Building Extensible Application Orchestration System

邓德源 郭维 章骏

deyuan@caicloud.io
guowei@cailcoud.io
jim.zhang@caicloud.io

Survey
Before Presentation

Catalog

- Container Orchestration on Kubernetes
- Application Orchestration and Helm
- Building Extensible Application Orchestration System

Orchestration ?

Orchestration（编排）

Orchestration（本意）：为管弦乐中的配器法，主要是研究各种管弦乐器的运用和配合方法，通过各种乐器的不同音色，以便充分表现乐曲的内容和风格。

Orchestration（计算机领域）：引申为描述复杂计算机系统、中间件（middleware）和业务的自动化的安排、协调和管理。

编配 or 编排？

Orchestration应该翻译为“编配”，为了方便符合大家的习惯，我们还是使用编排来描述下面问题

但我们习惯用编排来描述这个概念。

小知识：关于是使用“编配（Orchestration）”还是“编排（Choreography）”，这里有一篇文章，有兴趣可以看一下。链接地址

What is Container Orchestration ?

Why do we need container orchestration ?

在传统的单体式架构的应用中，我们开发、测试、交付、部署等都是针对单个组件。

而在云的时代，微服务和容器大行其道，为我们显示出了它们在敏捷性，可移植性等方面的巨大优势后，也为我们交付和运维带来了新的挑战：

1. 服务之间的依赖管理
2. 服务发现
3. 资源管理
4. 高可用
5. 等等

caicloud
才云

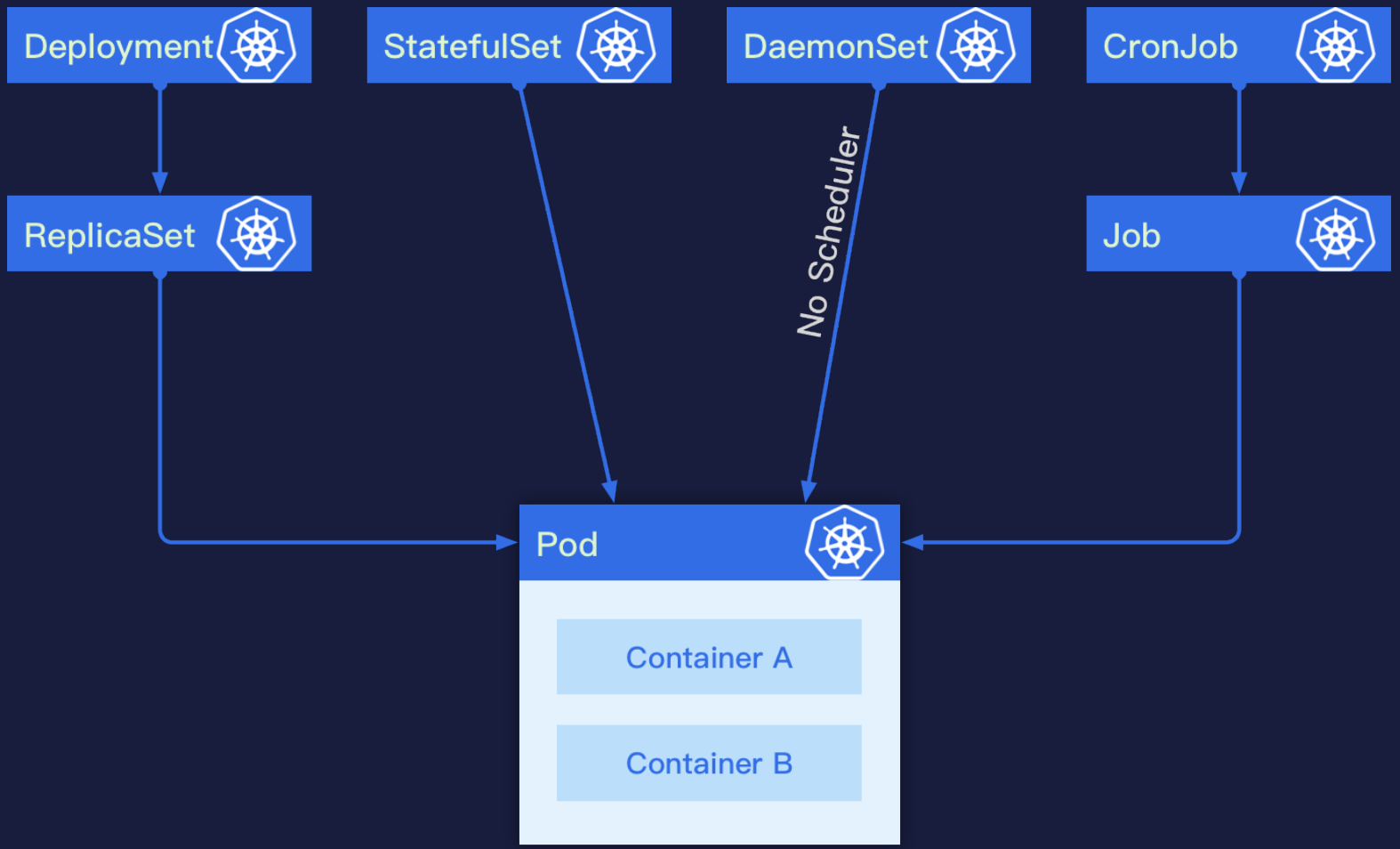
Container Orchestration on Kubernetes

在容器环境中，编排通常涉及以下几个方面：

- 资源编排： 负责合理分配调度 Node，CPU， Memory 等资源
- 工作负载编排： 负责在资源之间共享工作负载
- 服务编排：负责容器之间的服务发现和高可用等
- 弹性伸缩：负载自动调整计算资源来满足负载变化的情况

