# DEVELOPER DAY 2019

WEIQIANG ZHUANG

# OPERATOR FRAMEWORK IN PRACTICE

## Agenda

- Operators
- Operator Framework
  - operator sdk
  - operator lifecycle manager
- Create, Install an Operator and Deploy an application
  - Flow
  - Step-by-step example (Spark Cluster)
- Links

# Operators

- *Operators* are a method of packaging, deploying, and managing a Kubernetes application.
- Easy to manage complex stateful applications.
- An operator mainly consists of Kubernetes CustomResourceDefinitions (CRDs) and Controller logic.
  - either *Namespaced* or *Cluster* scoped (ie.single instance across the cluster)
  - eg. /apis/os.ibm.com/v1alpha1/namespaces/default/SparkApplication
- Good for Day 2 Operations
  - patch, update, scale up/down, upgrade and more
- But creating an operator is not easy until ...

# Operator Framework

- Operator Framework is an open source toolkit to manage Kubernetes native applications, called Operators, in an effective, automated, and scalable way.
  - operator-sdk – write, test and package operators
  - operator-courier – build, verify and push operator manifests (CRDs and CSVs)
  - operator-registry – store the manifest data in database and provide operator catalog data to Operator Lifecycle Manager
  - operator-lifecycle-manager – install, upgrade and RBAC control operators (operator of operators)
  - operator-metering – collect operational metrics of operators for day 2 management
  - operator-marketplace – an operator to register off-cluster operators
  - community-operators – host community created operators and publish to operatorhub.io

# Operator SDK

- SDK provides
  - High level APIs and abstractions to write the operational logic more intuitively
  - Tools for scaffolding and code generation to bootstrap a new project fast
  - Extensions to cover common operator use cases
- Three types of workflows to develop operators: Go, Helm, and Ansible

