KUBERNETES GITOPS WITH ARGOCD: BEST PRACTICES & PRACTICAL PATTERNS

MARYAM TAVAKKOLI

SENIOR CLOUD ENGINEER @ RELEX SOLUTIONS

DEVOPSSTAGE, NOVEMBER 2024

WHO AM I?

- Maryam Tavakkoli
- Senior Cloud Engineer @ RELEX Solutions
- CNCF Ambassador
- Microsoft MVP
- Kubernetes & CNCF Meetup Co-organizer
- LinkedIn: maryam-tavakkoli
- Medium: @maryam.tavakoli.3

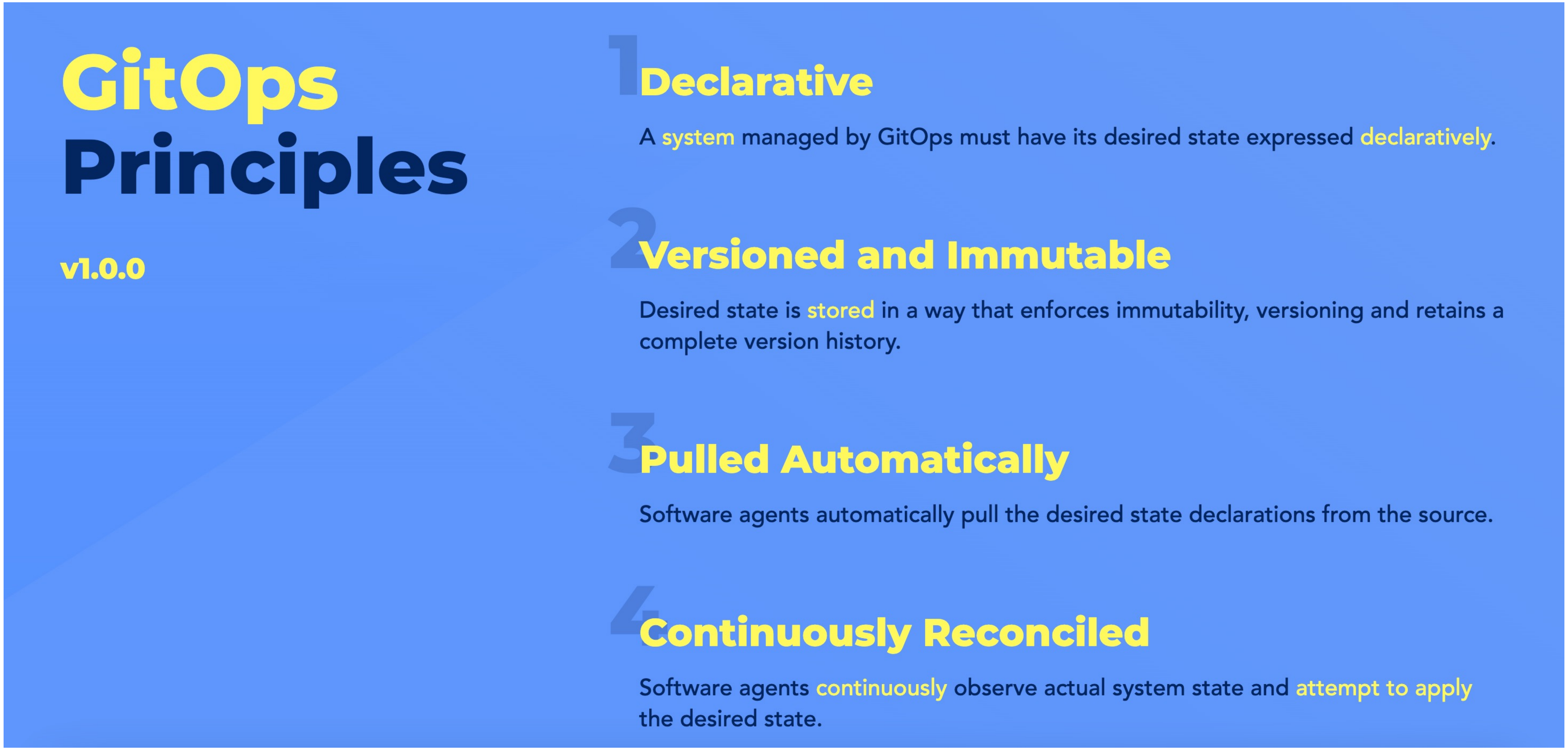
AGENDA

- What is GitOps?
- Benefits of GitOps
- Introducing ArgoCD?
- Demo: GitOps in Action with ArgoCD
- Best Practices & Patterns with ArgoCD
- Summary

WHAT IS GITOPS?

