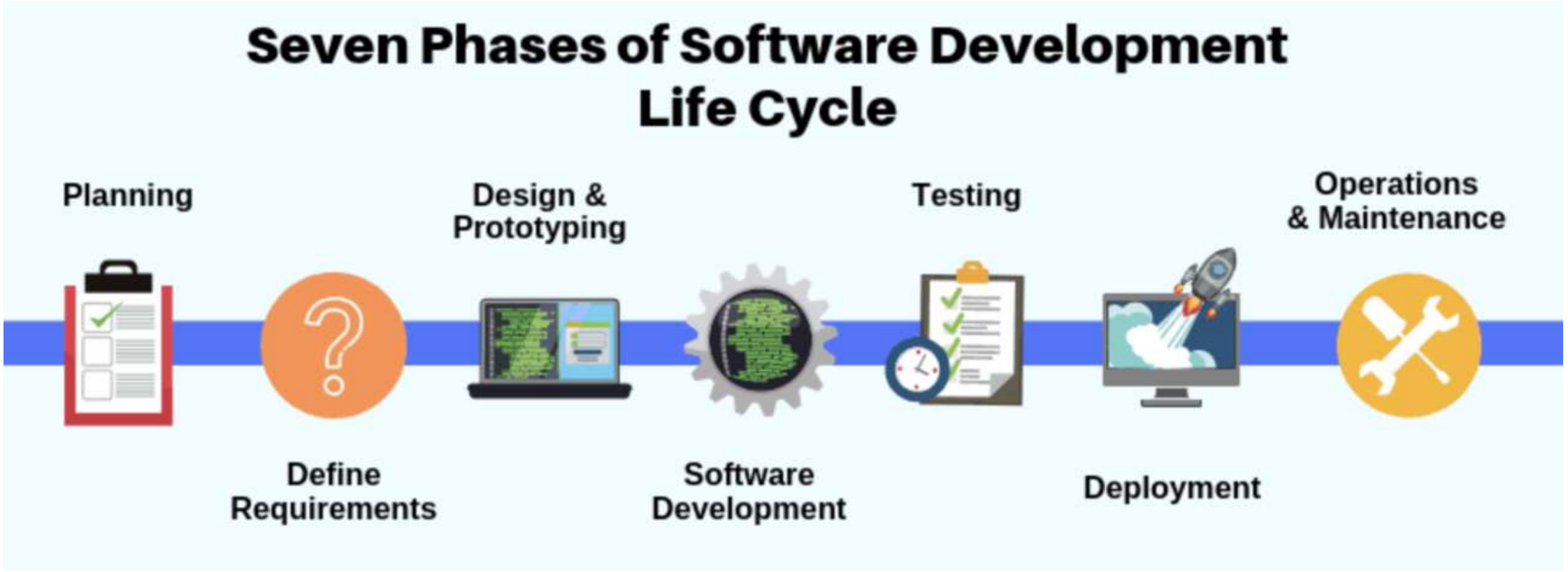
Implementing Delivery Strategies with Argo CD and Kubernetes