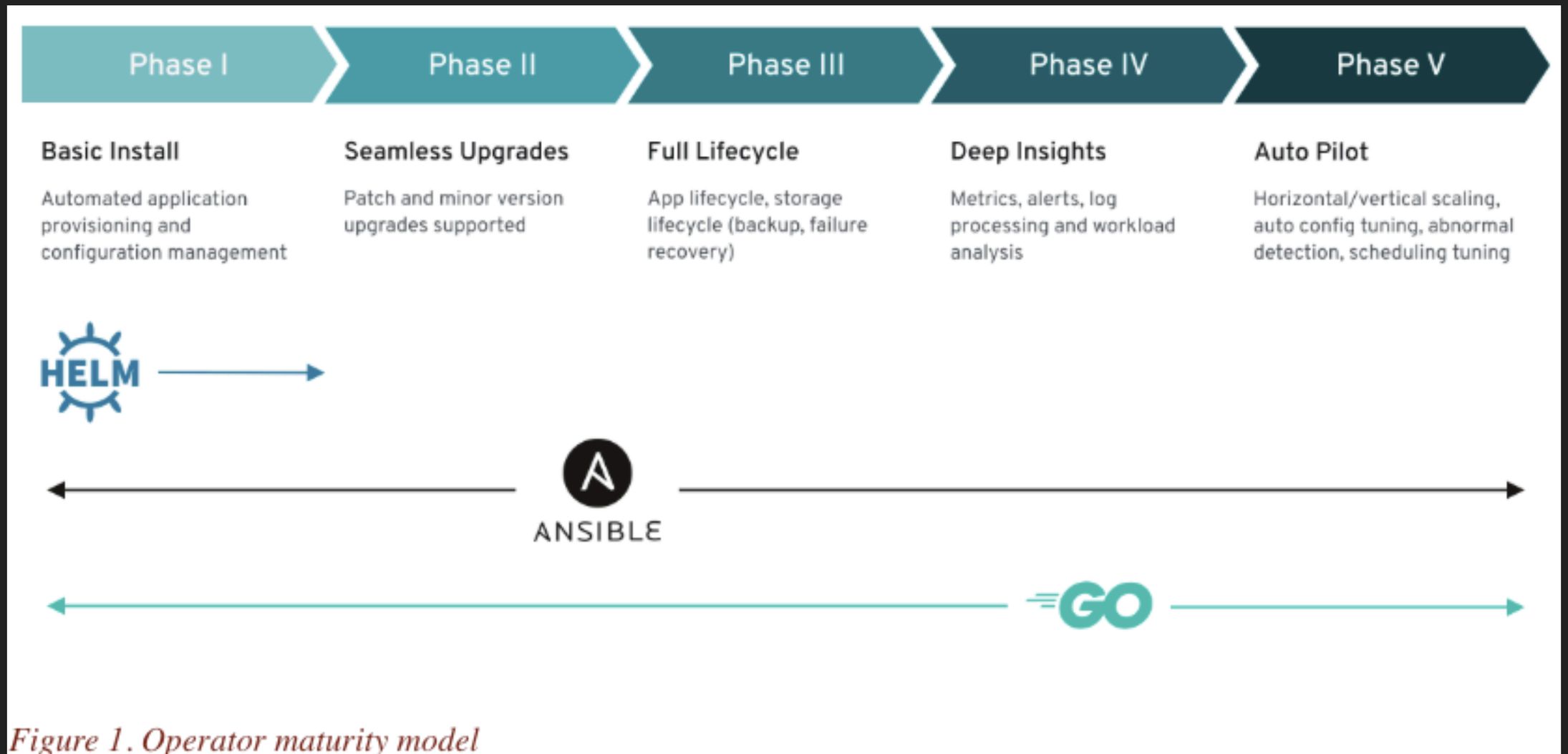



*Figure 1. Operator maturity model*

# Operator SDK

- Operator