The definition of GitOps, as outlined by gitops.tech:

The core idea of GitOps is having a Git repository that always contains declarative descriptions of the infrastructure currently desired in the production environment and an automated process to make the production environment match the described state in the repository. If you want to deploy a new application or update an existing one, you only need to update the repository - the automated process handles everything else. It’s like having cruise control for managing your applications in production.



GitOps Principles

v1.0.0

1 Declarative

A system managed by GitOps must have its desired state expressed declaratively.

2 Versioned and Immutable

Desired state is stored in a way that enforces immutability, versioning and retains a complete version history.

3 Pulled Automatically

Software agents automatically pull the desired state declarations from the source.

4 Continuously Reconciled

Software agents continuously observe actual system state and attempt to apply the desired state.

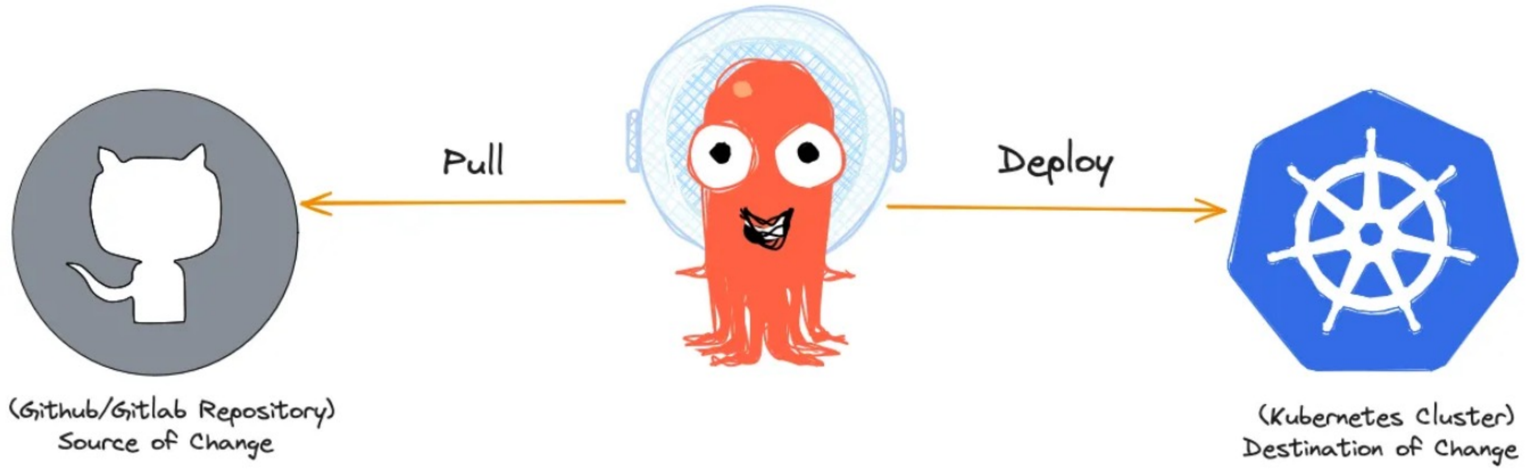
Source: https://opengitops.dev/

BENEFITS OF GITOPS

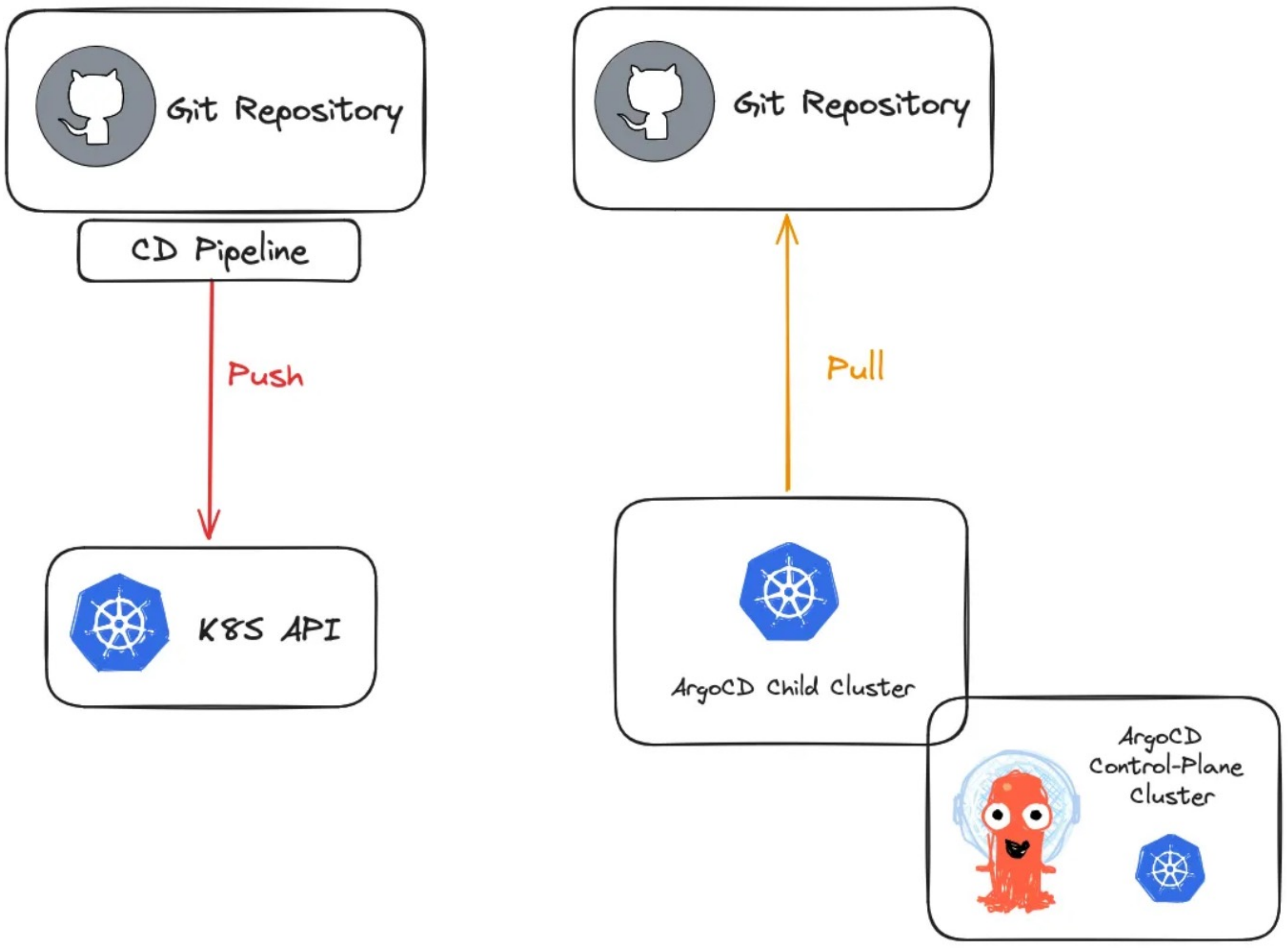
- **Consistency and Traceability**: Every change is tracked and stored in Git, providing a complete history of all modifications.
- **Automation and Efficiency**: Infrastructure and application deployments are automated, reducing the risk of human errors and speeding up delivery.
- **Improved Security**: By using Git as the central control point, only changes that are reviewed and approved via pull requests are applied, enhancing security and reducing unauthorized changes.
- **Easy Rollbacks**: Since all changes are versioned in Git, rolling back to a previous state is as simple as reverting a commit.

