Extending the Kubernetes API

2018-02-27

Jago Macleod
Engineering Manager, Google Cloud

Google Cloud

Agenda

The Kubernetes Resource Model, Properties and Conventions

Extending the Kubernetes API with Custom Resources

Industry Implications

Kubernetes - you've heard the pitch:

Improves developer productivity

Enables efficient bin-packing

Self healing

Enables orchestration of planet-scale applications

… **and so on and so forth.**

Not that talk.

“

Kubernetes is not just API-driven,

but is **API-*centric***.

– Brian Grant, Principal Engineer, Google

Kubernetes Resource Model

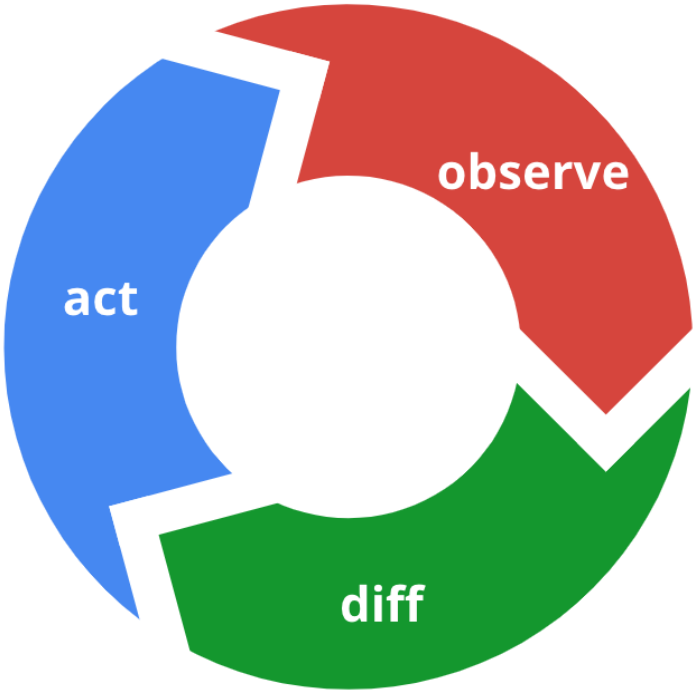
Humans and Automated Systems:

Observe the current state of the system;

Diff current (status) against the desired (spec) state;

Act to bring the system into alignment with the desired state.

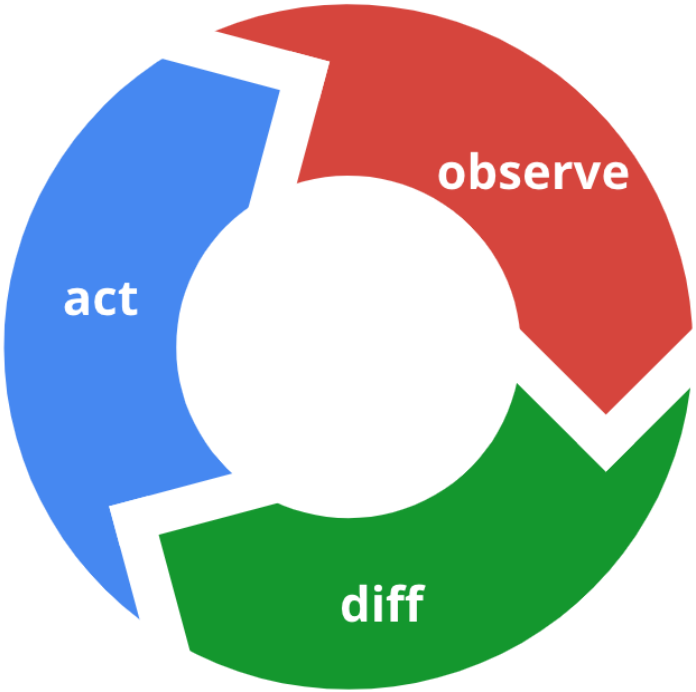
Repeat…



Kubernetes Resource Model

Humans and Automated Systems use the same APIs

- Observed state is Truth
- Multiple actors are assumed and supported
- Resources are not assumed to have a single, exclusive "owner"
- No strong ordering guarantees or transactions across multiple resources



Graceful tolerance over guarantees

Desired state is updated immediately,

but actuated asynchronously and eventually.

Accordingly, Kubernetes intentionally **does not support** strong ordering guarantees, pessimistic locking, atomic transactions across resources, strict resource ownership and does not enforce referential integrity.

