Red Hat OpenShift

Breakout Operators

Robert Bohne
SR. SPECIALIST SOLUTION ARCHITECT | OPENSHIFT
Twitter: @RobertBohne

Red Hat

What is an Operator?

Operator is a
automated software manager
that deals with the installation and life cycle of
an applications on top of Kubernetes/OpenShift.

Controller

Piece of software that deals with the installation and life cycle of an applications on top of OpenShift.

Custom Resource Definition (CRD)

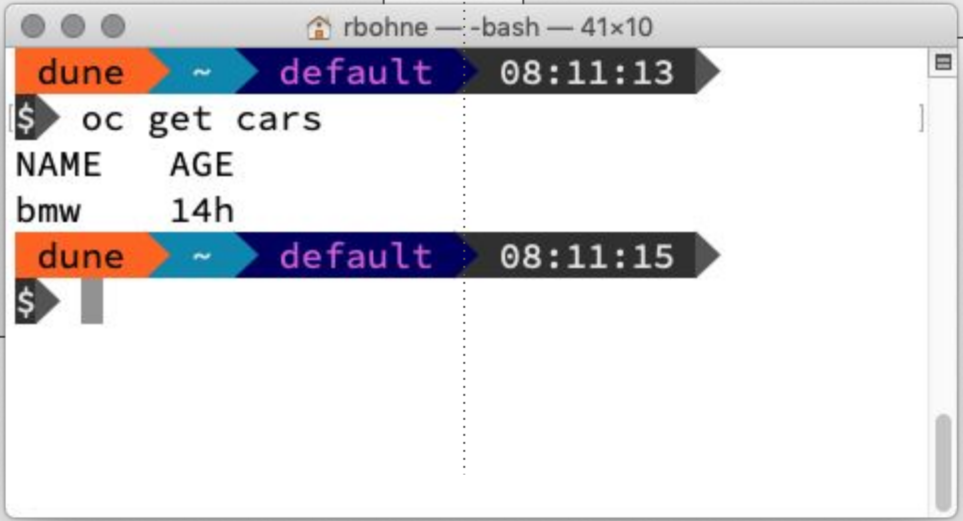
OpenShift API extension to interact and communicate with the Controller.

Custom Resource Definition (CRD)

```yaml
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: cars.openshift.pub
spec:
  group: openshift.pub
  names:
    kind: Car
    listKind: CarList
    plural: cars
    singular: car
  scope: Namespaced
  subresources:
    status: {}
  version: v1
```

Custom Resource (CR)

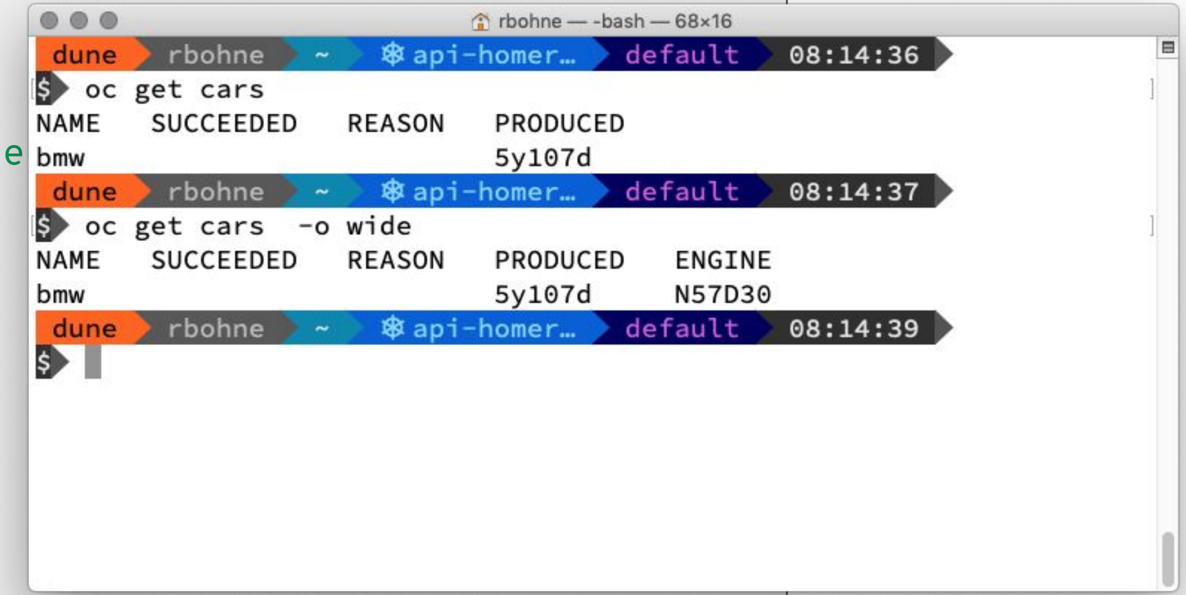
```yaml
apiVersion: openshift.pub/v1
kind: Car
metadata:
  name: bmw
spec:
  date_of_manufacturing: "2014-07-01T00:00:00Z"
  engine: N57D30
```



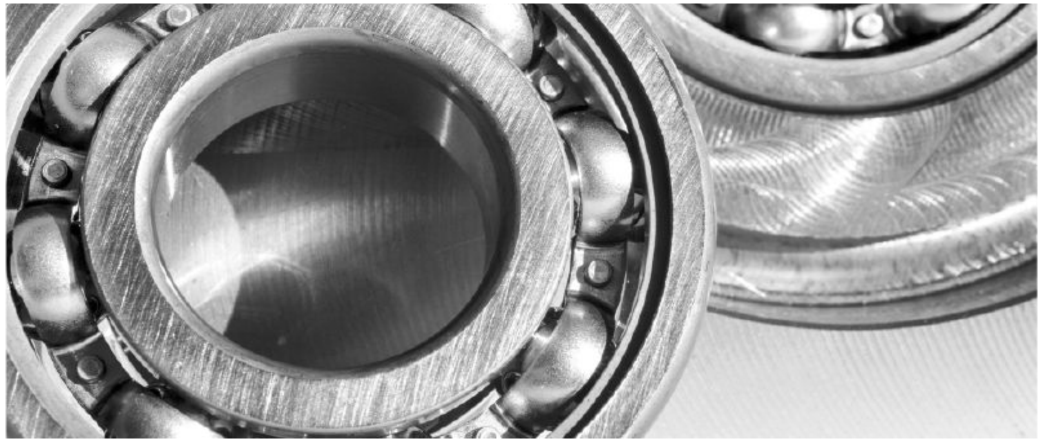
```
rbohne — -bash — 41×10
 dune  ~  default  08:11:13
$ oc get cars
NAME    AGE
bmw     14h
 dune  ~  default  08:11:15
$
```

Custom Resource Definition (CRD)

```
[..snipped..]
  additionalPrinterColumns:
  - JSONPath: .status.conditions[?(@.type=="Succeeded")].status
    name: Succeeded
    type: string
  - JSONPath: .status.conditions[?(@.type=="Succe
    name: Reason
    type: string
  - JSONPath: .spec.date_of_manufacturing
    name: Produced
    type: date
  - JSONPath: .spec.engine
    name: Engine
    type: string
    priority: 1
```



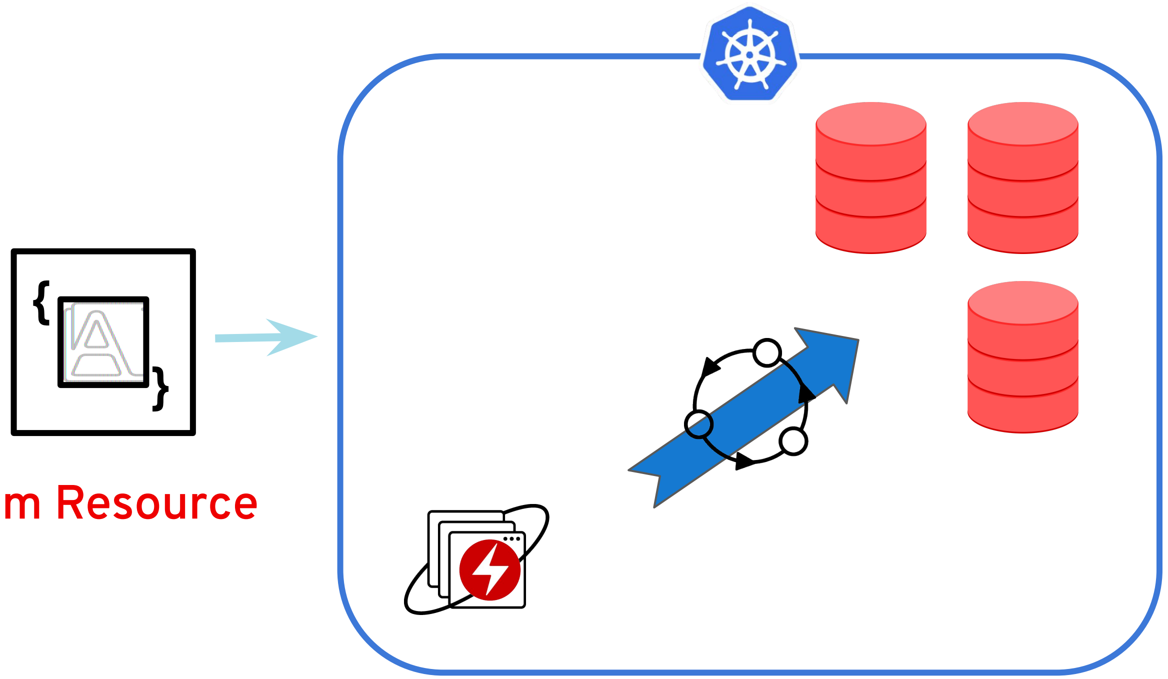
```
rbohne — -bash — 68×16
dune  rbohne  ~  api-homer…  default  08:14:36
$ oc get cars
NAME    SUCCEEDED   REASON   PRODUCED
bmw                          5y107d
dune  rbohne  ~  api-homer…  default  08:14:37
$ oc get cars  -o wide
NAME    SUCCEEDED   REASON   PRODUCED   ENGINE
bmw                          5y107d     N57D30
dune  rbohne  ~  api-homer…  default  08:14:39
$
```