Selvaraj Kuppusamy
DevOps Engineer
Grootan Technologies

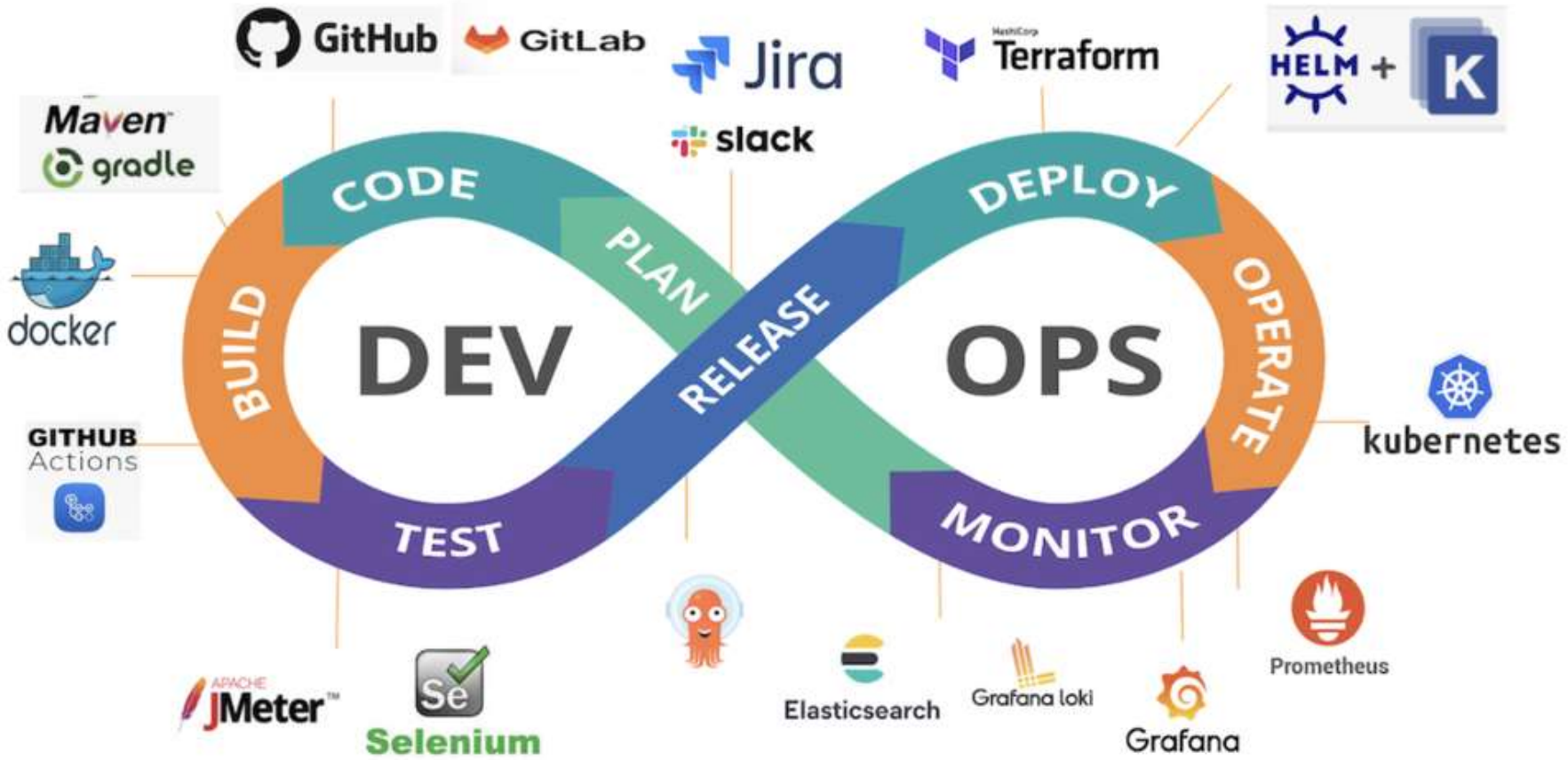
Agenda

- Traditional SDLC
- DevOps
  - Commonly available CI/CD Tools
- Microservices Deployment
  - Common challenges with traditional CI/CD Tools
- Argo CD
  - Why Argo CD
  - GitOps Pattern
  - Core concepts
    - Application
    - Project
    - ApplicationSet, etc..
- Demo

Traditional SDLC



DevOps Lifecycle

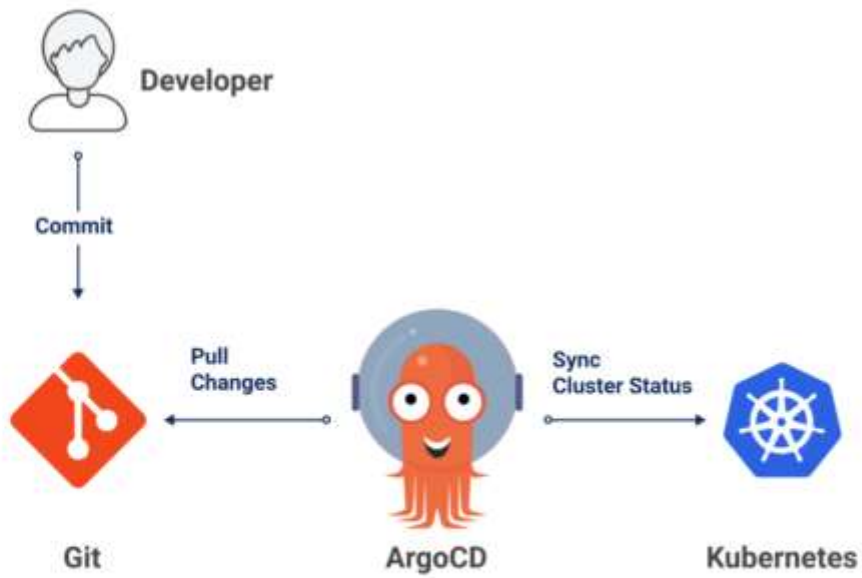


Challenges with other traditional CD tools

- **Complex Configuration Management:** Other CD tools may require complex scripting and manual setup of pipelines.
- **Limited GitOps Support:** Many CD tools aren't GitOps-based, requiring manual intervention to track changes and maintain state.
- **Lack of Visual Representation:** Tools like Jenkins, CircleCI, or GitLab CI/CD might not offer visual dashboards.
- **State Management and Rollbacks:** Some tools have poor state management and limited rollback options compared to GitOps tools like Argo CD.

ArgoCD

- ArgoCD Is a Gitops-based CD with pull model design



What is GitOps?

