



Kubernetes operator pattern

Fabrice Jammes, Karim Ammous, <https://k8s-school.fr>

Credits: Daniel Messer Product Manager, OpenShift - Guilherme Barros Product Manager, Cloud BU



<https://k8s-school.fr>

Operator across the industry

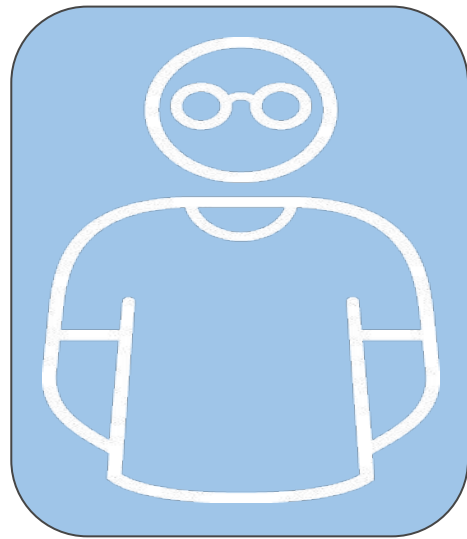
Operator across the industry

OperatorHub.io | The registry for Kubernetes Operators

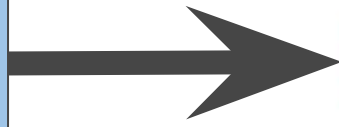


What is an Operator?

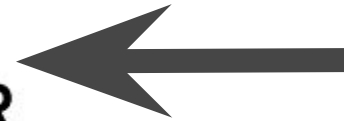
Operators embed ops knowledge from the experts



ops knowledge from the experts



**OPERATOR
SDK**



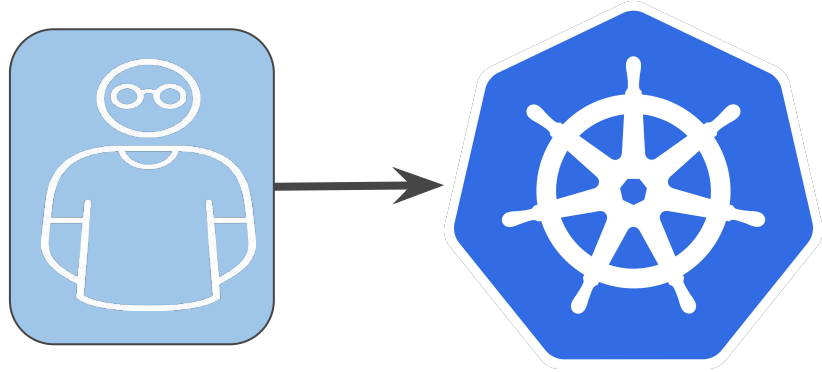
operator implementation
i.e. k8s controller

Deployments
StatefulSets
Autoscalers
Secrets
Config maps

See

- <https://kubernetes.io/docs/concepts/extend-kubernetes/operator/>
- <https://cloud.google.com/blog/products/containers-kubernetes/best-practices-for-building-kubernetes-operators-and-stateful-apps>

How does an operator works?



Software Developer
Kubernetes user

K8s API



Kubernetes operator

Native Kubernetes
resources

Custom resource

```
kind: ProductionReadyDatabase
apiVersion: database.example.com/v1alpha1
metadata:
  name: my-important-database
spec:
  connectionPoolSize: 300
  readReplicas: 2
  image: productionreadydatabase:v4.0.1
```

Custom Kubernetes controller

Watch Event

Reconcile

Custom resource definition

here ProductionReadyDatabase



Deployments
StatefulSets
Autoscalers
Secrets
Config maps

Why should you use an operator?

Operators: both sysadmin + application experts

⚙️ **Resize/Upgrade**

⚙️ **Reconfigure**

⚙️ **Backup**

⚙️ **Healing**



The Sysadmin

What make a good operator?