INTRODUCING ARGOCD?

- ArgoCD is a declarative, GitOps continuous delivery tool for Kubernetes. It is designed to facilitate the deployment and management of applications in a Kubernetes cluster using Git as the source of truth for the desired application state.



ArgoCD as a GitOps tool

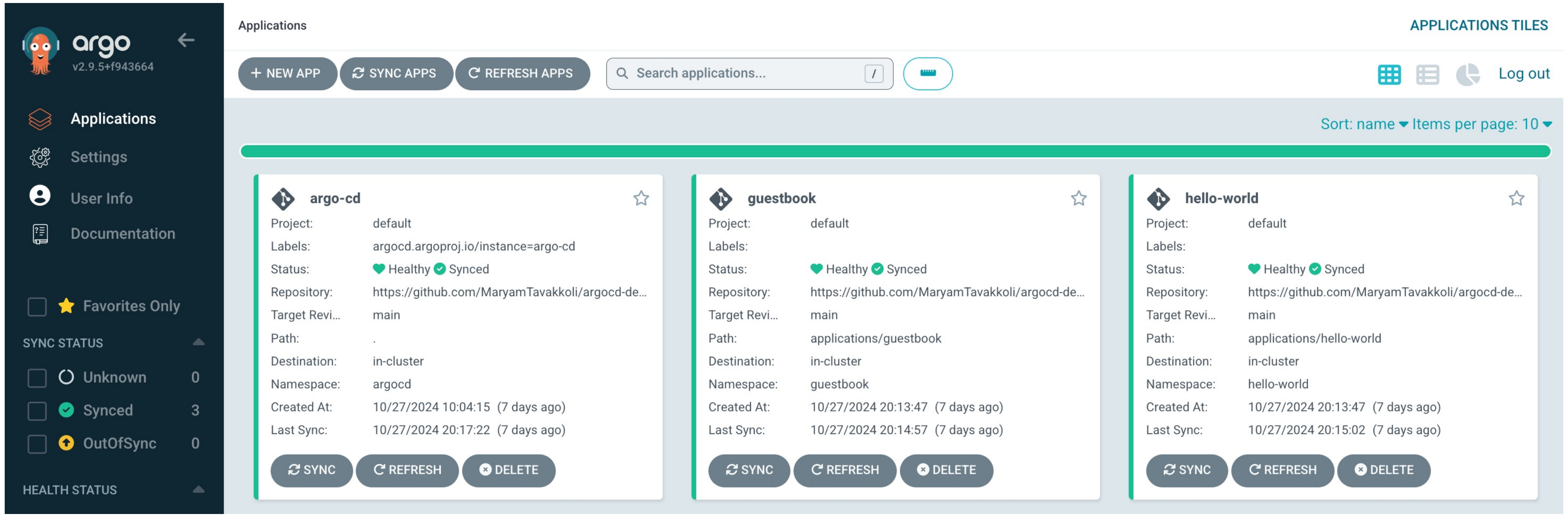


“Push Model” VS. GitOps “Pull Model”

DEMO

Resources

- https://github.com/MaryamTavakkoli/argocd-demo
- https://argo-cd.readthedocs.io/en/stable/getting_started/



argo-cd

Project:	default
Labels:	argocd.argoproj.io/instance=argo-cd
Status:	Healthy Synced
Repository:	https://github.com/MaryamTavakkoli/argocd-de...
Target Revi...	main
Path:	.
Destination:	in-cluster
Namespace:	argocd
Created At:	10/27/2024 10:04:15 (7 days ago)
Last Sync:	10/27/2024 20:17:22 (7 days ago)

guestbook

Project:	default
Labels:	
Status:	Healthy Synced
Repository:	https://github.com/MaryamTavakkoli/argocd-de...
Target Revi...	main
Path:	applications/guestbook
Destination:	in-cluster
Namespace:	guestbook
Created At:	10/27/2024 20:13:47 (7 days ago)
Last Sync:	10/27/2024 20:14:57 (7 days ago)

hello-world

Project:	default
Labels:	
Status:	Healthy Synced
Repository:	https://github.com/MaryamTavakkoli/argocd-de...
Target Revi...	main
Path:	applications/hello-world
Destination:	in-cluster
Namespace:	hello-world
Created At:	10/27/2024 20:13:47 (7 days ago)
Last Sync:	10/27/2024 20:15:02 (7 days ago)