Infrastructure as Code,
Configuration as Code

Collaboration + Change
Mechanism (Code Review)

Continues Integration and
Continues Delivery Pipelines

Why ArgoCD

- Git as the source of truth.
  - Developer and DevOps engineer will update the Git code only.
- Keep your cluster in sync with Git.
- Easy rollback.
- More security : Grant access to ArgoCD only.
- Disaster recovery solution : You easily deploy the same apps to any k8s cluster.

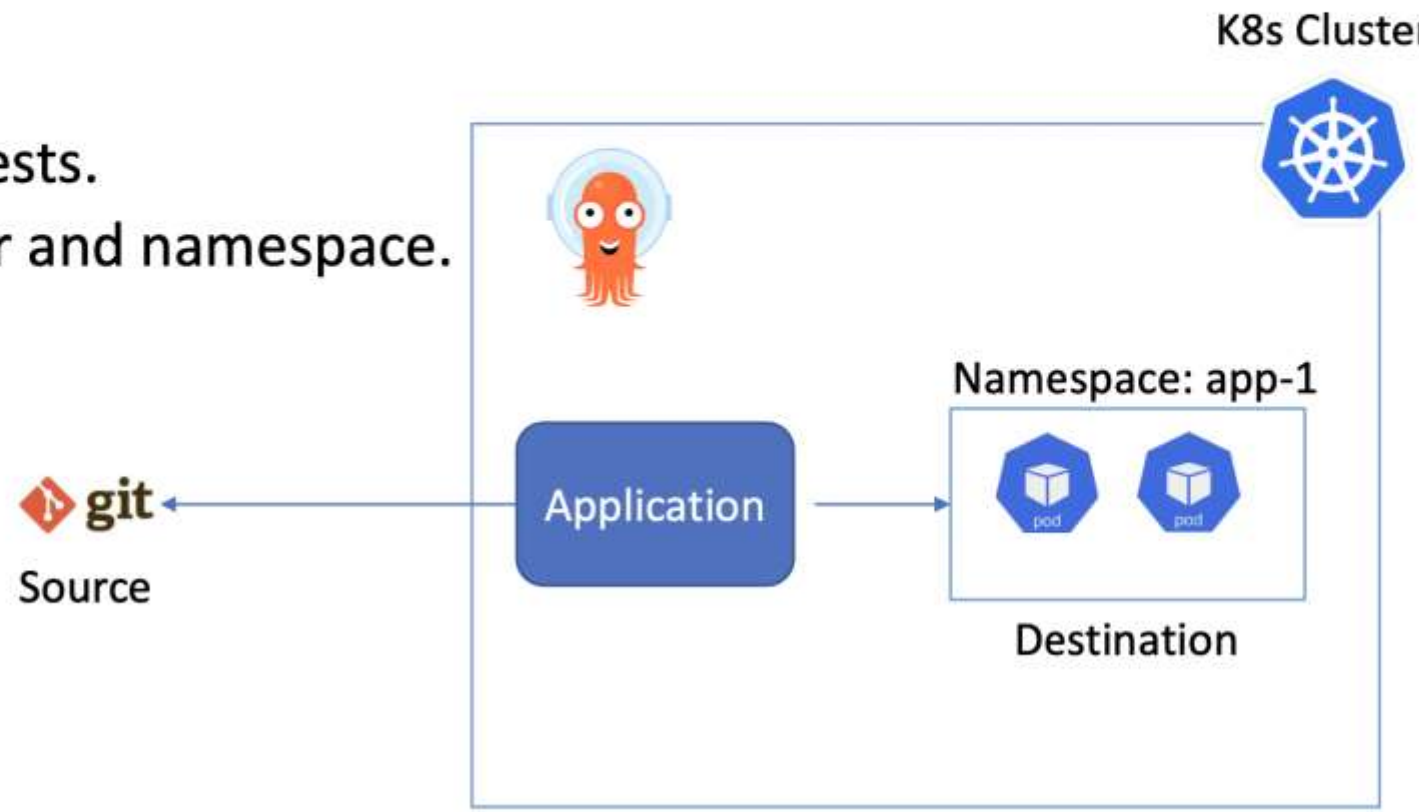
Application source - tools

- ArgoCD supports the below tools as source
  - Helm charts
  - Kustomize application
  - Directory of Yaml files.
  - Jsonnet

