KubeCon | CloudNativeCon

North America 2018

Introduction to GitOps Deployment to Kubernetes

by @sakajunquality. 10 December 2018

About me

Jun Sakata / @sakajunquality

- Google Developers Expert, Cloud
- Software Engineer at Ubie inc.
- From Japan
- Loves: #kubernetes and #beer

Ubie Inc.

- Medical Startup in Japan.
  - Diagnosis Assistance to Doctors
- Production Workloads on Kubernetes + GCP
  - Since Oct. 2018

Ubie

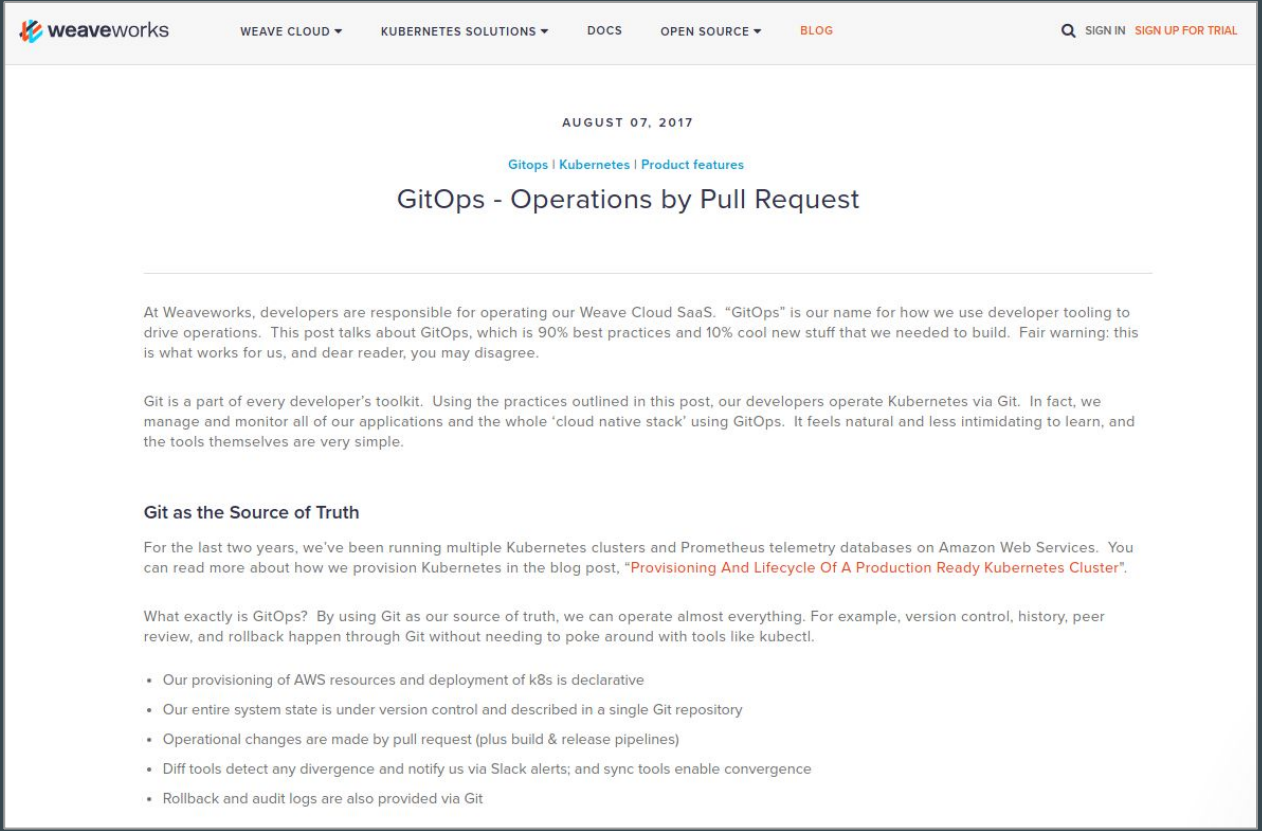
Agenda

- Concept of GitOps
- Very Prototype of GitOps in Ubie
- Future Perspectives

Google Cloud Platform

- As the company is using GCP, services used in the slides are products of GCP.
- But the whole story and idea, I believe, can be applied to any Cloud or On-Prem.

Concept of GitOps



AUGUST 07, 2017

Gitops | Kubernetes | Product features

GitOps - Operations by Pull Request

At Weaveworks, developers are responsible for operating our Weave Cloud SaaS. "GitOps" is our name for how we use developer tooling to drive operations. This post talks about GitOps, which is 90% best practices and 10% cool new stuff that we needed to build. Fair warning: this is what works for us, and dear reader, you may disagree.

Git is a part of every developer's toolkit. Using the practices outlined in this post, our developers operate Kubernetes via Git. In fact, we manage and monitor all of our applications and the whole 'cloud native stack' using GitOps. It feels natural and less intimidating to learn, and the tools themselves are very simple.

Git as the Source of Truth

For the last two years, we've been running multiple Kubernetes clusters and Prometheus telemetry databases on Amazon Web Services. You can read more about how we provision Kubernetes in the blog post, "Provisioning And Lifecycle Of A Production Ready Kubernetes Cluster".

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.