DEMO

Application

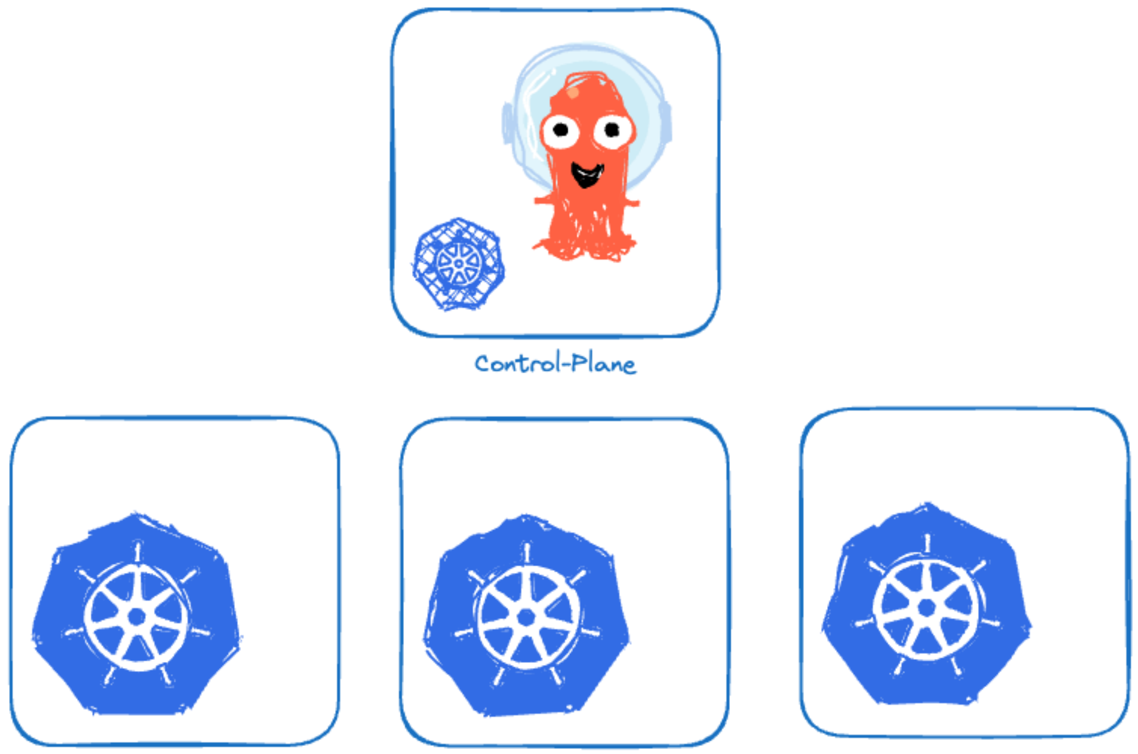
An Application in ArgoCD represents a Kubernetes manifest that is managed and deployed by ArgoCD. It defines the desired state of the application, including the source of the application manifests, the destination cluster where it should be deployed, and various synchronization policies.

ApplicationSet

ApplicationSet is a higher-level abstraction in ArgoCD that allows you to define and manage multiple applications based on a template. It enables you to dynamically generate and manage multiple instances of similar applications using a set of parameters.