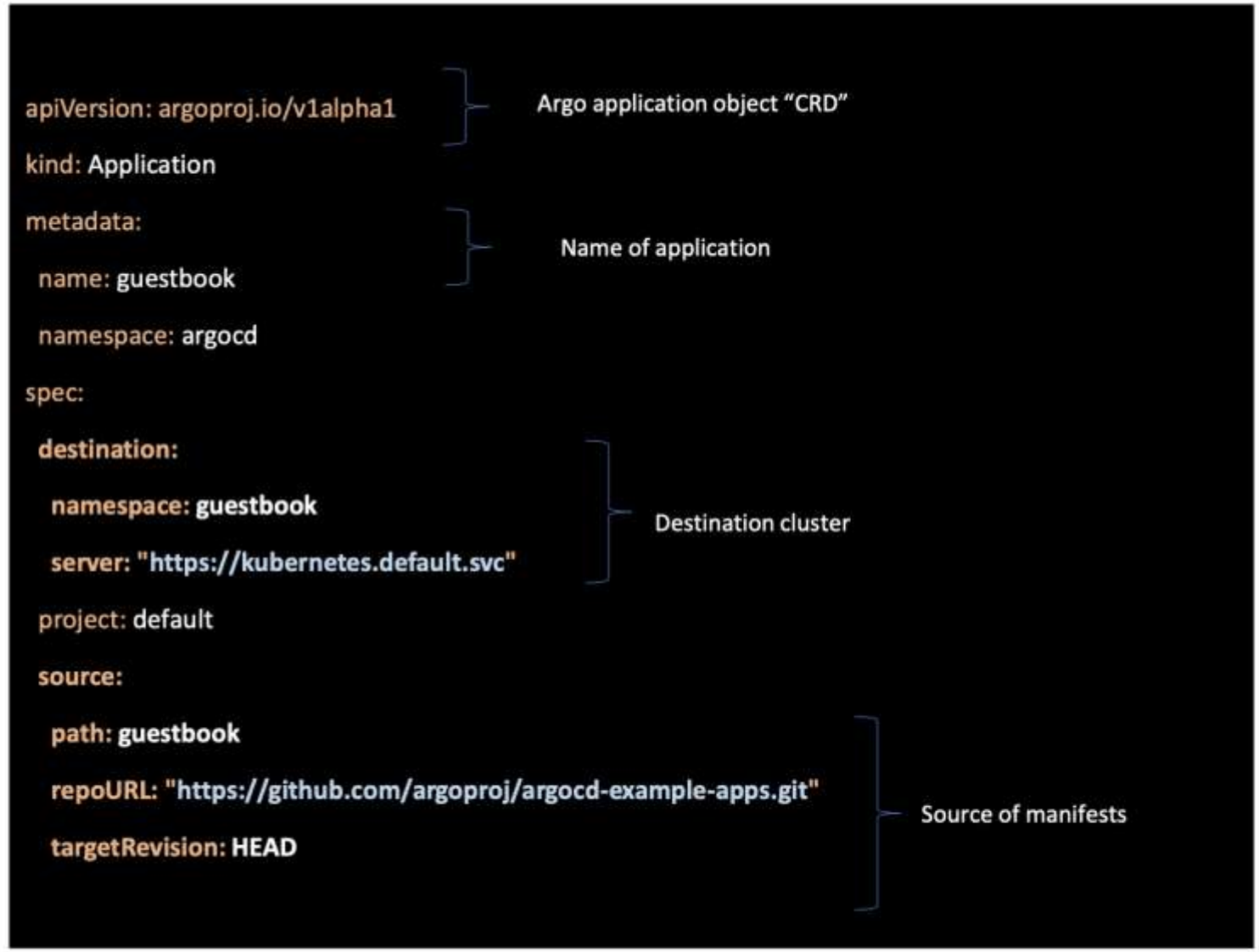
Core Concepts

Application

- Defines source and destination to deploy group of k8s resources.
  - Source : k8s manifests.
  - Destination: cluster and namespace.



Application
“declarative”



From Helm Repo

```yaml
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: guestbook
  namespace: argocd
spec:
  destination:
    namespace: guestbook
    server: "https://kubernetes.default.svc"
  project: default
  source:
    chart: sealed-secret
    repoURL: "https://bitnami-labs.github.io/sealed-secrets"
    targetRevision: 1.16.1  # For Helm, this refers to the chart version.
```

