CLOUD NATIVE

¿Por qué y cómo hacer Operators?

MIGUEL ÁNGEL GARCÍA

ARQCONF for IT

UTN FACULTAD REGIONAL CORDOBA
INGENIERÍA EN SISTEMAS DE INFORMACIÓN
AUDITORIO - ING. HECTOR AIASSA

Introducción

1. Kubernetes Operators
2. Qué problemas solucionan
3. Cómo funcionan
4. Implementación usando **Metacontroller**

https://github.com/flugel-it/arqconf-operators-talk

Kubernetes Operators

Los **operators** extienden el **API de Kubernetes** permitiendo simplificar el deploy y administración de aplicaciones complejas.

Encapsulan y abstraen los detalles referidos a la orquestación de los recursos de una aplicación.

Los usuarios pueden realizar el deploy y administración de aplicaciones:

- Sin tener que conocer los detalles
- Utilizando **kubectl**

Ejemplo deploy de una aplicación sin operator

Es necesario crear varios objetos: Namespace, Deployment, Service, ConfigMap, Ingress.

Problemas:

- Para hacer varios deploys hay que repetir el trabajo.
- El usuario tiene que saber como crear cada objeto y cómo interactúan.

Solución: deploy con operator

Usar un **operator** que permita al usuario especificar en un solo objeto los datos necesarios, los que varían de un deploy a otro.

- Usuario crea con kubectl:

```
apiVersion: customerapp.flugel.it/v1alpha1
kind: CustomerApp
metadata:
 name: customer1-customerapp
spec:
 customer: customer1
```

- El operator se encargará de crear y mantener: Namespace, Deployment, Service, ConfigMap, Ingress.

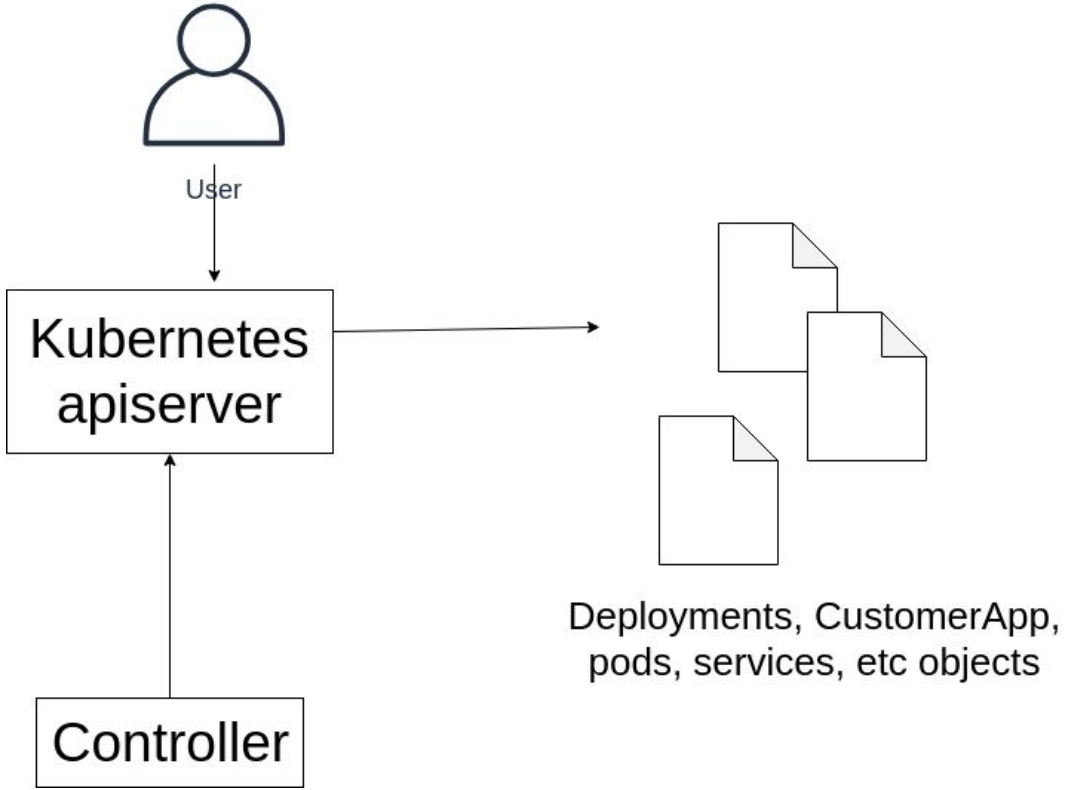
¿ Cómo funciona un operator ?

Extiende la API de Kubernetes. Al igual que el resto de la API de Kubernetes, la API que provee el operator es **declarativa**.