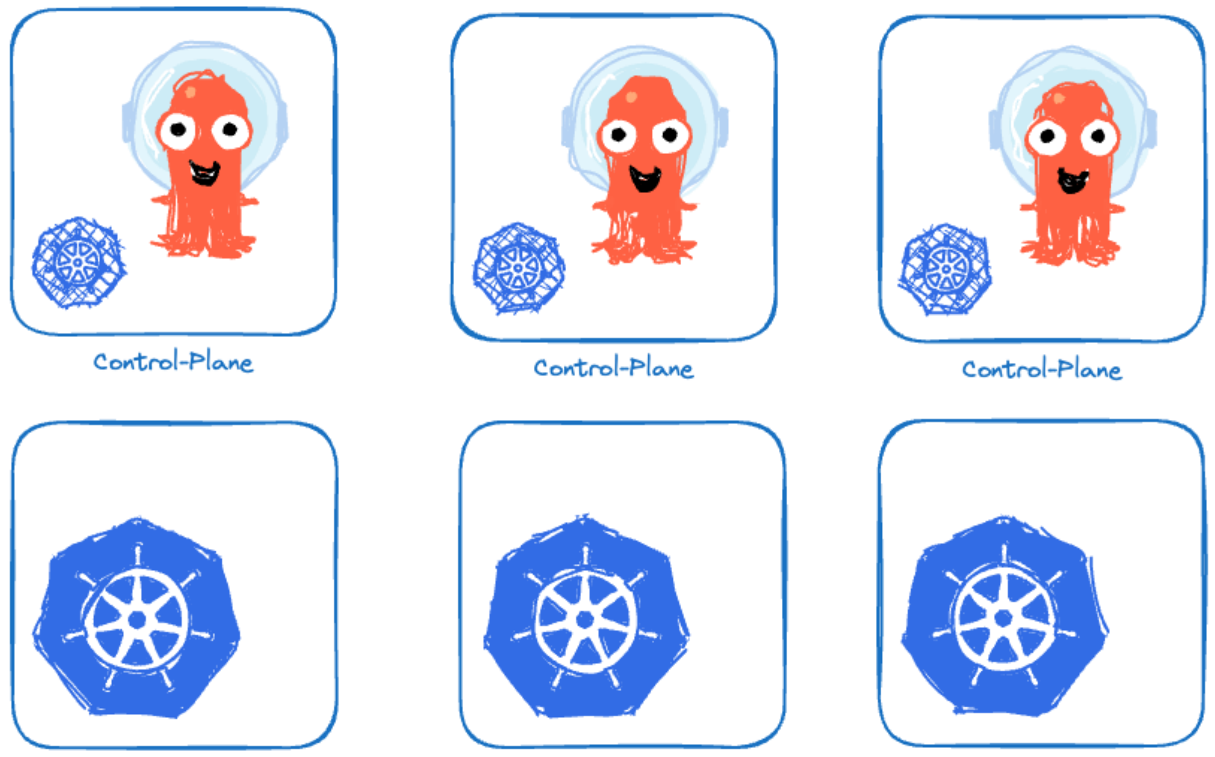
BEST PRACTICES & PATTERNS WITH ARGOCD

- ArgoCD control plane per cluster
  - One control-plane FOR ALL child clusters



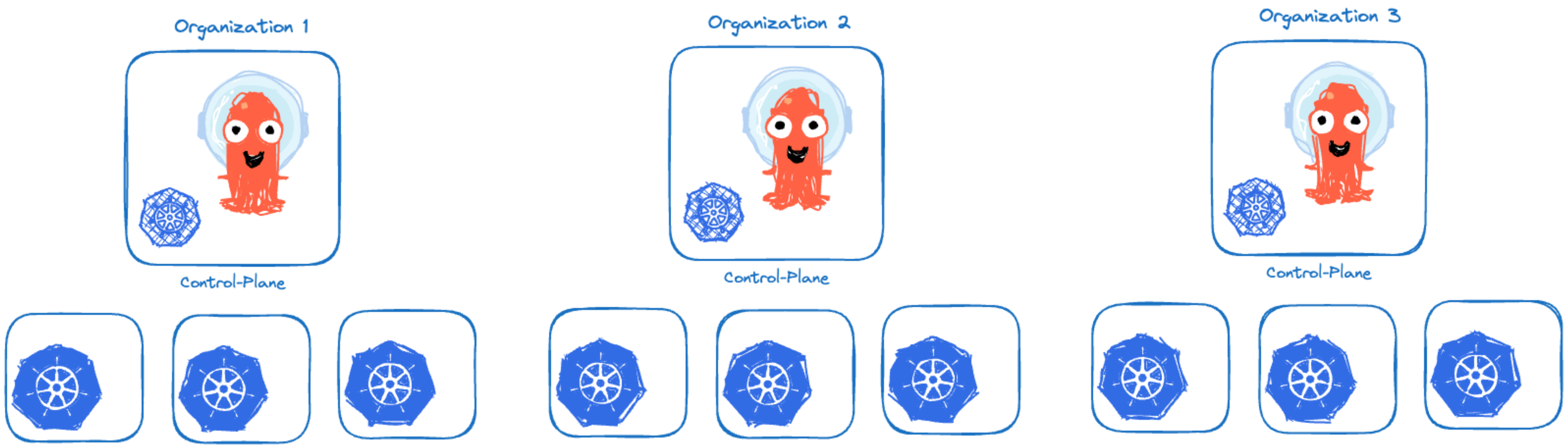
BEST PRACTICES & PATTERNS WITH ARGOCD

- ArgoCD control plane per cluster
  - One control-plane PER child cluster



BEST PRACTICES & PATTERNS WITH ARGOCD

- ArgoCD control plane per cluster
  - One control-plane PER organization (or any logical group)



BEST PRACTICES & PATTERNS WITH ARGOCD

- Manage ArgoCD with ArgoCD: App-of-Apps Pattern
  - You can install ArgoCD with a helm chart and let ArgoCD manage it after the first-time installation using App-of-Apps pattern.
  - App-of-apps pattern is to define a parent application that manages multiple child applications. You can use this to manage installation of ArgoCD.

BEST PRACTICES & PATTERNS WITH ARGOCD

- ApplicationSet is powerful feature
  - Leverage the full potential of ApplicationSets to automate the management of multiple applications in ArgoCD. ApplicationSets allow you to dynamically generate and manage ArgoCD applications based on external data sources like Git, lists or clusters.
  - Use ApplicationSets when you have multiple similar applications that require a consistent configuration across environments. For instance, if you're deploying microservices or managing environments for multiple teams, ApplicationSets let you define a single template to manage all applications uniformly.

BEST PRACTICES & PATTERNS WITH ARGOCD

- Use Helm or Kustomize as Application Sources
  - Leverage **Helm charts** or **Kustomize overlays** as sources in ArgoCD applications to simplify and standardize configurations.
