SKILUP DAYS
by: DevOps Institute
ADVANCING THE HUMANS OF DEVOPS

Hermetic Deployments -the Heart of GitOps

Understanding the GitOps model.

Tracy Ragan
CEO
DeployHub
@TracyRagan
Tracy@DeployHub.com

Agenda

GitOps Basics

GitOps Goal

GitOps Pros

GitOps Cons

Tracy Ragan
CEO
@TracyRagan

- CEO and Co-Founder – DeployHub, Inc.
- Founding Board member of the CD Foundation
- Founding Board Member Eclipse Foundation
- DevOps Institute Ambassador,
- Community Director – Ortelius Open Source
- 20+ DevOps Experience.

SKILup DAYS

What is GitOps?

GitOps is Continuous Deployment method for cloud native applications.

GitOps leverages a ‘repository,’ like Git, to create an immutable, air-tight deployment process.

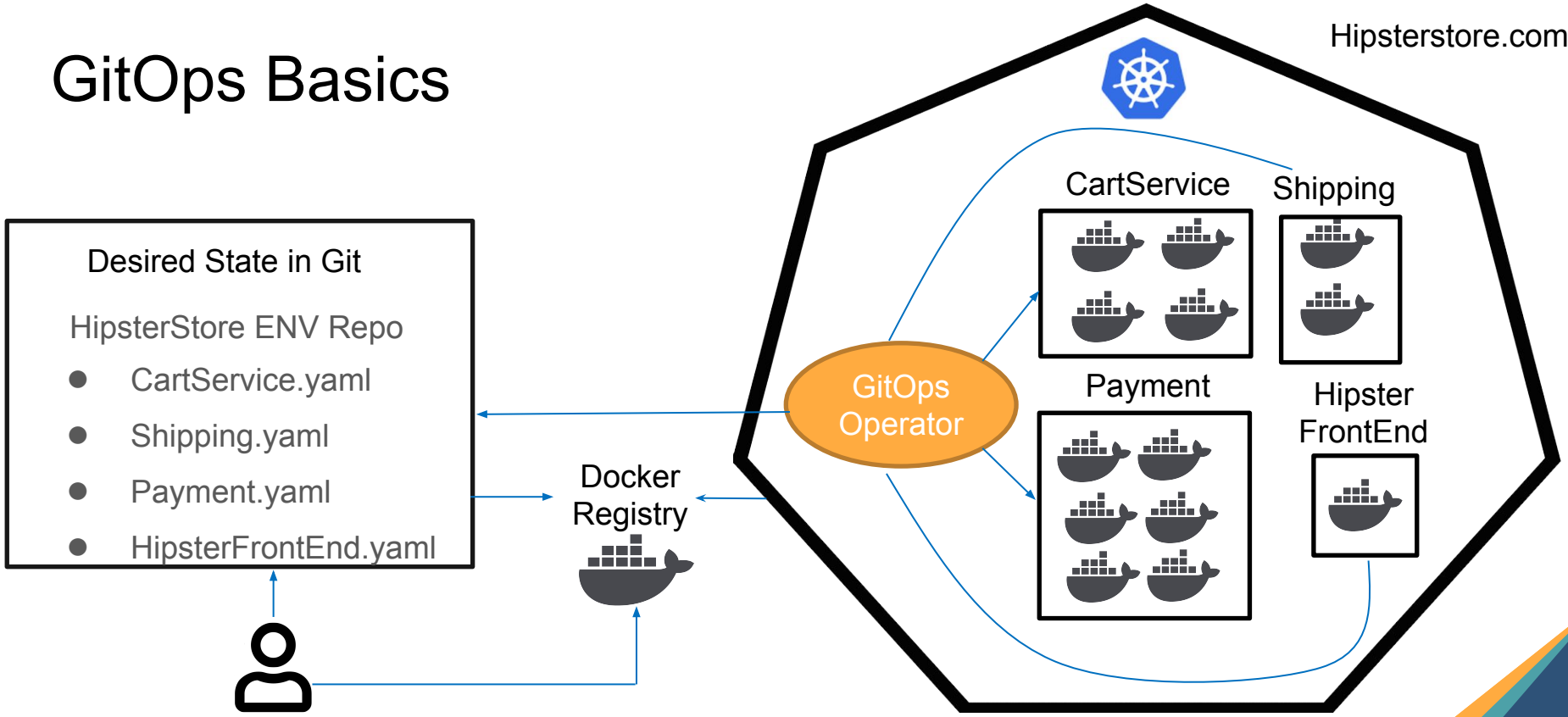
SKILup DAYS

GitOps Basics

- Core to GitOps is the ability to organize deployments around code repositories.
- GitOps is used to manage a Deployment ‘Environment’ repository where deployment .yaml is managed.
- Deployments are ‘pull’ based. A GitOps Operator interrogates the state of the cluster compared to the ‘Environment’ repository. When different, it pulls the updates.