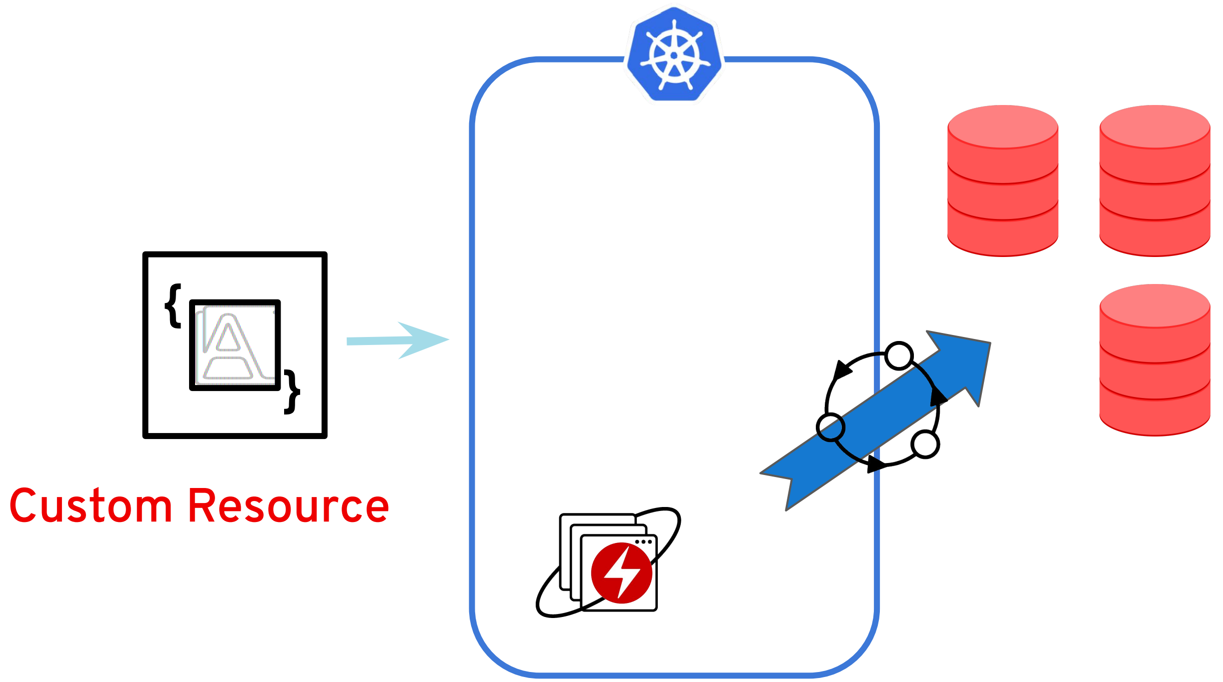


Controller

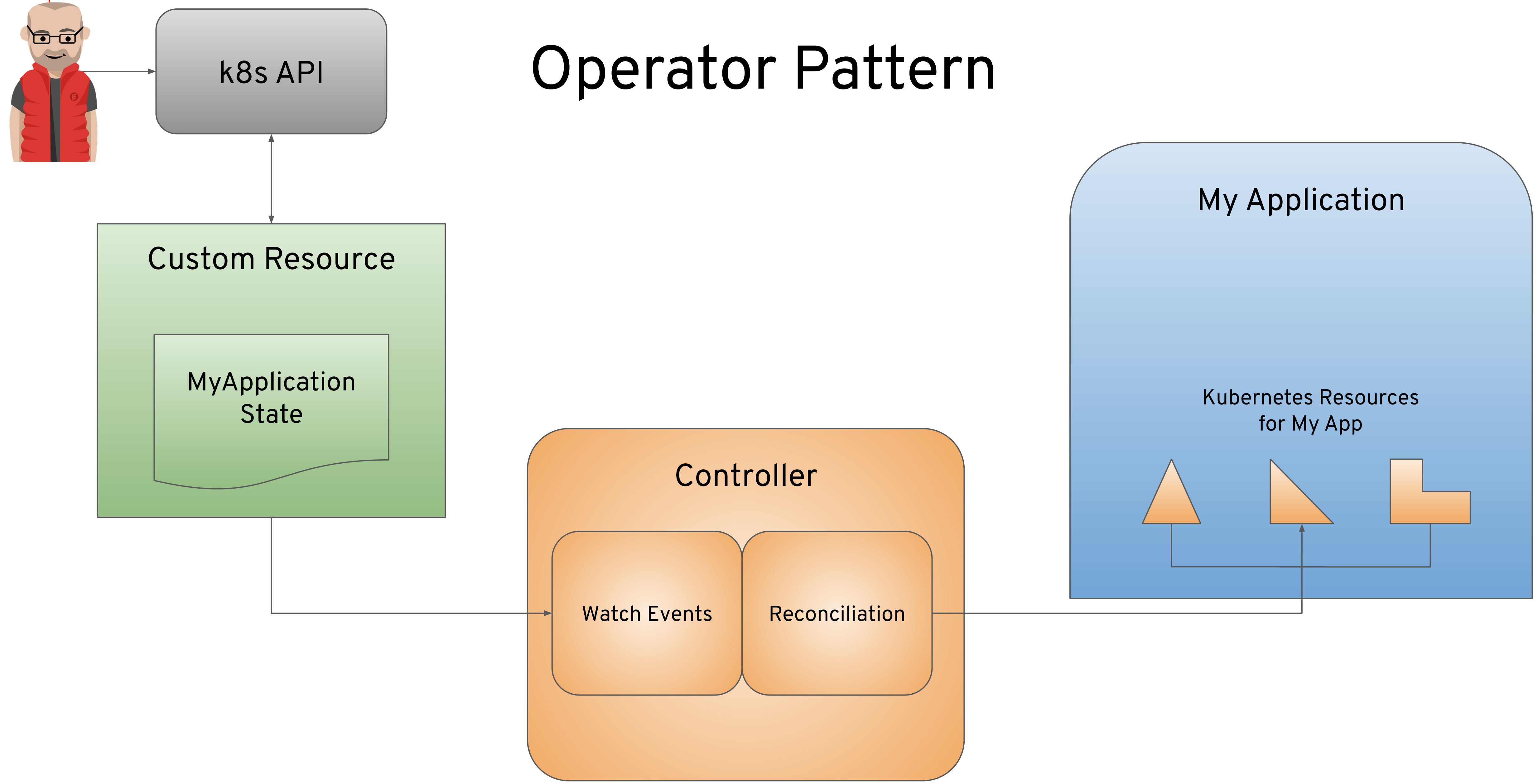
Piece of software running on top of OpenShift

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: operator
spec:
  replicas: 1
  selector:
    matchLabels:
      name: operator
  template:
    metadata:
      labels:
        name: operator
    spec:
      containers:
        - name: operator
          Image: "quay.io/org/operator:v0.0.1"
```


Custom Resource


Custom Resource

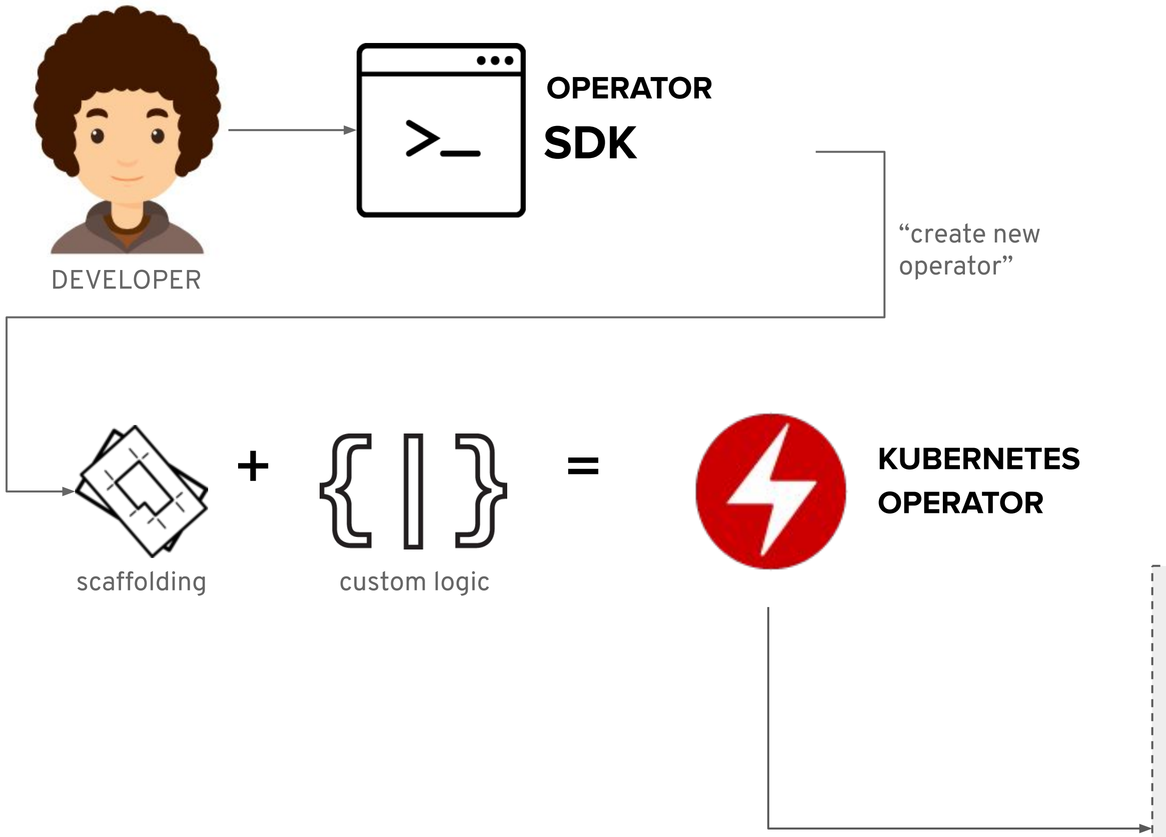
Operator Pattern



How to create an Operator?

OPERATOR SDK

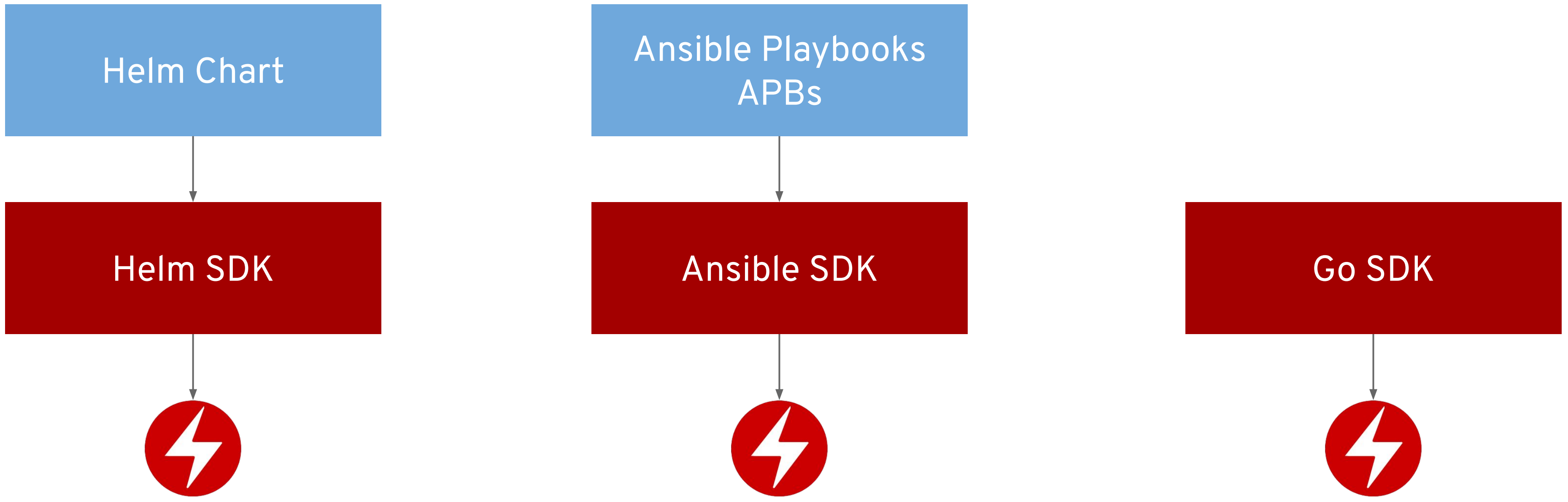
Operator Framework in Action



implemented as a container image plus:

```
Deployment
Role
ClusterRole
RoleBinding
ClusterRoleBinding
ServiceAccount
CustomResourceDefinition
```

Operator Adoption Patterns



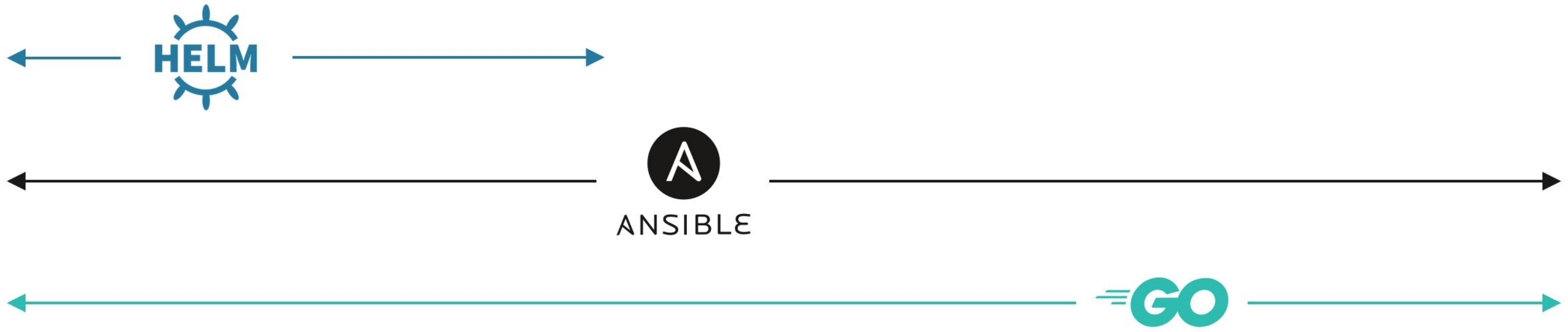
Build operators from Helm chart, without any coding

Build operators from Ansible playbooks and APBs

Build advanced operators for full lifecycle management

Types of Operators

Phase I	Phase II	Phase III	Phase IV	Phase V
Basic Install	**Seamless Upgrades**	**Full Lifecycle**	**Deep Insights**	**Auto Pilot**
Automated application provisioning and configuration management	Patch and minor version upgrades supported	App lifecycle, storage lifecycle (backup, failure recovery)	Metrics, alerts, log processing and workload analysis	Horizontal/vertical scaling, auto config tuning, abnormal detection, scheduling tuning



Use Cases ideal for operator development

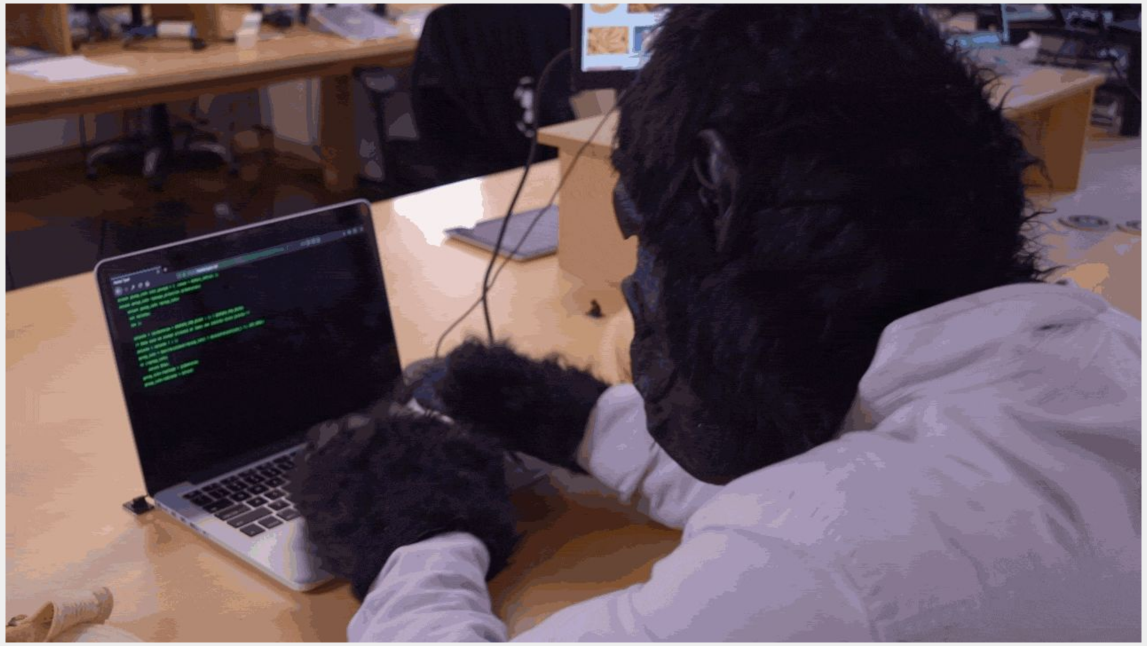
- stateful applications (such as databases)
- clustered or high-availability applications (clustered databases, key value stores such as etcd, in-memory cache clusters such as redis)
- multiservice applications (an application which needs dependent services to come online first)
- microservices (numerous small components that together make up a cohesive product or app)

Use cases less ideal for operator development

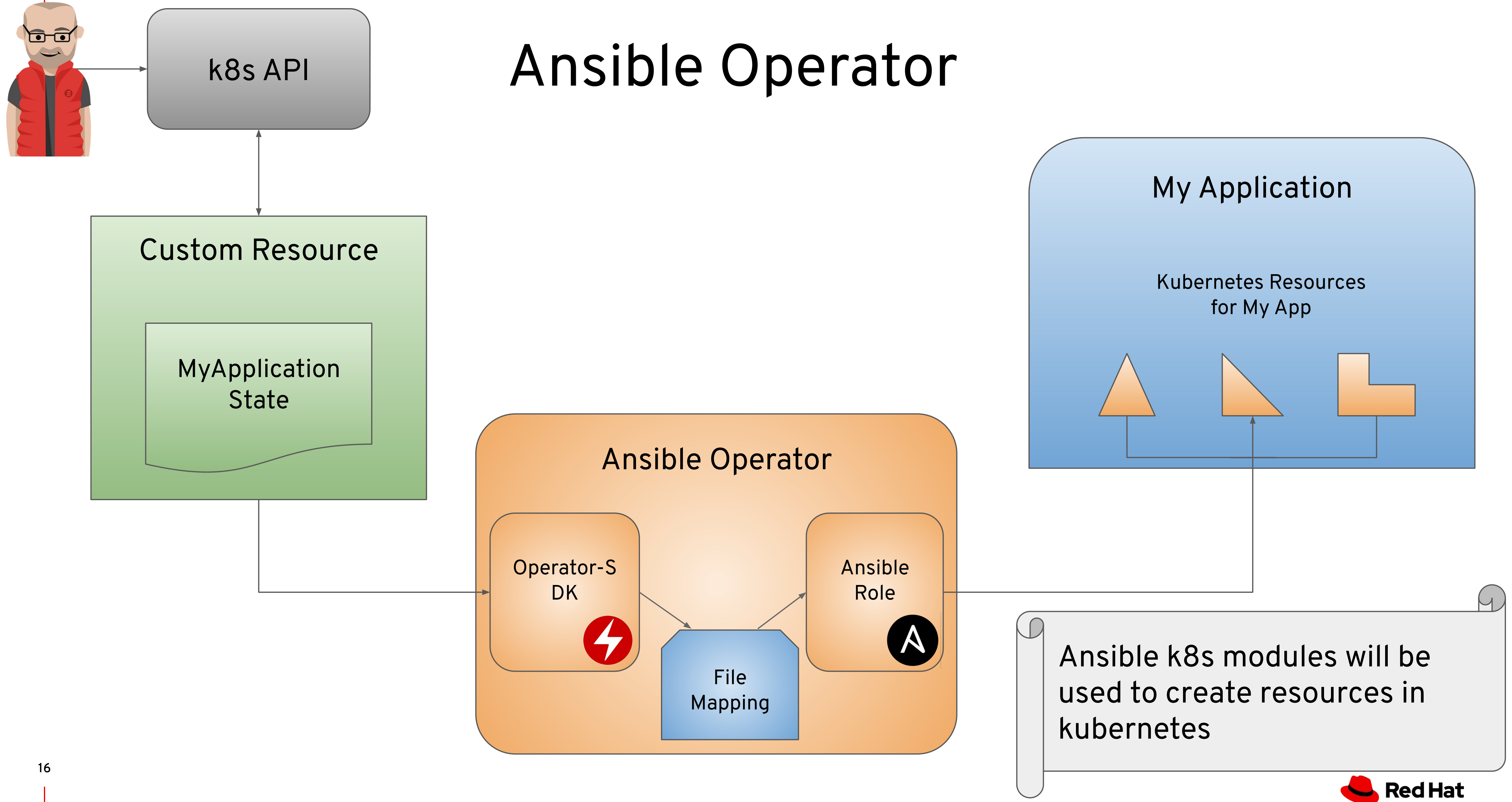
- stateless apps (most web apps or front-ends)
- infrastructure/host agents (monitoring agents or log forwarders)

Demo

https://examples.openshift.pub/operator#ansible-operator-example



Ansible Operator



Ansible k8s modules will be used to create resources in kubernetes

Red Hat

What you need to create an Ansible Operator

- A *CustomResourceDefinition* (CRD)
- An Ansible Playbook or Role
- A mapping from *CRD* to Ansible playbook / roles
- `operator-sdk`

Create the Operator with the SDK

```
$ operator-sdk new memcached-operator          \
    --api-version=cache.example.com/v1alpha1   \
    --kind=Memcached --type=ansible
```

Creates:

- Ansible Role
- Mapping File (watches.yaml)
- *Custom Resource Definition*
- Deploy manifest for the new Operator

Red Hat

Custom Resource (CR)

```
apiVersion: <Group/Version>
kind: <kind>
metadata:
  name: <name>
spec:
   <key>: <value>
   ….
status:
  <key>: <value>
  ….
```

Ansible Operator

Spec values will be translated to Ansible extra vars.

Status will be a generic status defined by the operator. This will use ansible runner output to generate meaningful output for the user.

Ansible Role

```
memcached/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml
```

Create a Role that deploys and manages your application

Mapping between *CRDs* and Ansible

Maps a Group Version Kind (GVK) to a role or playbook.

```
# watches.yaml
---
- version: v1alpha1
     - group: cache.example.com
        kind: Memcached
        playbook: /path/to/playbook
```

Build the Operator with the SDK

```
$ operator-sdk build memcached-operator:v0.0.1
```

Creates:

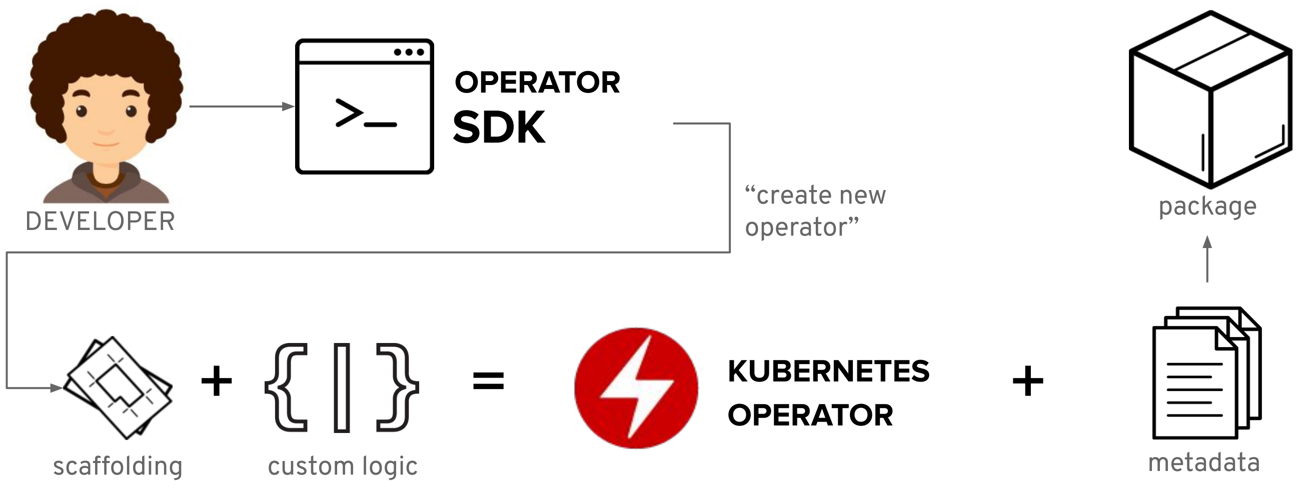
- A Dockerfile that creates the Operator
- Builds the container on top of ansible-runner image

What is an Operator?

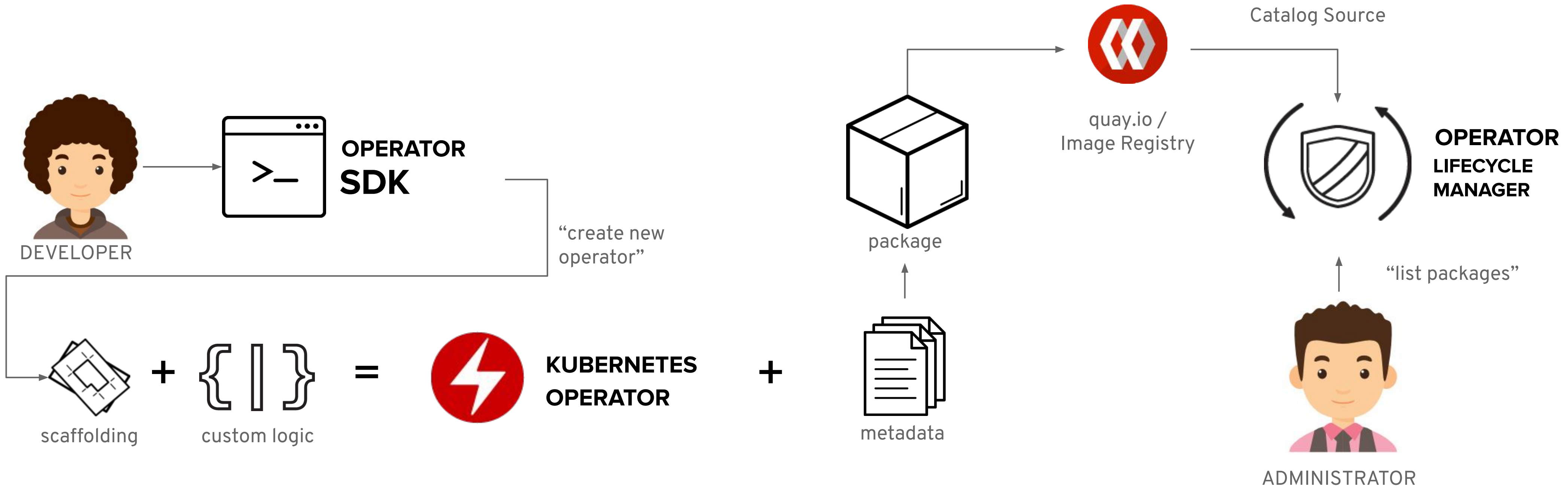
- Container Image (contains the software manager)

$$\left.\begin{array}{l}\bullet\ \text{Deployment}\\ \bullet\ \text{Custom Resource Definition (CRD)}\\ \bullet\ \text{Role Bindings}\\ \bullet\ \dots\end{array}\right\}\ \text{Metadata}$$

What is an Operator?



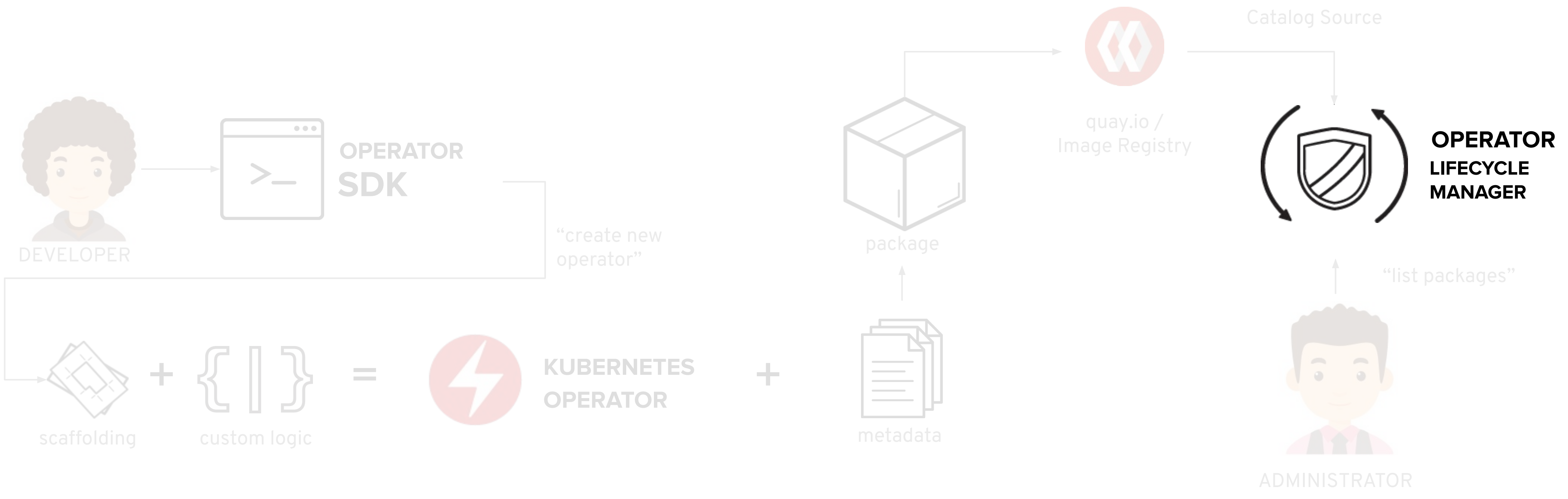
How to ship an Operator?



Package Discovery:

```
$ oc get packagemanifests
```

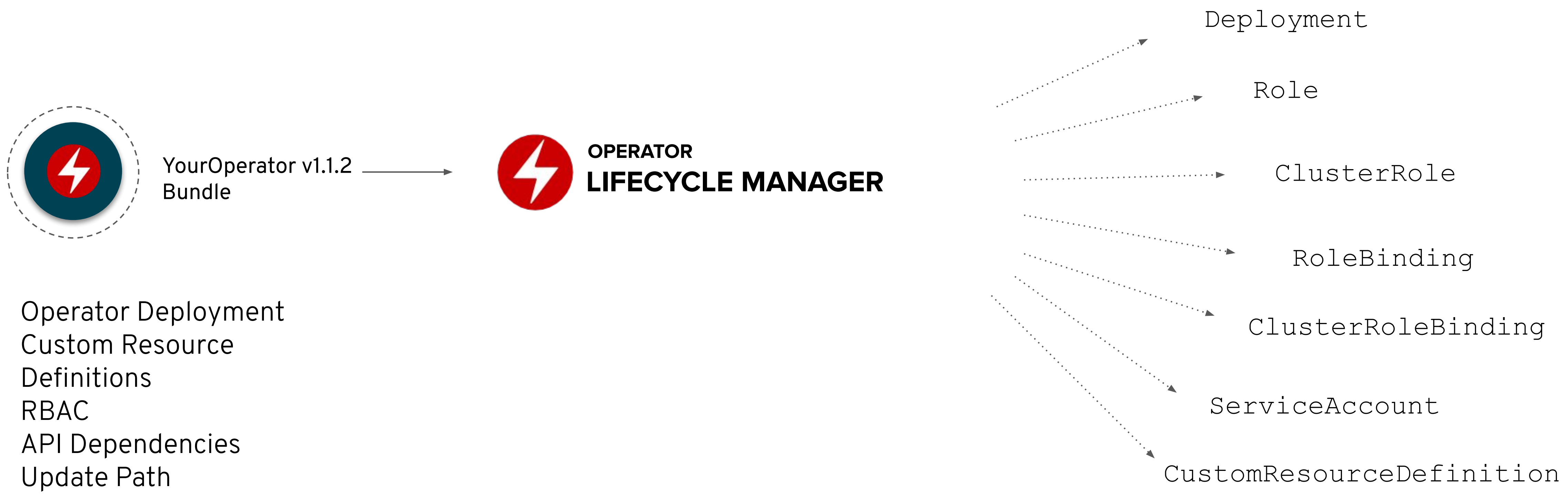
How to ship an Operator?



Package Discovery:

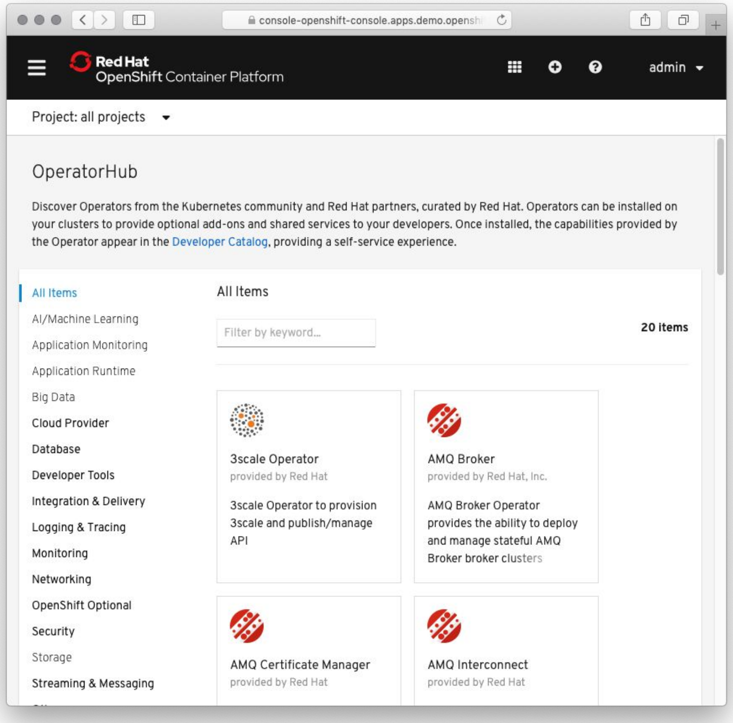
```
$ oc get packagemanifests
```

OPERATOR LIFECYCLE MANAGER



Operator Deployment
Custom Resource Definitions
RBAC
API Dependencies
Update Path
Metadata

OperatorHub



console-openshift-console.apps.demo.opensh

Red Hat OpenShift Container Platform — admin

Project: all projects

OperatorHub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. Operators can be installed on your clusters to provide optional add-ons and shared services to your developers. Once installed, the capabilities provided by the Operator appear in the Developer Catalog, providing a self-service experience.

All Items
AI/Machine Learning
Application Monitoring
Application Runtime
Big Data
Cloud Provider
Database
Developer Tools
Integration & Delivery
Logging & Tracing
Monitoring
Networking
OpenShift Optional
Security
Storage
Streaming & Messaging

All Items

Filter by keyword...

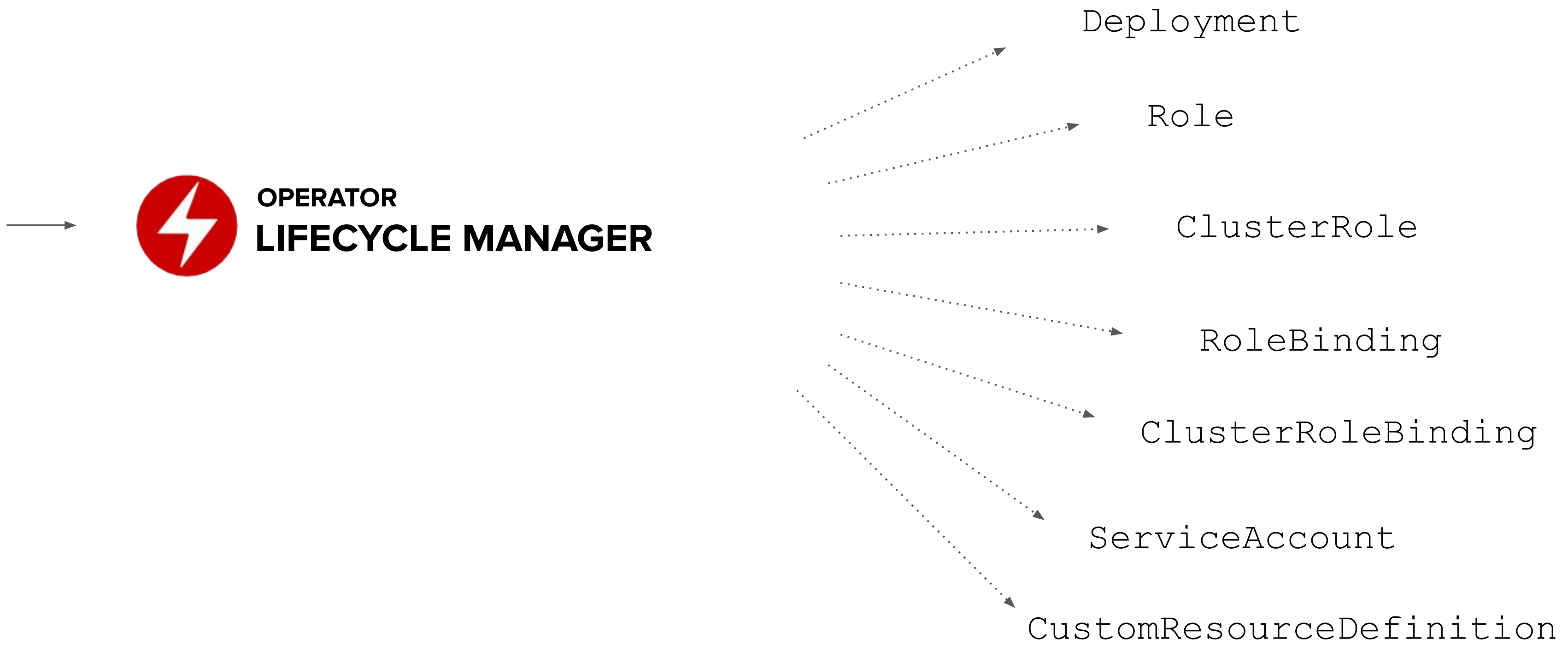
20 items

3scale Operator
provided by Red Hat
3scale Operator to provision 3scale and publish/manage API

AMQ Broker
provided by Red Hat, Inc.
AMQ Broker Operator provides the ability to deploy and manage stateful AMQ Broker broker clusters

AMQ Certificate Manager
provided by Red Hat

AMQ Interconnect
provided by Red Hat



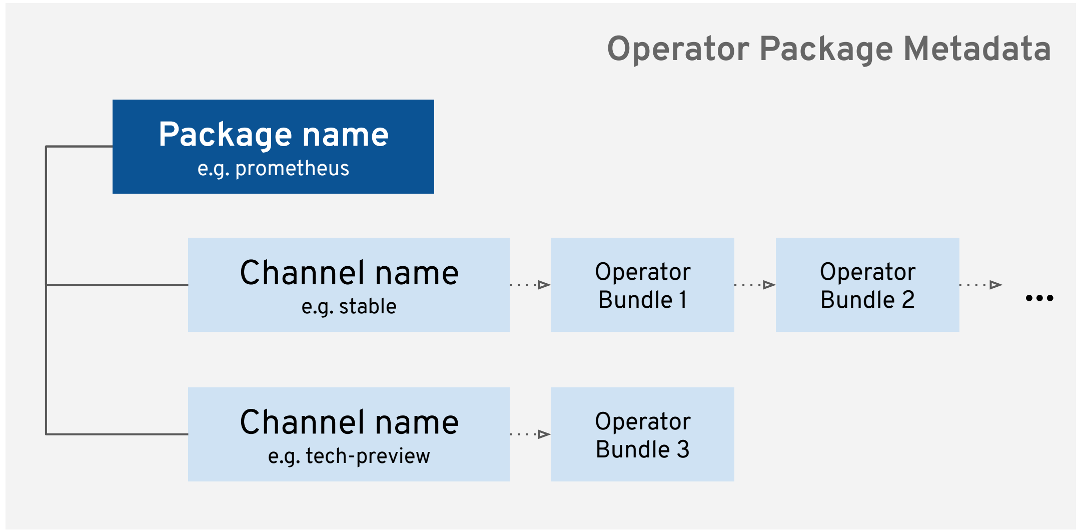
OperatorHub data sources

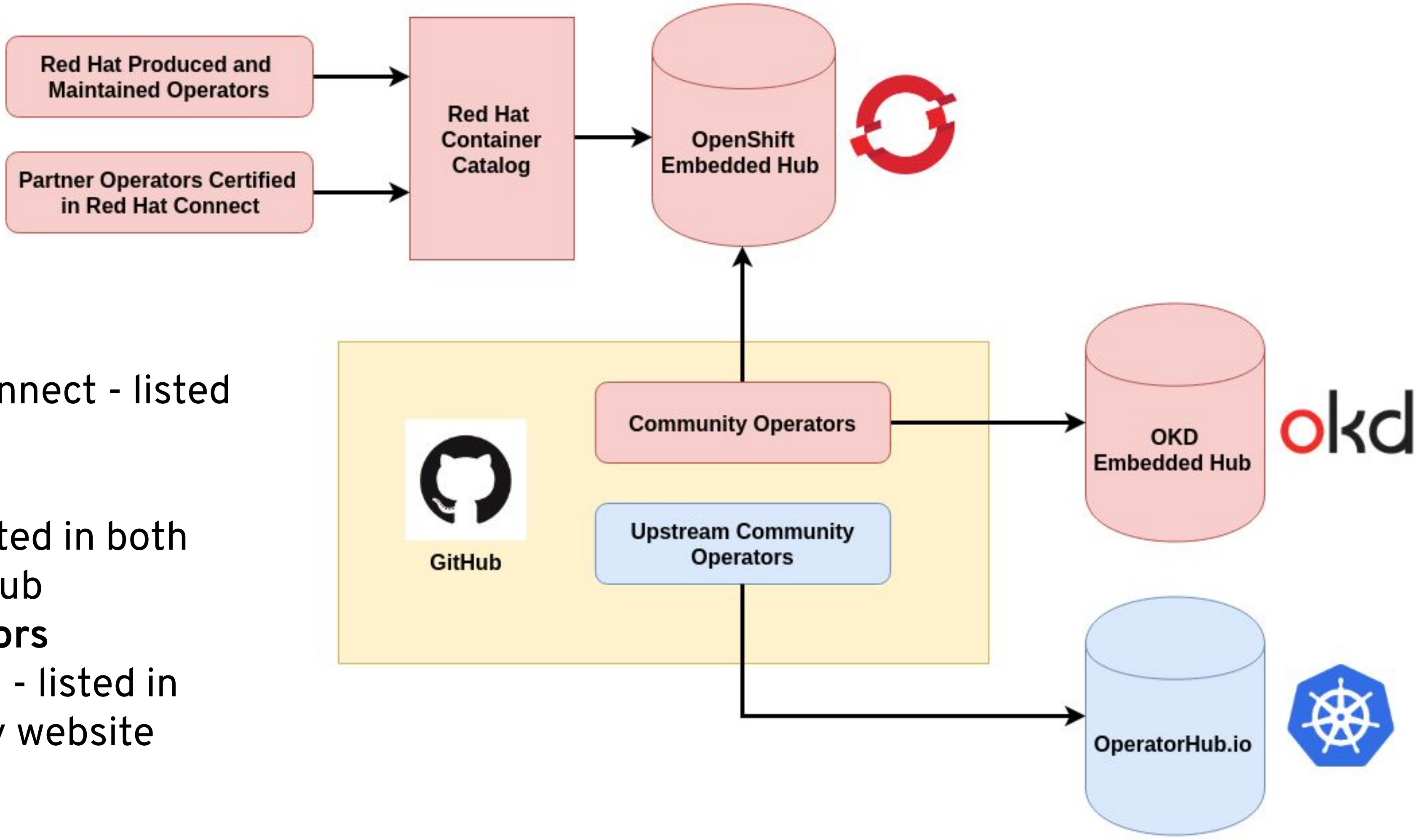
Operator Metadata from quay.io

- Backend for all default sources, cluster needs to be online
- Supplies Red Hat Operators, ISV Operators and Community Operator
- Custom sources supported in customer-owned quay.io namespaces

Operator Metadata in container images

- Already used internally used by OLM
- Operator package data is served from a SQlite database, bundled up in a container image
- Custom sources supported in customer-owned image registries
- Cluster can be disconnected / air-gapped





Red Hat Produced and Maintained Operators
Partner Operators Certified in Red Hat Connect
Red Hat Container Catalog
OpenShift Embedded Hub
GitHub
Community Operators
Upstream Community Operators
OKD Embedded Hub
okd
OperatorHub.io

- **Certified Operators**
  submitted through Red Hat Connect - listed in OpenShift OperatorHub
- **Community Operators**
  submitted through GitHub - listed in both OpenShift *and* OKD OperatorHub
- **Upstream Community Operators**
  also submitted through GitHub - listed in the OperatorHub.io community website

How Operator Catalogs are downloaded

Operator Metadata from quay.io

- Backend for all default sources, cluster needs to be online
- Supplies Red Hat Operators, ISV Operators and Community Operator
- Custom sources supported in customer-owned quay.io namespaces, data uploaded via operator-courier

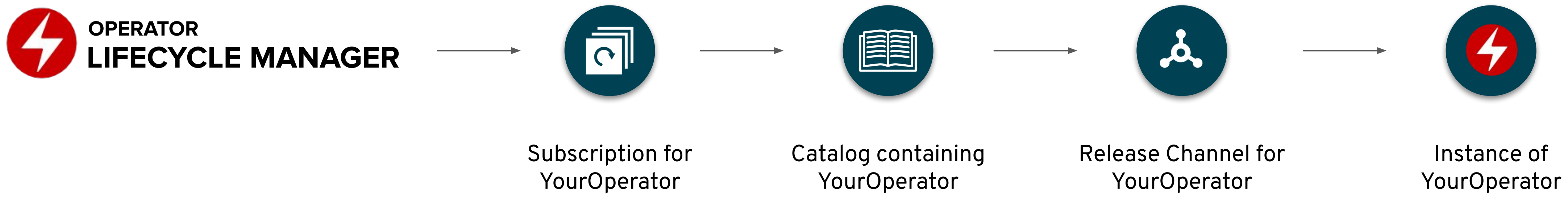
```
apiVersion: operators.coreos.com/v1
kind: OperatorSource
metadata:
  name: johndoe-operators
  namespace: marketplace
spec:
  type: appregistry
  endpoint: https://quay.io/cnr
  registryNamespace: johndoe
```

Operator Metadata in container images

- Already used internally used by OLM
- Operator package data is served from a SQlite database, bundled up in a container image (created via operator-registry)
- Custom sources supported in customer-owner image registries
- Cluster can be disconnected / air-gapped