SDK CLI – operator-sdk

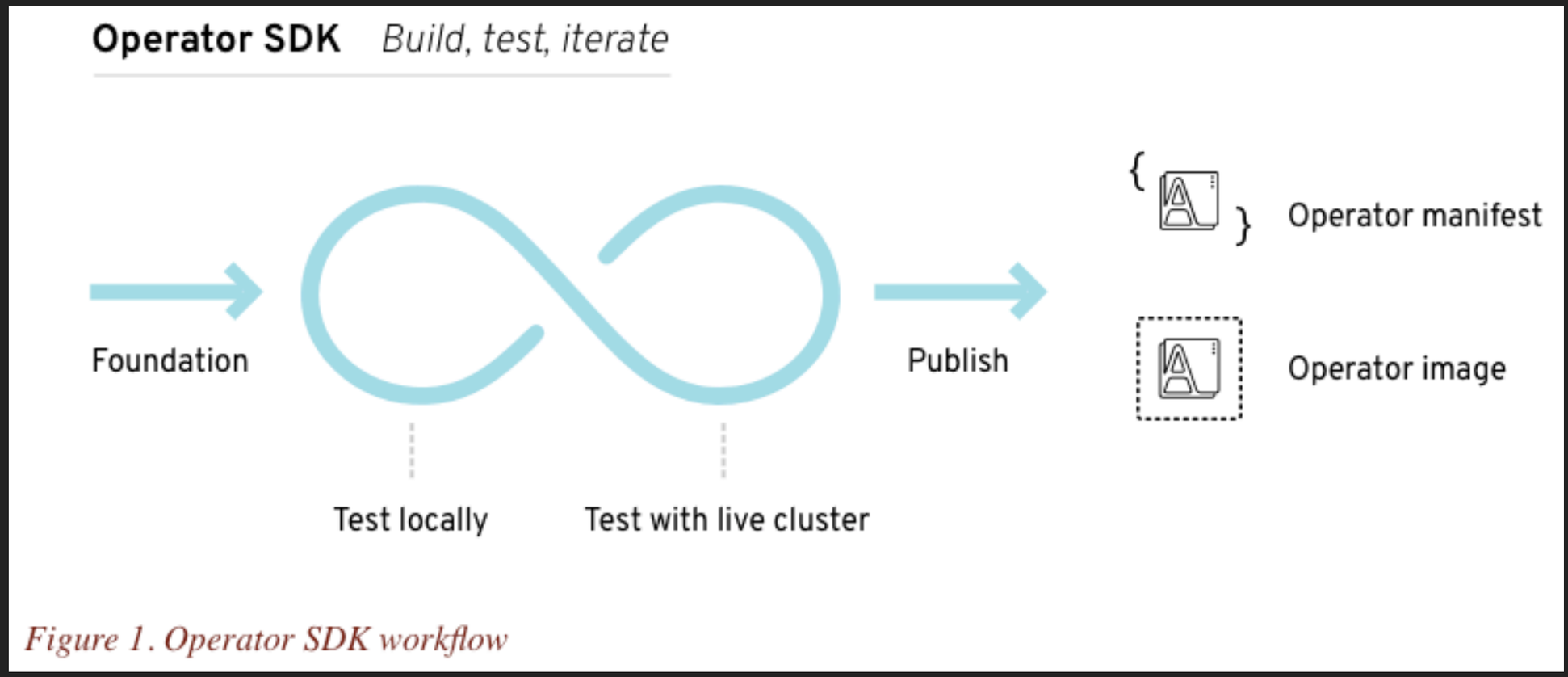
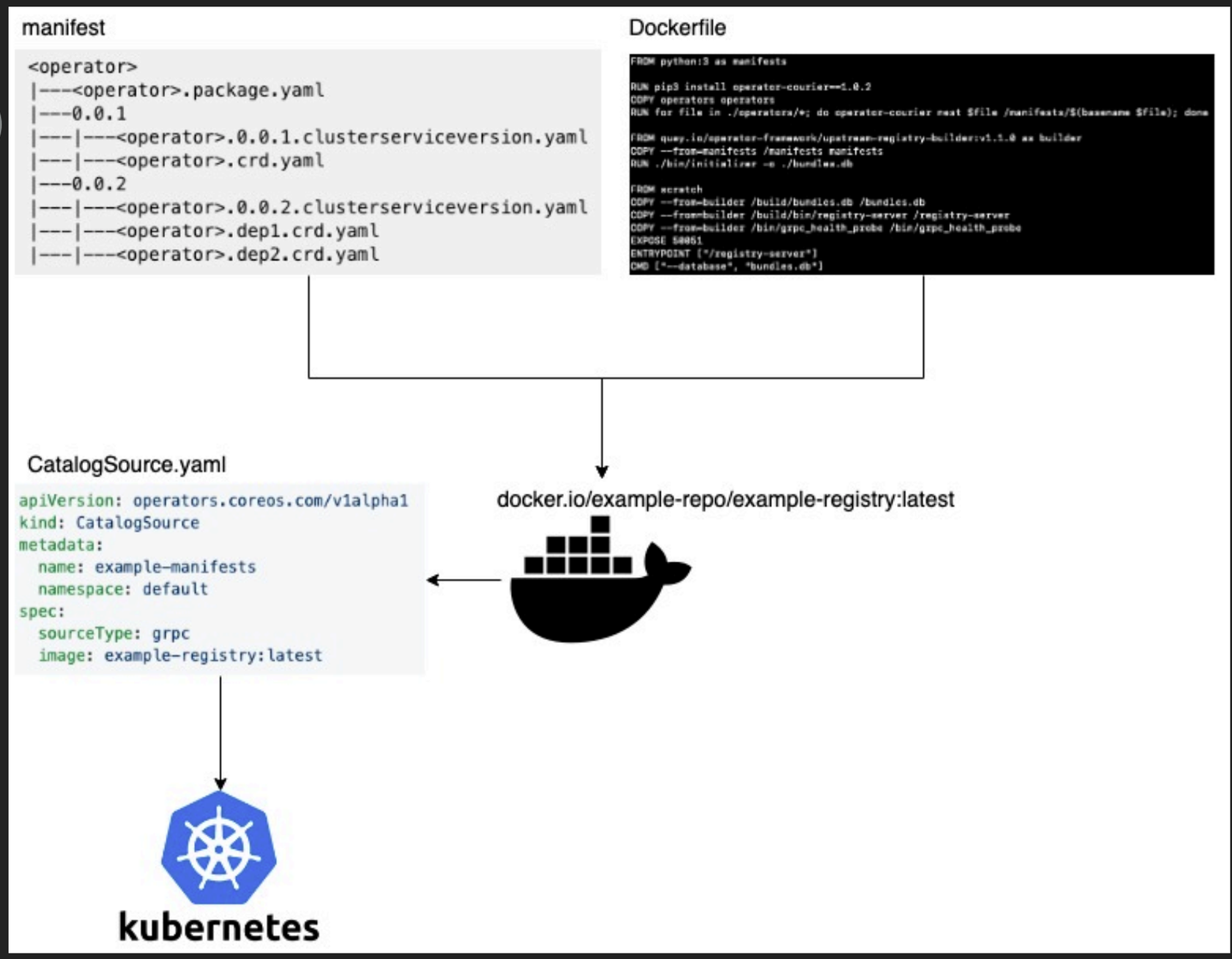