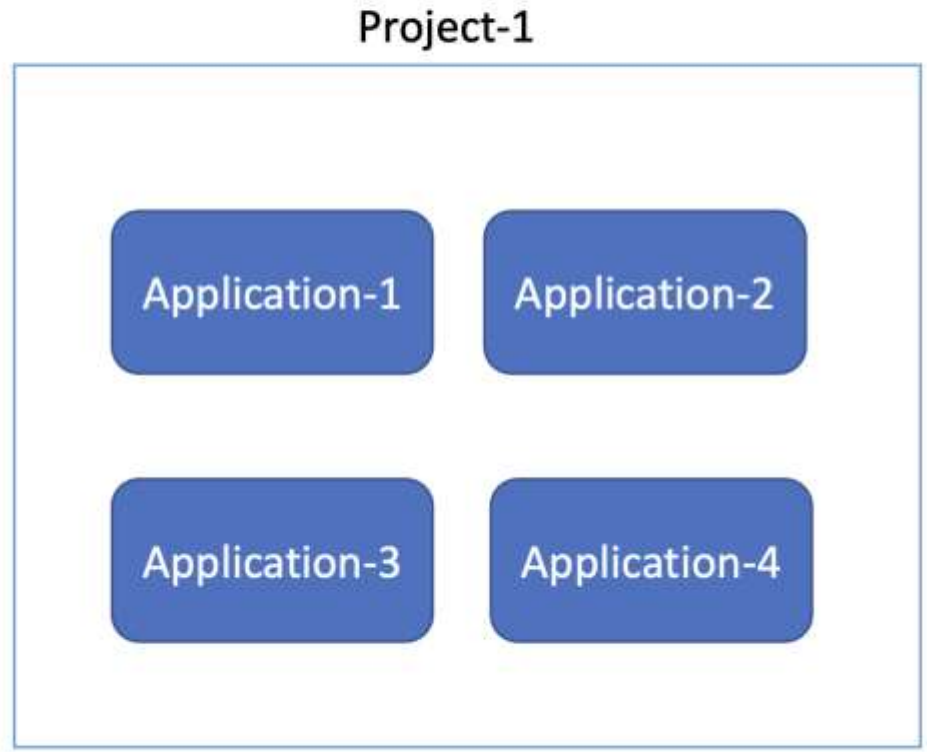
Chart name

Helm repo Url

Chart version

Project

- Projects provide a logical grouping of applications.
  - Useful when ArgoCD is used by multiple teams.
    - Allow only specific sources “trusted git repos”.
    - Allow apps to be deployed into specific clusters and namespaces.
    - Allow specific resources to be deployed “deployments, Statefulsets .. etc”.