```
apiVersion: operators.coreos.com/v1alpha1
kind: CatalogSource
metadata:
  name: johndoe-operators
  namespace: olm
spec:
  sourceType: grpc
  image: johndoe-catalog:latest
```

Operator Framework in Action

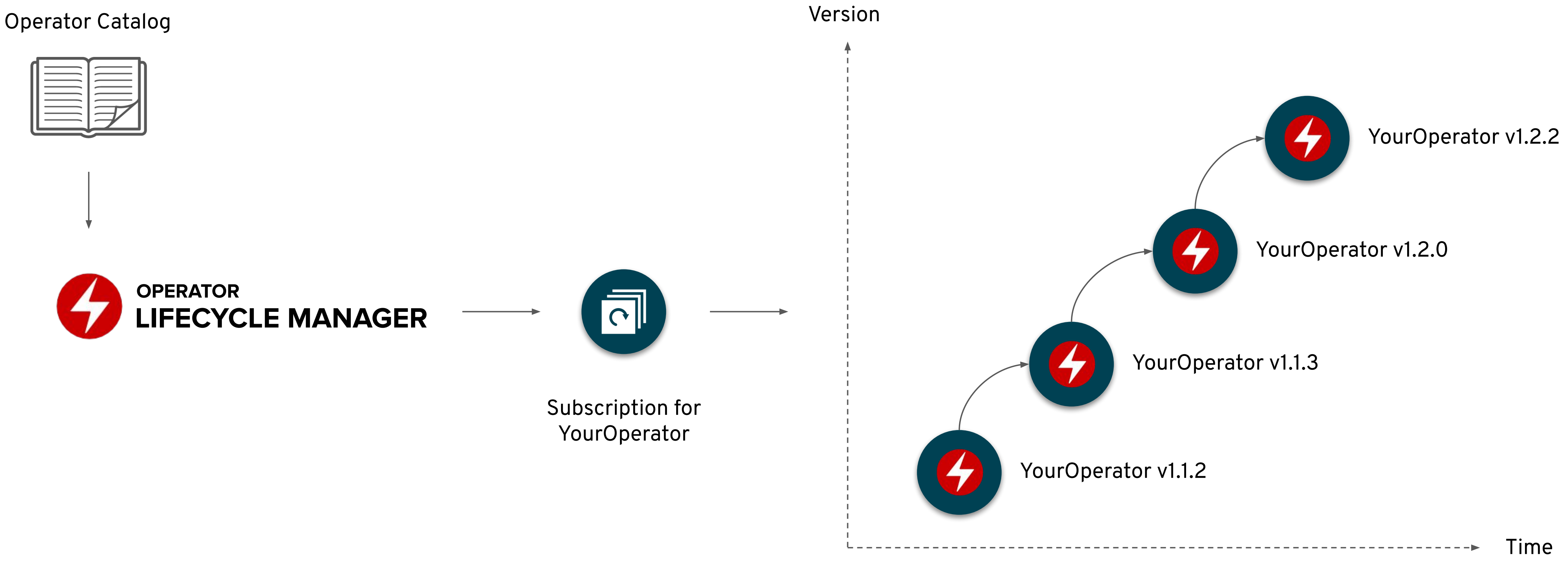

OPERATOR
LIFECYCLE MANAGER
Subscription for YourOperator
Catalog containing YourOperator
Release Channel for YourOperator
Instance of YourOperator

To install an Operator an administrator…

1. Picks an Operator from the Catalog
2. (Selects a distribution channel from the Operators package)
3. (Selects a version of the Operators from the channel)
4. Creates a Subscription pointing to the Catalog, Operator, Version and Channel
    a. If no channel is specified, the default channel is used
    b. If no version is specified, the latest is used

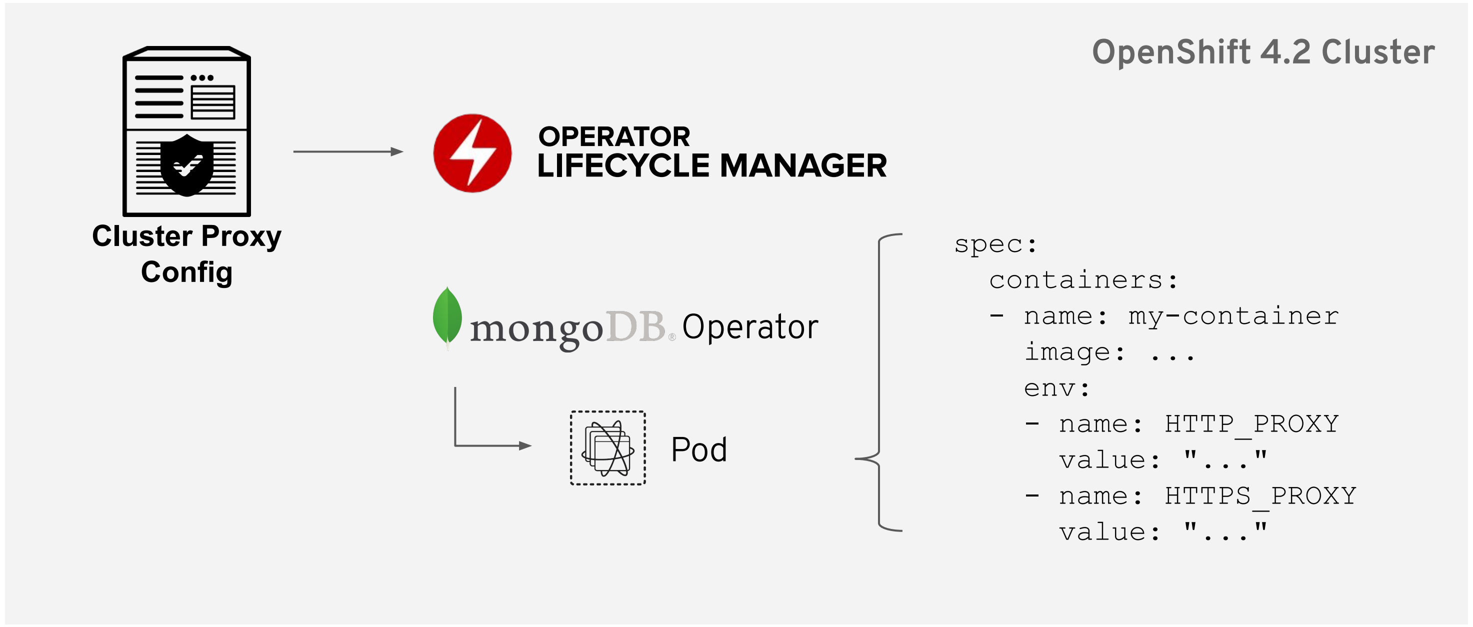
```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: johns-subsription
  namespace: default
spec:
  channel: stable
  name: my-operator
  source: johndoe-operators
  sourceNamespace: olm
```

Operator Updates



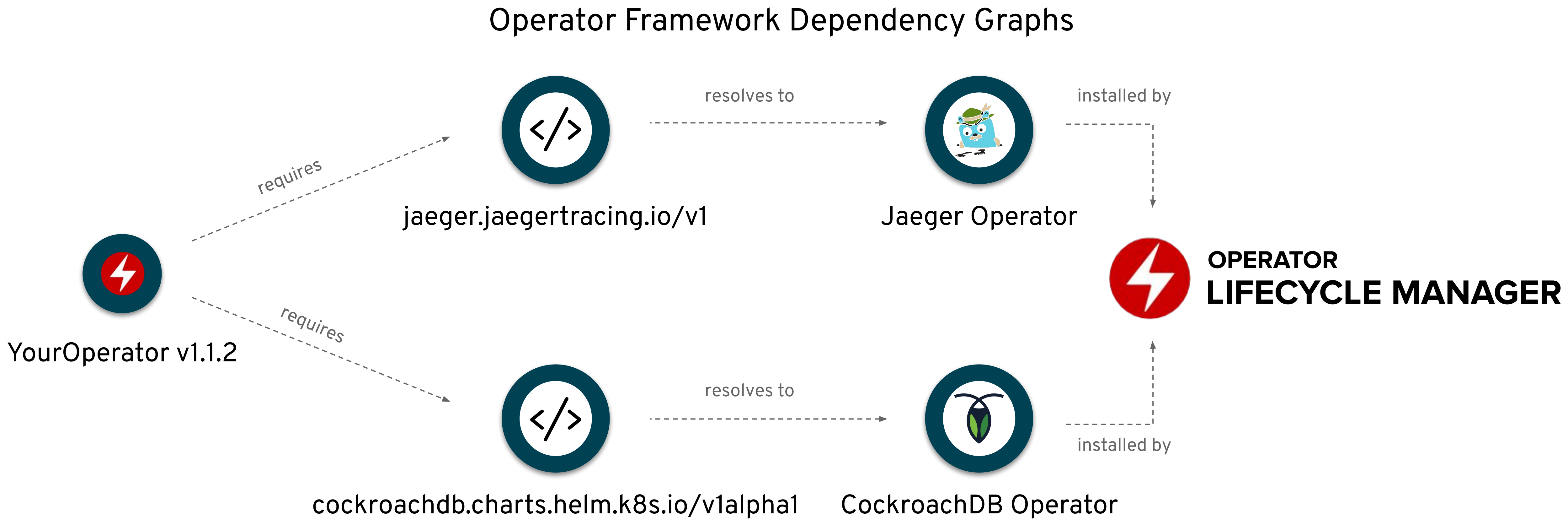
Red Hat

Proxy Support