Life of a K8s Request - mutation

1. Authentication
2. Authorization: Built-in and/or administrator-defined identity-based policies
3. Defaulting: API-version-specific default values are made explicit and persisted
4. Conversion: Auto-conversion between the client-requested and internal API versions
5. Admission control: Built-in and/or administrator-defined resource-type-specific policies
6. Validation: Resource field values are validated.
7. Idempotence: Resources are accessed via immutable client-provided, declarative-friendly names
8. Optimistic concurrency: Writes may specify a ‘check-and-set’ style precondition
9. Audit logging: Records the sequence of changes to each resource by all actors

Life of a K8s Request - deletion

1. Graceful termination: Some resources support delayed deletion, which is indicated by **deletionTimestamp** and **deletionGracePeriodSeconds** being set upon deletion
2. Finalization: A **finalizer** is a block on deletion placed by an external controller, and needs to be removed before the resource deletion can complete
3. Garbage collection: A resource may specify **ownerReferences**, in which case the resource will be deleted once all of the referenced resources have been deleted

Life of a K8s Request - get

1. List: All resources of a particular type within a particular namespace may be requested; response chunking is supported
2. Label selection: Lists may be filtered by their label keys and values
3. Watch: A client may subscribe to changes to listed resources using the resourceVersion returned with the list results

Portable by Design

Kubernetes runs on every major cloud provider, on prem and on bare metal. It will even run on a cluster of Raspberry Pis.

Currently over **50 Certified Kubernetes™** platforms and distributions.

Extensible by Design

Because the distinction between being part of the system and being built on top of the system is intentionally blurred, ecosystem developers can extend the Kubernetes API surface through the creation of **Custom Resources** and **Custom Controllers**.

Custom Resources

Extensibility

Aggregated API Servers

- Subordinate API Server that sits behind a proxy
- Provider is responsible for storage
- More flexible

Custom Resource Definitions

- User defines schema and, optionally, validation
- Gaining widespread adoption and evolving

CRDs

```yaml
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  # name must match the spec fields below, and be in the form: <plural>.<group>
  name: crontabs.stable.example.com
spec:
  # group name to use for REST API: /apis/<group>/<version>
  group: stable.example.com
  # version name to use for REST API: /apis/<group>/<version>
  version: v1
  # either Namespaced or Cluster
  scope: Namespaced
  names:
    # plural name to be used in the URL: /apis/<group>/<version>/<plural>
    plural: crontabs
    # singular name to be used as an alias on the CLI and for display
    singular: crontab
    # kind is normally the CamelCased singular type. Your resource manifests use this.
    kind: CronTab
    # shortNames allow shorter string to match your resource on the CLI
    shortNames:
    - ct
```

```
bash-3.2$ kubectl create -f resourcedefinition.yaml
customresourcedefinition "crontabs.stable.example.com" created
```

CRDs

```yaml
apiVersion: "stable.example.com/v1"
kind: CronTab
metadata:
  name: my-new-cron-object
spec:
  cronSpec: "* * * * */10"
  image: my-awesome-cron-image
```

```
bash-3.2$ kubectl create -f my-crontab.yaml
crontab "my-new-cron-object" created
```

CRDs

```
bash-3.2$ kubectl describe crontabs my-new-cron-object
Name:         my-new-cron-object
Namespace:    default
Labels:       <none>
Annotations:  <none>
API Version:  stable.example.com/v1
Kind:         CronTab
Metadata:
  Cluster Name:
  Creation Timestamp:  2018-02-25T20:25:38Z
  Resource Version:    2336
  Self Link:           /apis/stable.example.com/v1/namespaces/default/crontabs/my-new-cron-object
  UID:                 0ad79960-1a6a-11e8-969b-42010a8002f0
Spec:
  Cron Spec:  * * * * */5
  Image:      my-awesome-cron-image
Events:       <none>
```

Industry Implications

Kubernetes is **portable** and **extensible** by design.

The Kubernetes Resource Model has proven **effective**, and is being more **widely adopted** by other projects in the space.

...and support for a growing collection of languages is bringing new users.

More than Container Orchestration

More than Container Orchestration, it is the **Kubernetes Resource Model** that is transforming the way distributed systems are designed, built and operated.

There is still so much work to do. Please join the Kubernetes Community to find out how you can get involved!

Thank you

Google Cloud

References

Kubernetes Resource Management, Brian Grant

Extend the Kubernetes API with Custom Resource Definitions, Kubernetes authors

kubernetes.io documentation, Kubernets authors

Kubernetes API conventions, Kubernetes authors