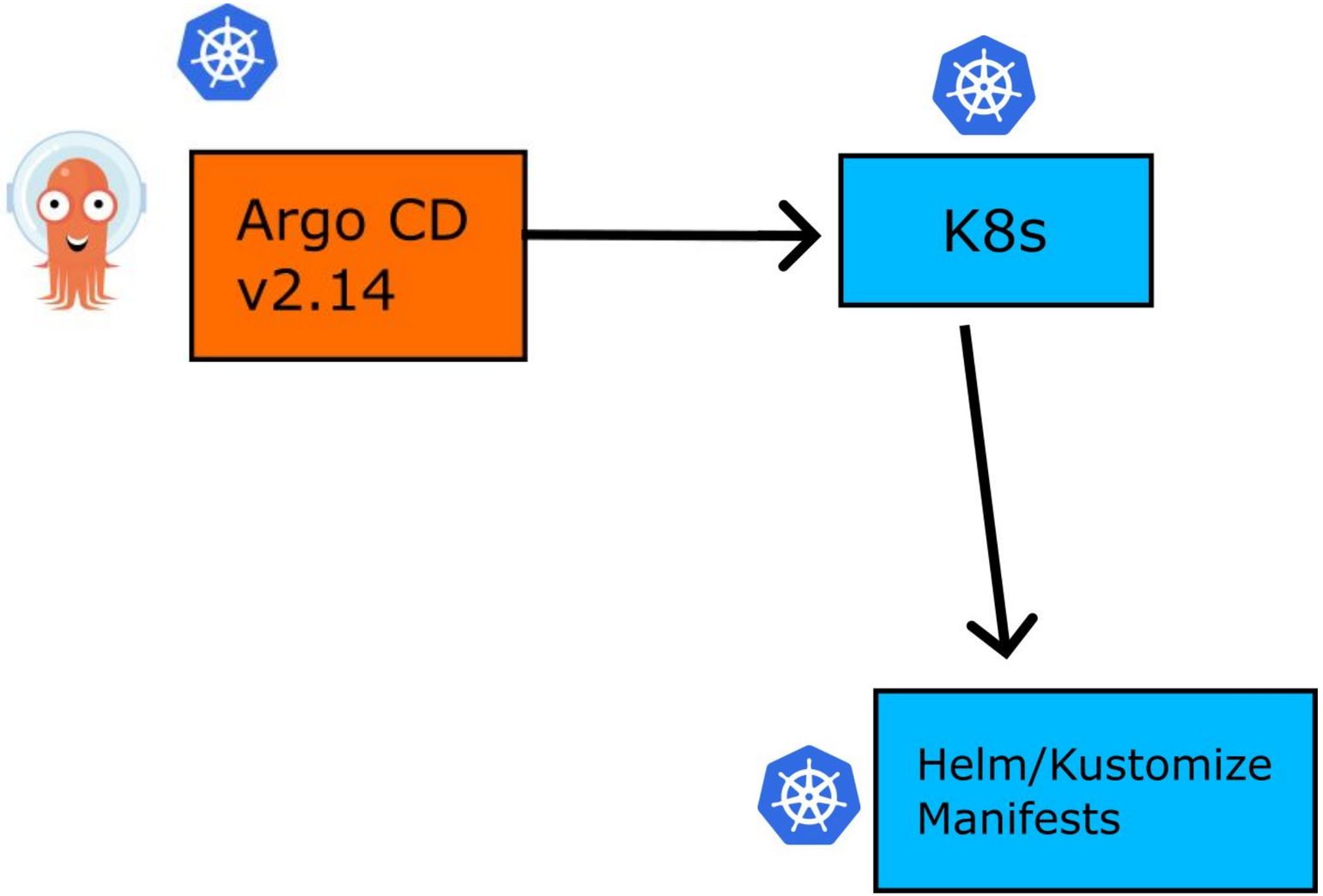
Use Finalizers

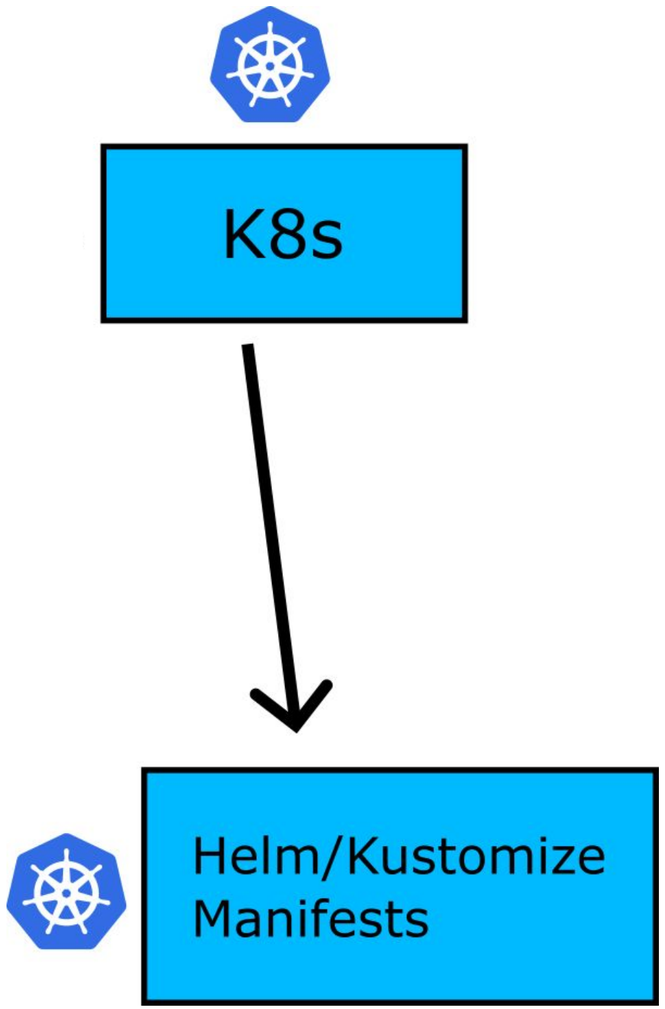
1. Migrate apps to different Argo CD instance (no downtime)
2. Upgrade Argo CD version (or K8s version) of admin cluster

