Config management for Kubernetes: GitOps + Helm

Tomasz Tarczyński, Gigaset

CLOUD NATIVE WARSAW

9TH-10TH SEPTEMBER 2019

JOIN CLOUD NATIVE WARSAW !

4 TRACKS:

CLOUD NATIVE
KEYNOTE:
PINI REZNIK

KUBERNETES
KEYNOTE:
IGNACY KOWALCZYK

CLOUD
KEYNOTE:
DARIUSZ DWORNIKOWSKI

SERVERLESS
KEYNOTE:
SEBASTIEN GOASGUEN

CALL FOR PROPOSALS: www.cloudnativewarsaw.com/cfp

EARLY BIRD TICKETS: cloudnativeconference2019.evenea.pl

cloudnativewarsaw.com

Agenda

1. **Configuration Management**
2. **Why GitOps**
3. **GitOps Demo:** Flux
4. **Demo 2:** GitOps + Helm

Configuration Management

- **Infrastructure as a Code**
- **GIT** – as a single source of truth
- **Tools:**

  Ansible / Chef / Puppet / Salt

Puppet

- **DSL** – a simple and constrained language
- **Resource Abstraction Layer**
- **Declarative:** describe the desired state
- **Templates:** ERB
- **Modules:** public and private

Kubernetes

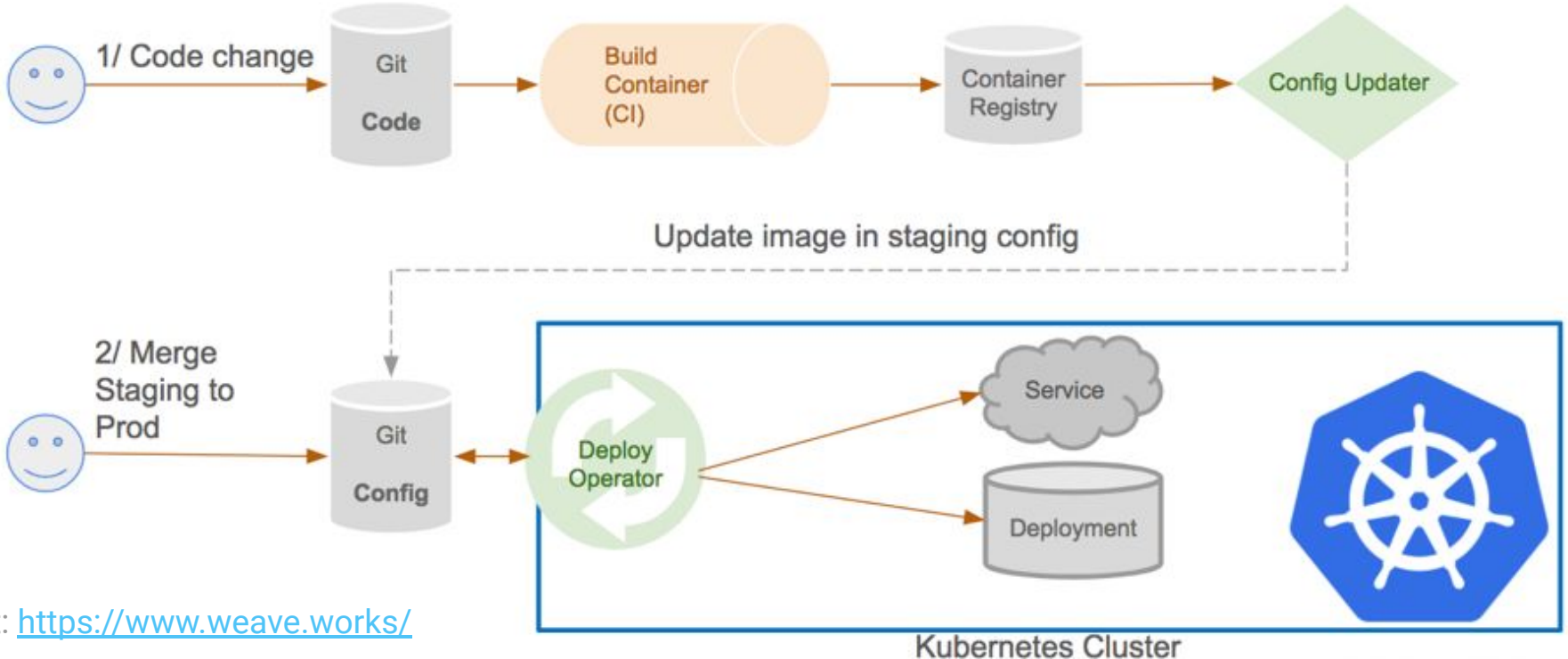
- **Control plane** provides a **Declarative API**
- Declare the **desired state**
- **Control plane** makes sure that the **actual state** converges to the **desired state**

Kubernetes: Why GitOps

- Can we **declare** all the state in **GIT**?
  And expect the system to **converge**.
- Can we have something like **modules**?

