KubeCon | CloudNativeCon

Europe 2018

Writing Kubernetes controllers for CRDs

Alena Prokharchyk, Rancher Labs

whoami

Alena Prokharchyk,

Principal Software Engineer @RancherLabs

@lemonjet

alena1108

So you need a new k8s feature

Writing a custom controller is the way to go when:

- Feature is not generic enough to become a part of the k8s platform

- You want to maintain feature development and release lifecycle

New feature implies new custom resource(s) that **user** can

- View
- Configure
- Monitor

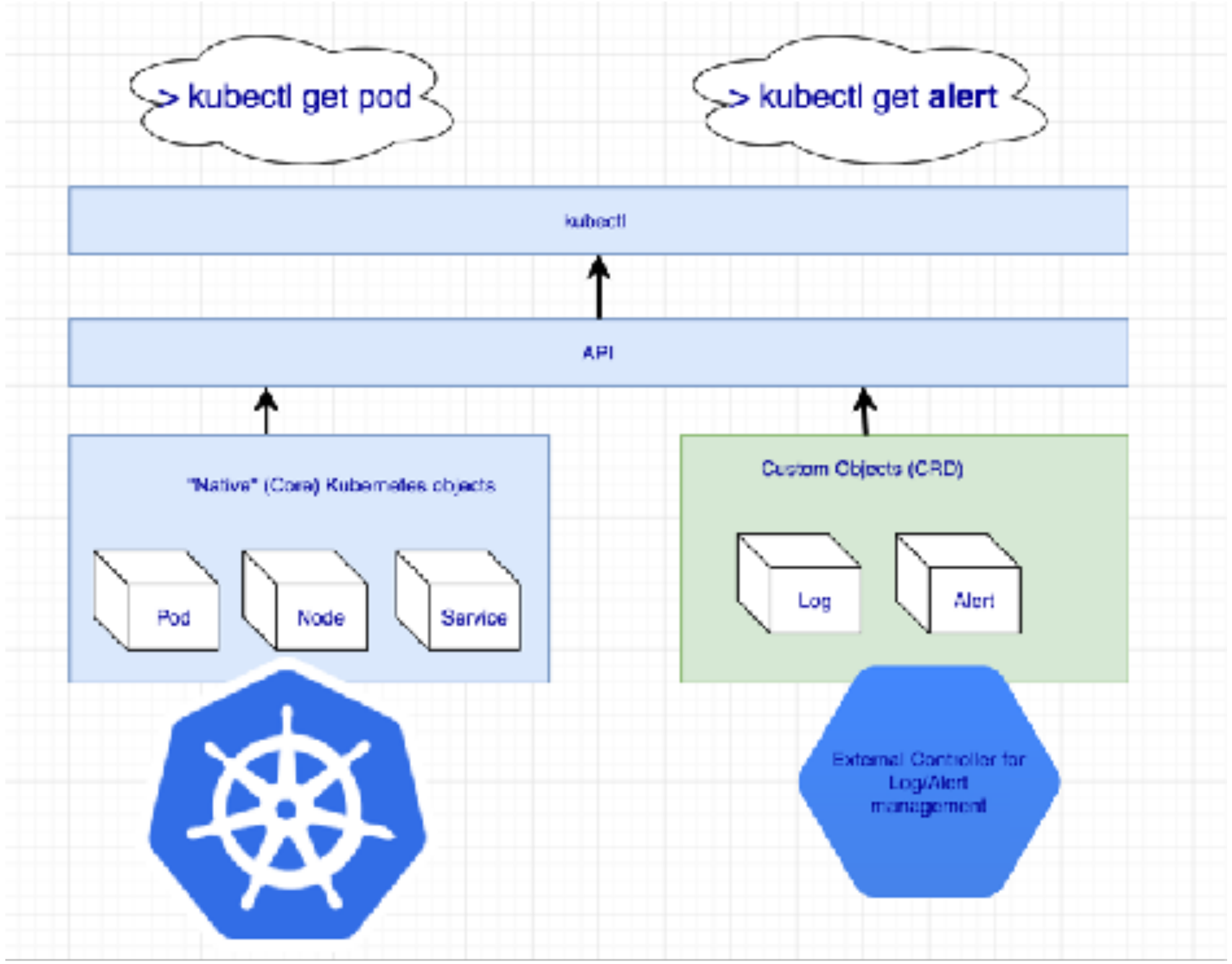
And controller can operate on the custom resource to:

- Run backend logic based on resource definition
- Update object to reflect the actual state of the resource