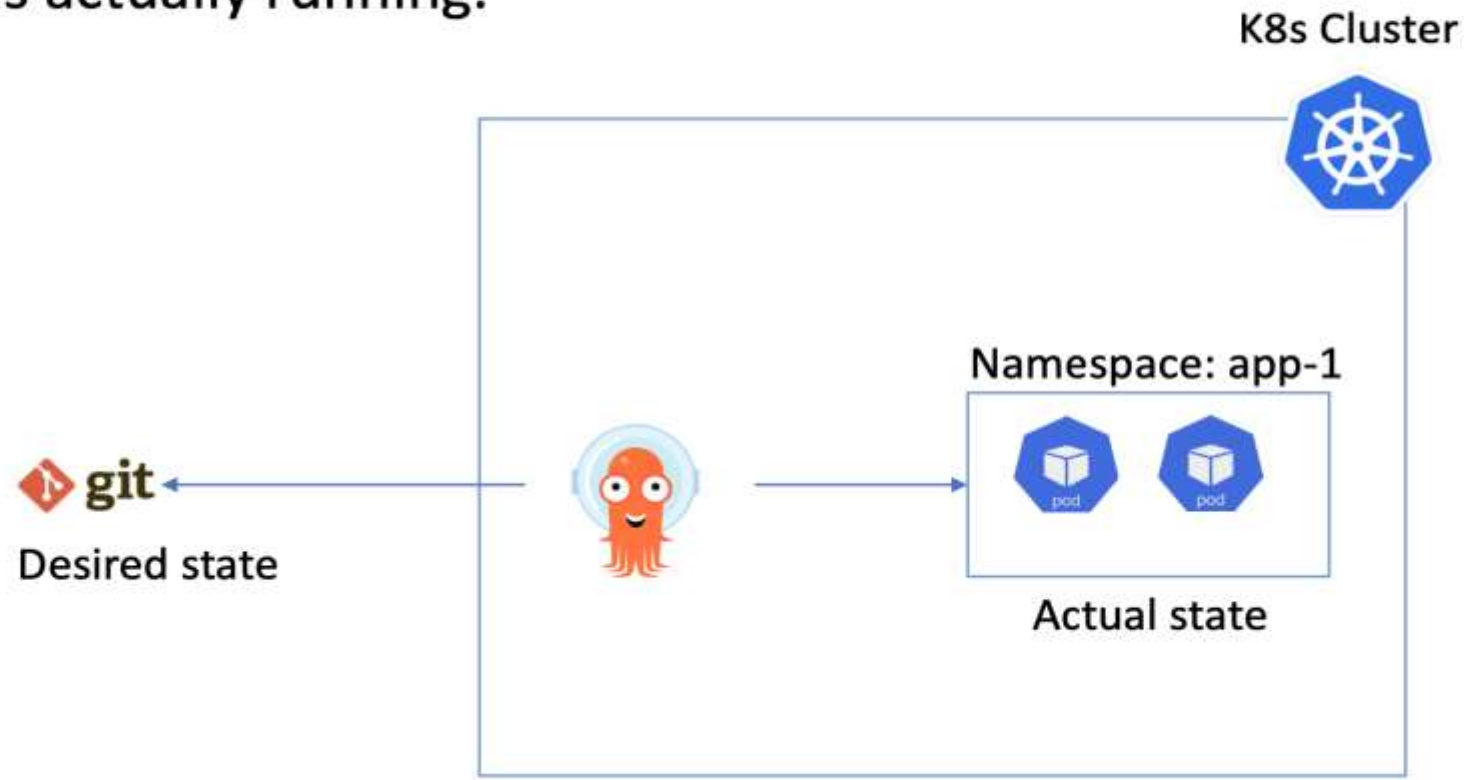


Project - Yaml

```yaml
apiVersion: argoproj.io/v1alpha1
kind: AppProject
metadata:
  name: project-1
  namespace: argocd
spec:
  description: project description
  sourceRepos:
  - "*"
  destinations:
  - server: "*"
    namespace: "*"
  clusterResourceWhitelist:
  - group: "*"
    kind: "*"
 namespaceResourceWhitelist:
  - group: "*"
    kind: "*"
```

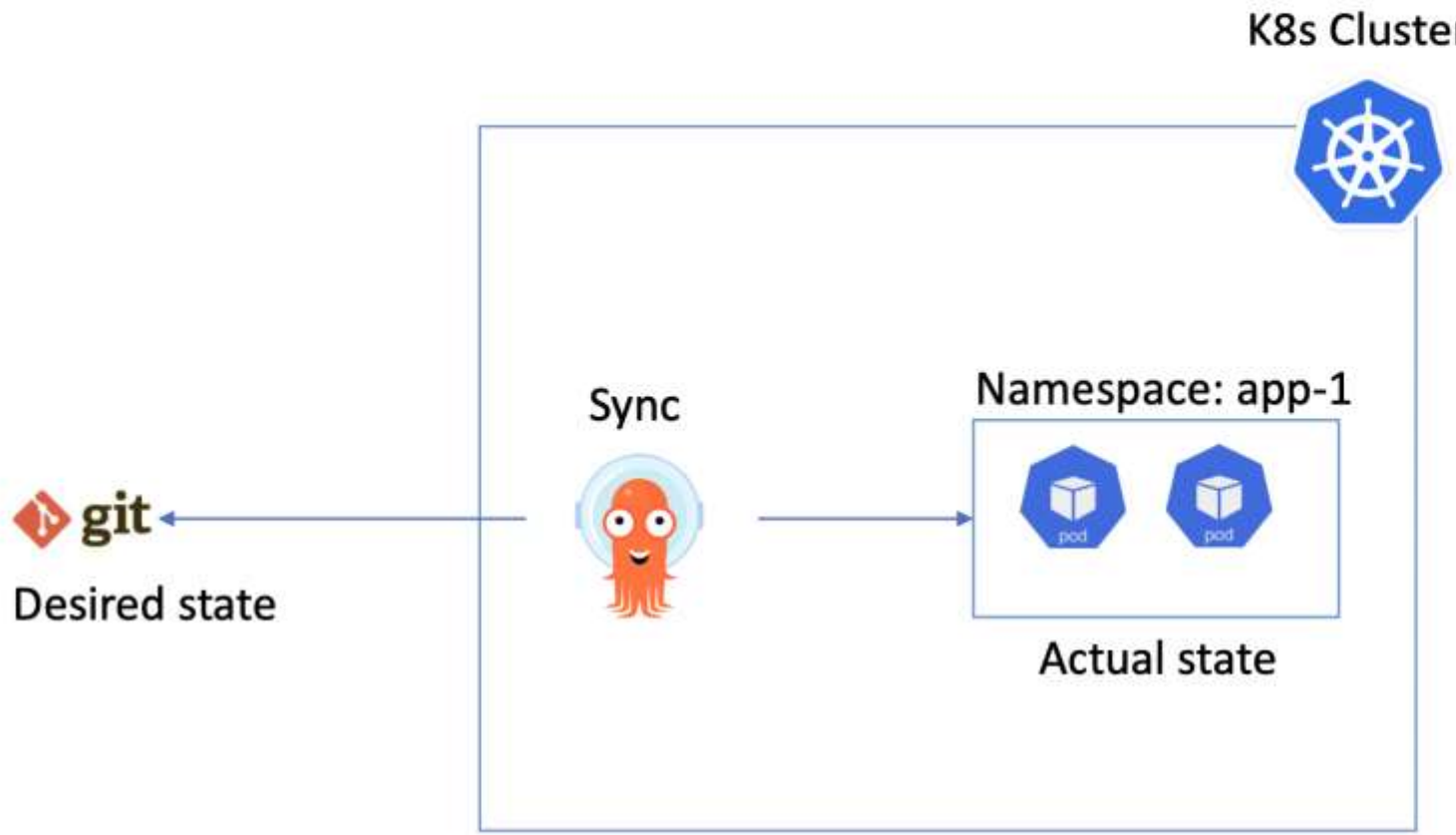
Desired state vs Actual state

- Desired state : described in git.
- Actual state: what is actually running.