*Figure 1. Operator SDK workflow*

- Operator scope
  - namespace scoped: watches and manages resources in a single namespace
  - cluster scoped: watches and manages cluster-wide

# Operator Lifecycle Manager (OLM)

- Install, update, and manage the lifecycle of all operators and their associated services running across their clusters.
- ClusterServiceVersion (CSV)
  - metadata + install strategy + CRDs
- Catalog Registry
  - Stores CSVs, CRDs, metadata of packages and channels



# Operator Lifecycle Manager (OLM)

- Two operators
  - OLM operator – operator of operators
    - ClusterServiceVersion, ClusterResourceDefinition, OperatorGroup
    - Deployments, (Cluster)Role, (Cluster)RoleBinding, ServiceAccount
  - Catalog operator – discover and install CSVs and CRDs
    - InstallPlan, CatalogSource, Subscription
- Metrics
  - csv_count, install_plan_count, subscription_count, csv_upgrade_count

# Create, Install an Operator and Deploy Applications

- Flow

1. Prepare artifacts for the application, including docker images etc.
2. Run operator-sdk command to create an Ansible type operator.
3. Modify the scaffolding code to build the operator:

- Add the tasks to ansible playbook
- Modify or add CRDs and CRs
- Modify the role and rolebinding
- Build docker image for the operator
- Update Deployment to use the docker image

4. Two approaches to install the operator and deploy application.

- Install manually
  - create serviceaccount
  - create role and rolebinding
  - create the operator deployment
  - deploy the application with customresource
- Install through OLM
  - generate clusterserviceversion
  - update csv and verify with operator-courier
  - build image for operator-registry
  - create catalogsource using the registry image
    - if the sourcenamespace has an operatorgroup watching the targetnamespaces, then the operator can be installed to the targetnamespace
    - otherwise, need to create an operatorgroup
  - create the operator deployment with a subscription
  - deploy the application by creating an instance of the provided API

# Create, Install an Operator and Deploy Applications

- Spark Cluster operator example

  ➡ https://github.ibm.com/wzhuang/dday2019/blob/master/Keynote.md

- Example usage in a Pachyderm pipeline

  ➡ https://github.ibm.com/wzhuang/dday2019/blob/master/pachyderm/SPARK-PIPELINE.md

# Links

- this talk: https://github.ibm.com/wzhuang/dday2019
- OpenShift 4.1 documentation:
  - https://docs.openshift.com/container-platform/4.1/applications/operators/olm-what-operators-are.html
  - https://docs.openshift.com/container-platform/4.1/applications/operator_sdk/osdk-getting-started.html
- Operator Framework github: https://github.com/operator-framework/
  - operator-sdk: https://github.com/operator-framework/operator-sdk
  - operator-lifecycle-manager: https://github.com/operator-framework/operator-lifecycle-manager
  - operator-courier: https://github.com/operator-framework/operator-courier
  - operator-registry: https://github.com/operator-framework/operator-registry
- OperatorHub: http://operatorhub.io
  - github: https://github.com/operator-framework/community-operators
- OpenDataHub: http://opendatahub.io/