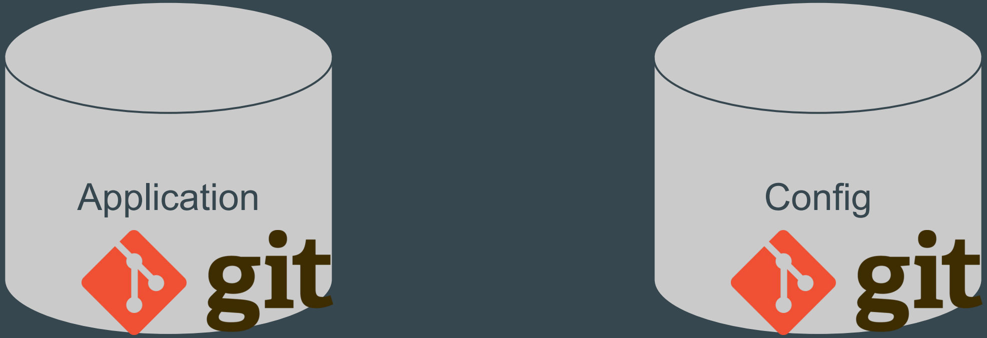
- Our provisioning of AWS resources and deployment of k8s is declarative
- Our entire system state is under version control and described in a single Git repository
- Operational changes are made by pull request (plus build & release pipelines)
- Diff tools detect any divergence and notify us via Slack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

GitOps - Operations by Pull Request

https://www.weave.works/blog/gitops-operations-by-pull-request

GitOps Basics

- Two different types of git repository.
  - Application Repo: Application source code
  - Config Repo: Declarative manifest for configuration



Concept of GitOps

- All the manifest is managed declaratively in Git.
- Any “apply” is through CI.

Concept of GitOps - In Other Words...

- Manifest in the Git represents the current state of the infrastructure.
- Any kind of manual “apply” is prohibited.

Very Prototype of GitOps in Ubie

Infrastructure in Ubie

- Several services are running on Kubernetes cluster.
    - Frontend
    - Several backend microservices
- Kubernetes (in Ubie) = Google Kubernetes Engine.
    - All the workloads are on Google Cloud Platform.
    - Migrated from Heroku on Oct. 2018.

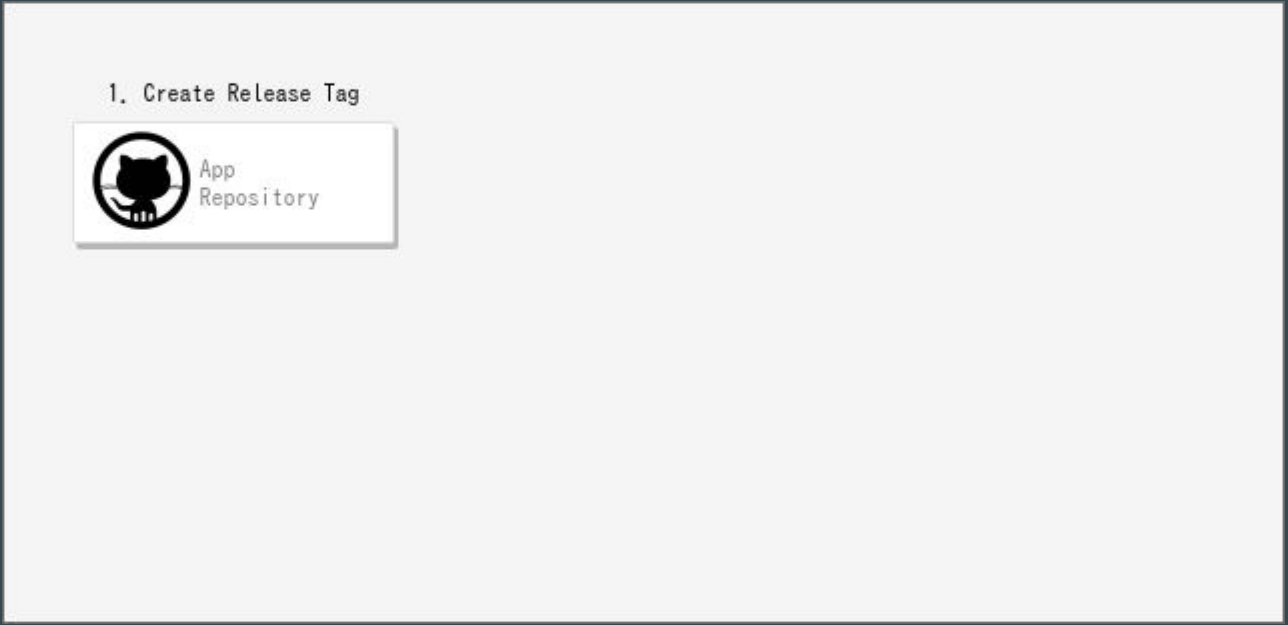
My GitOps Philosophy in Ubie

- Workflow itself should be simple.
- Each components should be decoupled.
- New application should be easily integrated.