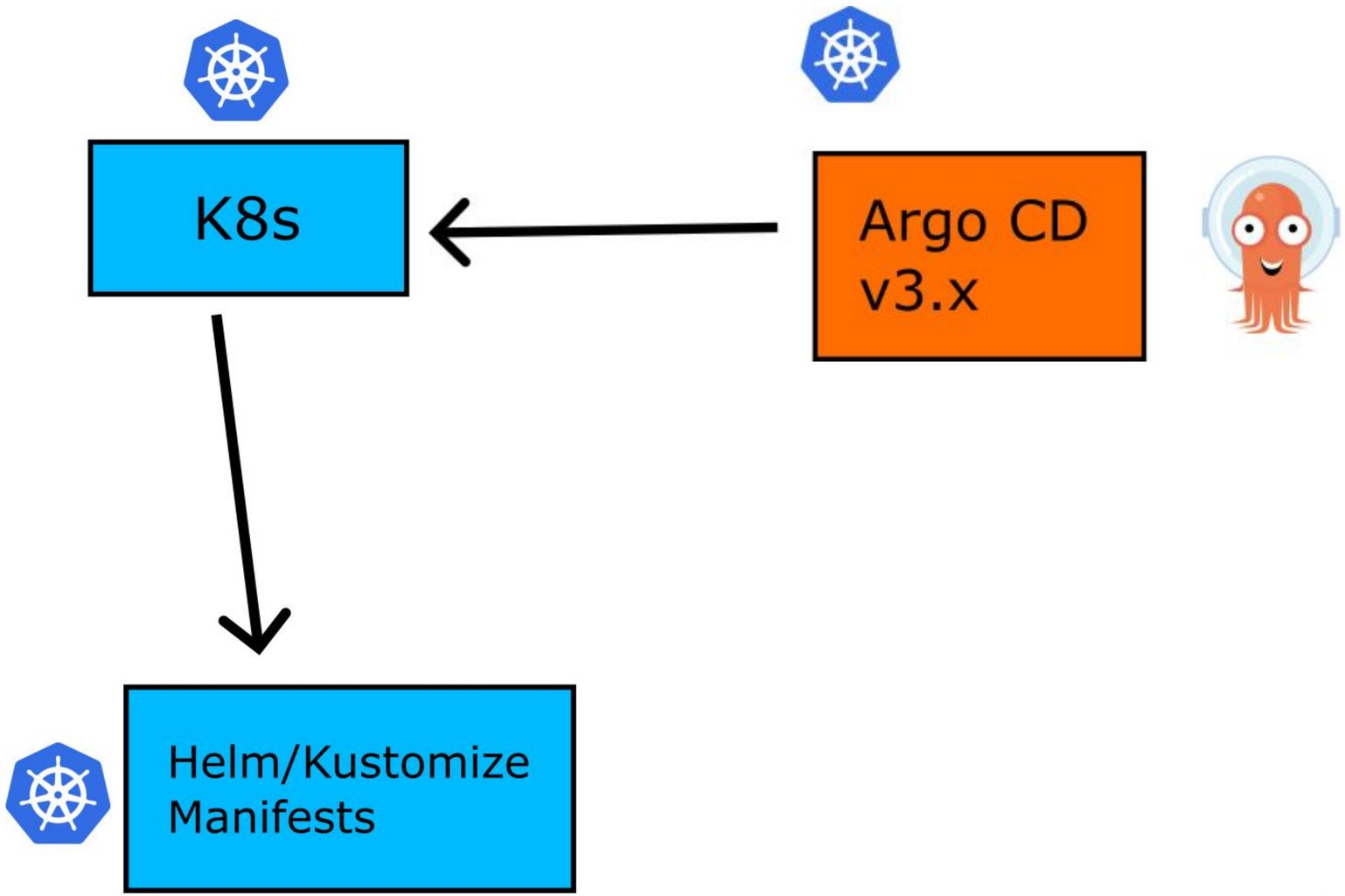

Argo CD
A
K8s
Helm/Kustomize
Manifests

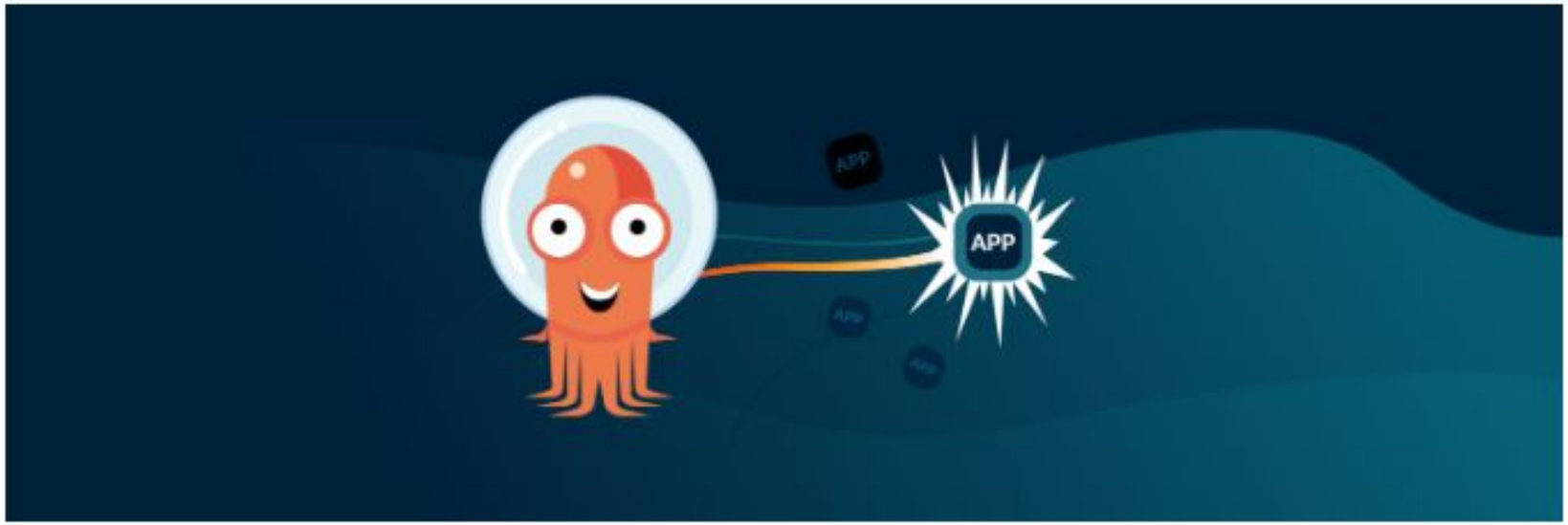

Argo CD
A
K8s
Argo CD
B
Helm/Kustomize
Manifests


K8s
Argo CD
B
Helm/Kustomize
Manifests


Argo CD
v2.14
K8s
Helm/Kustomize
Manifests


K8s
Helm/Kustomize
Manifests


K8s
Argo CD
v3.x
Helm/Kustomize
Manifests



BEST PRACTICES

Everything You Ever Wanted to Know About Deletion and Argo CD Finalizers but Were Afraid to Ask

8 min read

https://codefresh.io/blog/argocd-application-deletion-finalizers/

Octopus Deploy

Enterprise support for Argo from Argo maintainers

Support when you need it and priority bug fixes for all Argo users across Argo CD, Argo Rollouts, Argo Workflows, and Argo Events.

Contact Support

https://octopus.com/support/enterprise-argo-support

Thank you!

Questions: kostis.kapelonis@octopus.com

GitOps/Argo CD certification learning.octopus.com

CNCF Slack https://slack.cncf.io/

Support:
https://octopus.com/support/enterprise-argo-support

Octopus Deploy