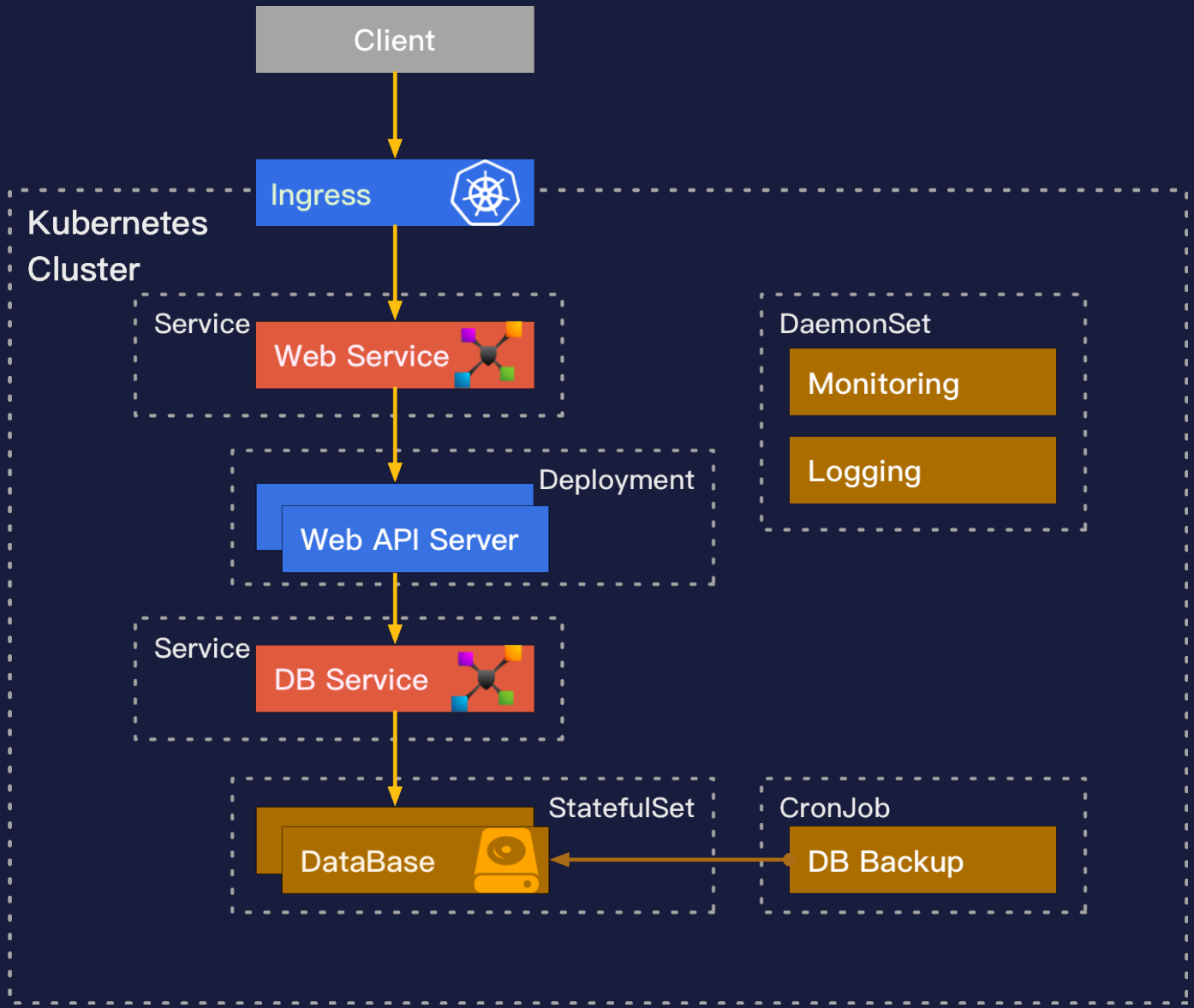
caicloud
才云

Kubernetes Workloads



caicloud
才云

For Example



caicloud
才云

Kubernetes 好处都有啥？

基础资源封装

- Service
- ConfigMap
- PV/PVC
- Workloads: pods deployments statefulset ...

自动化运维机制

- HPA
- VPA
- Rolling Update

简化的部署方式

- 资源可以用 yaml 表示
- 提供命令行简化了操作

等等。。。

caicloud
才云

But ...
How to Deploy a Mysql?

But ...