Sync

- The process of making desired state = actual state.



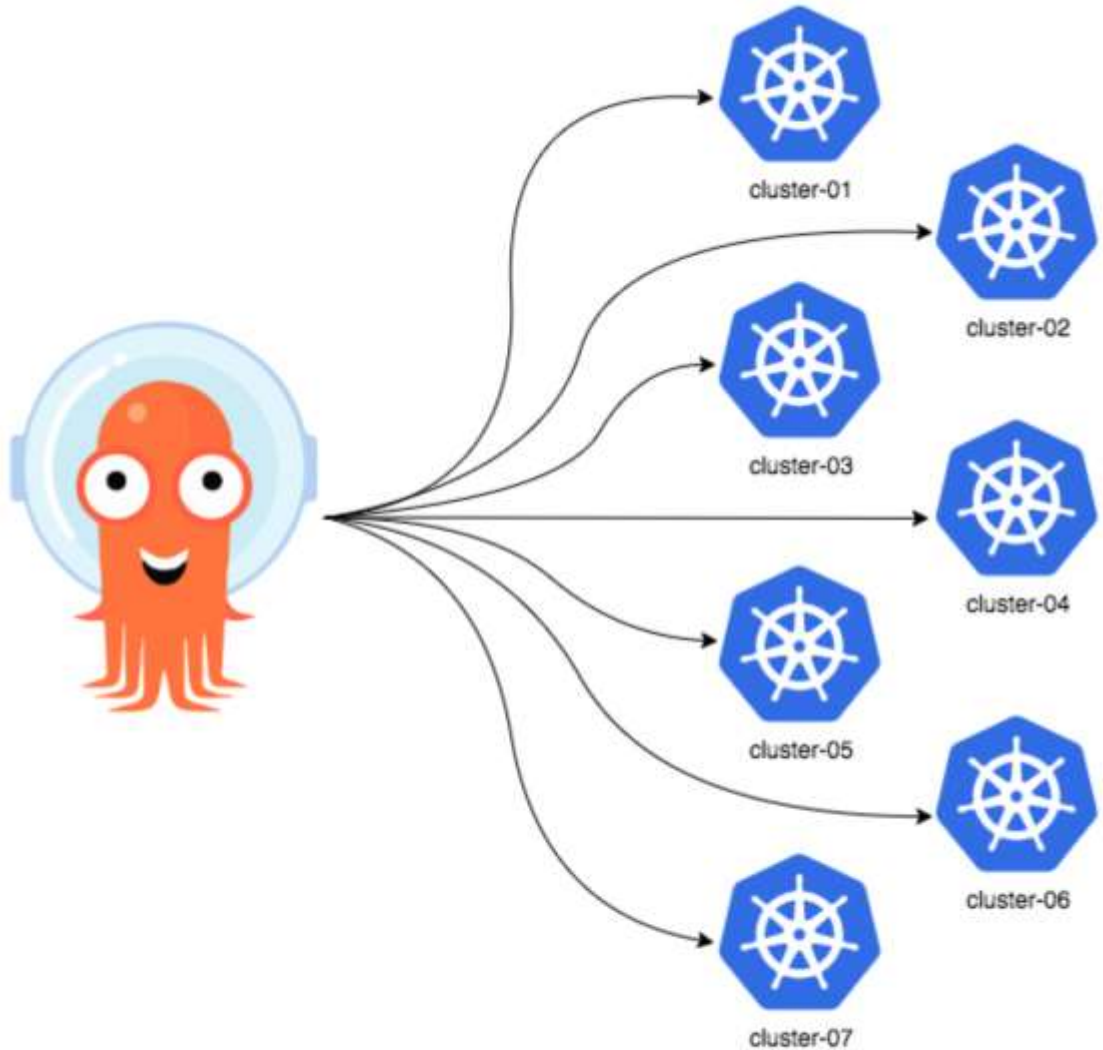
Refresh (Compare)

- Compare the latest code in Git with the live state. Figure out what is different.
- ArgoCD automatically refreshes every 3 minutes.

ApplicationSet

Its a controller and CRD that provides:

- Automating the applications generations.
- More Flexibility when managing Argo CD Applications across a large number of clusters.
- Ability to make self-service usage on multitenant clusters.