Kubernetes Ecosystem

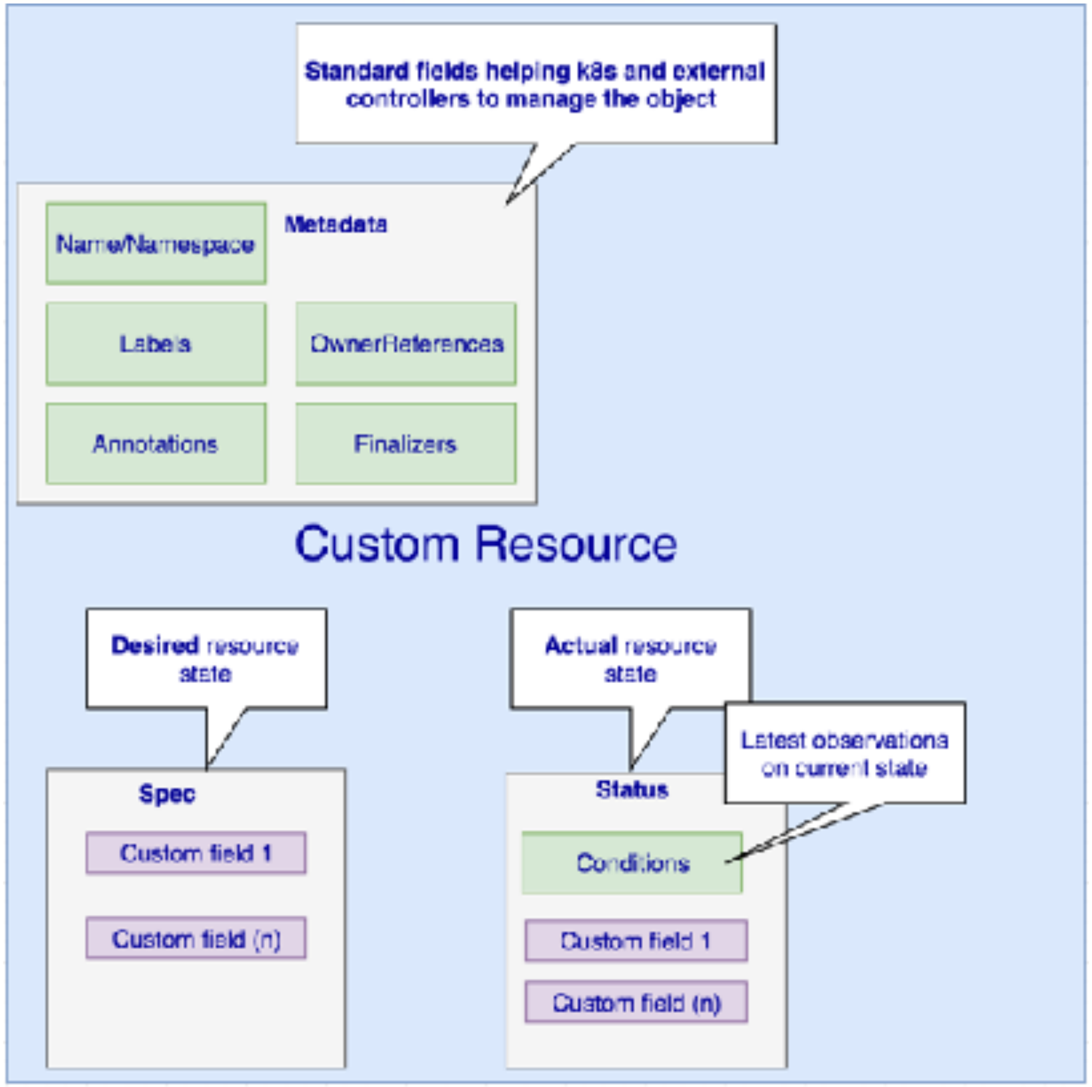
Is made of custom controllers

Custom Resources



- Strongly typed
- Top-level support from API and kubectl
- Ability to subscribe to resource change events

What makes a custom resource



- Metadata, spec, status are recommended fields to have to leverage k8s capabilities like Garbage Collection, pre-delete hooks, etc
- Status.conditions is advised to have as an alternative to a single state field
- The rest of the fields are custom, and solely driven by external controller implementation/use

Lets build…

Kubernetes clusters management tool that will let user:

- Create/view/delete Kubernetes clusters by operating on custom resource **cluster** using kubectl

- Access provisioned cluster by using custom resource **kubeconfig** fetched using kubectl

Things we are going to demo

- Client generation for custom resources cluster and kubeconfig
- Handling cluster create/update events by calling cluster installer tool
- Utilize resource Conditions field to reflect cluster state
- Use Finalizer to execute pre-delete hook on cluster.remove
- Leverage k8s garbage collection using ownerReferences field on child resource

Tools used

- K8s code-generator to create client/informers/other useful functions for the custom resource https://github.com/kubernetes/code-generator
- RKE - open source Kubernetes installer https://github.com/rancher/rke
- Demo controller logic can be found here: https://github.com/alena1108/kubecon2018

Demo time!

Demo reflections

Be careful with update logic

Eliminate infinite updates by either:

* comparing current spec with the previous spec

* for update that are meant to run only once, introduce Condition to reflect whether the update happened(-ing)

K8s 1.10 offers new construct reducing update problems - Object Status as a Custom Object: https://blog.openshift.com/kubernetes-custom-resources-grow-up-in-v1-10/

Conditions

KubeCon CloudNativeCon Europe 2018

* Each condition should represent a certain **single** functionality state. For multi functionality reflection, consider introducing more conditions

* Avoid updating the same condition by multiple controllers.

Finalizer

Set on the object, so on its removal controller(s) get a chance to run a custom cleanup logic

Owner reference

Nice way to delegate “child” objects cleanup to k8s Garbage Collector

Thank you!