- 手写 yaml 真的累：我们很难编辑和维护如此多的服务和配置
- 服务依赖难搞定：发布应用的时候未能将这些服务作为整体发布，需要理解他们之间的关系，然后依次启动
- 服务无法复用：两个服务都需要使用到相同的数据库类型时，我们只能拷贝一份修改后再启动
- 不支持应用整体的版本管理
- 没有应用级别的状态追踪和健康检查

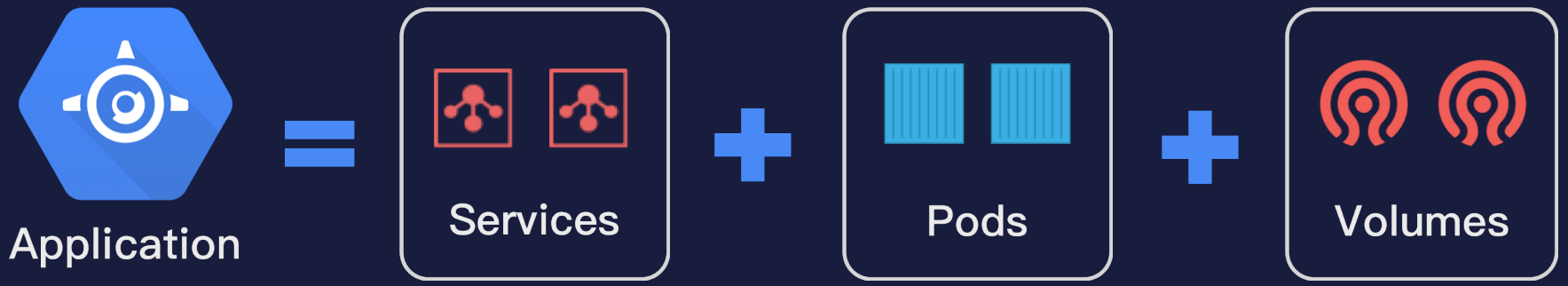
caicloud
才云

Catalog

- Container Orchestration on Kubernetes
- Application Orchestration and Helm
- Building Extensible Application Orchestration System

What is Application?