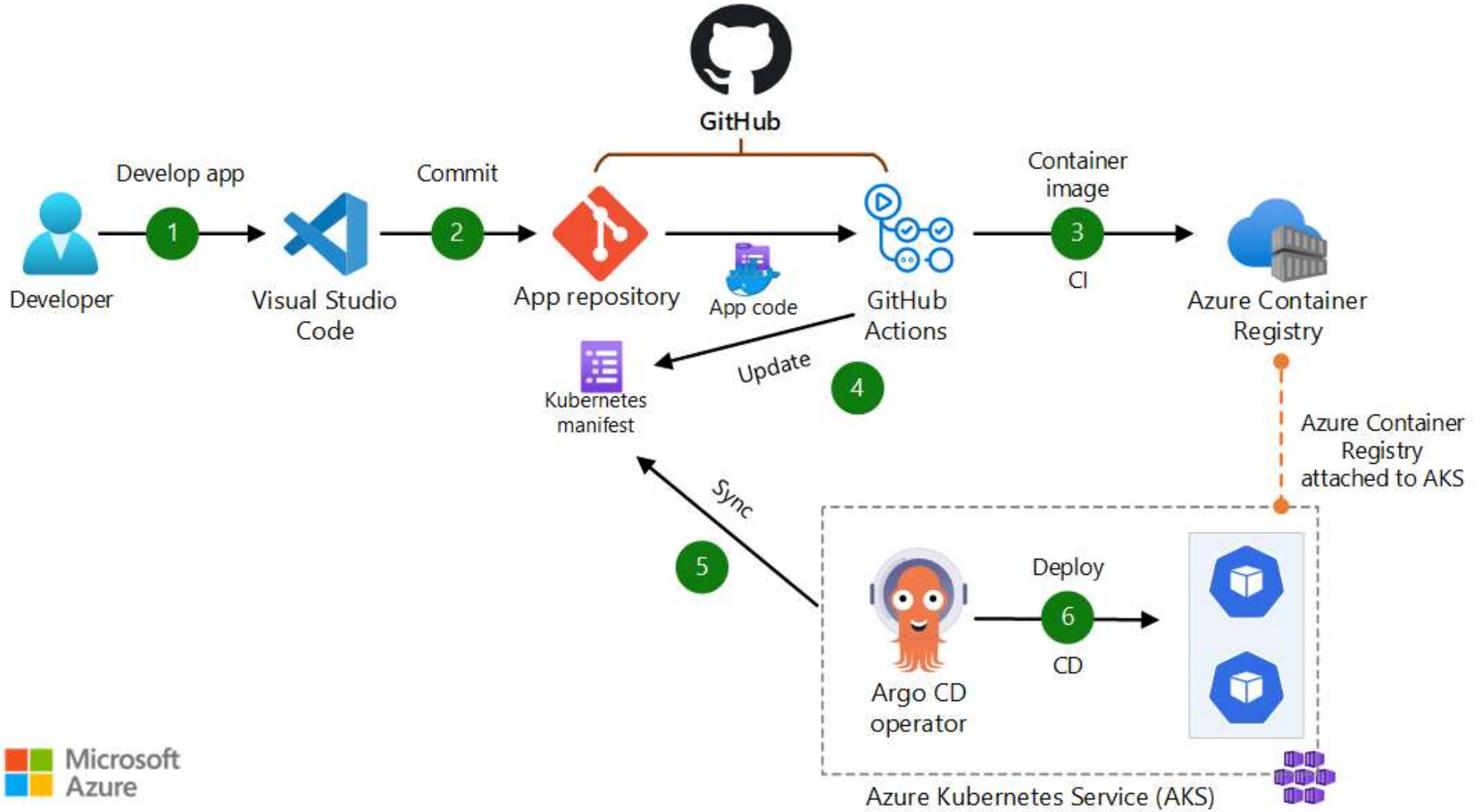


ApplicationSet yaml structure

```
apiVersion: argoproj.io/v1alpha1
kind: Application
metadata:
  name: guestbook
  namespace: argocd
spec:
  destination:
    namespace: guestbook
    server: "https://kubernetes.default.svc"
  project: default
  source:
    path: guestbook
    repoURL: "https://github.com/argoproj/argocd-example-apps.git"
    targetRevision: HEAD
```

```
apiVersion: argoproj.io/v1alpha1
kind: ApplicationSet                          # Argo applicationSet object "CRD"
metadata:
  name: guestbook                             # Name of applicationSet
spec:
  generators:
  - list:                                     # List generator
      elements:
      - cluster: engineering-prod-1           # List of elements (params)
        url: https://1.2.3.4
      - cluster: engineering-prod-2
        url: https://2.4.6.8
  template:
    metadata:
      name: '{{cluster}}-guestbook'           # Name of application
    spec:
      destination:                            # Destination cluster
        server: '{{url}}'
        namespace: guestbook
      source:                                 # Source of manifests
        repoURL: https://github.com/argoproj/argocd-example-apps.git
        targetRevision: HEAD
        path: guestbook/{{cluster}}
```

DevOps Workflow



Demo