Custom Resource: Usando **kubectl** el usuario puede leer y escribir este resource para establecer el **estado deseado** y consultar el **estado real**.

Controller: proceso encargado de **reconciliar** el estado real con el estado deseado. Utiliza la API de Kubernetes para escuchar por eventos acerca de cambios en objetos y operar sobre resources.

¿ Cómo funciona un operator ?



Cómo implementarlo

1. **Crear Custom Resource Definition**
- Herramientas que ayudan a general el YAML
- A mano

2. **Crear controller**: Imagen Docker + Deployment
- Usando directamente la API de Kubernetes
- Operator SDK
- **Metacontroller**

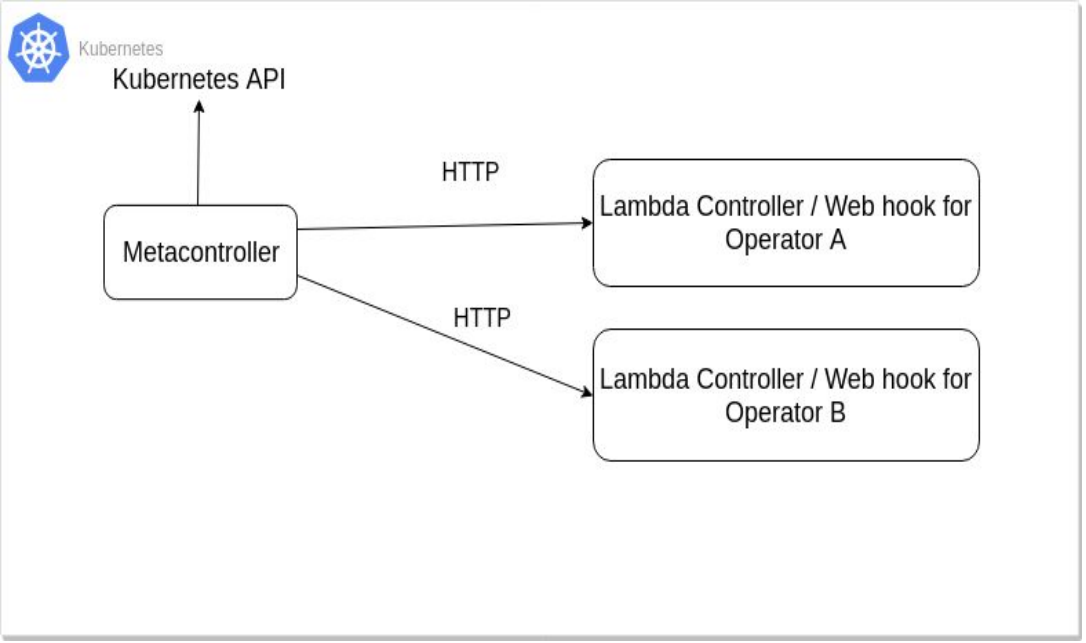
Metacontroller

Add-on para Kubernetes que simplifica el desarrollo de controllers. Proyecto open source iniciado por Google. (https://metacontroller.app/)

Ventajas:

- Abstrae la detección de eventos, consulta de estado actual y ejecución de operaciones para efectuar cambios.
- Permite usar casi cualquier lenguaje de programación.
- El desarrollador sólo tiene que proporcionar la funcionalidad que calcula a partir del estado actual el estado deseado.

Metacontroller - funcionamiento



1. Metacontroller listen events relevantes a los controller registrados
2. Invoca Web hook pasandole el estado actual
3. Web hook retorna el estado deseado
4. Metacontroller hacer “apply” de los cambios

Demo: Instalación de Metacontroller

Demo: Operator para deploy de aplicación

Caso de uso: proveer una app no multi-tenant a varios clientes. Para cada cliente require un Namespace, Deployment, Service, Ingress y ConfigMap. Lo único que varía para cada instancia es **CustomerId**.

Objetivo: Permitir al usuario realizar el deploy de una instancia especificando solamente CustomerId.

- Custom Resource: Custom Resource Definition
- **Registro en Metacontroller**: Declaración de parent,children y service
- **Web hook**: Servicio para calcular el estado deseado
- Prueba

Demo: auto publish de services en ingress

Caso de uso: publicar vía Ingress services que cumplan ciertas condiciones.

Objetivo: Publicar automáticamente vía Ingress los services que tengan cierta annotation en un path. (services.example.com/{service}.{namespace}/)

Además de los **CompositeController**, Metacontroller provee **DecoratorController** para extender la funcionalidad de resources existente.

- ~~**Custom Resource**~~
- Registro en Metacontroller
- Web hook
- Prueba

¡ Gracias !

¿ Preguntas ?

https://github.com/flugel-it/arqconf-operators-talk