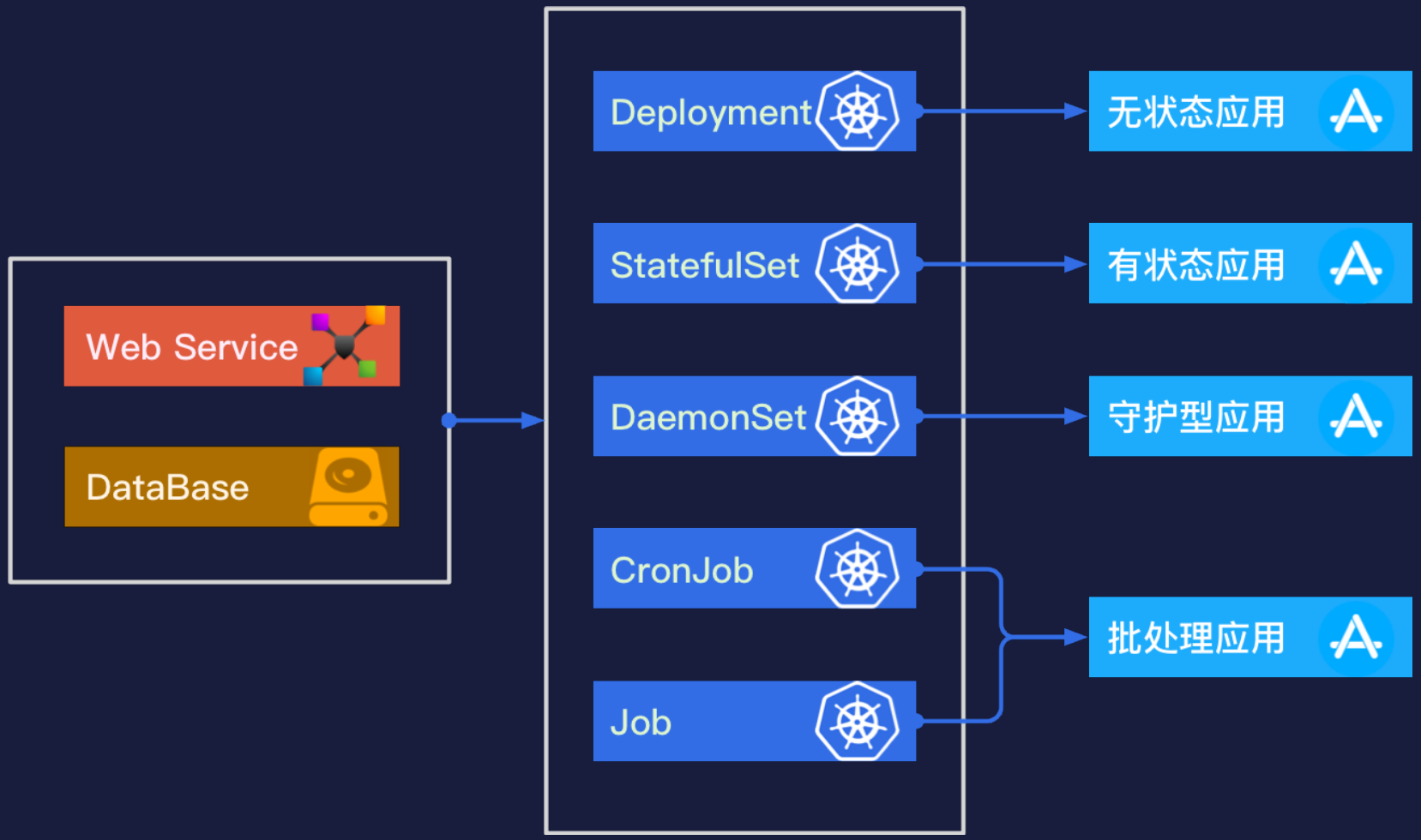
Application



应用 网络 计算 存储

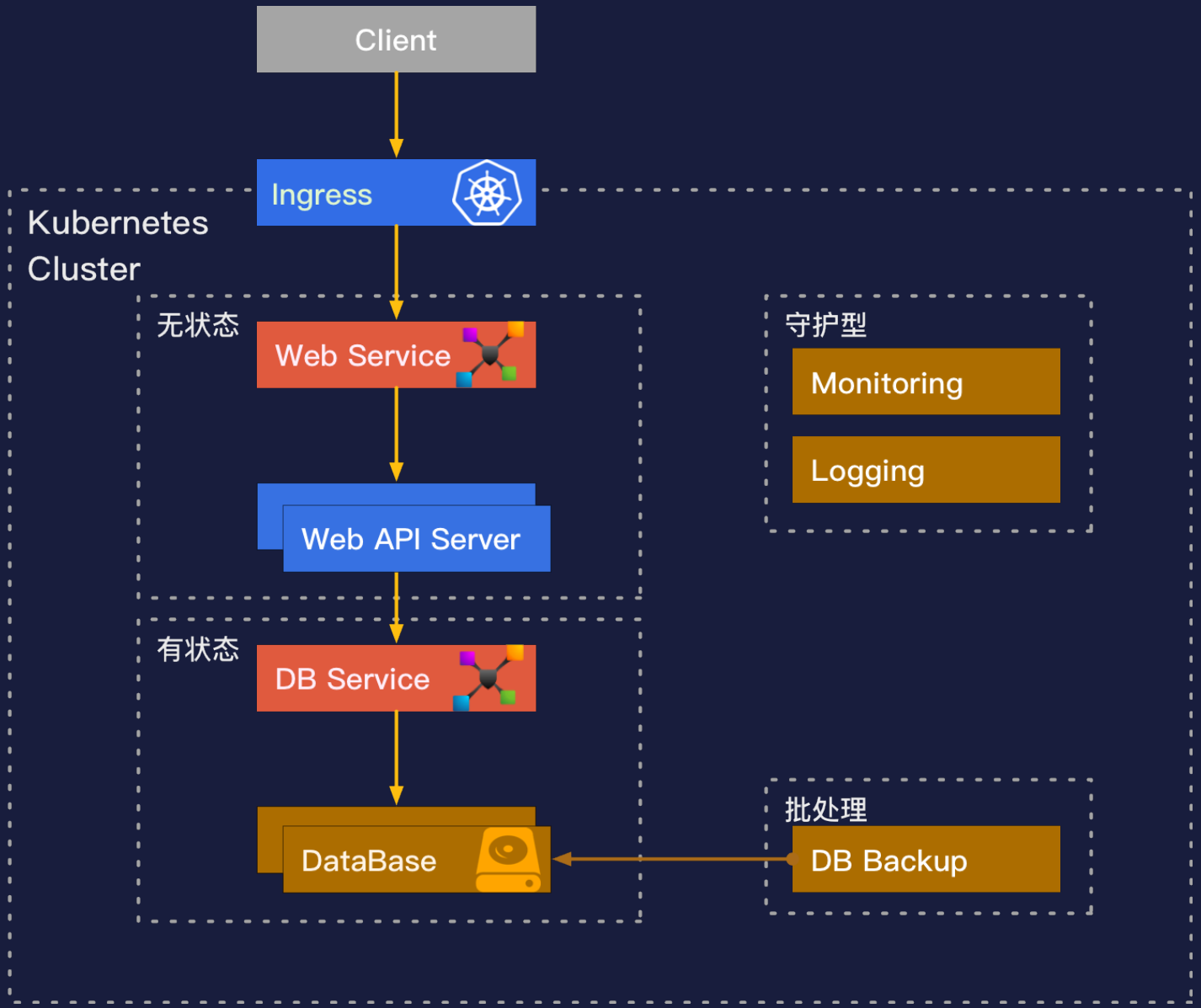
caicloud
才云

Application on Kubernetes



caicloud
才云

Review the example



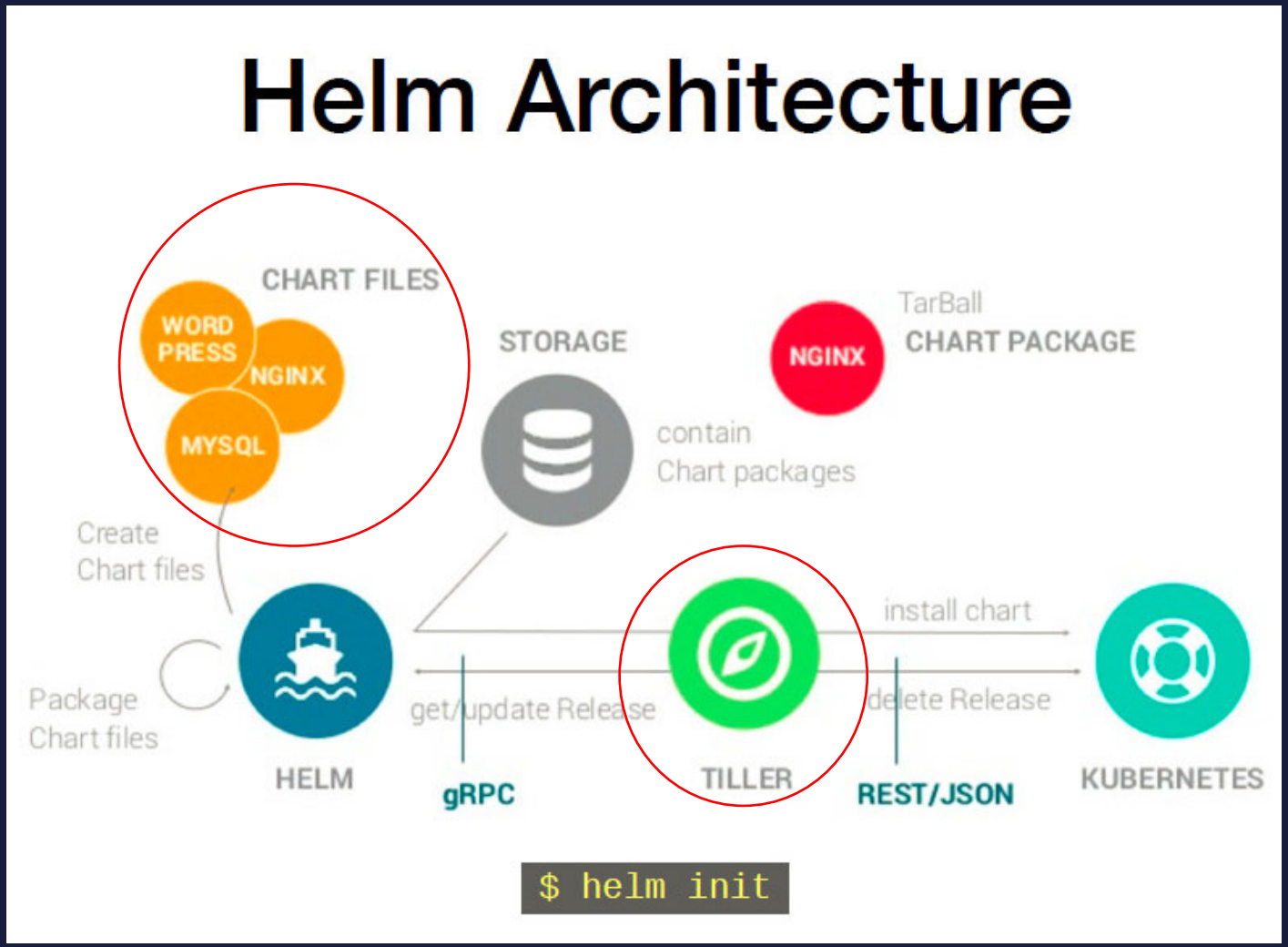
caicloud
才云

Four Issues of Application

1. Package Format And Architecture
2. Dependency And Version Management
3. Package Storage
4. Runtime Management

Helm

Helm



Reference: https://www.slideshare.net/alexLM/helm-application-deployment-management-for-kubernetes

caicloud
才云

Tiller, what is the problem

1. 没有内建的认证授权机制，Tiller 跑在 kube-system 分区下，拥有整个集群的权限。
2. Tiller 将 release 安装到 Kubernetes 集群中后并不会主动继续追踪他们的状态
3. Helm + Tiller 的架构并不符合 Kubernetes 的设计模式，这就导致它的拓展性比较差
4. Tiller 创建的 Release 是全局的并不是在某一个分区下，这就导致多用户/租户下，不能进行隔离
5. Tiller 的回滚机制是基于更新的，每次回滚会使版本号增加，这不符合用户的直觉

Helm 3 Design Proposal
https://github.com/kubernetes-helm/community/blob/master/helm-v3/000-helm-v3.md

Catalog

- Container Orchestration on Kubernetes
- Application Orchestration and Helm
- **Building Extensible Application Orchestration System**

Rudder

An application package orchestration runtime system based on kubernetes controller pattern.

Two Custom Resource Definition:

- Release
- Release History

One Controller

- Rudder aka Release Controller

caicloud
才云

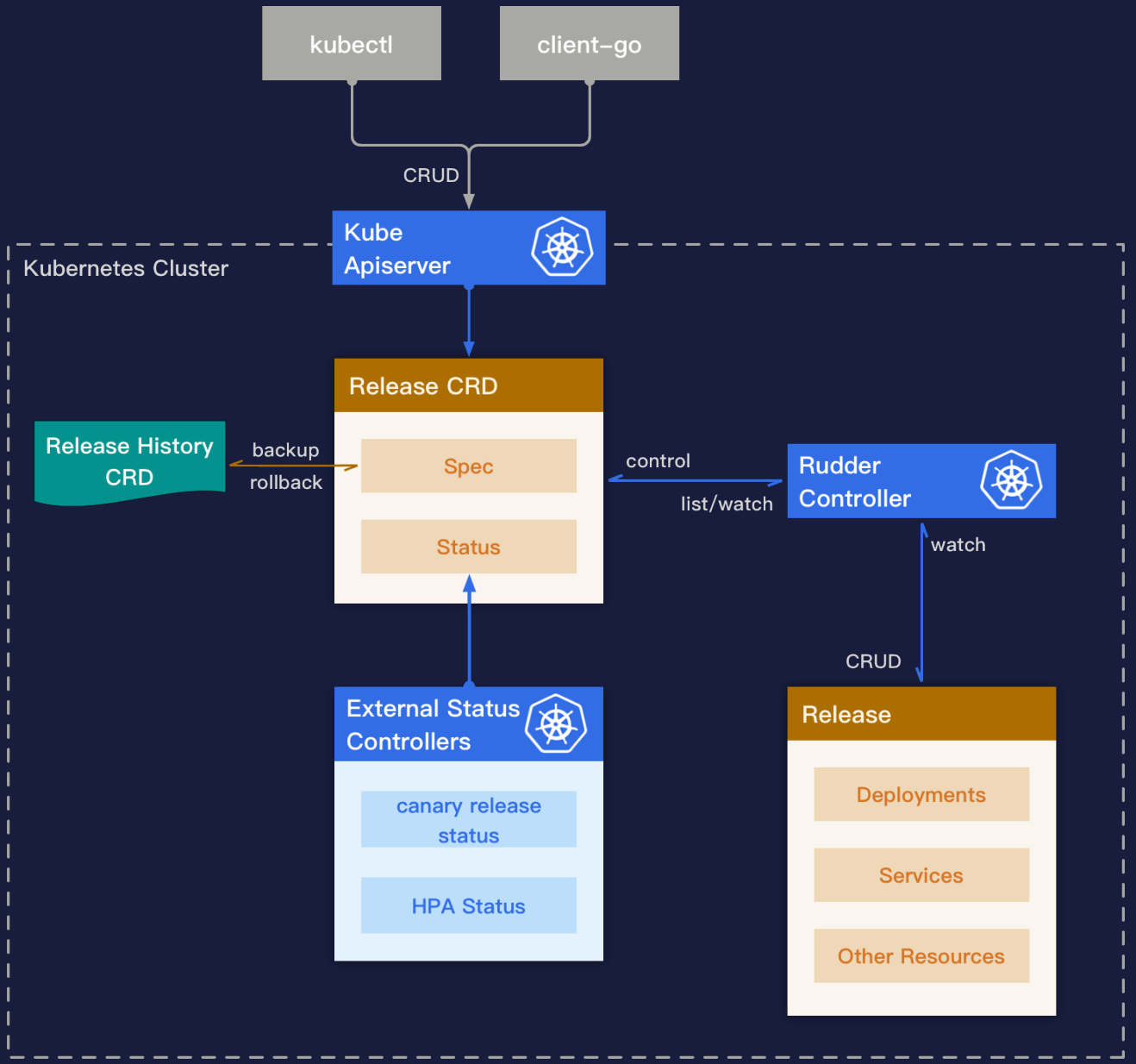
Release

Release 的概念与 Helm 中的 release 相似，它也包含了 Values 和 Templates

只不过 Release 是个 CRD 被集群里面的 Rudder Controller 监听着，由 Rudder 来控制 Release 下面子资源的生命周期，完成了类似于 Tiller 的工作。

caicloud
才云

Rudder Architecture



caicloud
才云

The Good

1. Isolation：release 使用 namespace 隔离，适用于多用户/租户
2. Security：所有的操作都是基于 Kubernetes 的 Resource，所以可以重复利用 Kubernetes 内置的认证鉴权模块，如ABAC, RBAC
3. Readability：Rudder controller 会主动追踪每个 Release 的子资源的状态，你可以很容易的看到整个应用现在处于什么状态中
4. Version Control：可以很容易地回退到某一个确定的版本
5. Extensibility：整个架构是遵循 Kubernetes 的 controller pattern，具有良好的可拓展性，可以基于它做二次开发

caicloud
才云

Extensibility

Open Source

- https://github.com/caicloud/rudder
- https://github.com/caicloud/canary-release
- https://github.com/caicloud/charts
- https://github.com/caicloud/helm-registry

caicloud
才云

The Orchestration
is not only a Technology,
but also an Art.

caicloud

才云

Thanks For Watching