GitOps

Operations by Pull Request

CloudNative.directory

GitOps - An Operating Model for Building Cloud Native Applications

Weaveworks has pioneered and defined 'GitOps' best practices:

"1. An operating model for Kubernetes and other cloud native technologies, providing a set of best practices that unify deployment, management and monitoring for containerized clusters and applications.

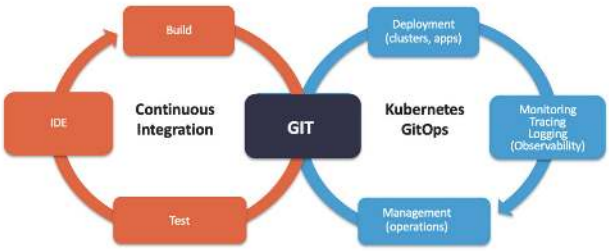
2. A path towards a developer experience for managing applications; where end-to-end CICD pipelines and git workflows are applied to both operations, and development. "

As the term suggests it's an approach based on using Git as the central, single source of truth for application development and deployment. there is a 'source of truth' for both your infrastructure and application code, allowing development teams to increase velocity and improve system reliability.

GitOps.tech offers this intro guide, where they state:

"The fundamental idea of GitOps can be summarized as operations managed and performed in a declarative way with Git as the source-of-truth system."

Your system configuration is defined and stored in a version control system, with the use of software agents to detect when this changes and automatically update the production environment to match it.



This approach brings many benefits:

GitOps - An Operating Model for Building Cloud Native Applications

- Your apps can be easily deployed and rolled back to and from Kubernetes. And even more importantly, when disaster strikes, your cluster's infrastructure can also be dependably and quickly reproduced. This trivializes rollbacks; where you can use a `Git revert` to go back to your previous application state.
- When you use Git workflows to manage your cluster, you automatically gain a convenient audit log of all cluster changes outside of Kubernetes. An audit trail of who did what, and when to your cluster can be used to meet SOC 2 compliance and ensure stability.
- Continuous deployment automation with an integrated feedback control loop speeds up Mean Time to Deployment. Your team can ship 30-100 times more changes per day, increasing overall development output 2-3 times.

Your system configuration is defined and stored in a version control system, with the use of software agents to detect when this changes and automatically update the production environment to match it.

High Velocity Kubernetes

There is a 'single source of truth' for both your infrastructure and application code, allowing development teams to increase velocity and improve system reliability and achieve High velocity CICD for Kubernetes.

"When we say "high velocity" we mean that every product team can safely ship updates many times a day — deploy instantly, observe the results in real time, and use this feedback to roll forward or back. The goal is for product teams to use continuous experimentation to improve the customer experience as fast as possible.

GitOps - An Operating Model for Building Cloud Native Applications

We recommend that you use the operator pattern to listen for and orchestrate service deployments to your Kubernetes cluster. This approach is described by William Denniss in slides 15-21 of our Kubecon presentation (video, slides). Using the operator, an agent can act on behalf of the cluster to listen to events relating to custom resource changes and apply them consistently. In other words the operator performs reconciliation between Git and the cluster."

Create a high velocity CICD pipeline

Operations by Pull Request

At the heart of GitOps is an model of 'Operations by Pull Request', explained in this presentation from Alexis Richardson, Founder and CEO of Weaveworks.

and in their blog:

"What exactly is GitOps? By using Git as our source of truth, we can operate almost everything. For example, ***version control, history, peer review, and rollback*** *happen through Git without needing to poke around with tools like kubectl.*

- ◦
  - *Our provisioning of AWS resources and deployment of k8s is declarative*
  - *Our entire system state is under version control and described in a single Git repository*
  - *Operational changes are made by pull request (plus build & release pipelines)*
  - *Diff tools detect any divergence and notify us via Slack alerts; and sync tools enable convergence*
  - *Rollback and audit logs are also provided via Git "*

Alex explains the role of the Weaveworks technology:

GitOps - An Operating Model for Building Cloud Native Applications

"Our product Weave Cloud provides tools for cloud native applications using GitOps patterns. The core of our GitOps machinery is the CICD tooling. For us, the critical piece is continuous deployment (CD) and release management. This is based on our open source project Weave Flux which supports Git-cluster synchronisation, and so is designed for version controlled systems and declarative application stacks."

And how

"In the "GitOps" model, we use Git to solve for divergence and convergence, aided by a set of "diff" and "sync" tools that compare intended with actual state. A full write up is here."

You can use a variety of CI tools, such as Jenkins, TravisCI, Shippable and Google Container Builder.

The GitOps Pipeline

In their blog The GitOps Pipeline Weaveworks explain that:

"GitOps is a way to do Continuous Delivery. It works by using Git as a source of truth for declarative infrastructure and applications. Automated delivery pipeline automatically roll out changes to your infrastructure when changes are made to Git.

But the idea goes further – using tools to look at the actual production state, and tell you when what's source code doesn't match the real world, giving you the ability to spot the differences and fix problems accordingly. In other words GitOps extends pipelines with a feedback loop for observing and controlling the system."

and that

"To do this, GitOps aims to make developers more productive by applying familiar tools to the hard things: operations management and monitoring. Every developer can use Git and make pull requests; now they can use Git to accelerate and simplify operational tasks for Kubernetes etc. The benefits are far reaching:

- ◦
  - *a model for cloud native CICD pipelines;*
  - *faster mean time to deployment and mean time to recovery,*
  - *actionable alerting,*
  - *stable rollbacks (ie., revert/rollback/fork as per Git);*
  - *and an overall coherent approach to understanding, observing and managing apps."*

Weave Flux

Weave Flux enables GitOps deployment and natively understands how to manage deployments on Kubernetes, automating the staging and release of containers to Kubernetes, as well as services, deployments, network policies and even Istio routing rules.

It can be seen as a leaner, Kubernetes-native alternative to Spinnaker.

Weaveworks enables the GitOps core machinery is in its CI/CD tooling with the critical piece being continuous deployment (CD) that supports Git-cluster synchronization, and they offer training and professional services to support it's implementation.

The GitOps Application Ecosystem

In addition to the core mechanics of GitOps enabled by technologies like Weaveworks, there is also a set of apps that build upon it to manage the surrounding work.

An especially interesting dimension to GitOps is that you can extend the model to include 'people management' related activities, the collaboration, scheduling and project management surrounding the core operations automation.

In their blog The GitOps Pipeline, Weaveworks start to touch on this:

"User friendly social features like user names, auth, permissions, teams, private and public repos and pull request collaboration. This makes Git more intelligible to humans and enables teamwork."

Three vendors are referenced to explain further: Zenhub, Aha.io and SonarCloud. Each work "within" Github and augment its functionality with new features.

SonarCloud

SonarCloud decorates your pull requests, giving you the feedback you need, right in your GitHub repositories. It checks your pull requests and provides a clear Go / No-Go Quality Gate indicator before you merge, providing a barometer on the quality of your PR, so you'll have a clear vision of what you're about to merge.

SonarCloud finds the issues in your pull requests, so you can fix them while the code is still fresh.

Zenhub

ZenHub is a project system for engineering and product management that operates within GitHub, integrates features like Backlog Management and Sprints directly into Github. It is natively integrated into GitHub, using Issues and GitHub's underlying data to keep project progress up-to-date and your software roadmap on track.

The GitOps Application Ecosystem

In their blog How to use GitHub for Project Management, they describe their core approach of mapping agile concepts and working models on to the functionality of their app, such as Product And Sprint Backlogs – The team can feed in all the outstanding work and assign tasks to sprints, adding Estimates to issues (by time or complexity), requirements, and attach a Milestone.

Zenhub emphasize the cost of 'context switching', where workers have to change between tools to continue their work. A study by the Journal of Experimental Psychology found that people who multitask see a 40% drop in productivity. When interrupted from a task, it takes 20-30 minutes to refocus.

Zenhub eliminates this need to switch, by embedding the functionality of their application within another, in this case Github, thus augmenting the features that environment provides to your developers.

Aha.io

Aha.io is an app for strategic roadmapping, cascading down from the top most level of market and product strategy planning through the feature release scheduling to implement those strategies.

As they describe here, and in this article, Aha! provides a two-way integration with GitHub Enterprise that allows teams to send their planned work from Aha! to their team working in GitHub.

Sprints, or "iterations", are mirrored in GitHub with Milestones. Simply set a start and end date (typically two or four weeks), and add user stories to begin sprinting.

The GitOps Application Ecosystem

GitOps for People – Connecting Business and IT

Software teams do not exist in isolation, the code they deploy is consumed by business and end user customers; it is essential this whole system is considered.

It's also been clearly identified that closer interaction between business users and developers during the full software engineering process is key to positive outcomes and success.

Harnessing 'GitHub for People' achieves this literally, merging the activities of marketing, product and technology teams into a single interface and set of workflows, and also extending the core use of the Github premise to this broader scope of work.

Requirements enter the developers realm as new Issues, where they can be progressed to a point of conclusion deployment to the GitOps environment. The collaboration features of the integrated Team Management apps cater for keeping every one 'on the same page' as it progresses through this cycle, which in turn can update their strategic product roadmaps.

Critically this establishes a formal two-way dynamic between 'Business' and 'IT', one that is enabled in real terms through the synthesis of these two main application sets – The strategy and planning of work, and the implementation and maintenance of it.

Apps like Aha.io cater for business users, in their case the ability to create large-scale strategy & product management roadmaps. This captures strategy at the highest organization level, then cascades it down into what are in essence Epics and Sprints, rolling up Requirements into Milestone delivery plans.

The GitOps Application Ecosystem

At this point the apps connect to exchange the main unit of work, the Github issue, and thus there is an end-to-end assignment of work linked to strategy, and back again. The Business team can break down their strategy into specific deliverables required, feeding the DevOps work stream in the form of user stories, with reporting on their progress fed back up the line via the same integration.

Establishing this ‘closed loop’ system the flow through it can be assigned measurement tags, leveraging factory line type optimization principles to find ways to expand and accelerate Deployment Throughput.