SKILup DAYS

GitOps Basics



Developer updates code repository, registers container image and updates the deployment .yaml in the Environment Repository to use the new container version referenced by a tag or SHA.

Goals of GitOps

Updates are:

- Automated and Repeatable - eliminate manual touch.
- Audited - track changes via versions.
- Deterministic with convergence – Kubernetes state is determined by what is in the repository.

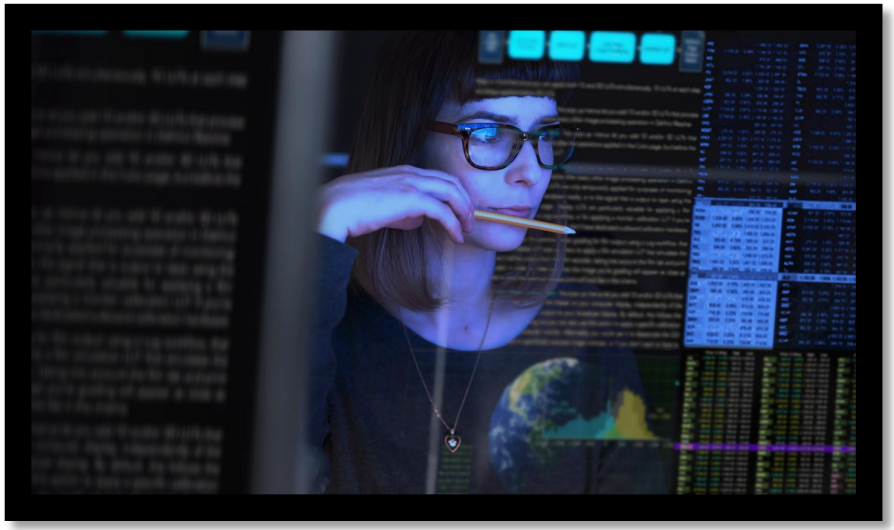
A Deployment that is 100% hermetic and immutable.

Pros - Automated

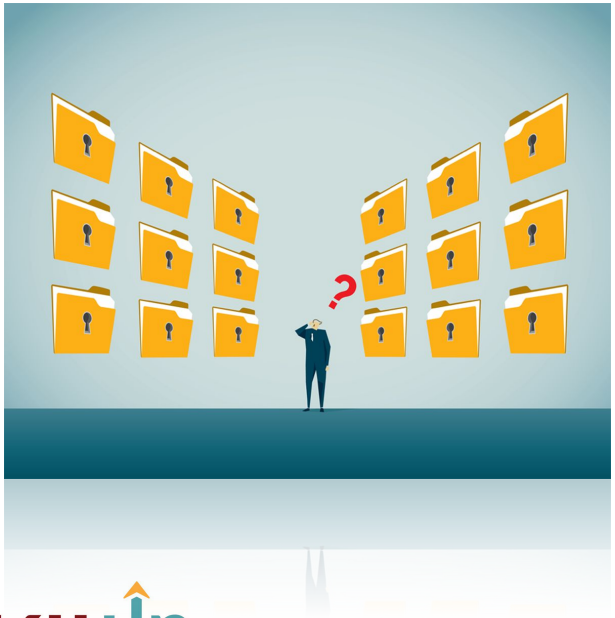
- Airtight must be automated and is naturally repeatable.
  - The GitOps Operator makes all changes to the cluster. No manual changes or manual actions are made to the Kubernetes Environment, or any Applications.
  - The management of the .yaml in a source repository is what creates the ‘airtight’ or hermetic and immutable deployments.
  - With the .yaml file locked down in a source repository, you know that no new change has been introduced before a deployment. This creates a high degree of repeatability.

Pros - Audited

- Tracking who did what, when and why is the benefit of a source repository.
- With the deployment .yaml managed this way, you have visibility into what changed between any two .yaml files giving some insight into differences.



Pros - Deterministic with Convergence

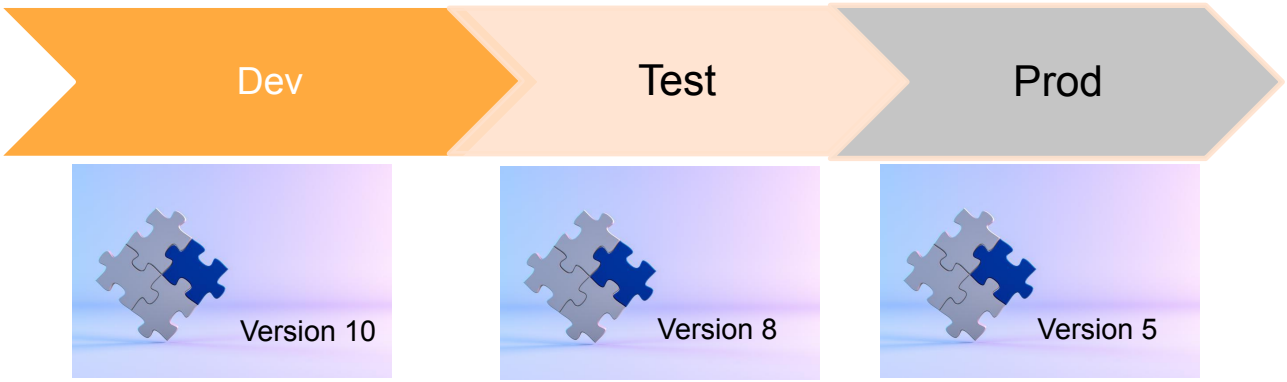


- The State of Kubernetes is determined by the ‘Environment’ Repository.
- The GitOps Operator always brings the system back into sync. If a manual update is made in the Kubernetes system, it is corrected and brought back into the correct state based on the ‘Environment’ repository.

SKILup DAYS

Cons - Pipeline Progression

GitOps does not handle a ‘promotion’ of changes from Development through Production. While we may well eventually move away from this model, today it is core to managing a pipeline.

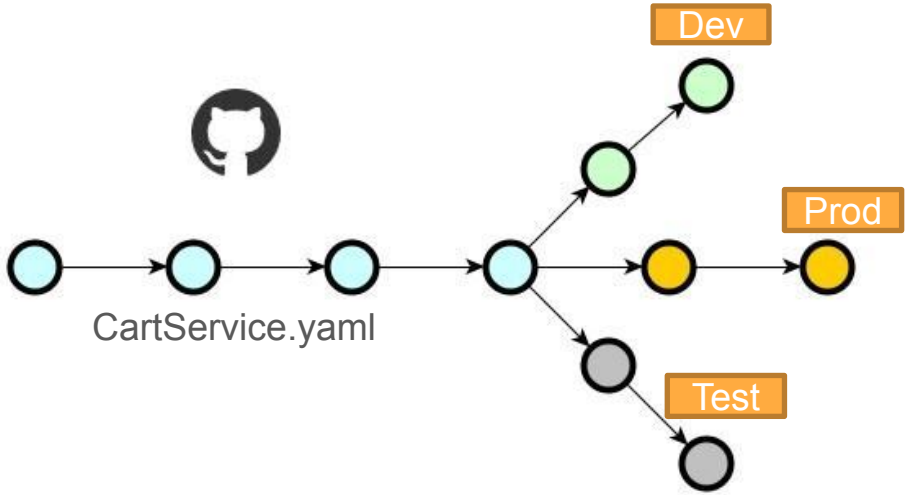


GitOps Environment Repository Branches and the Pipeline

A different version of the Cart service is represented by branches.

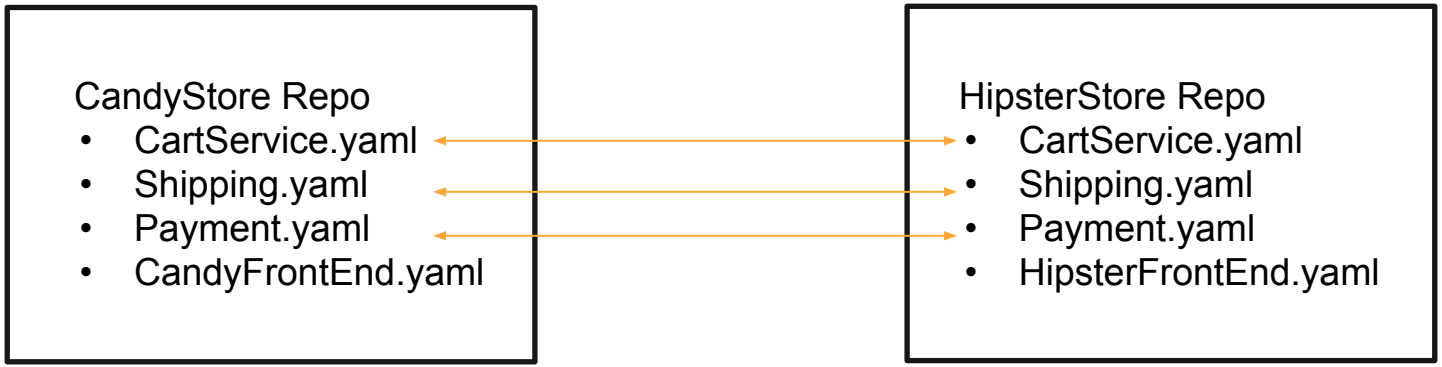
Each cluster, regardless of what the cluster is used for, is managed by a branch.

In another words, if you have many Test Environments, you may have many branches.



SKILup DAYS

Cons - Sharing



Each Application's deployment manifest is managed (.yaml) in separate 'Environment' repositories which creates duplicate .yaml files for a single microservice.

For a shared microservice, this causes drift between the application teams. One team's update can then impact another team's application if they are running in the same cluster. If different clusters, you have branched the microservice unintentionally.

SKILup DAYS

Cons - Security

- While the operator makes the changes to the Kubernetes cluster, anyone with access to the repository can update the .yaml.

- To fix this, a process to approve a ‘pull request’ or a way to manage an update to a .yaml file for a change is required (handled via CD Pipeline).

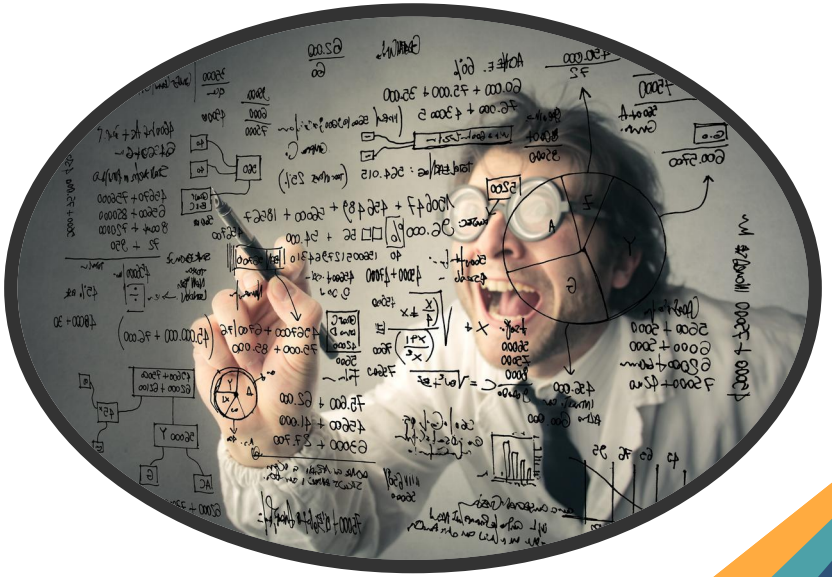
Cons – Scaling and Reporting

Critical data is managed in the .yaml scripts that cannot be easily reported:

- Relationship data and the blast radius of a single microservice update is hidden.
- Inventory reports across all clusters are not provided.
- A single “Environment” repository does not support multiple clusters that may require unique values. In this model we find repository sprawl to support an “Environment” repository per cluster. More .Yaml scripts required.

Con – Not Unscripted

- The GitOps model requires the manual updates of .yaml files. As we move into managing thousands of microservices, this method will bring along thousands of .yaml files with multiple versions.