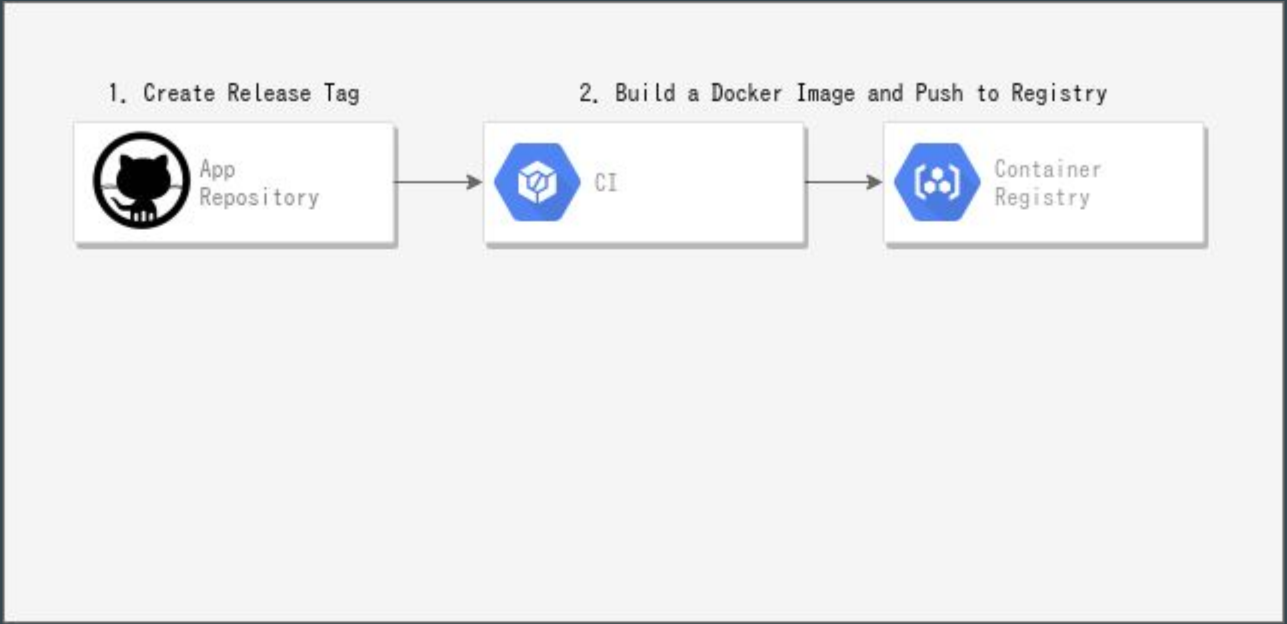
(as much as possible)

GitOps Steps in Ubie

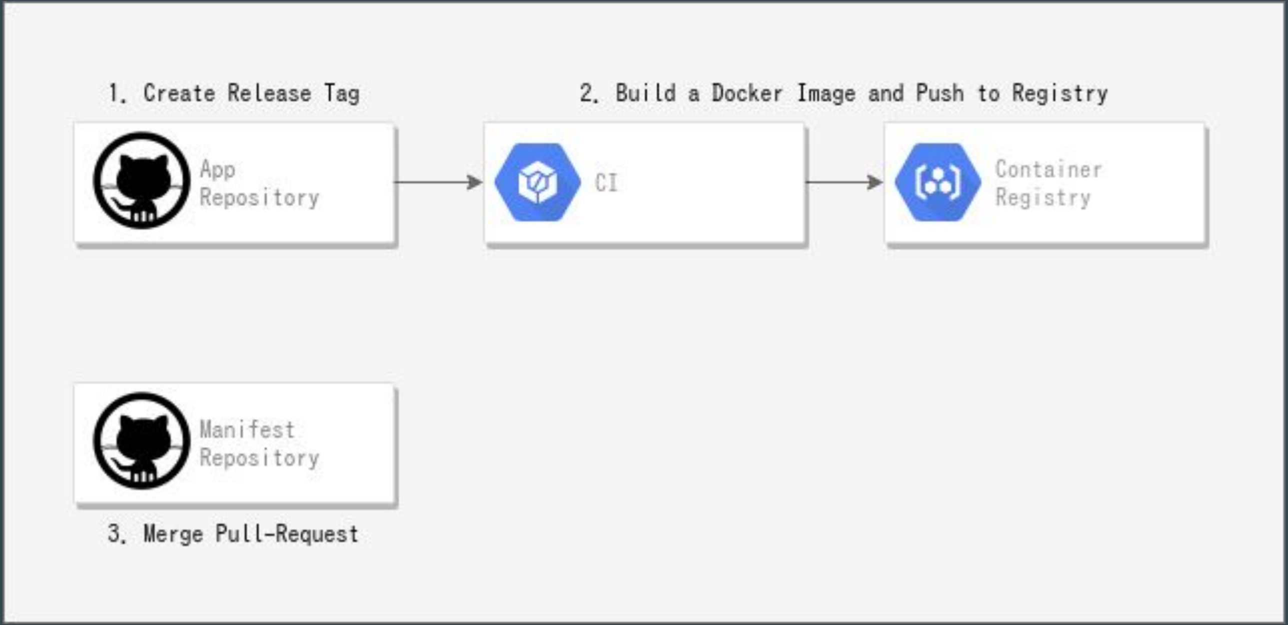
GitOps Steps in Ubie



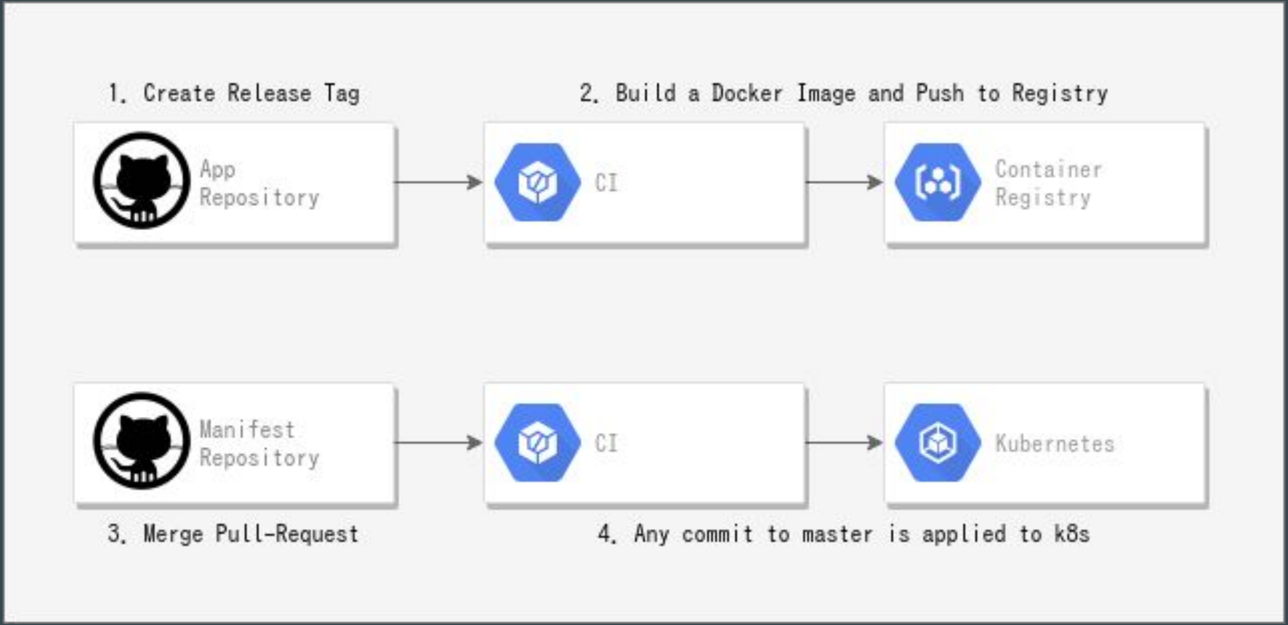
GitOps Steps in Ubie



GitOps Steps in Ubie



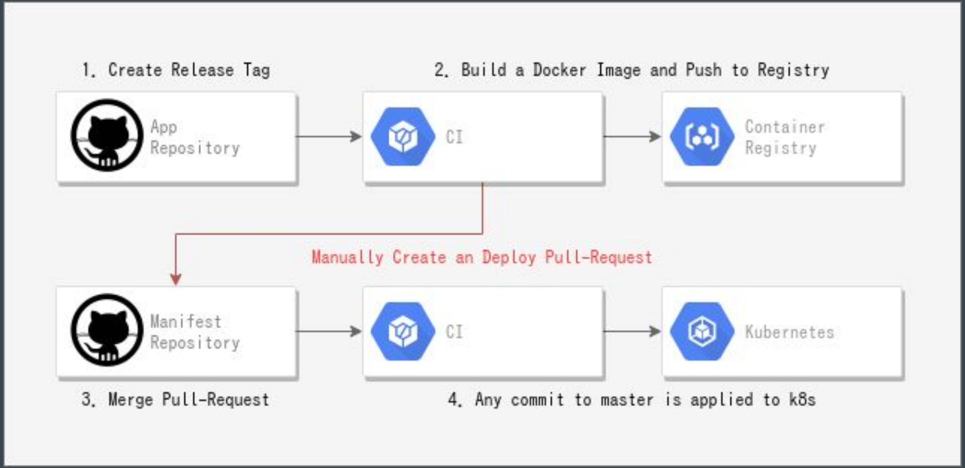
GitOps Steps in Ubie



GitOps First Step

GitOps First Step

- Commit and Push to the manifest repo manually.
- Create an release Pull-Request manually.
- Merge the Pull-Request to deploy.


1. Create Release Tag
2. Build a Docker Image and Push to Registry
App Repository
CI
Container Registry
Manually Create an Deploy Pull-Request
Manifest Repository
CI
Kubernetes
3. Merge Pull-Request
4. Any commit to master is applied to k8s

GitOps First Step: Problems

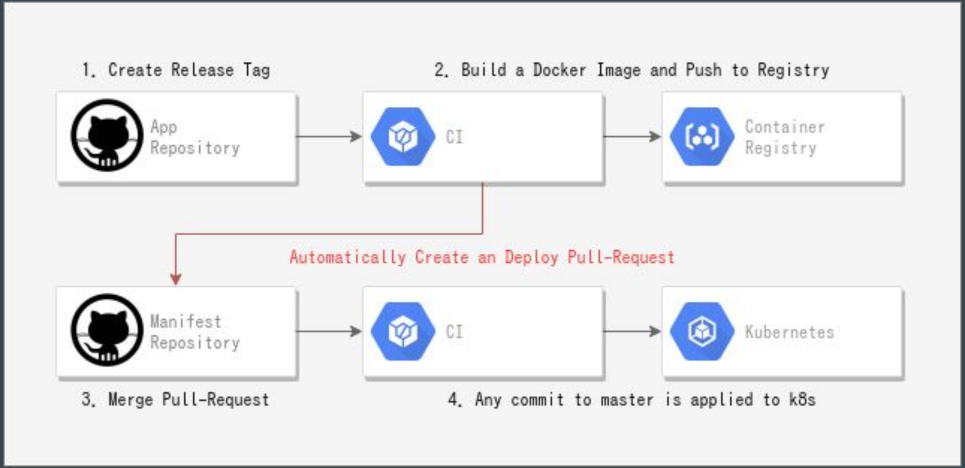
Obviously there are problems,

- We make mistakes.
- Difficult to make changes to manifest repo for engineers.