GitOps

Example GitOps Pipeline



Credit: https://www.weave.works/

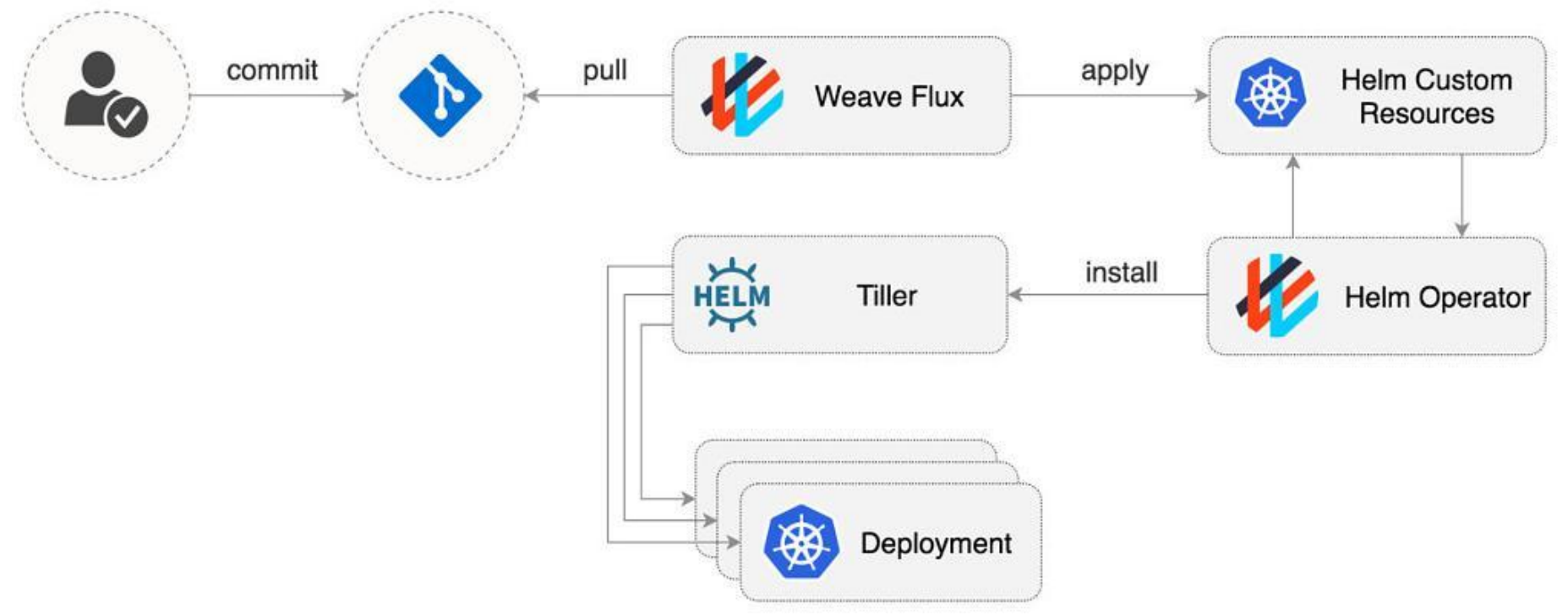
Demo

https://github.com/ttarczynski/gitops-demo

GitOps: Why Helm

- We **declared** all the state in **GIT**
- But can we have something like **modules**?

GitOps + Helm



Credit: https://www.weave.works/

Demo

https://github.com/ttarczynski/gitops-demo

GitOps

- The **entire system** is described **declaratively**
- The canonical **desired** system **state** is **versioned** (with Git)
- **Approved changes** to the desired state are **automatically applied** to the system
- **Software agents** ensure **correctness**

GitOps: Flux docs

- github.com/weaveworks/flux
- github.com/stefanprodan/gitops-helm

Thanks!

Tomasz Tarczynski
@ttarczynski

Gigaset