Operators: best practices for development

- One Operator per managed application
- Write an Operator-of-Operators for complex, multi-tier application stacks
- CRD can only be owned by a single Operator, shared CRDs should be owned by a separate Operator
- One controller per custom resource definition
- Use an SDK like Operator SDK
- Do not hard-code namespaces or resources names
- Make watch namespace configurable
- Use semver / observe Kubernetes guidelines on versioning APIs
- Use OpenAPI spec on CRDs

Operators: best practices for development

- Does not run as root
- Does not self-register CRDs
- Writes meaningful status information on Custom Resources objects
- Is capable of updating from a previous version of the Operator
- Is capable of managing an Operand from an older Operator version
- Does not deploy other Operators
- Uses CRD conversion (webhooks) if API/CRDs change
- Uses Admission Webhooks to reject invalid CRs
- Should always be able to deploy and come up without user input
- Offers configuration via an “Configuration CR”

Multiple operator frameworks

Operators

- **kudo**: simple, no need to code, not so popular:
<https://github.com/kudobuilder/operators/tree/master/repository>
- **metacontroller**: simple, no need to code, started at Google

Based on [kubernetes-sigs/controller-runtime: Repo for the controller-runtime subproject of kubebuilder \(sig-apimachinery\)](#)

and [kubernetes-sigs/controller-tools: Tools to use with the controller-runtime libraries](#)

- **operator-framework**: complex, code in golang, popular, well-documented (book)
- **kubebuilder**: complex, code in golang, popular, well-documented (book)

See <https://gist.github.com/tiewei/d98c663cf76b61bf835c1ebf87b36999>

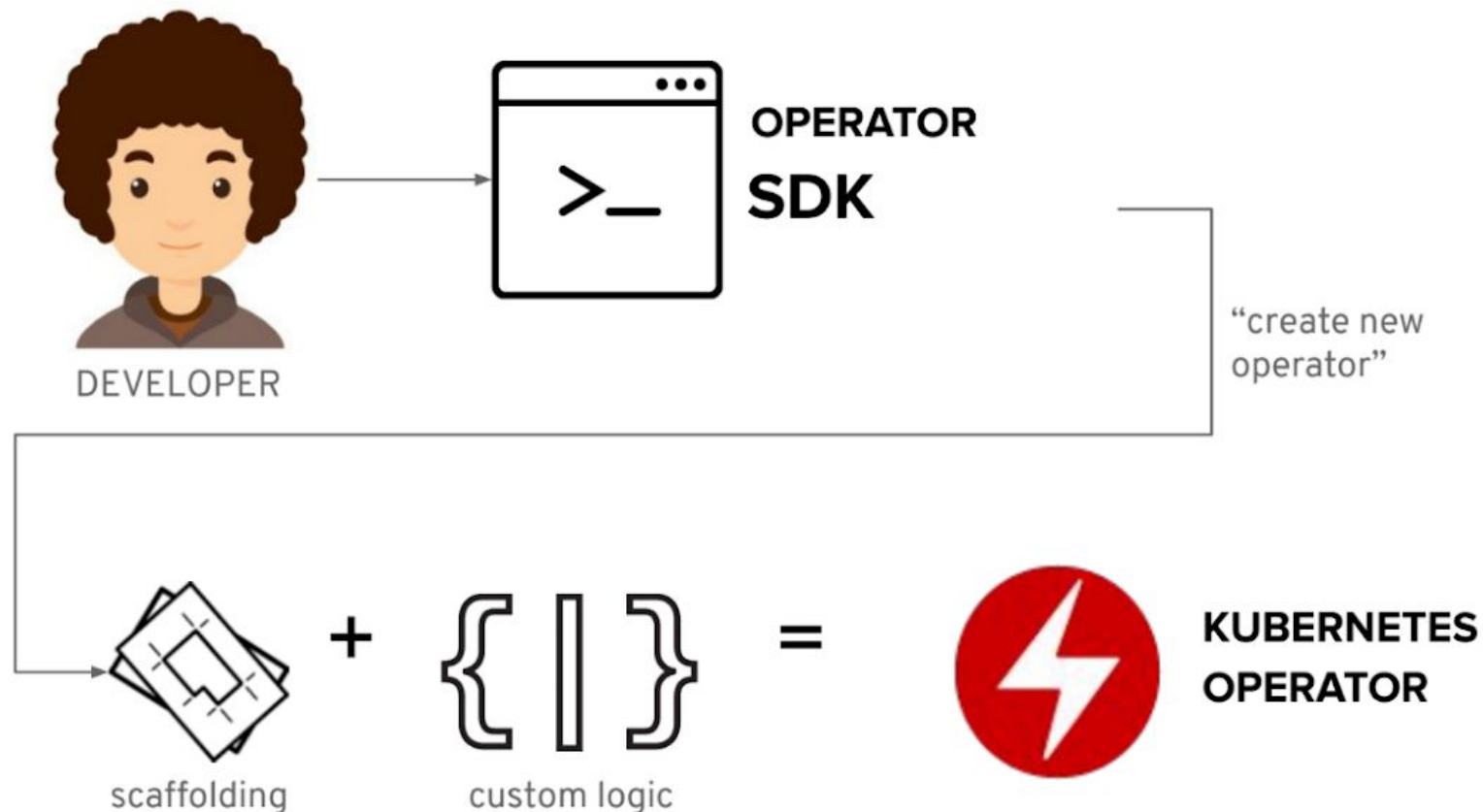
operator-framework

Operator framework in action

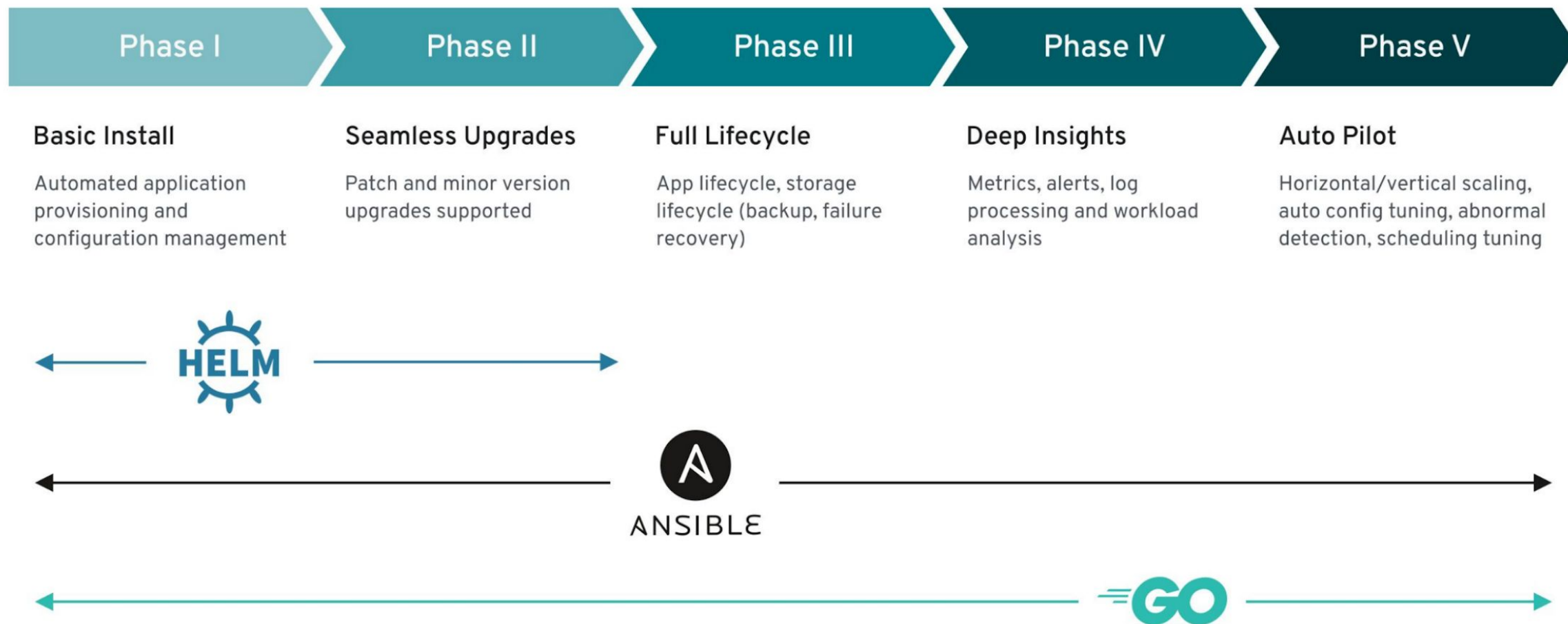
Based on KubeBuilder libraries:

[kubernetes-sigs/controller-runtime](https://github.com/kubernetes-sigs/controller-runtime): Repo for the controller-runtime subproject of kubebuilder (sig-apimachinery)

[kubernetes-sigs/controller-tools](https://github.com/kubernetes-sigs/controller-tools): Tools to use with the controller-runtime libraries



Operator SDK: types of operators



OperatorHub: the MongoDB example

OperatorHub.io

Search OperatorHub...

Contribute ▾

Home > MongoDB

MongoDB

The MongoDB Enterprise Kubernetes Operator enables easy deploys of MongoDB into Kubernetes clusters, using our management, monitoring and backup platforms, Ops Manager and Cloud Manager.

The Operator has beta support for a containerized Ops Manager with the MongoDBOpsManager custom resource.

Install

CHANNEL

stable

VERSION

1.4.1 (Current) ▾

CAPABILITY LEVEL ⓘ

- Basic Install
- Seamless Upgrades
- Full Lifecycle
- Deep Insights
- Auto Pilot

PROVIDER

MongoDB, Inc

LINKS

[Documentation](#) ↗

REPOSITORY

[https://github.com/mongodb/mo](https://github.com/mongodb/mongodb-kubernetes-operator)

Before You Start

To start using the operator you'll need an account in MongoDB Cloud Manager or a MongoDB Ops Manager deployment.

- [Create a Secret with your OpsManager API key](#)
- [Create a ConfigMap with your OpsManager project ID and URL](#)

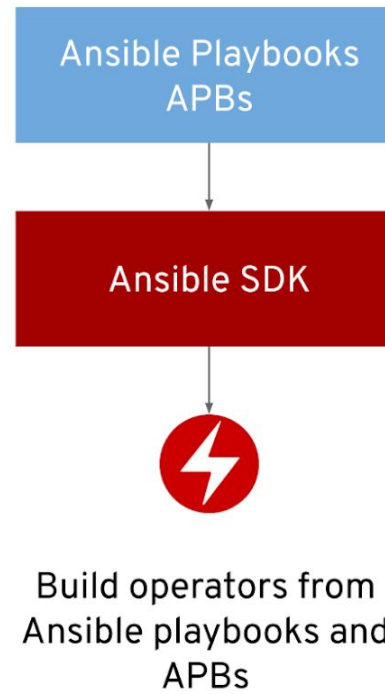
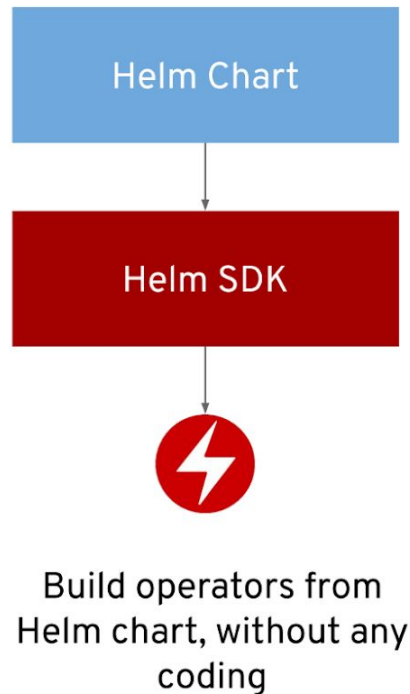
By installing this integration, you will be able to deploy MongoDB instances with a single simple command.

Required Parameters

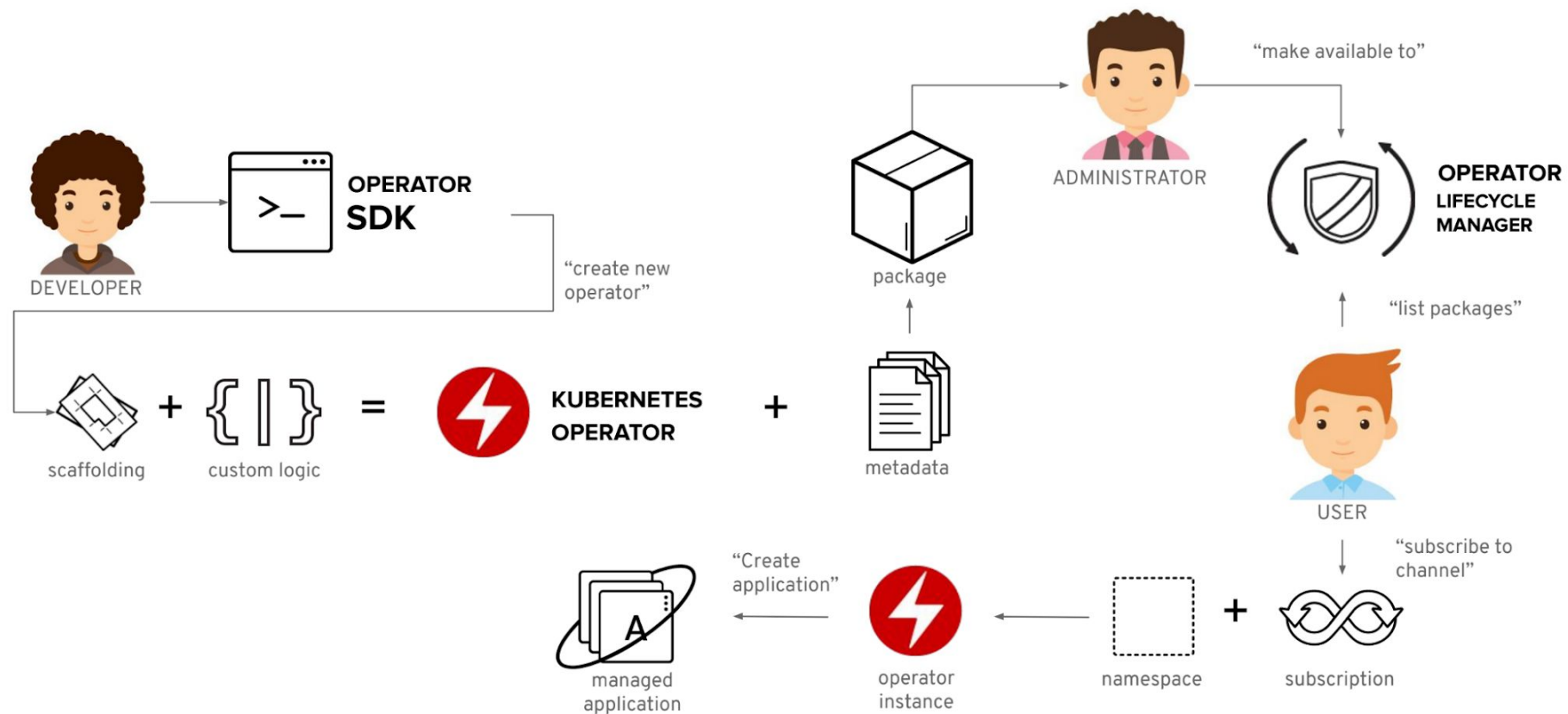
Operator SDK: types of operators

Perfectly integrated with Openshift ;-)

CNCF project : incubating



Operator framework: the full stack



Summary

What we have seen:

- Operators **ease application delivery and management** over Kubernetes.
- Operator goal is to **automate sysadmins tasks**.
- **Multiple operator frameworks** are competing right now.
- **For complex application, KubeBuilder and Operator-SDK** are serious tracks.
- **OperatorHub.io** provides lots of operators, with possible source code examples.

Demos

Redis with KubeDB: <https://travis-ci.com/k8s-school/kubedb-example>

Qserv-operator: <https://travis-ci.org/lsst/qserv-operator/builds/651390166>

Questions?

Merci!

[@fjammes](#) on Github



<https://k8s-school.fr>