GitOps Second Step

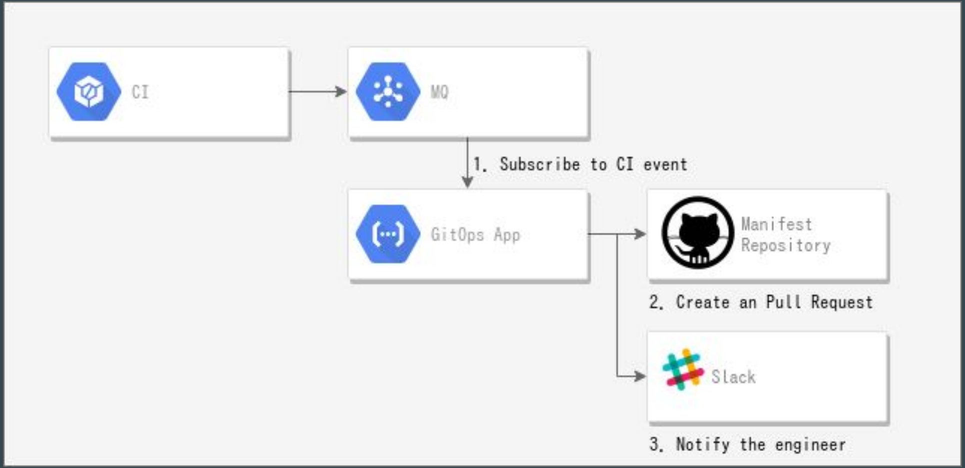
GitOps Second Step

- Commit to the manifest repo and Create an release Pull-Request automatically.
- Merge the Pull Request to deploy.


1. Create Release Tag
2. Build a Docker Image and Push to Registry
App Repository
CI
Container Registry
Automatically Create an Deploy Pull-Request
Manifest Repository
CI
Kubernetes
3. Merge Pull-Request
4. Any commit to master is applied to k8s

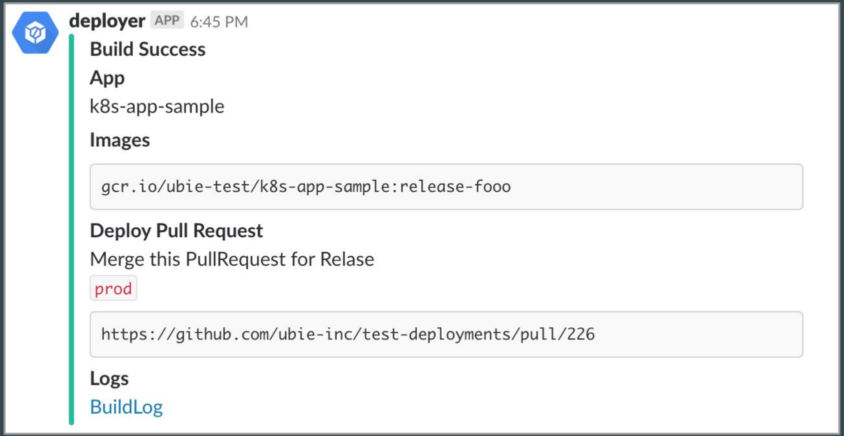
GitOps Second Step: GitOps App

- App that subscribes event from CI (Cloud Build) through MQ (Cloud Pub/Sub),
  - Create an Release Pull-Request on Github.
  - Notify the Pull-Request via Slack.



GitOps Second Step: GitOps App - Slack Notification

After docker image is finished, Pull-Request url is notified via slack.



deployer APP 6:45 PM

Build Success

App

k8s-app-sample

Images

```
gcr.io/ubie-test/k8s-app-sample:release-fooo
```

Deploy Pull Request

Merge this PullRequest for Relase

`prod`

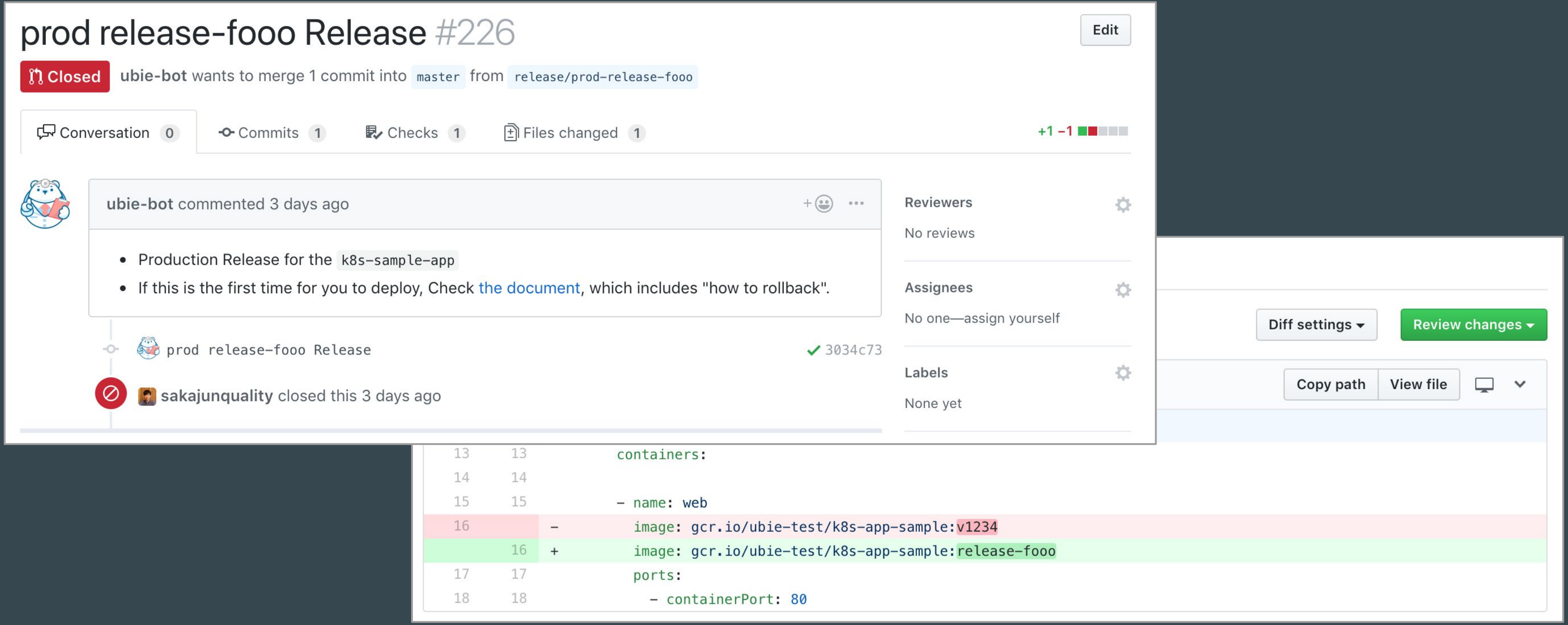
```
https://github.com/ubie-inc/test-deployments/pull/226
```

Logs

BuildLog

GitOps Second Step: GitOps App - Github Pull-Request

Engineer just need to merge the Pull-Request.



prod release-fooo Release #226

Closed ubie-bot wants to merge 1 commit into `master` from `release/prod-release-fooo`

ubie-bot commented 3 days ago

- Production Release for the `k8s-sample-app`
- If this is the first time for you to deploy, Check the document, which includes "how to rollback".

`prod release-fooo Release` ✓ 3034c73

sakajunquality closed this 3 days ago

Reviewers: No reviews

Assignees: No one—assign yourself

Labels: None yet

```
        containers:

        - name: web
-         image: gcr.io/ubie-test/k8s-app-sample:v1234
+         image: gcr.io/ubie-test/k8s-app-sample:release-fooo
          ports:
            - containerPort: 80
```

GitOps Second Step: GitOps App - Rollback

When you need to rollback,

- Revert the merged Pull-Request.
- Merge the reverted Pull-Request.

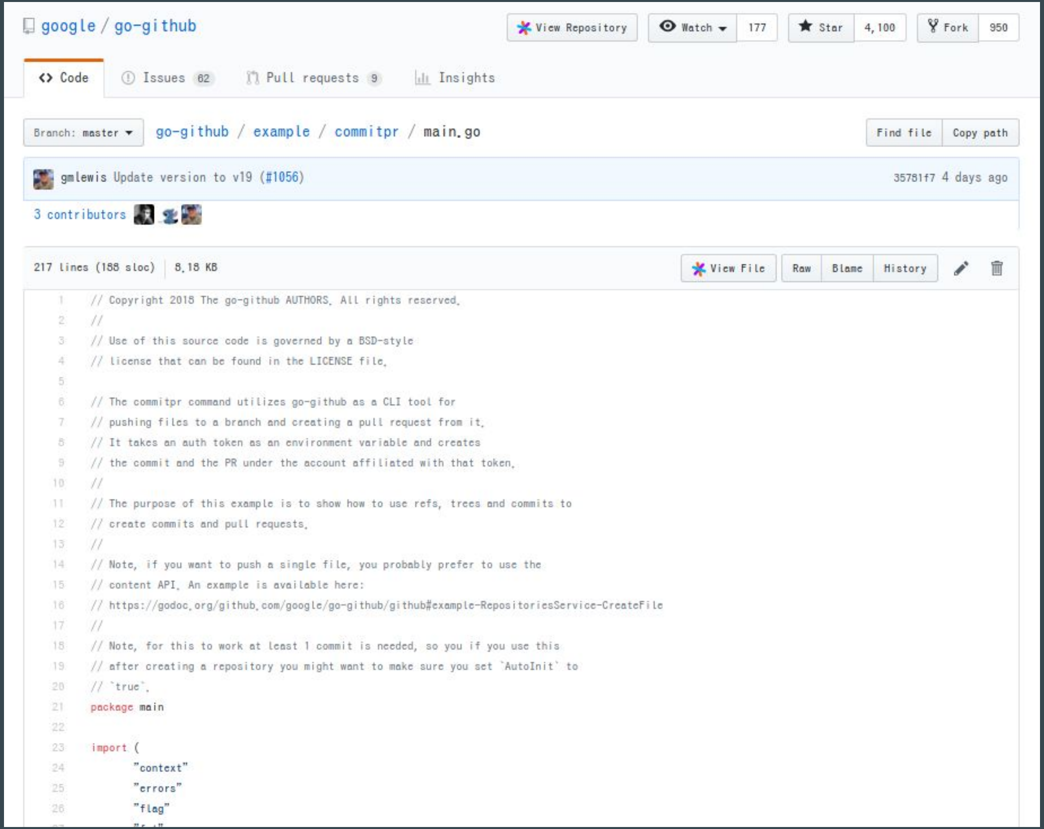
No manual changes to the manifest

(in terms of application release)

GitOps Agent

GitOps Agent

- Using custom app written in Go.
  - https://github.com/sakajunquality/flow
  - No docs at the moment…
- OSS exists though.
  - https://github.com/weaveworks/flux



```go
// Copyright 2018 The go-github AUTHORS. All rights reserved.
//
// Use of this source code is governed by a BSD-style
// license that can be found in the LICENSE file.

// The commitpr command utilizes go-github as a CLI tool for
// pushing files to a branch and creating a pull request from it.
// It takes an auth token as an environment variable and creates
// the commit and the PR under the account affiliated with that token.
//
// The purpose of this example is to show how to use refs, trees and commits to
// create commits and pull requests.
//
// Note, if you want to push a single file, you probably prefer to use the
// content API. An example is available here:
// https://godoc.org/github.com/google/go-github/github#example-RepositoriesService-CreateFile
//
// Note, for this to work at least 1 commit is needed, so you if you use this
// after creating a repository you might want to make sure you set `AutoInit` to
// `true`.
package main

import (
	"context"
	"errors"
	"flag"
```

Example in google/go-github is helpful to create a GitOps App

https://github.com/google/go-github/blob/master/example/commitpr/main.go

Future Perspective

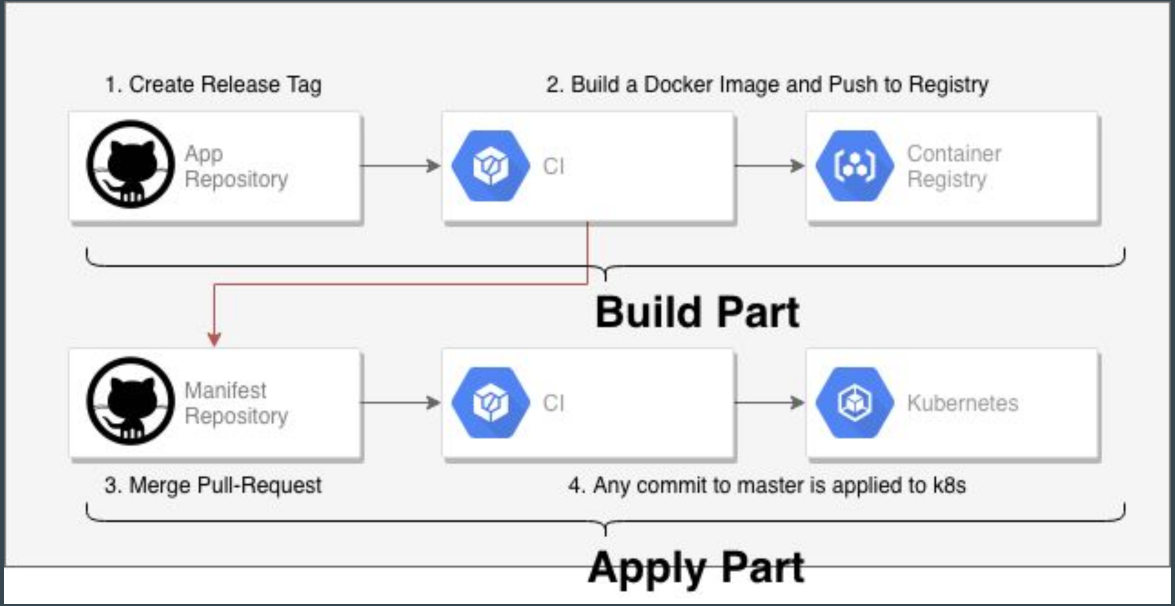
Some Improvements from the Prototype

- Support for pre/post jobs like migration.
- Support for ad-hoc pre/post jobs.
  - Must consider rollback!
- Deployment notification
  - Must be easy for developers.
- Strategic Release
  - Canary Release / Release Analytics
  - Blue/Green
  - etc.

Currently working on it...

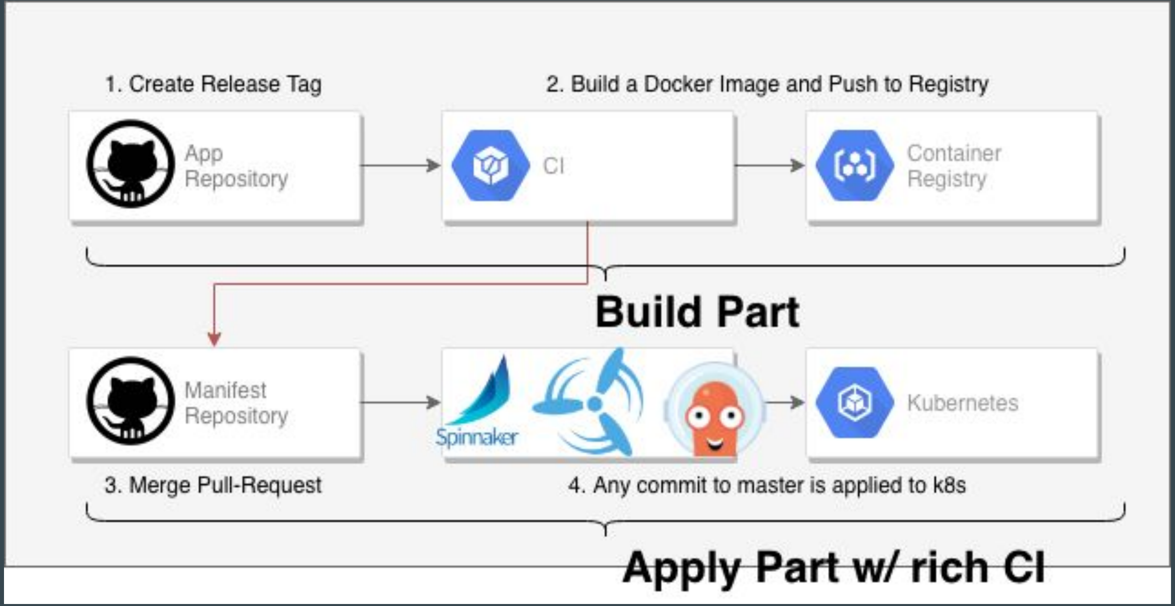
Some Improvements from the Prototype

Our pipeline is separated into two parts: Build and Apply



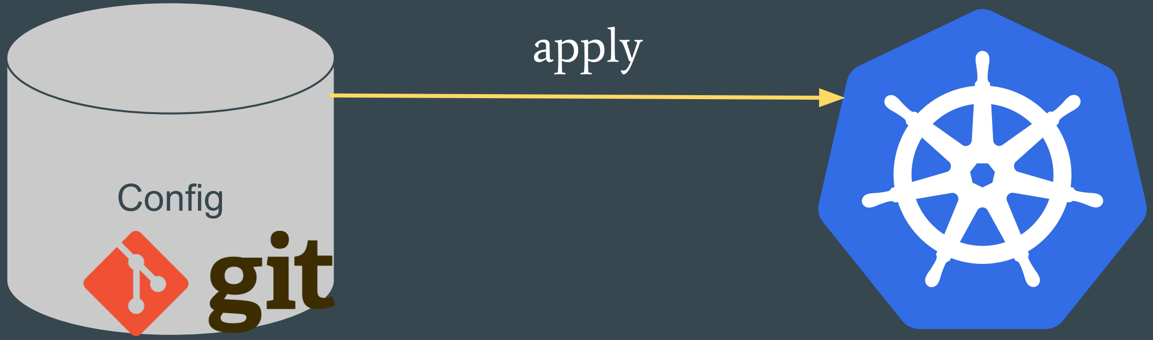
Some Improvements from the Prototype

Apply Part can be replaced with more “Rich” CIs to run more complex jobs.



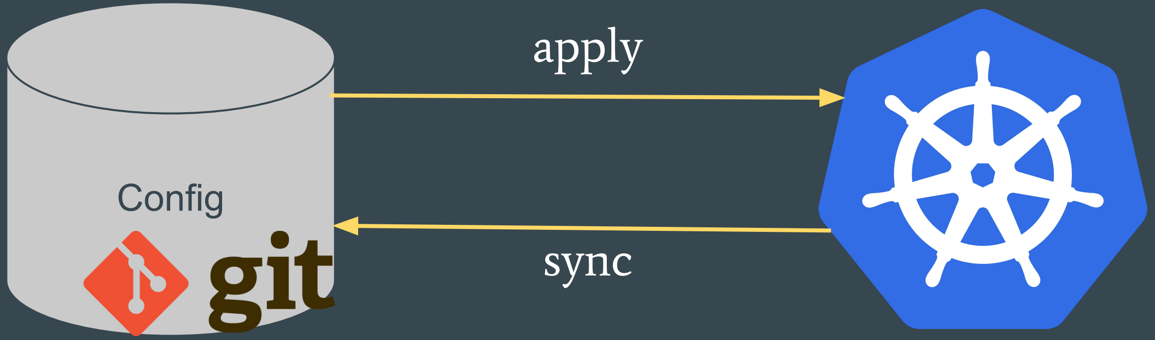
GitOps App: Sync

- Currently Ops is unidirectional: Config repo to Cluster Only



GitOps App: Sync

- Currently Ops is unidirectional: Manifest repo to Cluster Only
- Considering auto-scaling or any updates within a cluster, bidirectional ops should be implemented in the future.
    - flux is bidirectional



Conclusion

Conclusion

- By GitOps, workflow for Kubernetes can be simple.
- GitOps can be introduced step by step.
- Let’s start simply :)

For more info

I will publish an article with more detail, and share on my twitter:

@sakajunquality

Thank you.

KubeCon | CloudNativeCon

North America 2018