    - **Helm**: Ideal for managing parameterized configurations, enabling templating, and sharing deployment templates across teams or environments.
    - **Kustomize**: Useful for creating layered configurations, allowing you to apply environment-specific overlays without duplicating manifest files.

BEST PRACTICES & PATTERNS WITH ARGOCD

- Use Automatic Syncing with Caution
  - ArgoCD supports both **automatic syncing** (automatically applying changes as soon as they are detected in Git) and **manual syncing**. For ***production*** environments, you may want to use ***manual sync*** to ensure that changes are reviewed before deployment.
  - For ***development*** **and** ***staging environments***, enable ***auto-sync*** to speed up testing cycles.

BEST PRACTICES & PATTERNS WITH ARGOCD

- Handle Secrets Securely
  - Never store sensitive data like secrets directly in Git. Use **external secret management tools** to handle sensitive information.
  - Use tools like **Azure Key Vault, HashiCorp Vault**, or **AWS Secrets Manager**, and integrate them with ArgoCD using tools like **External Secrte Operator (ESO)** or **Sealed Secrets**.

SUMMARY

- GitOps: A method to manage infrastructure and applications using Git, providing a single source of truth.
- ArgoCD: A GitOps tool that automates deployment by continuously syncing Git changes to clusters, ensuring they always reflect the desired state.
- Demo Recap: Showcased how changes in Git are automatically deployed to Kubernetes, highlighting GitOps' efficiency and simplicity.
- Key Best Practices: Secure secrets management, cautious use of auto-sync, and leveraging the App-of-Apps pattern to scale and organize deployments effectively.

THANK YOU

Maryam Tavakkoli

Senior Cloud Engineer @ RELEX Solutions |
CNCF Ambassador | Microsoft MVP |...