SKILup DAYS

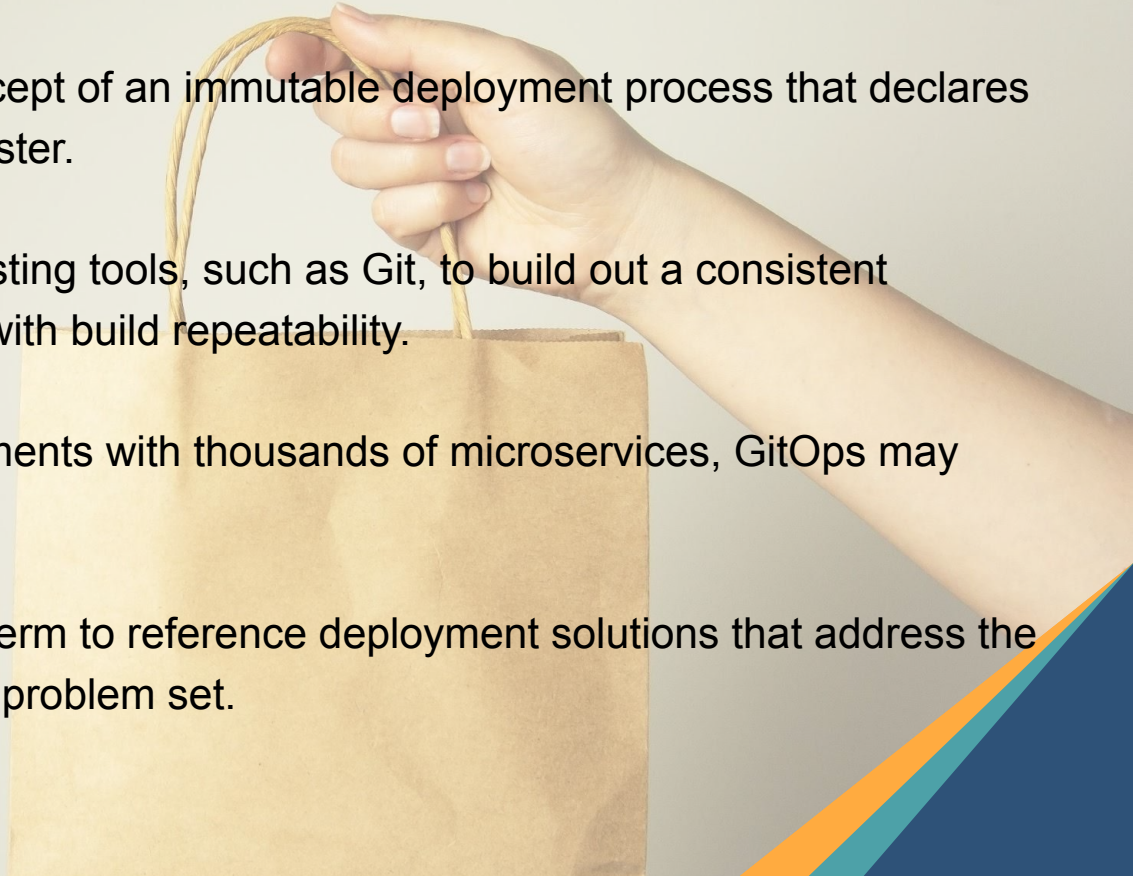
CNCF GitOps Working Group

https://github.com/gitops-working-group/gitops-working-group

A GitOps System Defined

- “The runtime manages itself autonomously, and the Administrator interacts with the Repositories if they wish to modify the desired state of the system.”
  - One or more Runtime environments consisting of resources under management.
  - In each Runtime, management Agents to act on resources according to security policies.
  - One or more software Repositories for storing deployable artifacts that may be loaded into the runtime environments, eg. configuration files, code, binaries and packages.
  - One or more Administrators who are responsible for operating the runtime environments i.e., installing, starting, stopping and updating software, code, configuration, etc.
  - A set of policies controlling access and management of repos, deployment, runtimes.

Key Takeaways



- At the core of GitOps is the concept of an immutable deployment process that declares the state of your Kubernetes cluster.
- GitOps leverages the use of existing tools, such as Git, to build out a consistent continuous deployment model, with build repeatability.
- As we move into larger environments with thousands of microservices, GitOps may struggle with scaling.
- GitOps will become a common term to reference deployment solutions that address the broader continuous deployment problem set.

THANK YOU!

Meet Me in the Network Chat Lounge for Questions

LinkedIn: https://www.linkedin.com/in/tracy-ragan-oms/
Twitter: @TracyRagan
Calendar: https://drift.me/tracyragan/meeting/coffeechat
Email: Tracy@DeployHub.com
Dig In at: DeployHub.com or Ortelius.io

SKILUP DAYS