```
spec:
  containers:
  - name: my-container
    image: ...
    env:
    - name: HTTP_PROXY
      value: "..."
    - name: HTTPS_PROXY
      value: "..."
```

4.2 Automated Dependency Resolution



Red Hat

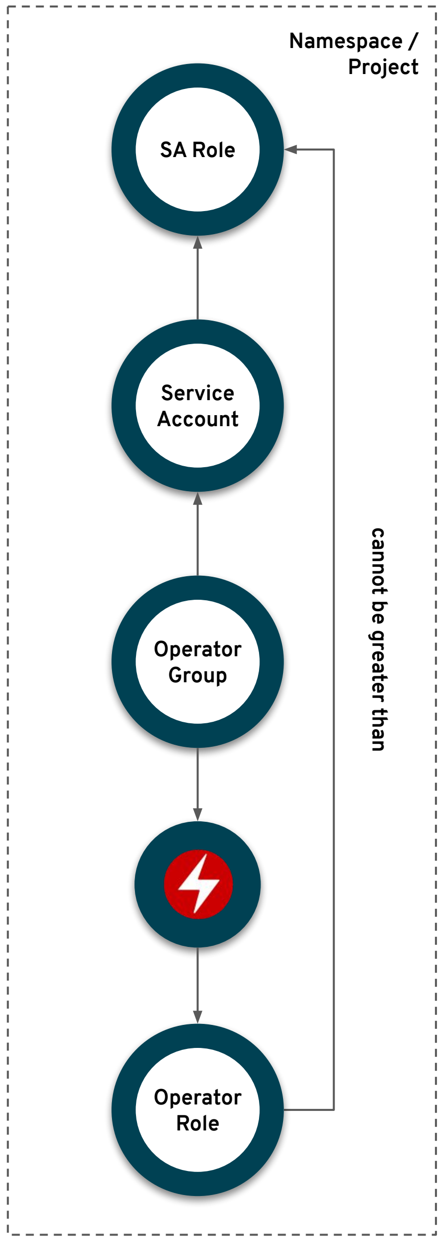
Allow regular users to install Operators

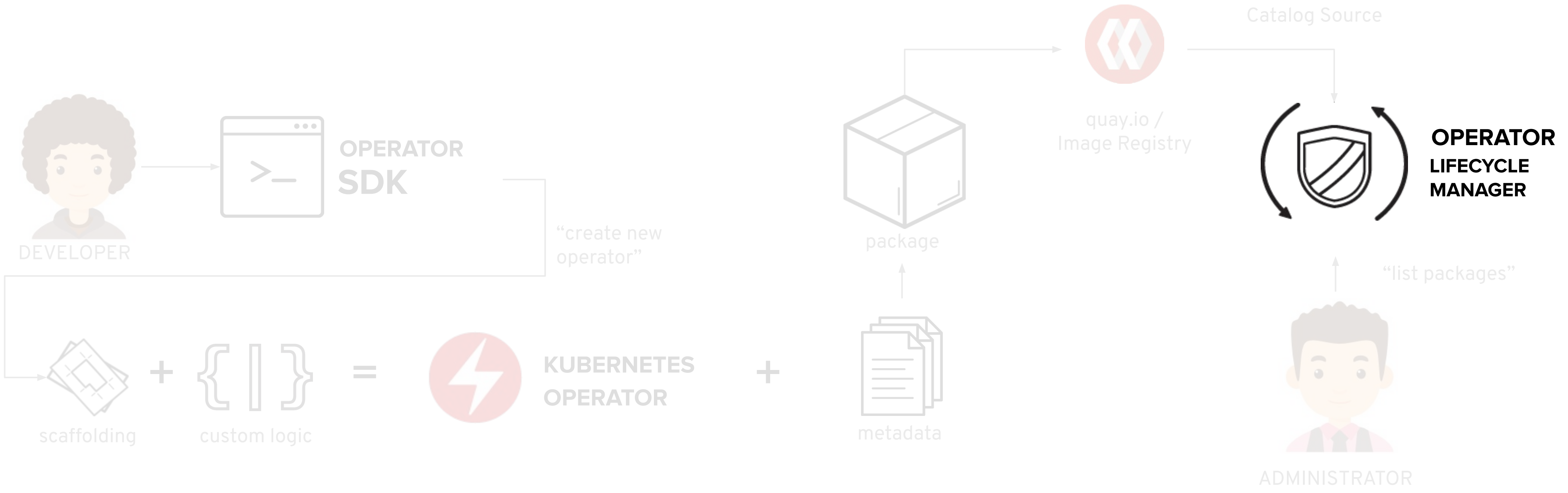
- **In 4.1:** only users carrying `cluster-admin` roles are allowed to install Operators

- **In 4.2:** administrators can delegate install to users

  - `cluster-admin` select namespaces in which namespace admins can install operators self-sufficiently
  - `cluster-admin` defines `ServiceAccount` in this namespace
  - all installed Operators in this namespace get equal or lower permissions of this `ServiceAccount`

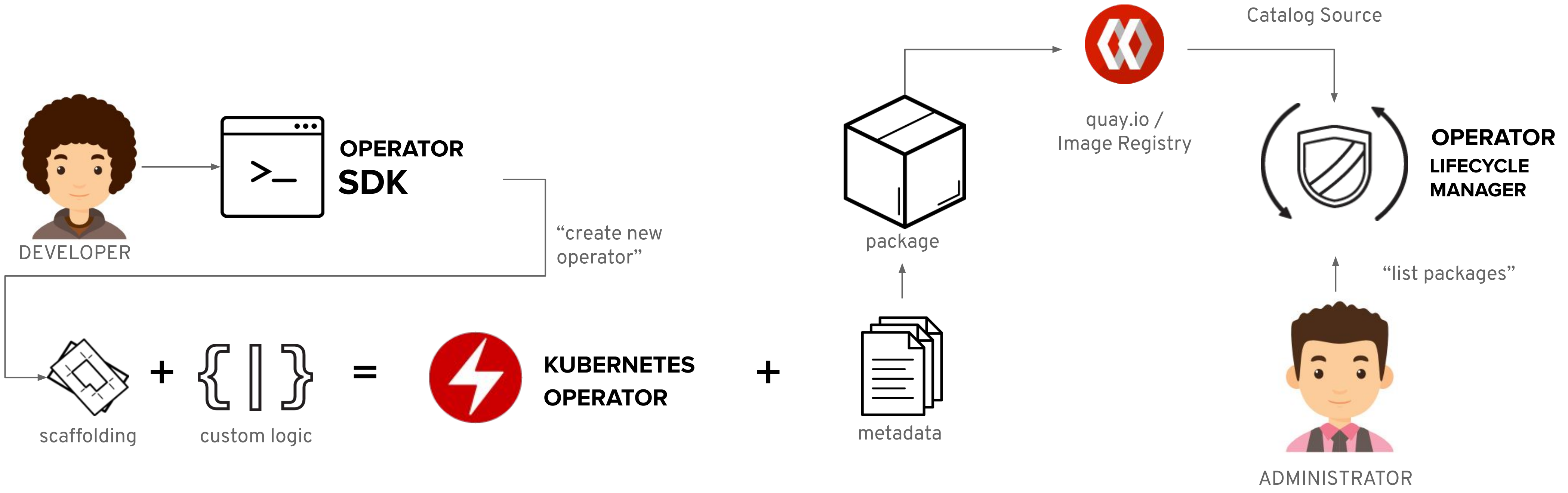
    - RBAC is typically limited to this namespace





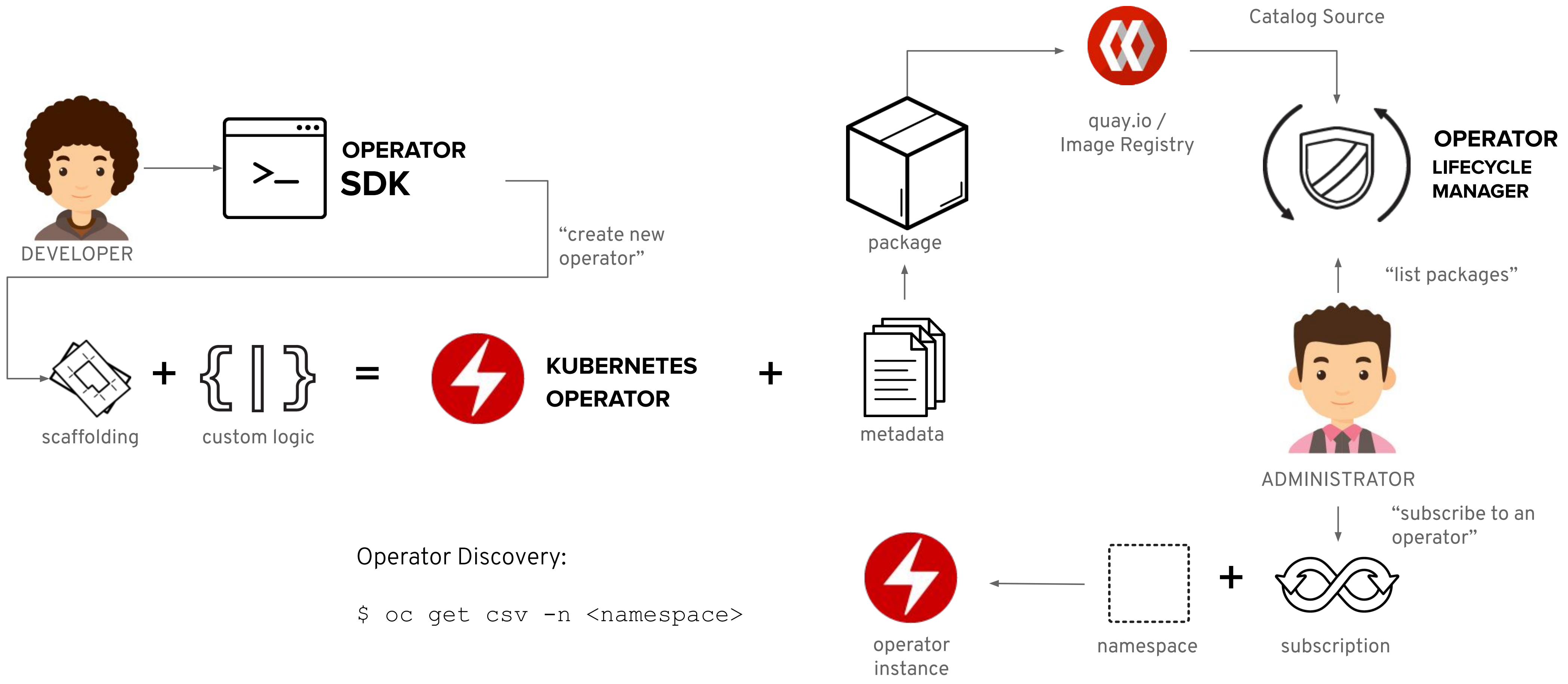
Package Discovery:

```
$ oc get packagemanifests
```



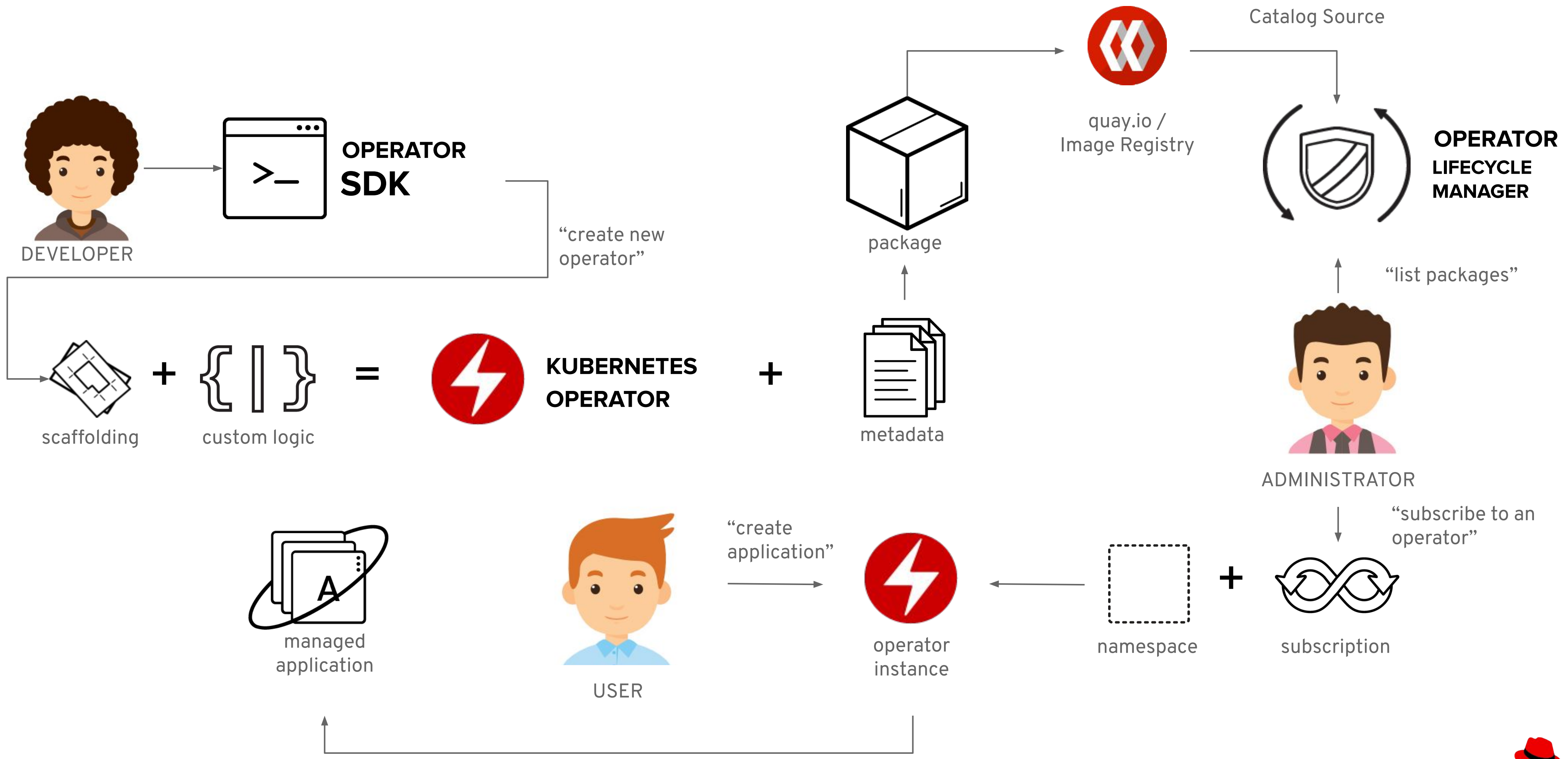
Package Discovery:

```
$ oc get packagemanifests
```



Operator Discovery:

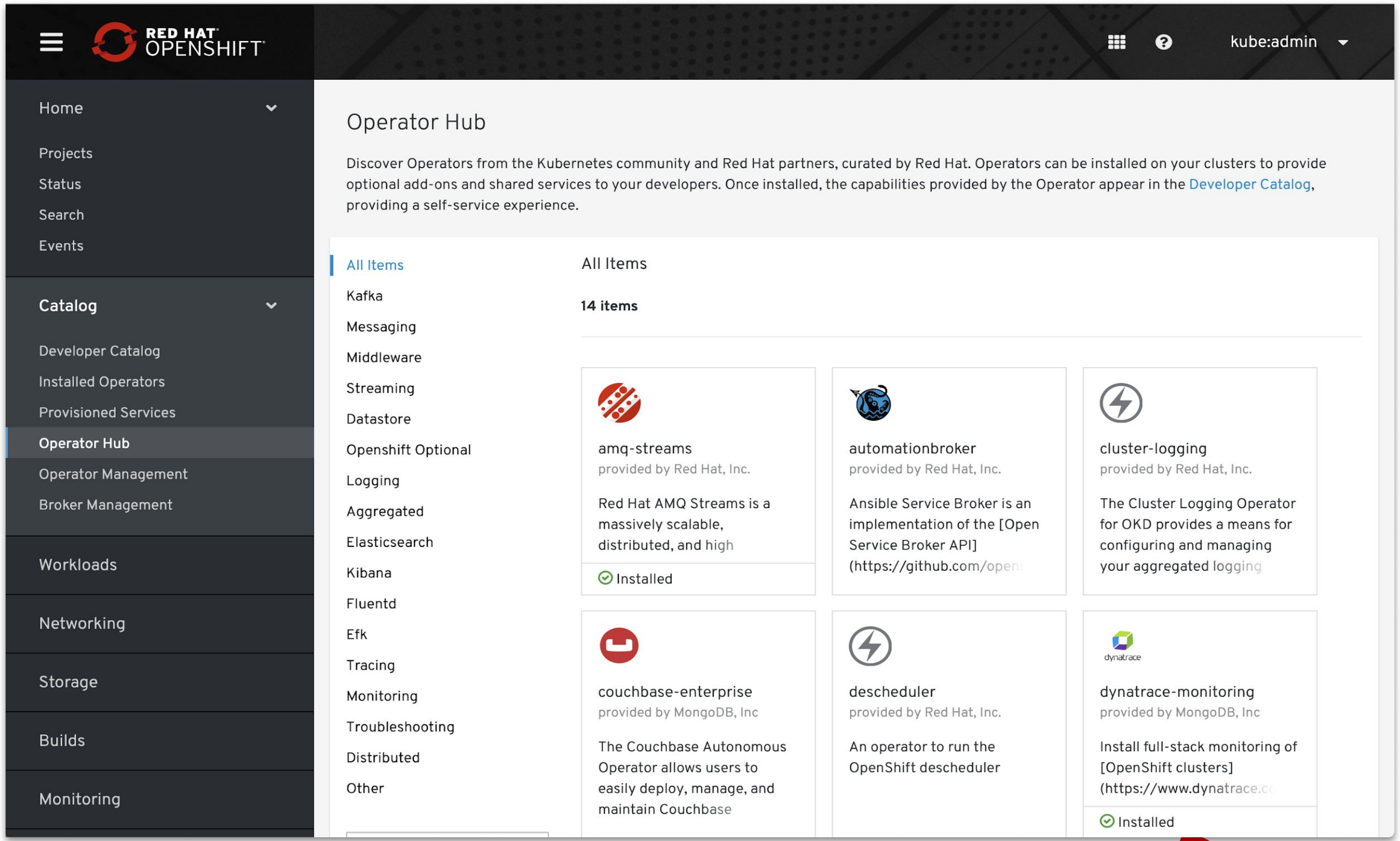
```
$ oc get csv -n <namespace>
```


DEVELOPER
OPERATOR SDK
"create new operator"
scaffolding
+
custom logic
=
KUBERNETES OPERATOR
+
metadata
package
quay.io / Image Registry
Catalog Source
OPERATOR LIFECYCLE MANAGER
"list packages"
ADMINISTRATOR
"subscribe to an operator"
subscription
+
namespace
operator instance
"create application"
USER
managed application

Red Hat

Operators in OpenShift

Operator Hub - Allows administrators to selectively make operators available from curated sources to users in the cluster.



Operator Hub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. Operators can be installed on your clusters to provide optional add-ons and shared services to your developers. Once installed, the capabilities provided by the Operator appear in the Developer Catalog, providing a self-service experience.

All Items

14 items

amq-streams
provided by Red Hat, Inc.
Red Hat AMQ Streams is a massively scalable, distributed, and high
Installed

automationbroker
provided by Red Hat, Inc.
Ansible Service Broker is an implementation of the [Open Service Broker API] (https://github.com/oper

cluster-logging
provided by Red Hat, Inc.
The Cluster Logging Operator for OKD provides a means for configuring and managing your aggregated logging

couchbase-enterprise
provided by MongoDB, Inc
The Couchbase Autonomous Operator allows users to easily deploy, manage, and maintain Couchbase

descheduler
provided by Red Hat, Inc.
An operator to run the OpenShift descheduler

dynatrace-monitoring
provided by MongoDB, Inc
Install full-stack monitoring of [OpenShift clusters] (https://www.dynatrace.c
Installed

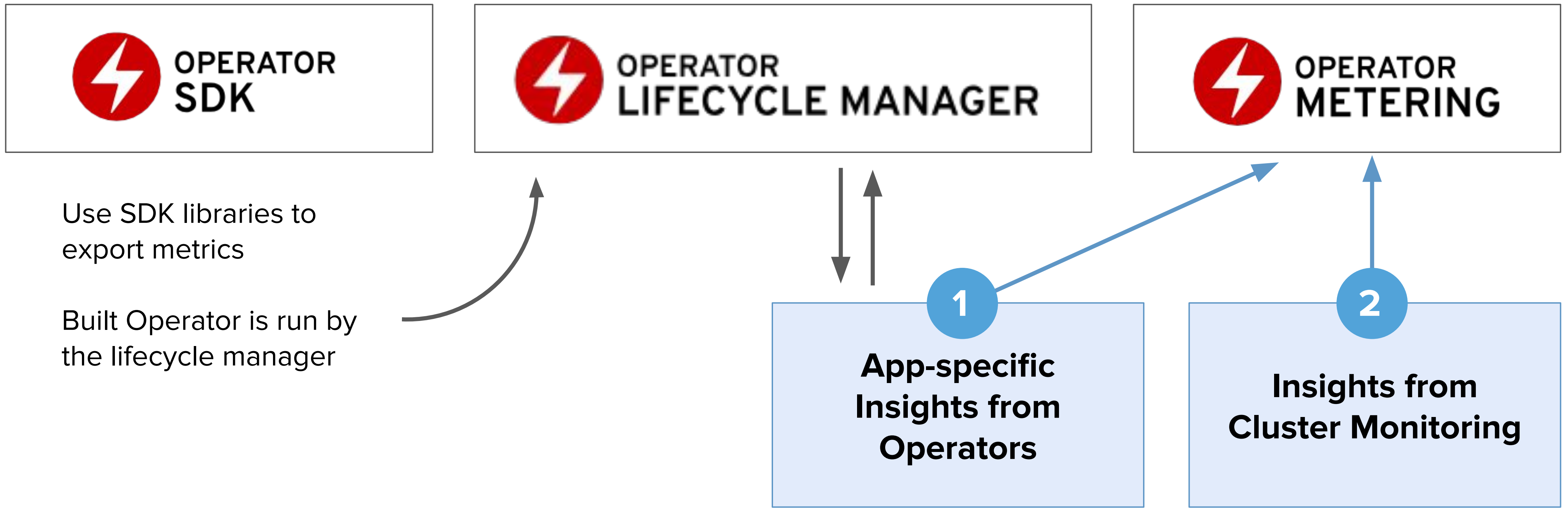
OPERATOR FRAMEWORK

- **Operator SDK** - Allows developers to build, package and test an Operator based on your expertise without requiring all the knowledge of Kubernetes API complexities

- **Operator Lifecycle Manager** - Helps you to install, and update, and generally manage the lifecycle of all of the Operators (and their associated services) running across your clusters

- **Operator Metering** - Enable usage reporting for Operators and resources within Kubernetes

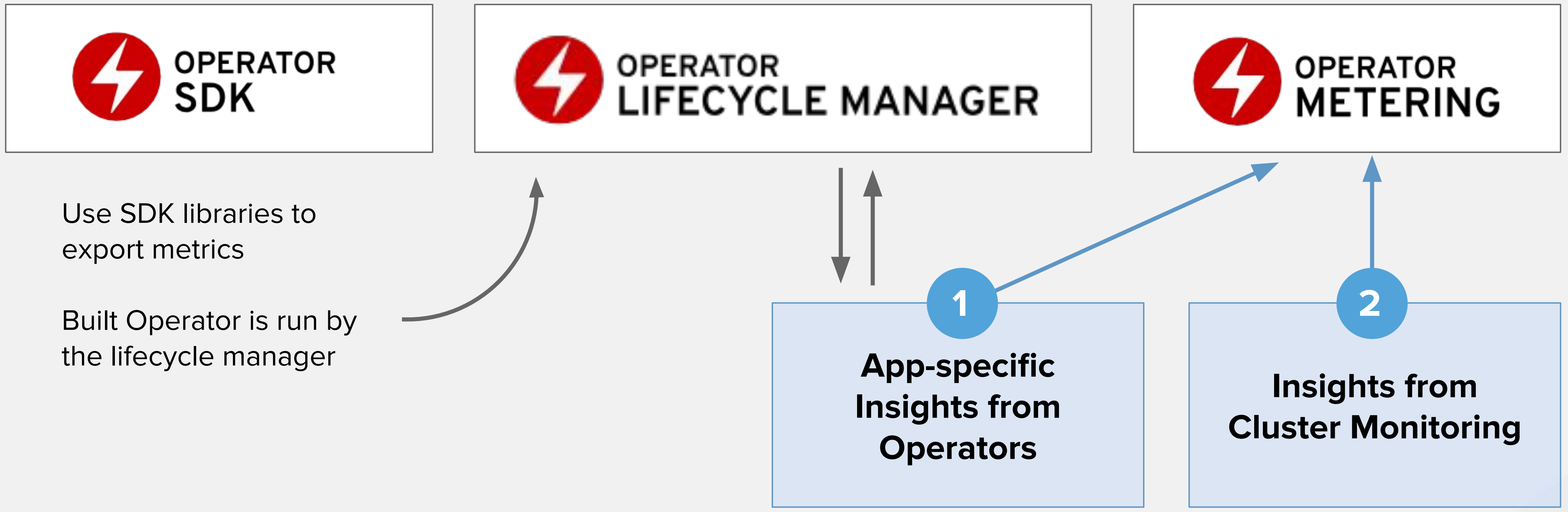
OPERATOR METERING


OPERATOR SDK
OPERATOR LIFECYCLE MANAGER
OPERATOR METERING
1
App-specific Insights from Operators
2
Insights from Cluster Monitoring

Use SDK libraries to export metrics

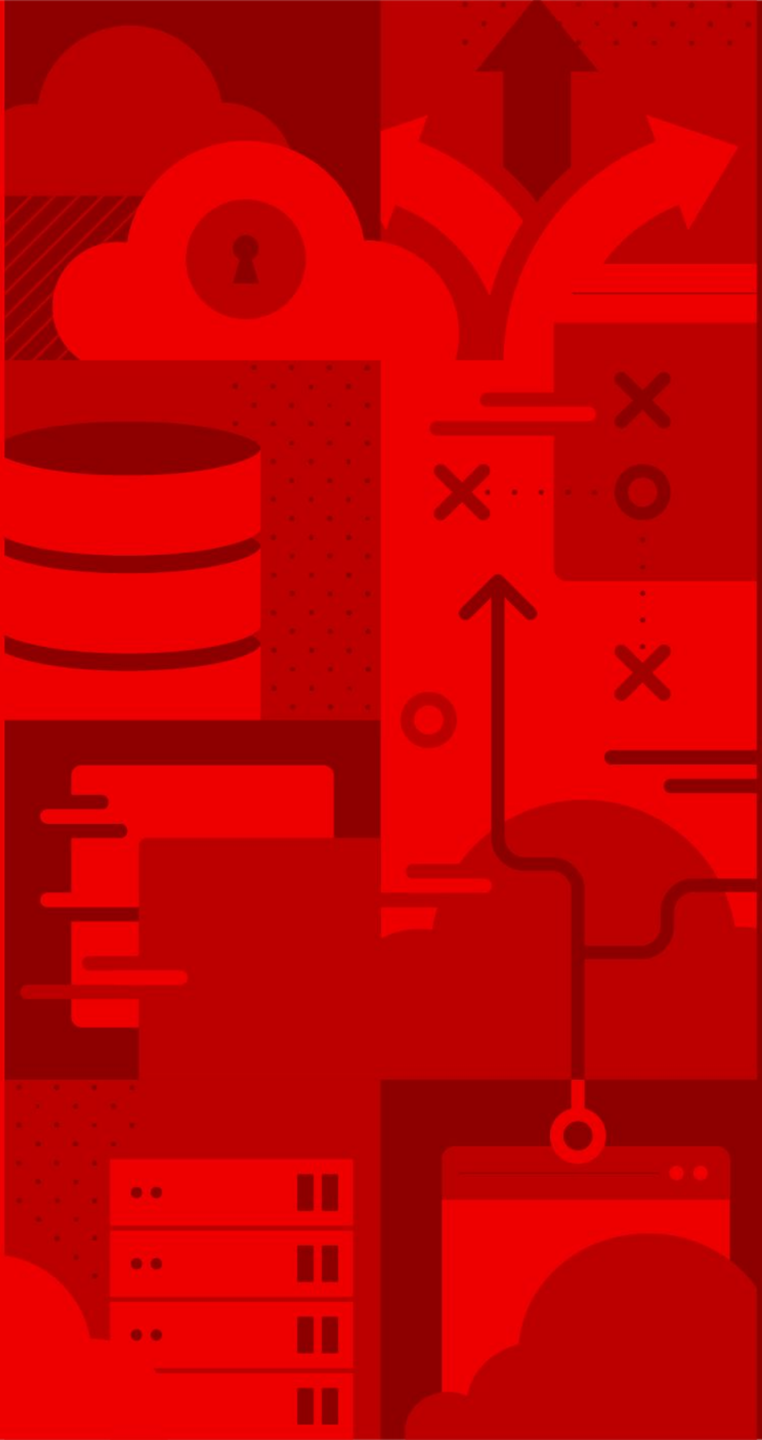
Built Operator is run by the lifecycle manager

Metering Goals


OPERATOR SDK
OPERATOR LIFECYCLE MANAGER
OPERATOR METERING
1
App-specific Insights from Operators
2
Insights from Cluster Monitoring

Use SDK libraries to export metrics

Built Operator is run by the lifecycle manager



Roadmap for Operator Framework

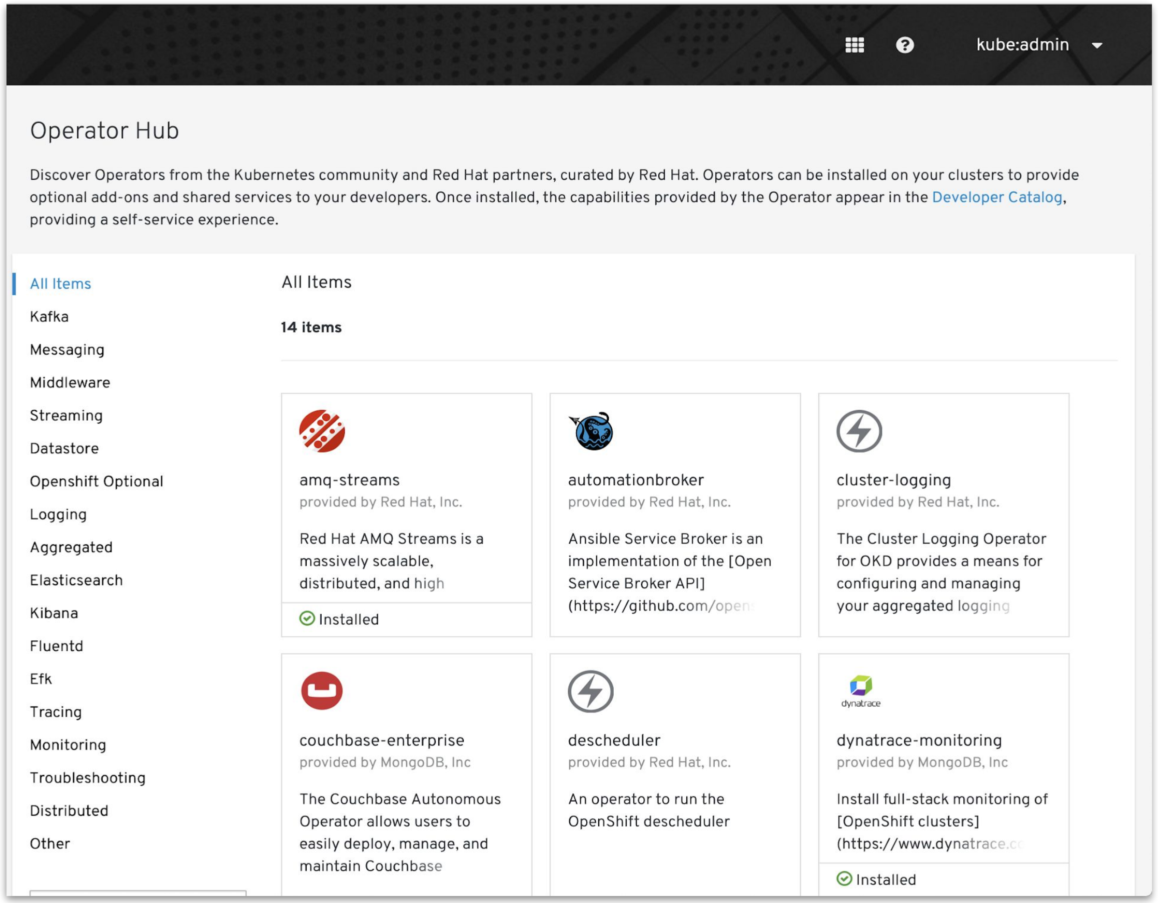
Custom Operators Catalogs

Operator Metadata from quay.io

- Backend for all default sources, cluster needs to be online
- Supplies Red Hat Operators, ISV Operators and Community Operator
- Custom sources supported in customer-owned quay.io namespaces

Operator Metadata in container images

- Already used internally used by OLM
- Operator package data is served from a SQlite database, bundled up in a container image
- Custom sources supported in customer-owner image registries
- Cluster can be disconnected / air-gapped



Operator Hub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. Operators can be installed on your clusters to provide optional add-ons and shared services to your developers. Once installed, the capabilities provided by the Operator appear in the Developer Catalog, providing a self-service experience.

All Items

14 items

amq-streams
provided by Red Hat, Inc.
Red Hat AMQ Streams is a massively scalable, distributed, and high
Installed

automationbroker
provided by Red Hat, Inc.
Ansible Service Broker is an implementation of the [Open Service Broker API] (https://github.com/open

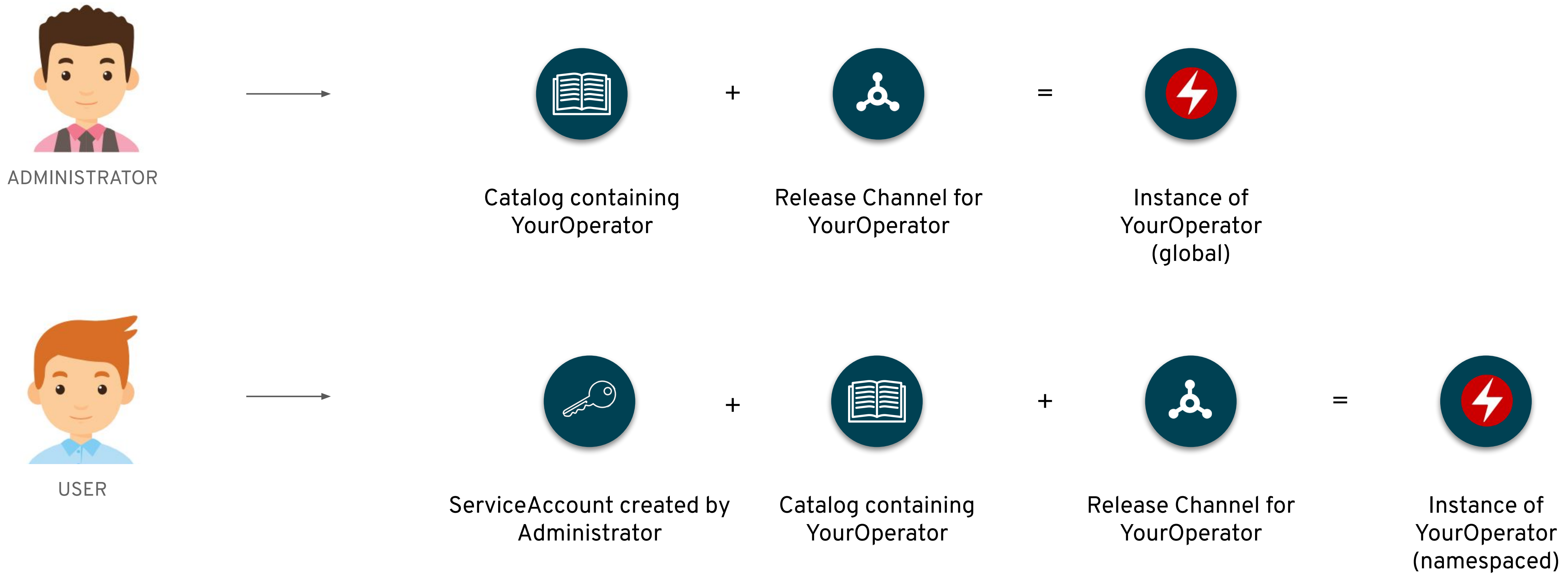
cluster-logging
provided by Red Hat, Inc.
The Cluster Logging Operator for OKD provides a means for configuring and managing your aggregated logging

couchbase-enterprise
provided by MongoDB, Inc
The Couchbase Autonomous Operator allows users to easily deploy, manage, and maintain Couchbase

descheduler
provided by Red Hat, Inc.
An operator to run the OpenShift descheduler

dynatrace-monitoring
provided by MongoDB, Inc
Install full-stack monitoring of [OpenShift clusters] (https://www.dynatrace.c
Installed

Regular users can install Operators


ADMINISTRATOR
Catalog containing YourOperator
+
Release Channel for YourOperator
=
Instance of YourOperator (global)
USER
ServiceAccount created by Administrator
+
Catalog containing YourOperator
+
Release Channel for YourOperator
=
Instance of YourOperator (namespaced)

Operator Framework Roadmap

Q4 CY19

OLM

Automatic Dependency Resolution

Delegation of Operator installation to regular users

Proxy / Disconnected Support

Custom Operator Catalogs

SDK

Helm v2.14 support

Kubernetes 1.14 support

Prometheus metrics

UBI as base-image

1H CY20

OLM

Single Object to represent an Operator

Container Images as Operator Bundles

Container Registries as Operator Catalogs

SDK

Kubebuilder support

OLM integration (“install” / “run”)

2H CY20

OLM

Investigation support for helm and OCI as bundle formats

SDK

Investigation of Java and Python support

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat