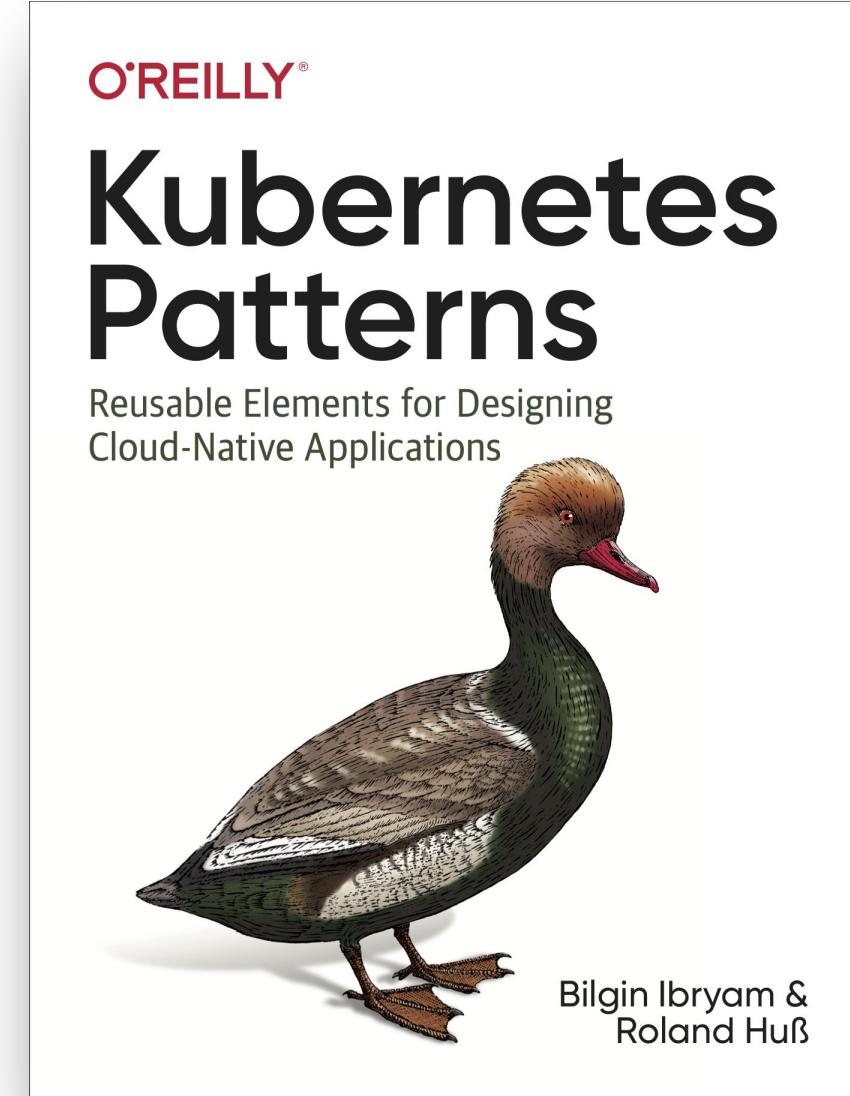


# Kubernetes Patterns

Reusable Elements for Designing Cloud-Native Applications

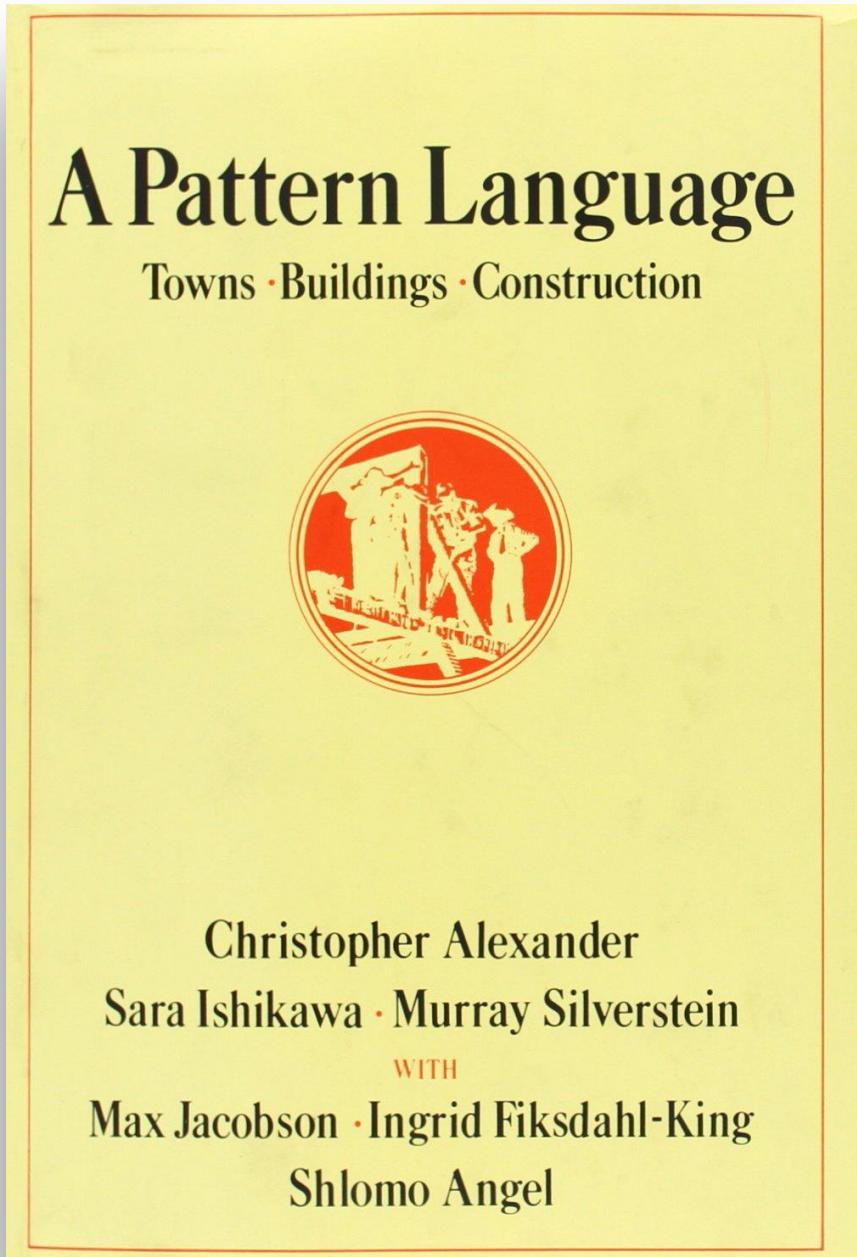
Dr. Roland Huß  
Principal Software Engineer  
@ro14nd  
<https://k8spatterns.io>



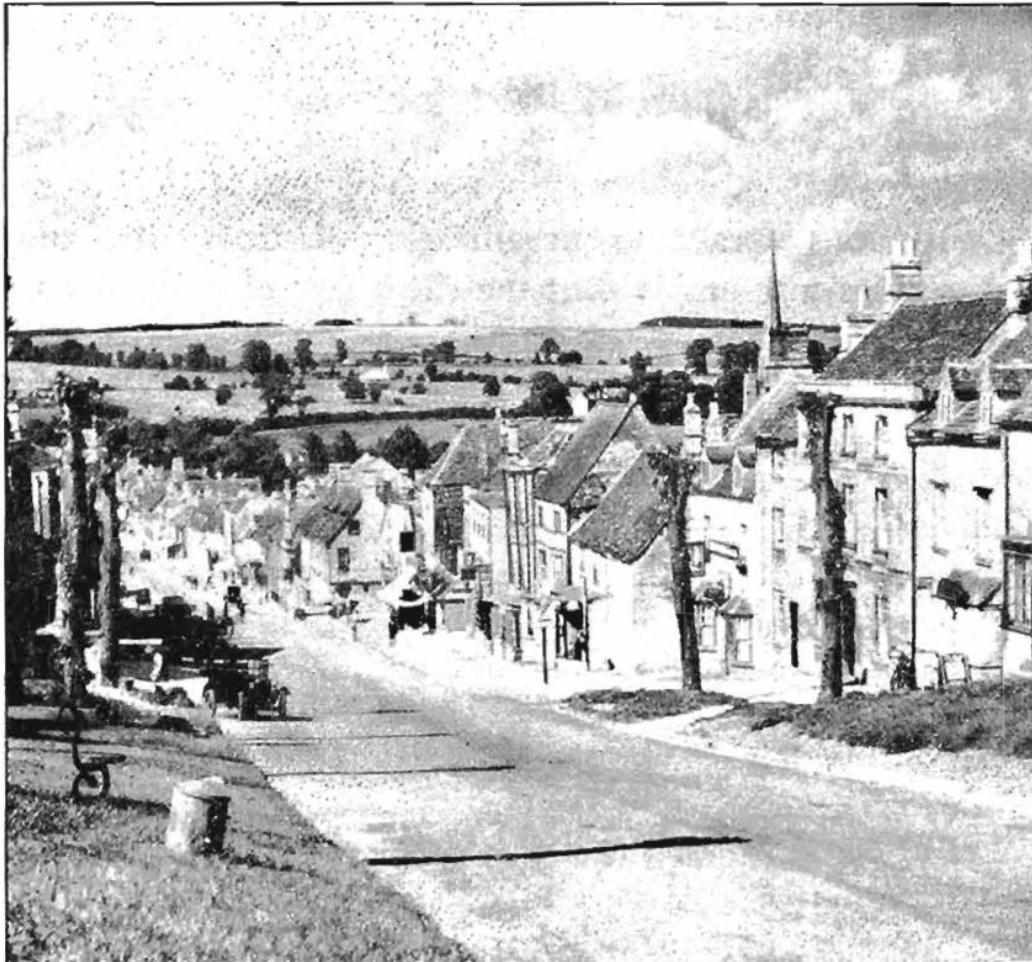
<https://k8spatterns.io>

The background of the image is a intricate geometric pattern. It features a repeating motif of overlapping diamond shapes, creating a sense of depth and movement. The colors used are various shades of brown, tan, and black, which are arranged in a way that suggests a three-dimensional perspective. The overall effect is one of a sophisticated and artistic design.

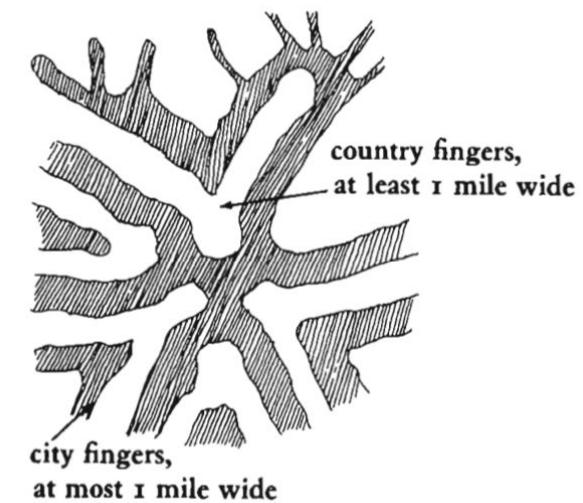
# Patterns

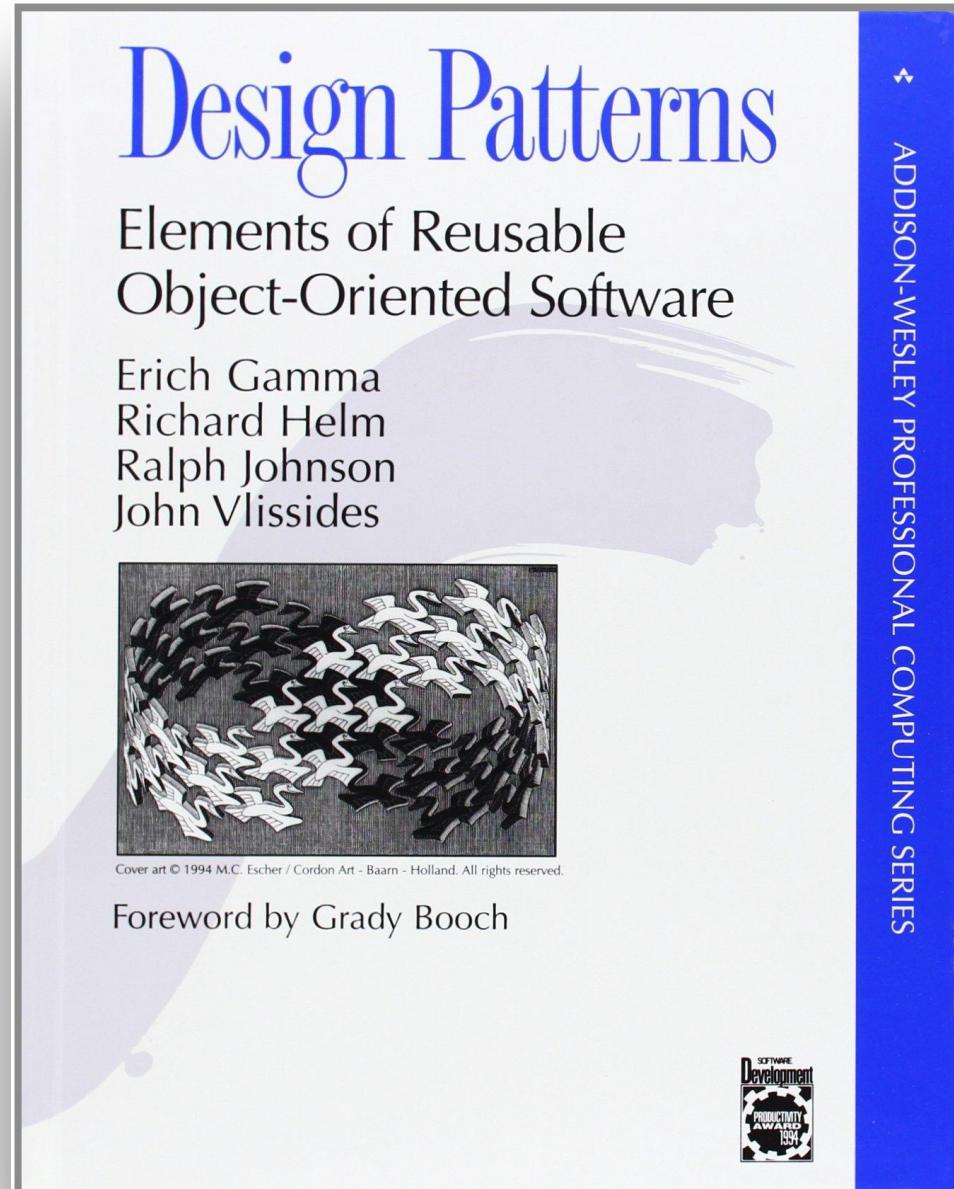


### 3 CITY COUNTRY FINGERS\*\*



*When the countryside is far away  
the city becomes a prison.*

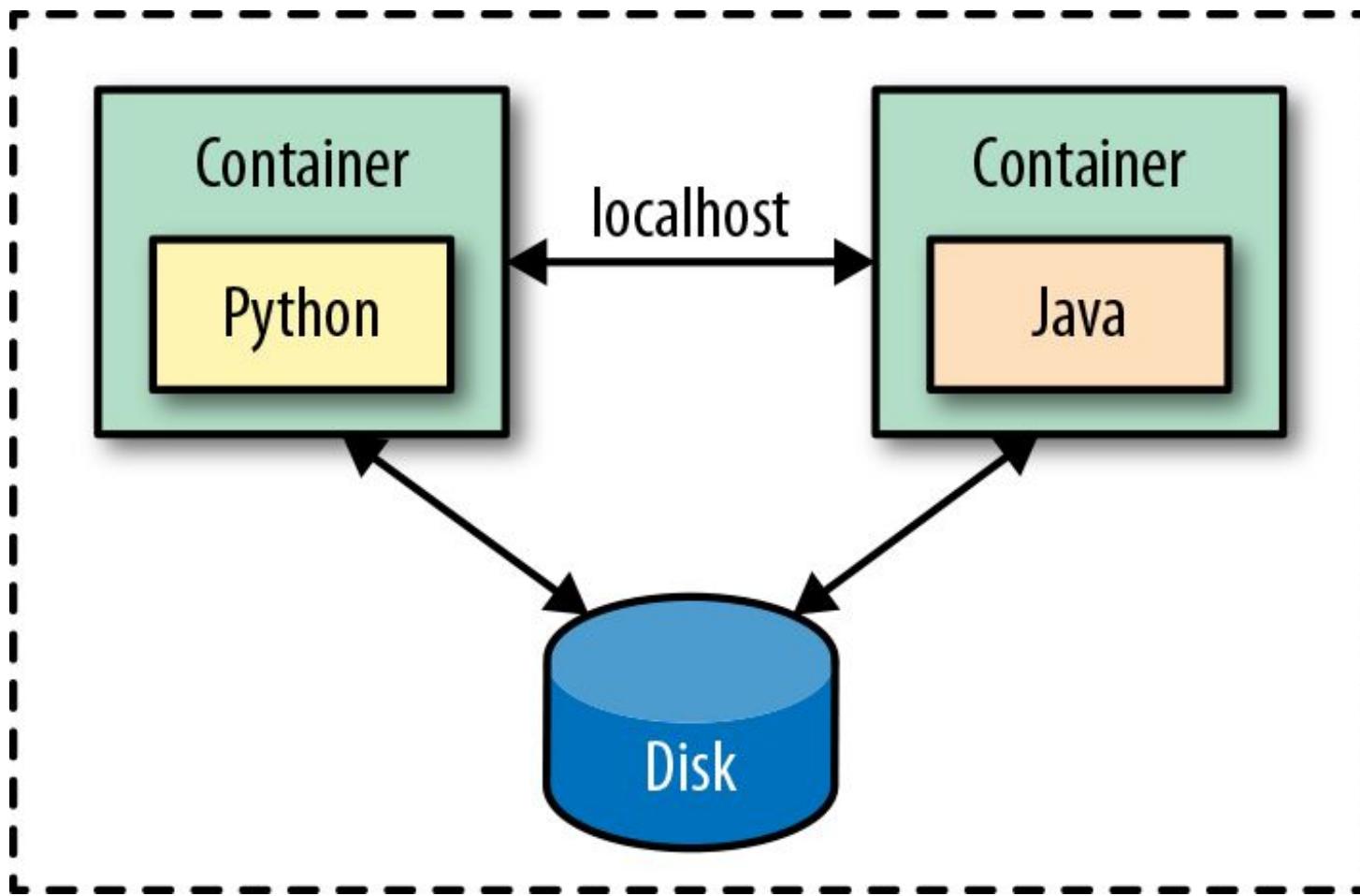


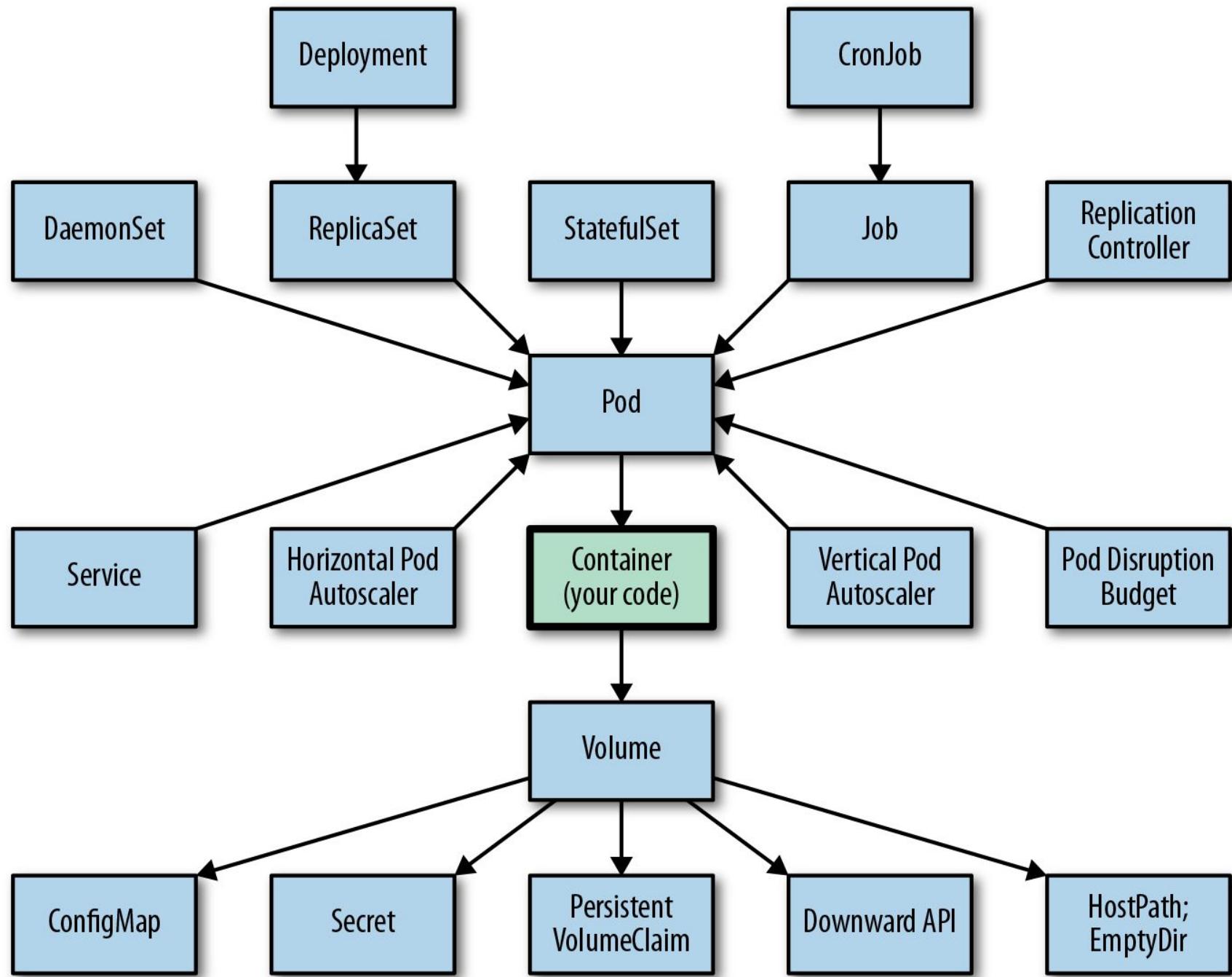


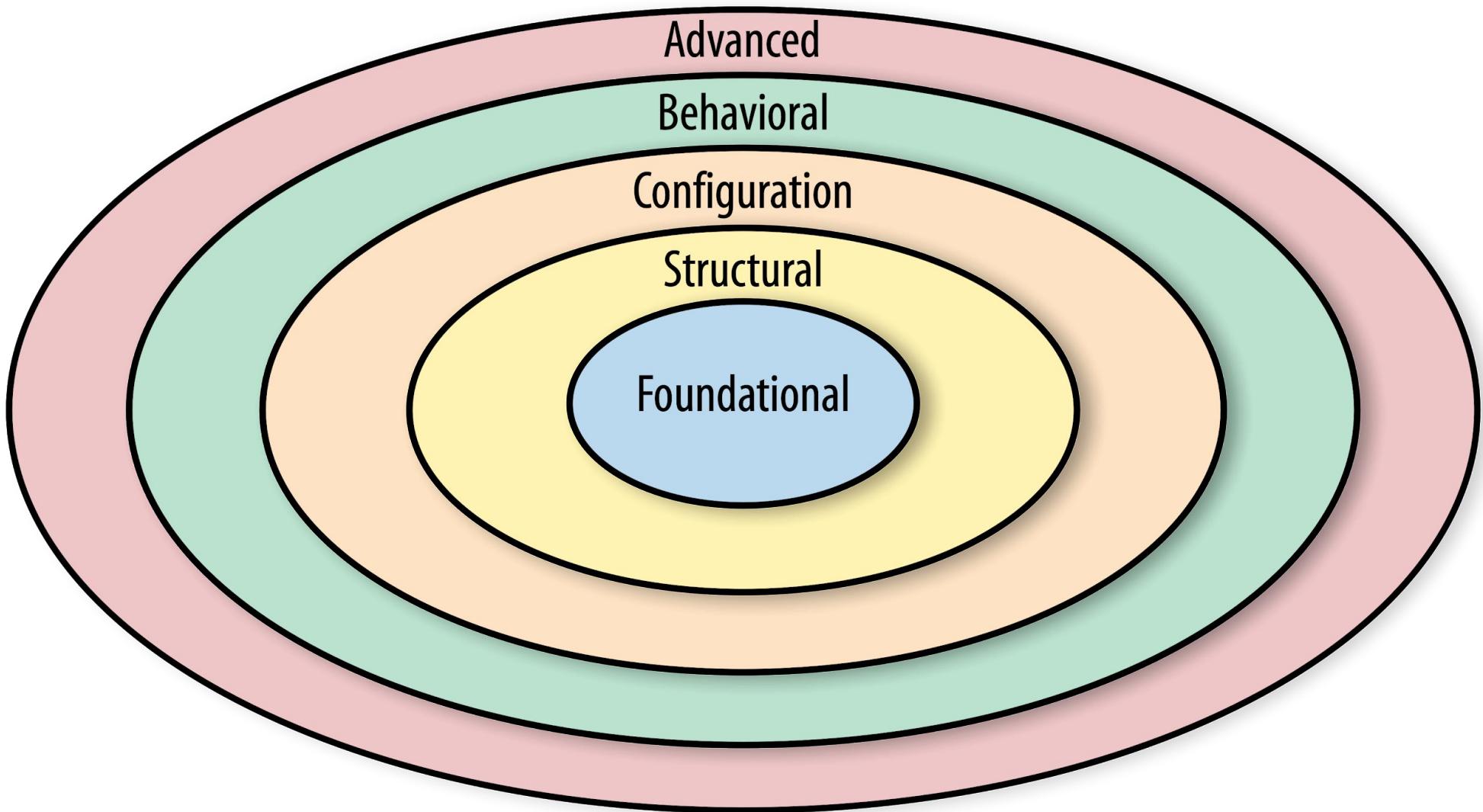
A large, dark wooden ship's wheel is positioned in the foreground, angled towards the left. A rectangular metal plaque is attached to the right side of the wheel, featuring the words "U.S. COAST GUARD" in both English and Spanish. The background is a blurred photograph of a sunset or sunrise over the ocean, with warm orange and yellow hues reflecting on the water.

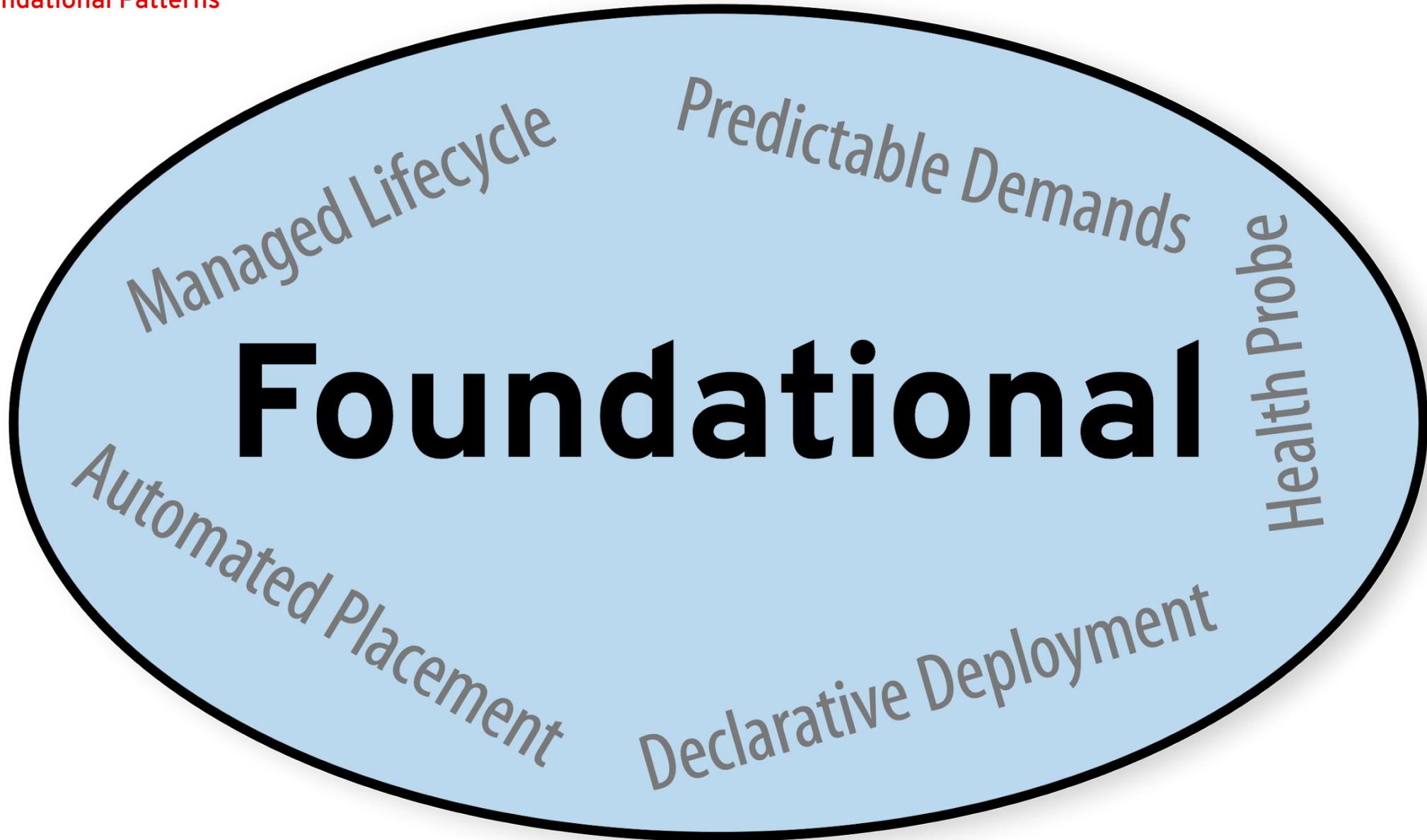
# Kubernetes

# Pod











# Declarative Deployment

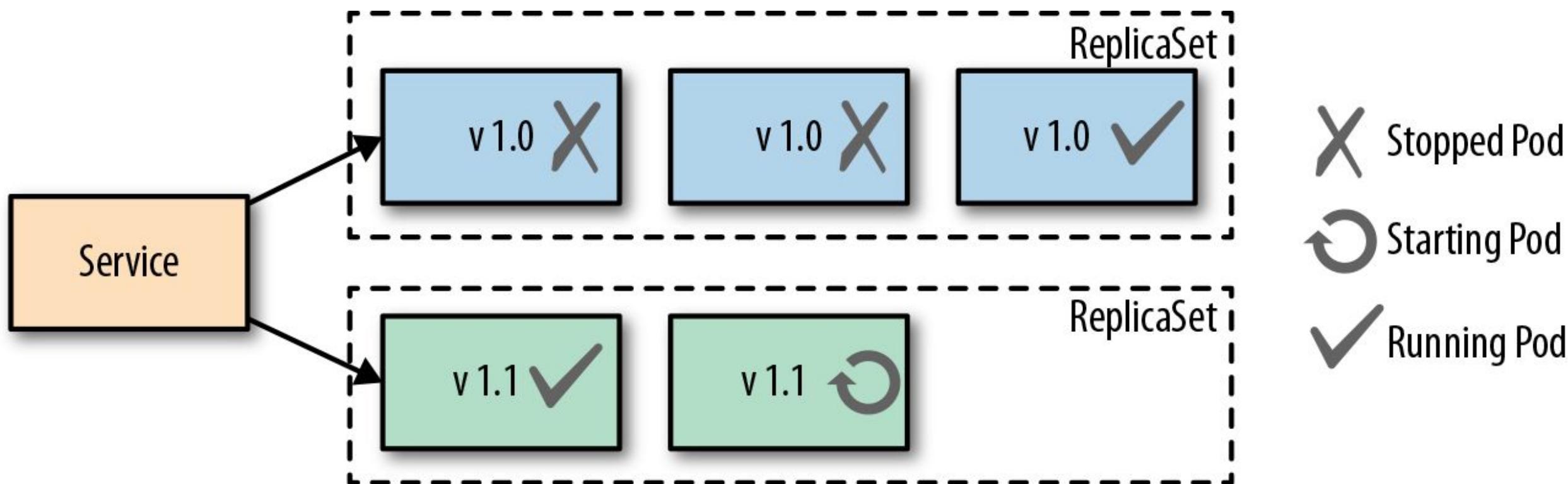
---

# How to perform application installations and upgrades by configuration.

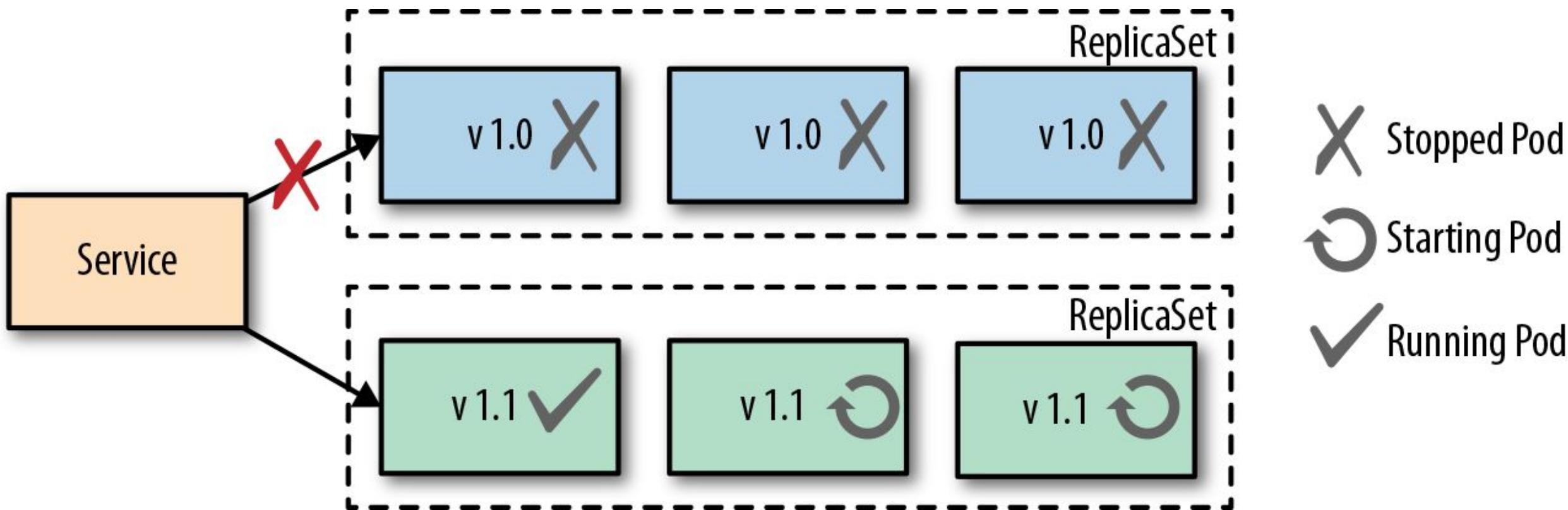
# Deployment

- **Declarative** vs. **Imperative** deployment
- Deployment Kubernetes Resource:
  - Holds template for Pod
  - Creates ReplicaSet on the fly
  - Allows rollback
  - Update strategies are declarable

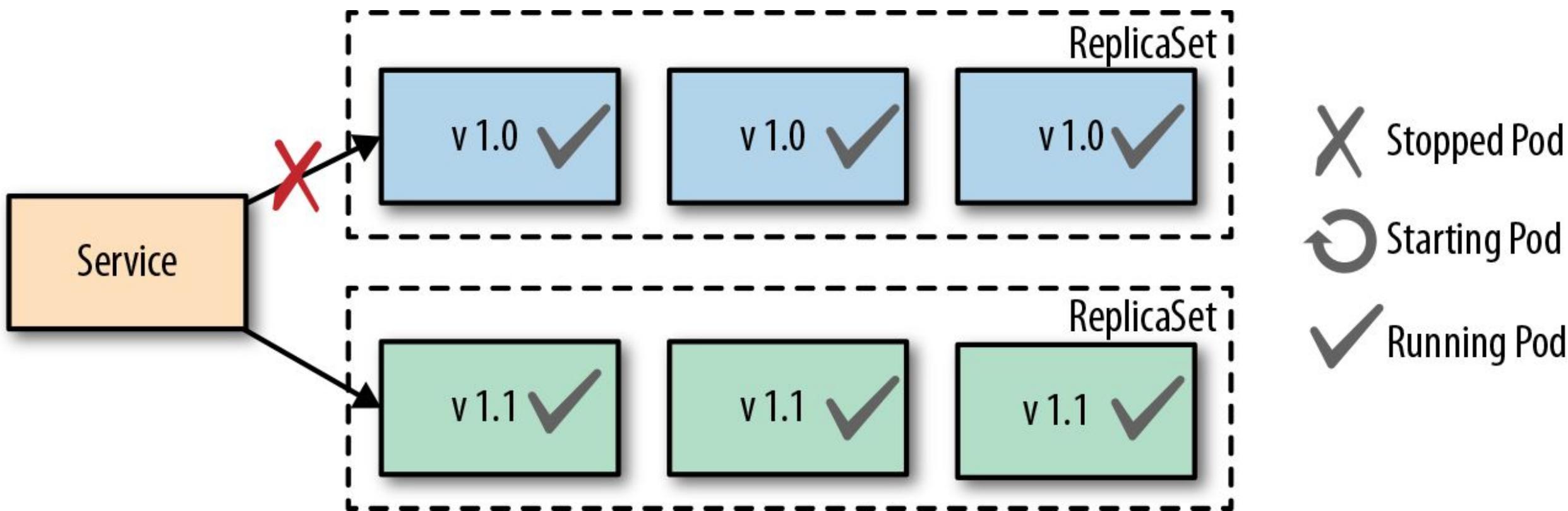
# Rolling Deployment



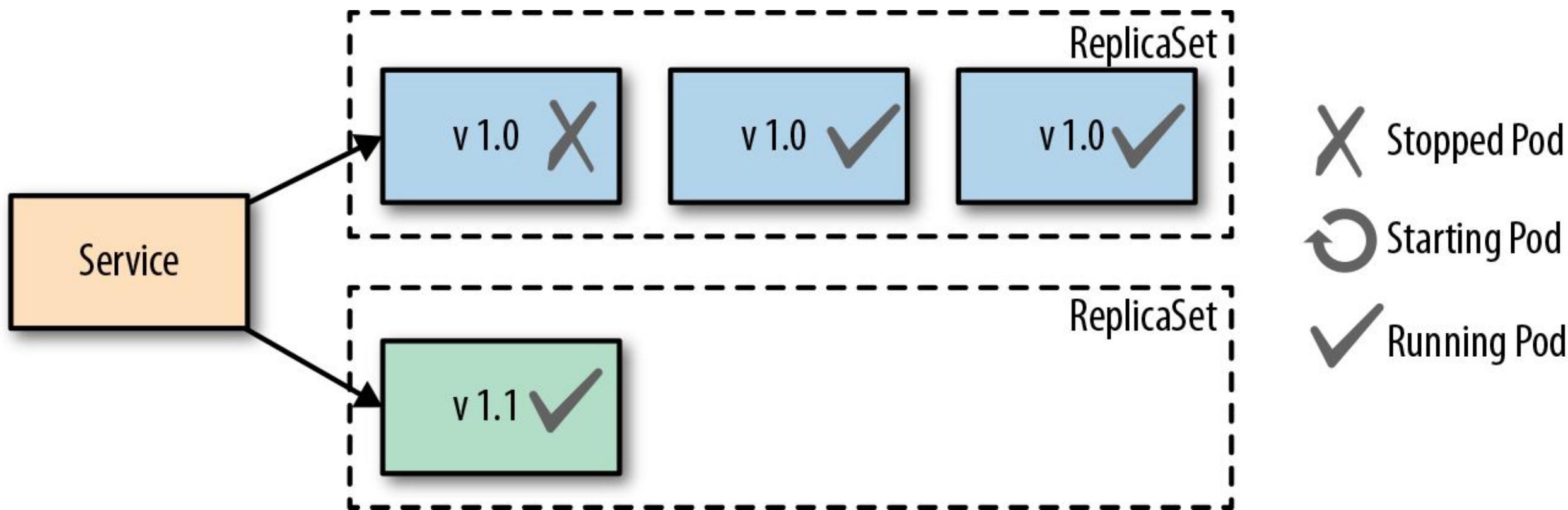
# Fixed Deployment



# Blue-Green Release

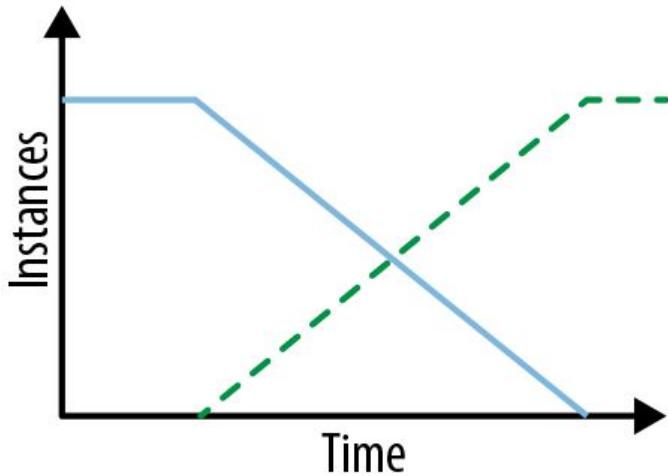


# Canary Release

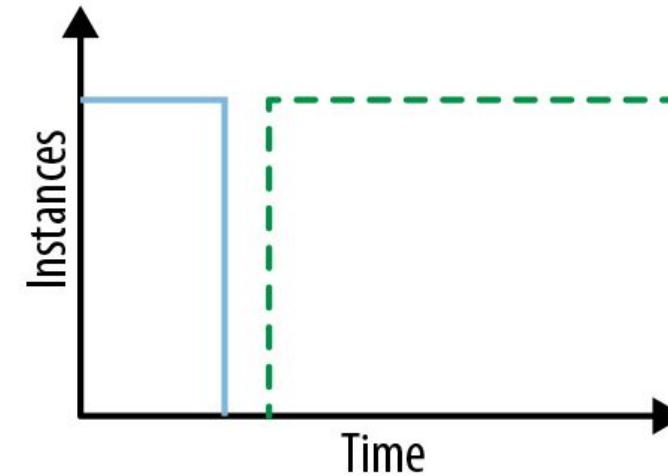


## Declarative Deployment

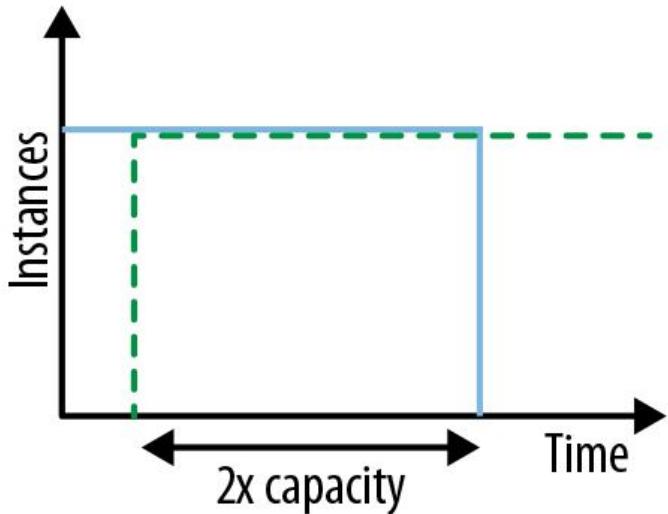
**Rolling deployment**



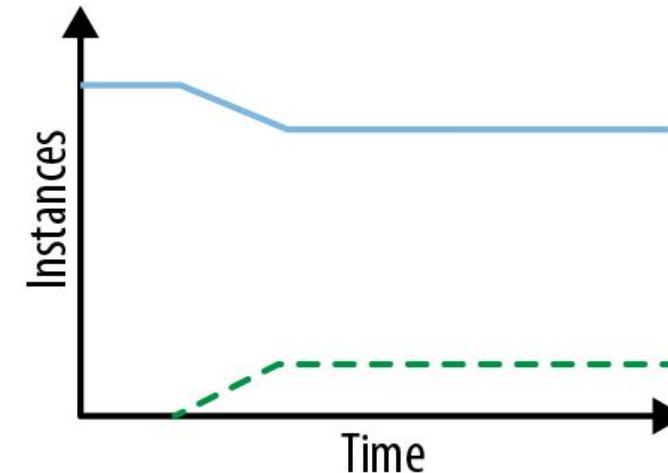
**Fixed deployment**

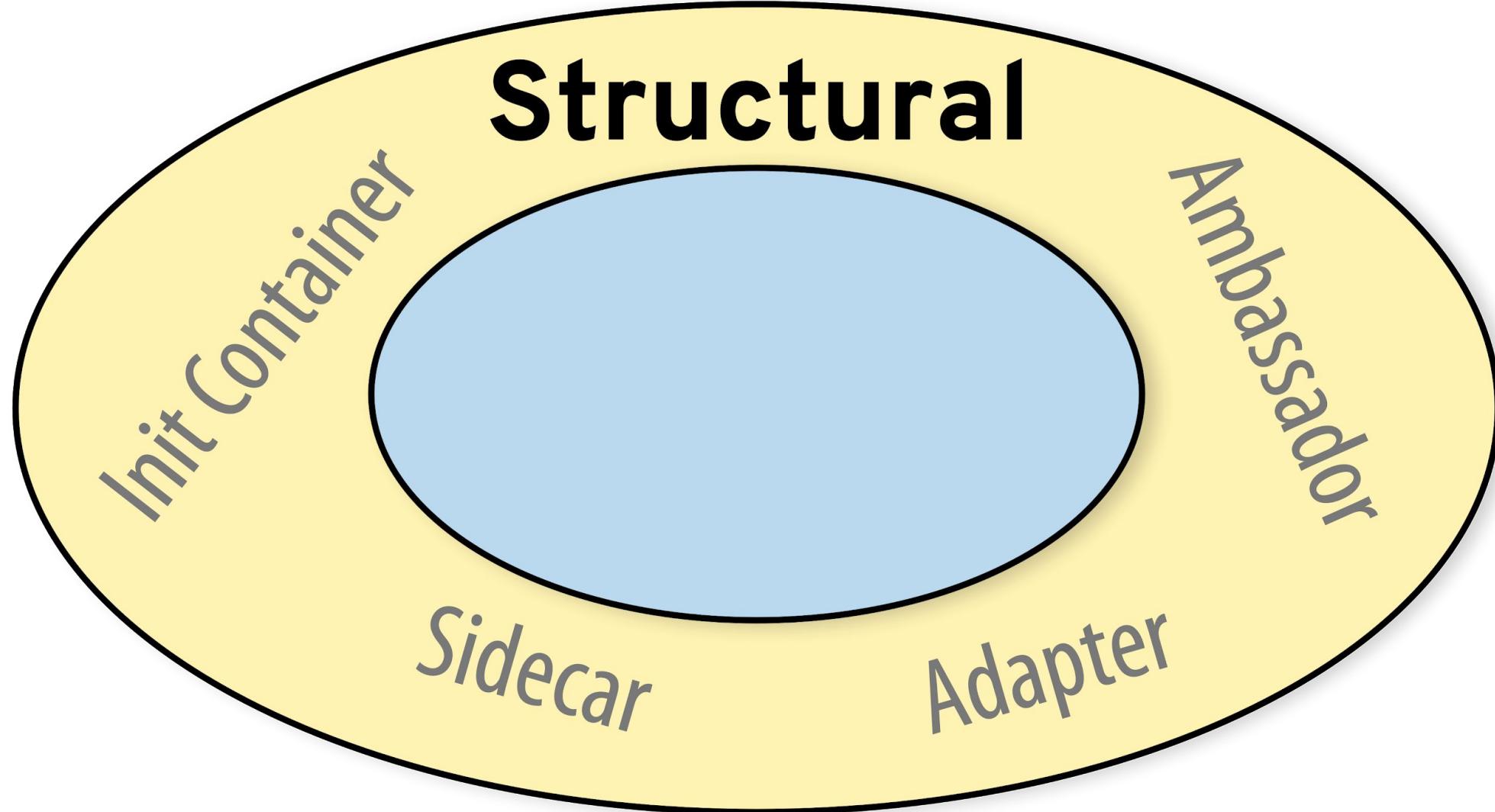


**Blue-green release**



**Canary release**





The image is a collage of three photographs related to sidecar racing. The top photograph shows a rider in a purple and white patterned suit and helmet riding a sidecar through mud, with another rider visible behind them. The middle photograph is a close-up of a sidecar, heavily covered in mud, with its number plate partially visible. The bottom photograph shows a rider in a dark suit and helmet leaning into a turn on a muddy track.

Sidecar

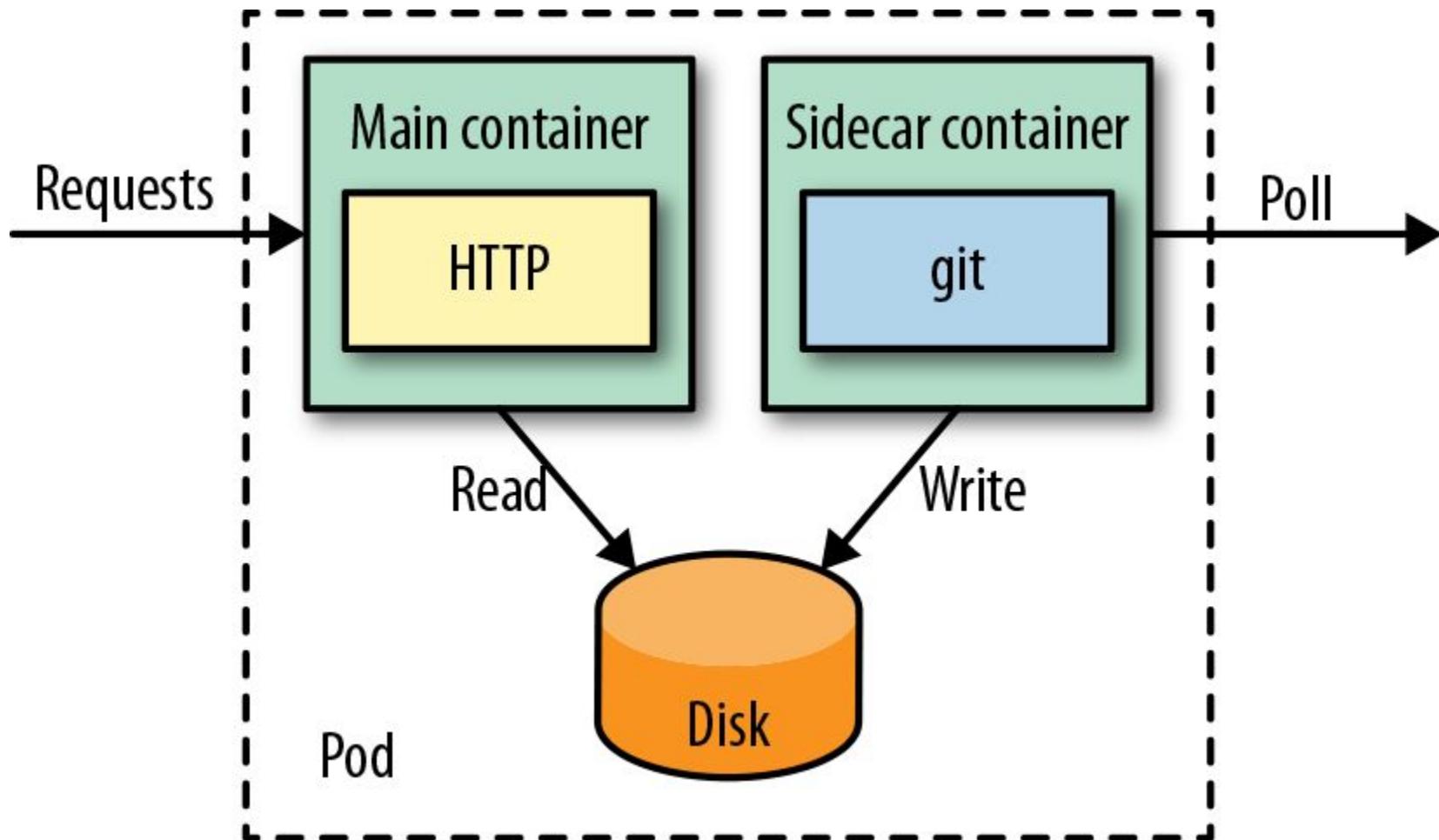
---

# How to enhance the functionality of an application without changing it

# Sidecar

- Runtime collaboration of containers
- Connected via shared resources:
  - Network
  - Volumes
- Similar what AOP is for programming
- Separation of concerns

# Sidecar



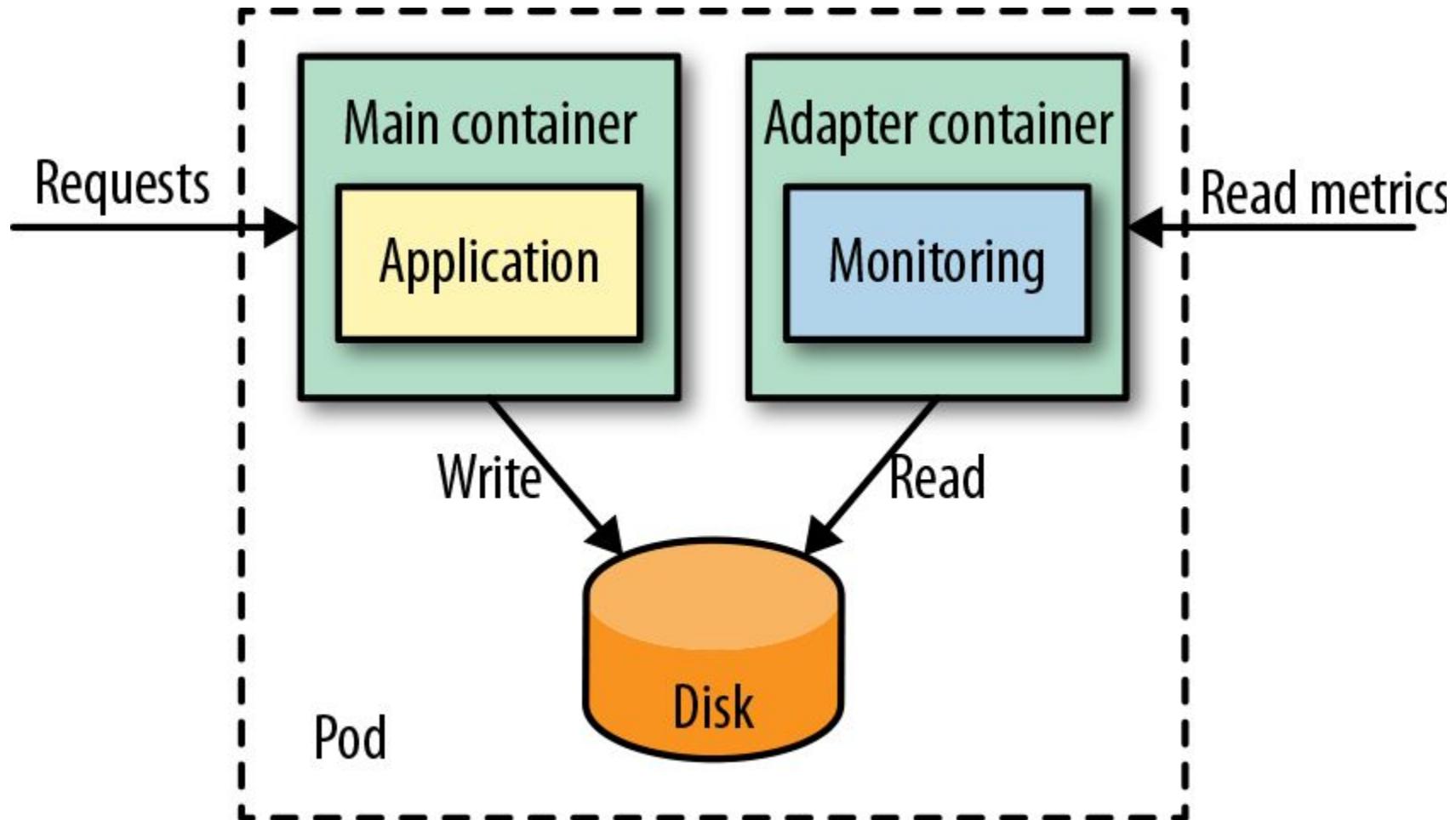
# Adapter



---

# How to decouple access to a container **from** the outside world

# Adapter



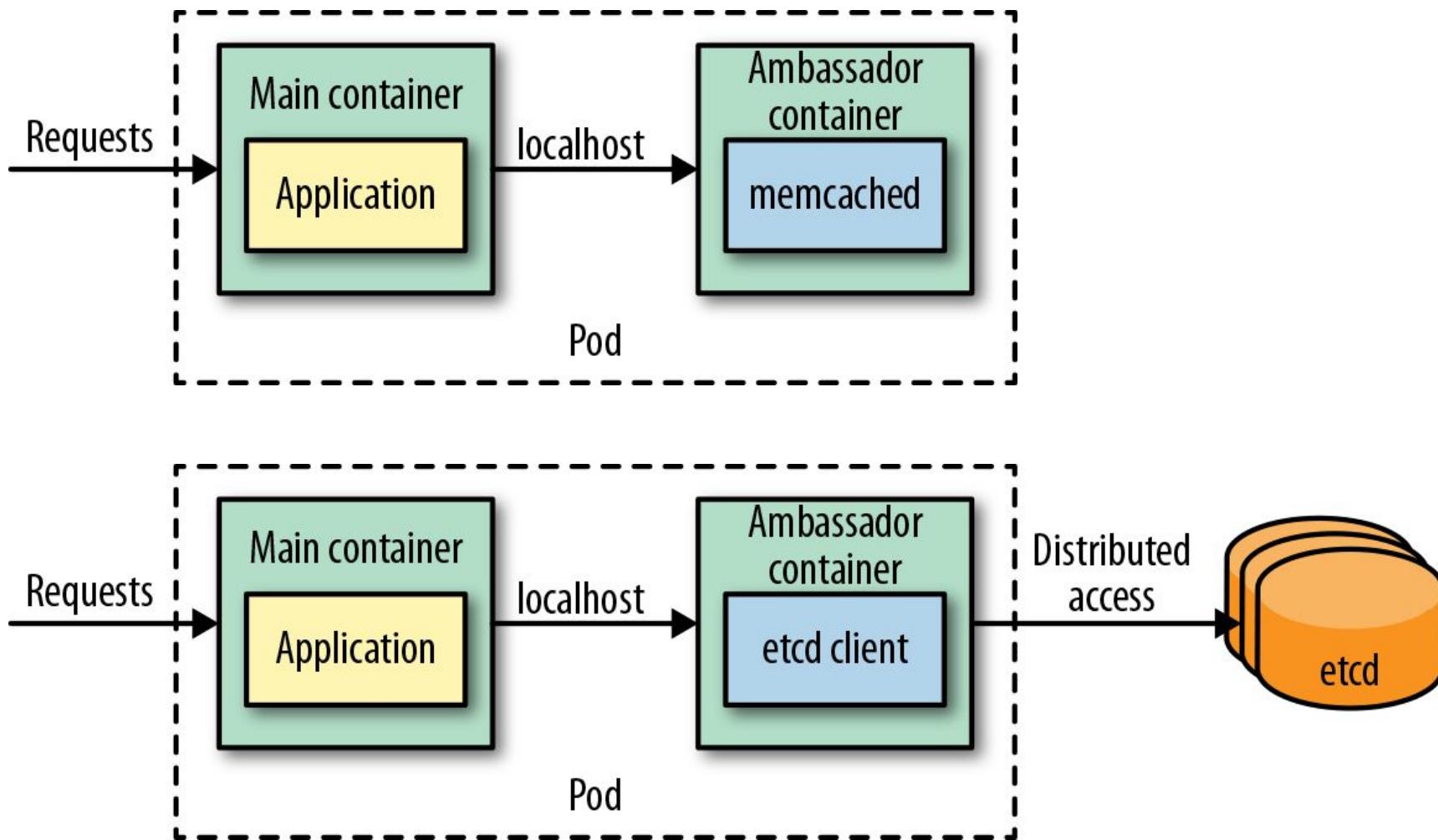


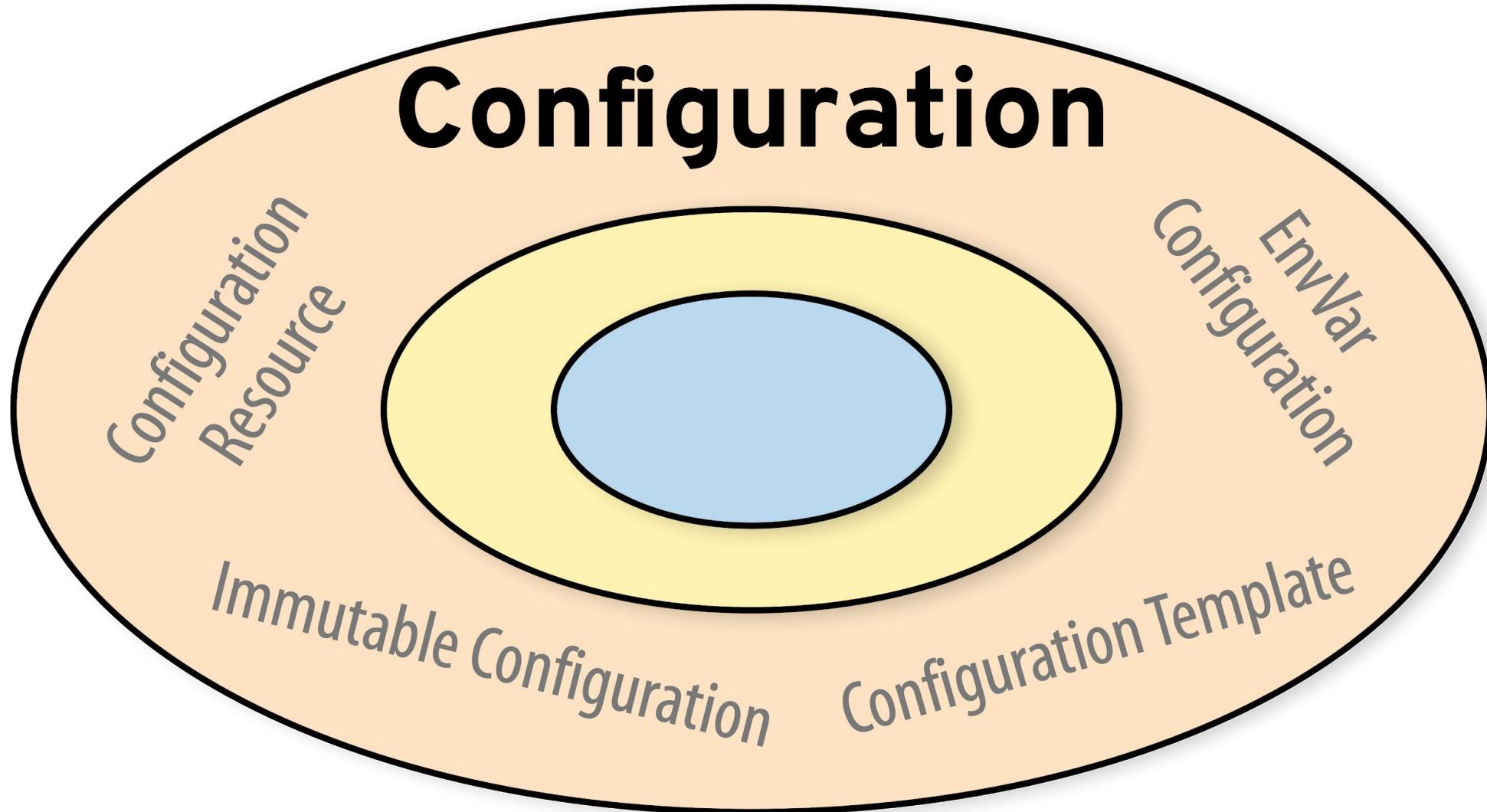
# Ambassador

---

# How to decouple a container's access **to** the outside world

# Ambassador







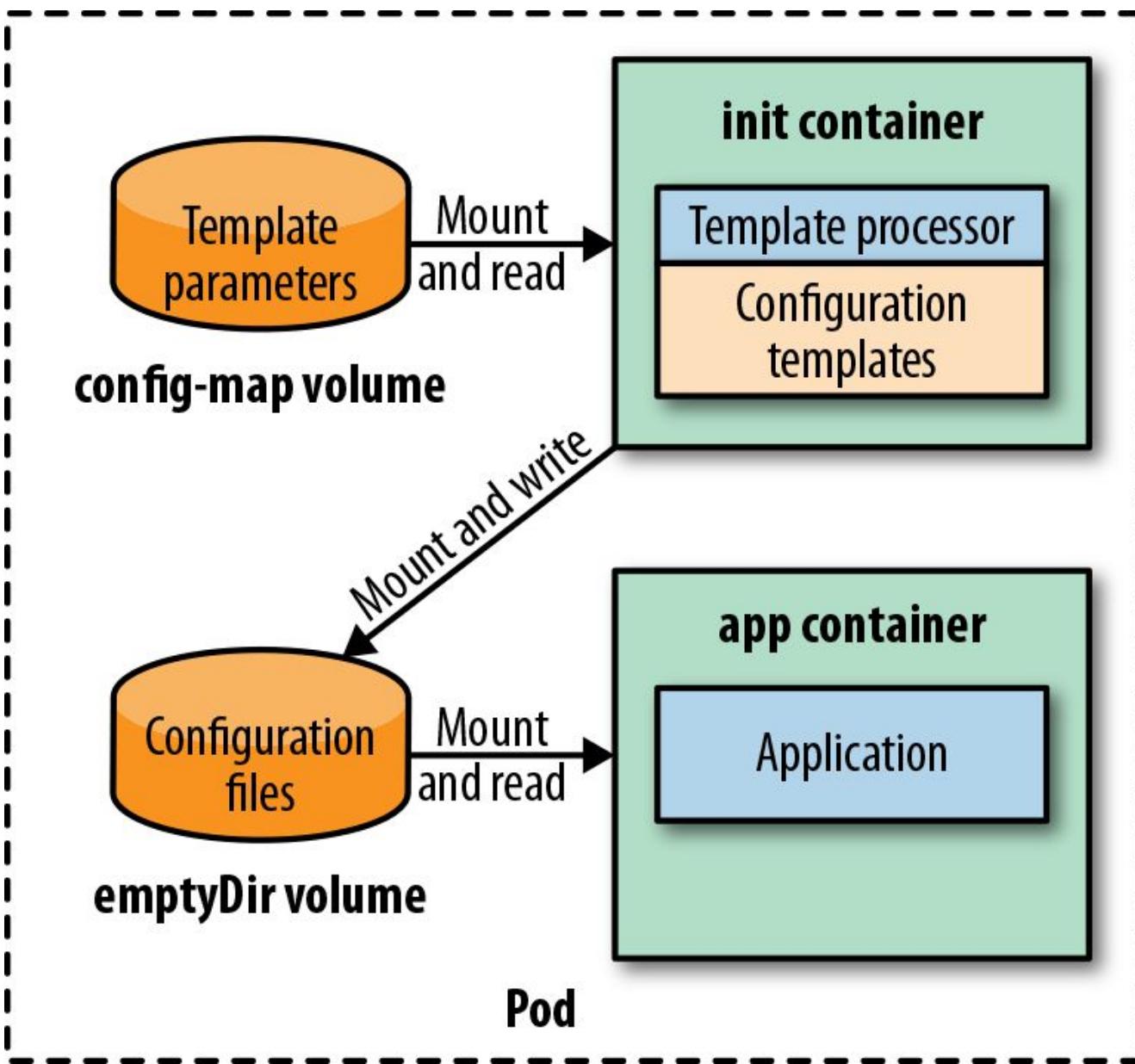
# Configuration Template

---

# How to manage large and complex similar configuration data

# Preparing Configuration during Startup

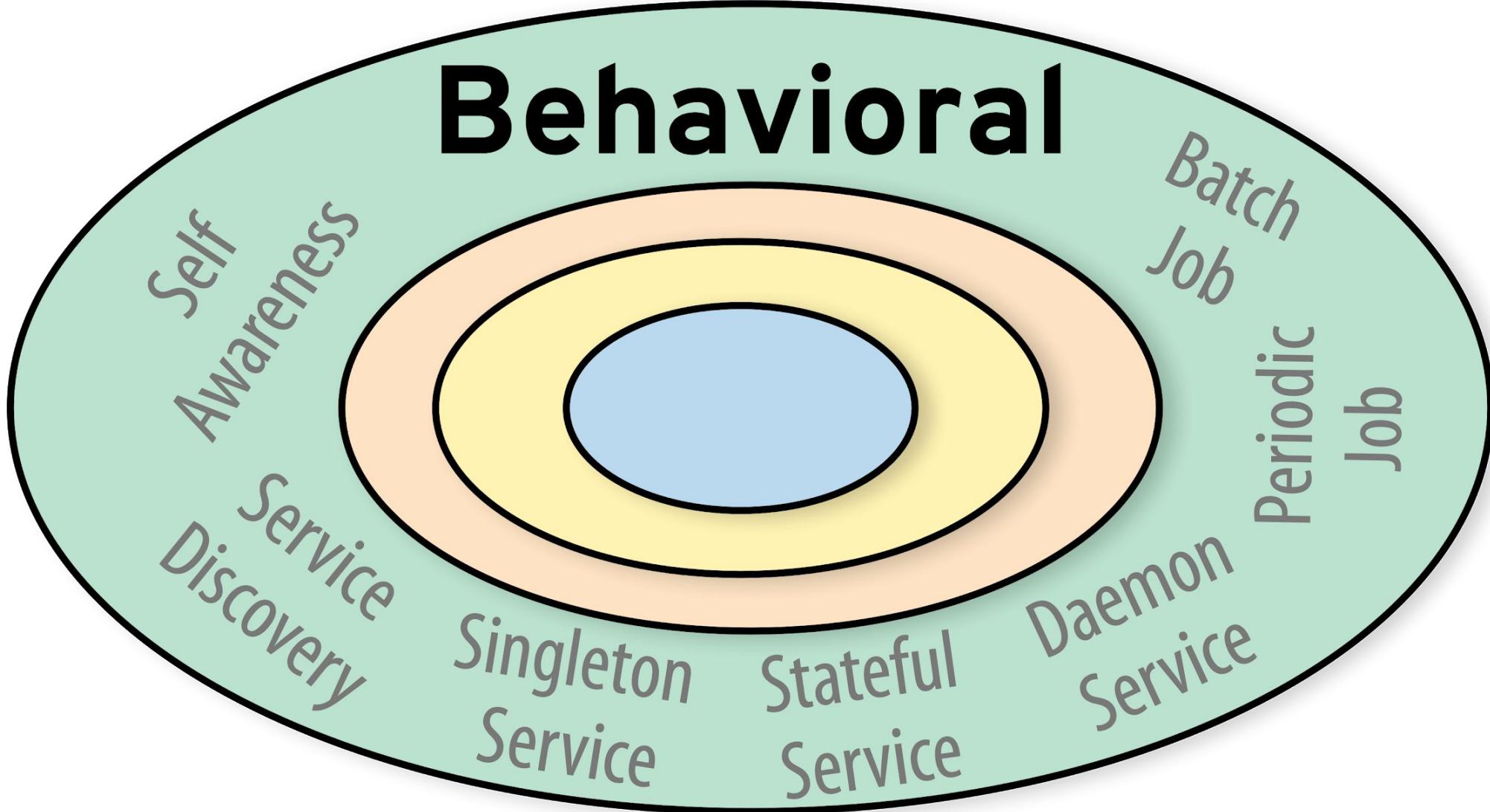
- Init Container ...
  - ... contains a template processor
  - ... holds the configuration template
  - ... picks up template parameter from a ConfigMap
  - ... stores final configuration on a shared volume
- Main Container ....
  - ... accesses created configuration from shared volume



## Configuration Template

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: wildfly-cm-template
spec:
  replicas: 1
  template:
    spec:
      initContainers:
        - image: k8spatterns/config-init
          name: init
          volumeMounts:
            - mountPath: "/params"
              name: wildfly-parameters
            - mountPath: "/out"
              name: wildfly-config
```

```
containers:
- image: jboss/wildfly:10.1.0.Final
  name: server
  volumeMounts:
    - mountPath: "/config"
      name: wildfly-config
  volumes:
    - name: wildfly-parameters
      configMap:
        name: wildfly-params-cm
    - name: wildfly-config
      emptyDir: {}
```



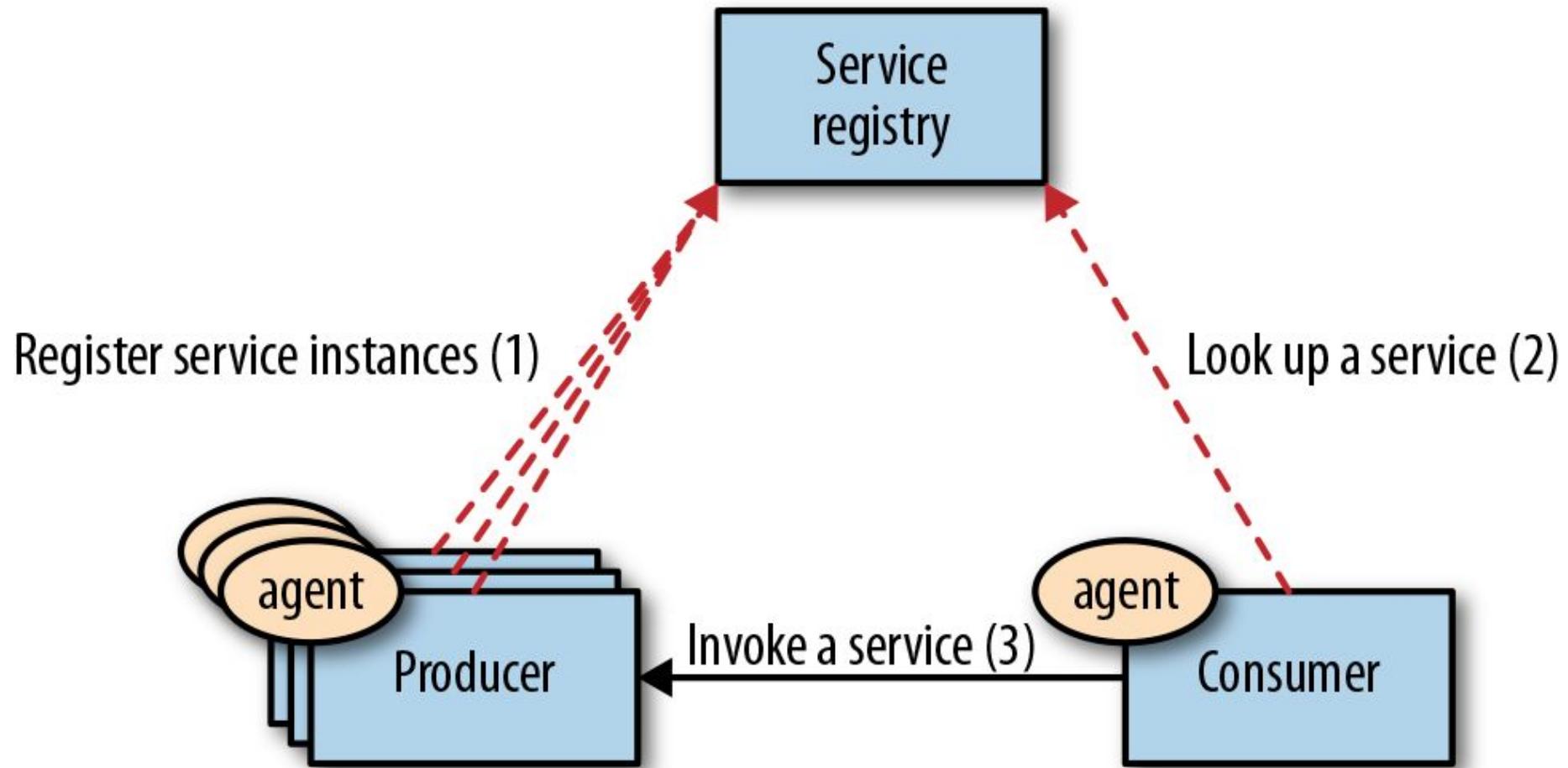


# Service Discovery

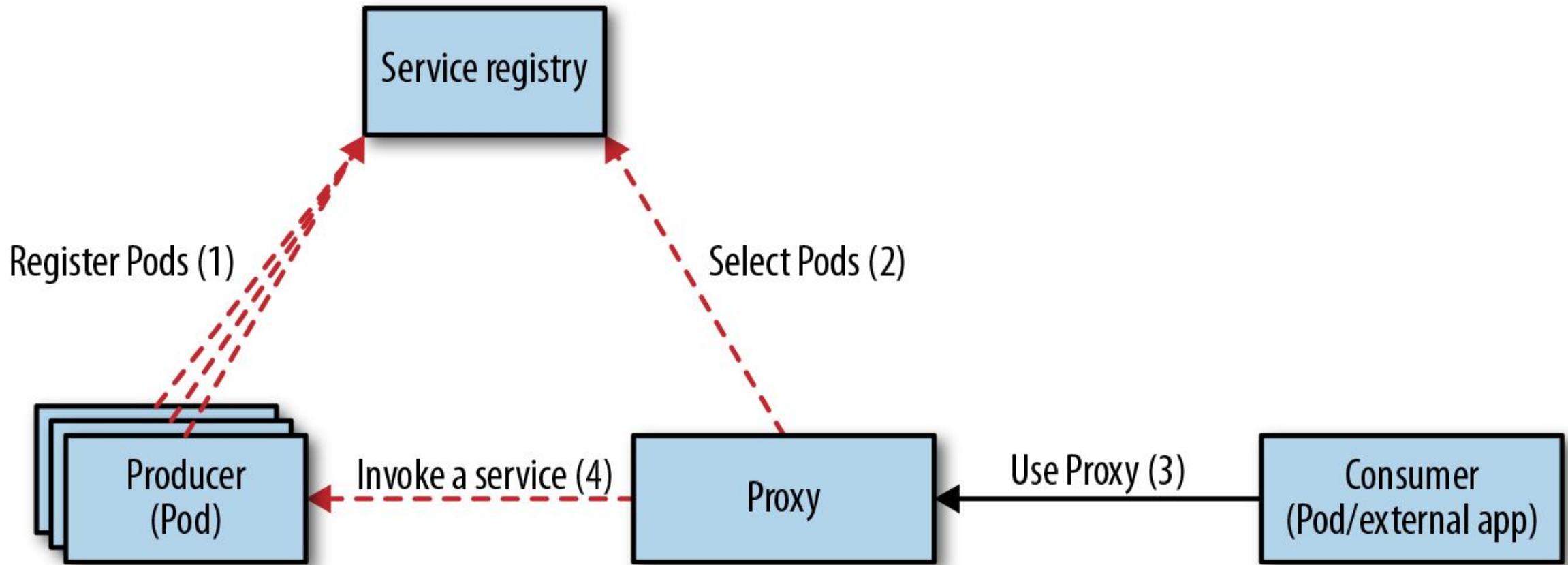
---

# How to discover and use services

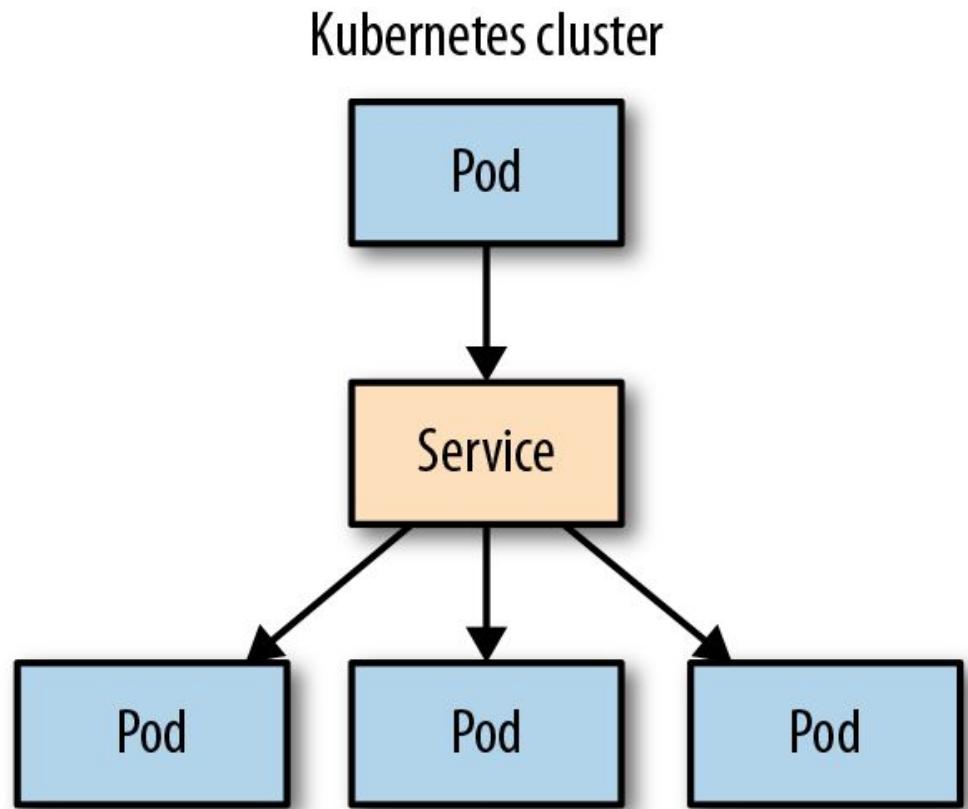
# Client-side Service Discovery (non Kubernetes)



# Server-side Service Discovery (Kubernetes)

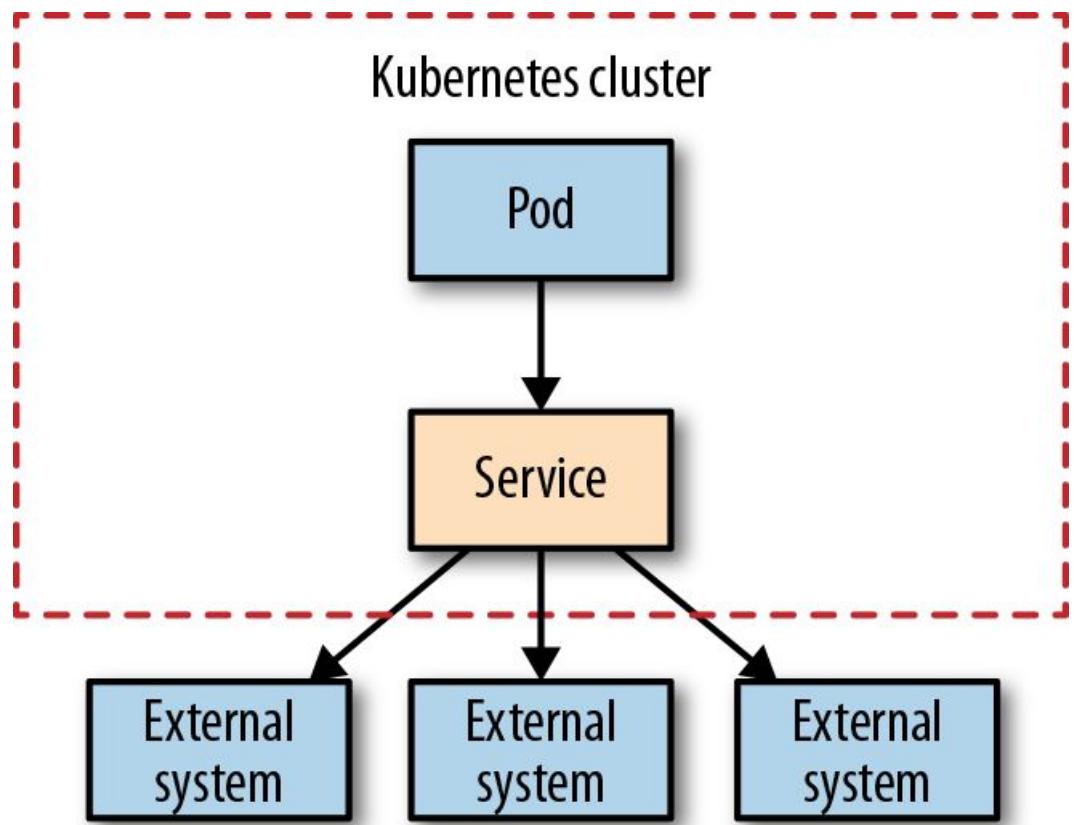


# Internal Service Discovery



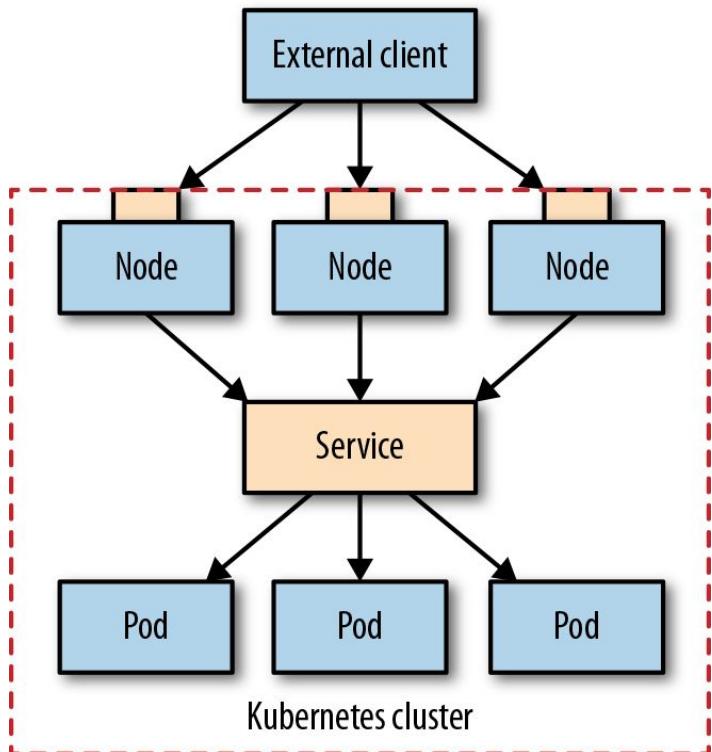
- Discovery through DNS lookups
- Pods picked by label selector
- Multiple ports per Service
- Session affinity on IP address
- Successful readiness probes required for routing
- Virtual IP address for each Service

# Manual Service Discovery

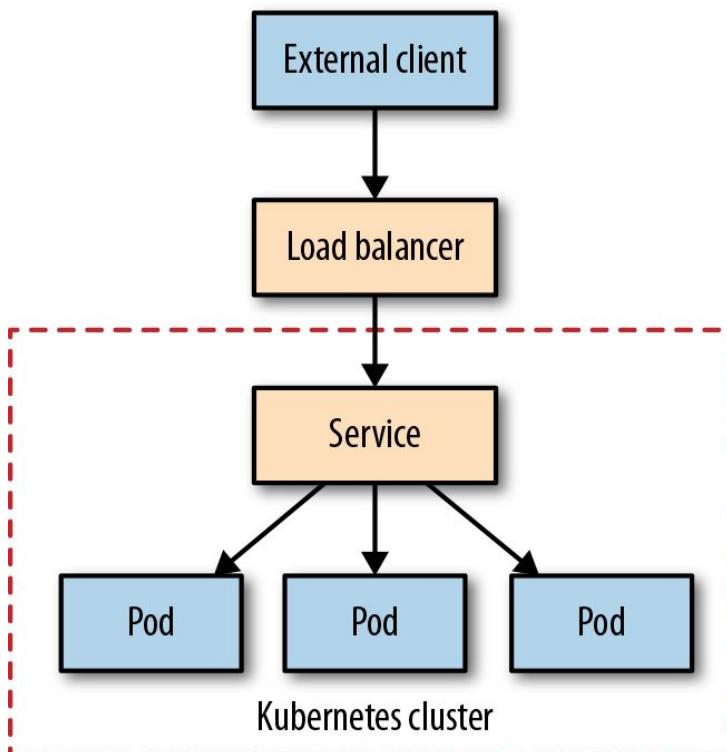


- Service without selector
- Manually creating Endpoint resource with the same name as the Service
- Service of type **ExternalName** map are registered as DNS CNAMEs

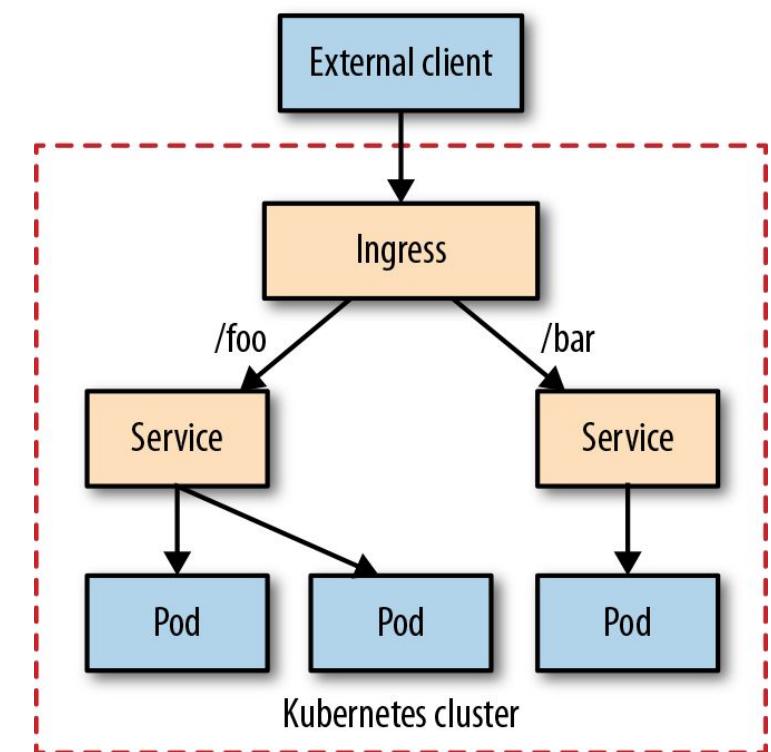
## Node Port



## Load Balancer



## Ingress

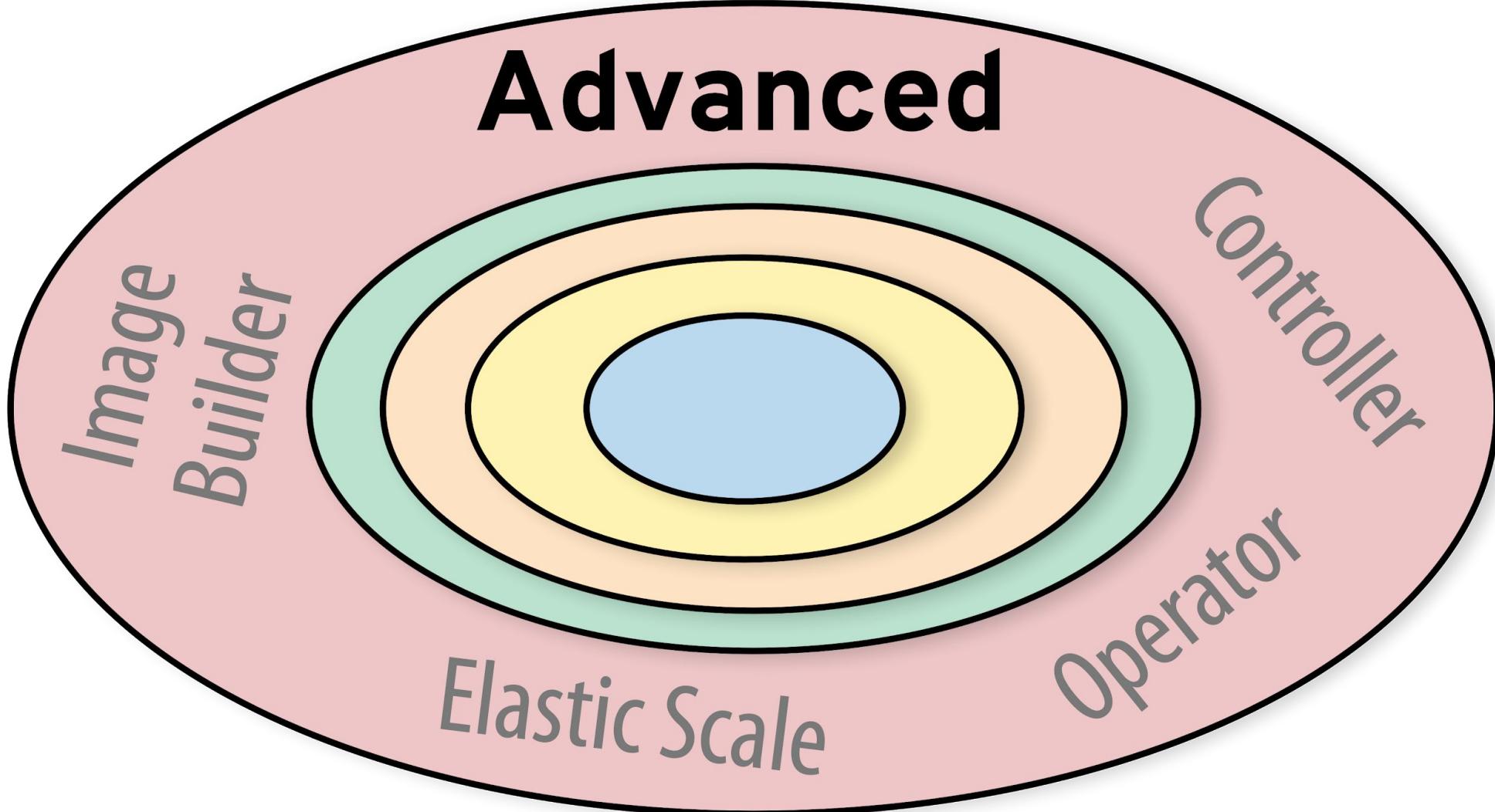


# Ingress

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: random-generator
spec:
  rules:
  - http:
    paths:
    - path: /
      backend:
        serviceName: random-generator
        servicePort: 8080
    - path: /cluster-status
      backend:
        serviceName: cluster-status
        servicePort: 80
```

## Service Discovery

Name	Configuration	Client type	Summary
ClusterIP	type: ClusterIP .spec.selector	Internal	The most common internal discovery mechanism
Manual IP	type: ClusterIP kind: Endpoints	Internal	External IP discovery
Manual FQDN	type: ExternalName .spec.externalName	Internal	External FQDN discovery
Headless Service	type: ClusterIP .spec.clusterIP: None	Internal	DNS-based discovery without a virtual IP
NodePort	type: NodePort	External	Preferred for non-HTTP traffic
LoadBalancer	type: LoadBalancer	External	Requires supporting cloud infrastructure
Ingress	kind: Ingress	External	L7/HTTP-based smart routing mechanism



# Controller



---

# How to get from the current state to a declared target state

# State Reconciliation

- Kubernetes as distributed state manager
- Make the **actual** state more like the declared **target** state.



**Observe** - Discover the actual state

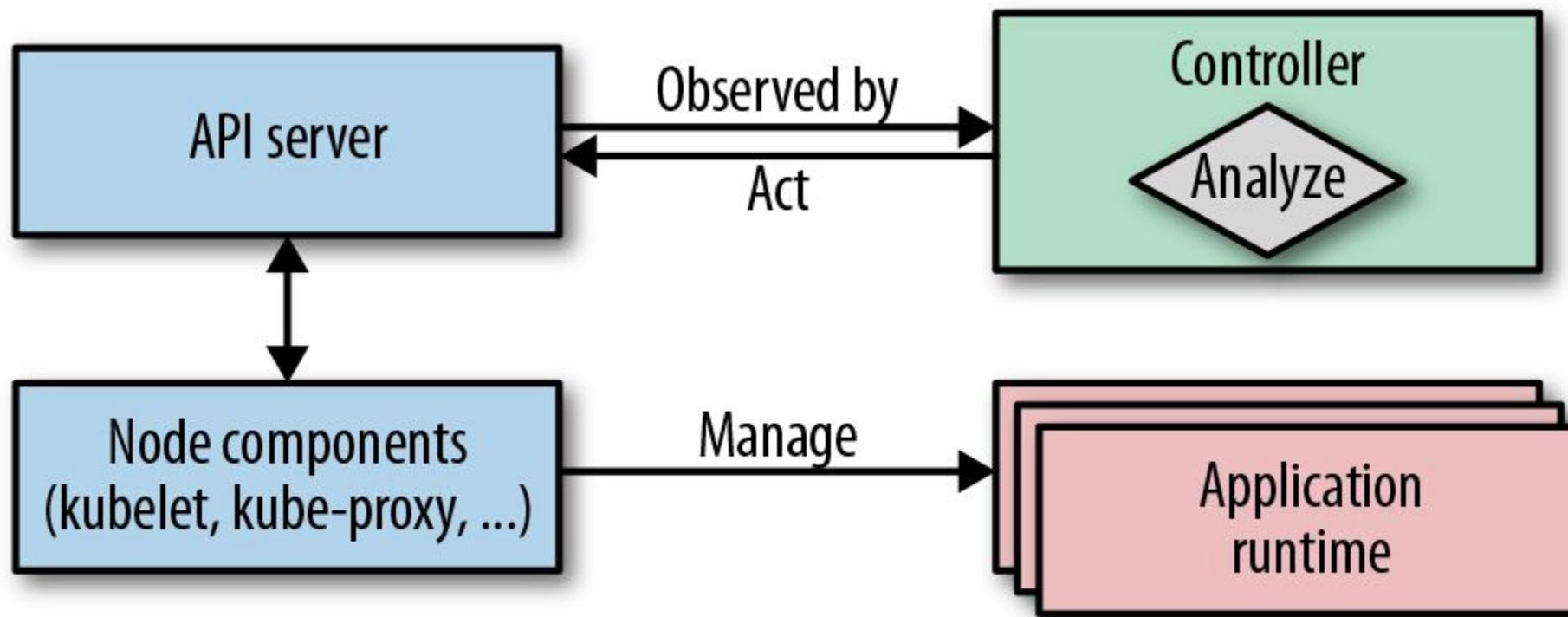


**Analyze** - Determine difference to target state



**Act** - Perform actions to drive the actual to the desired state

# Observe - Analyze - Act



# Common Triggers

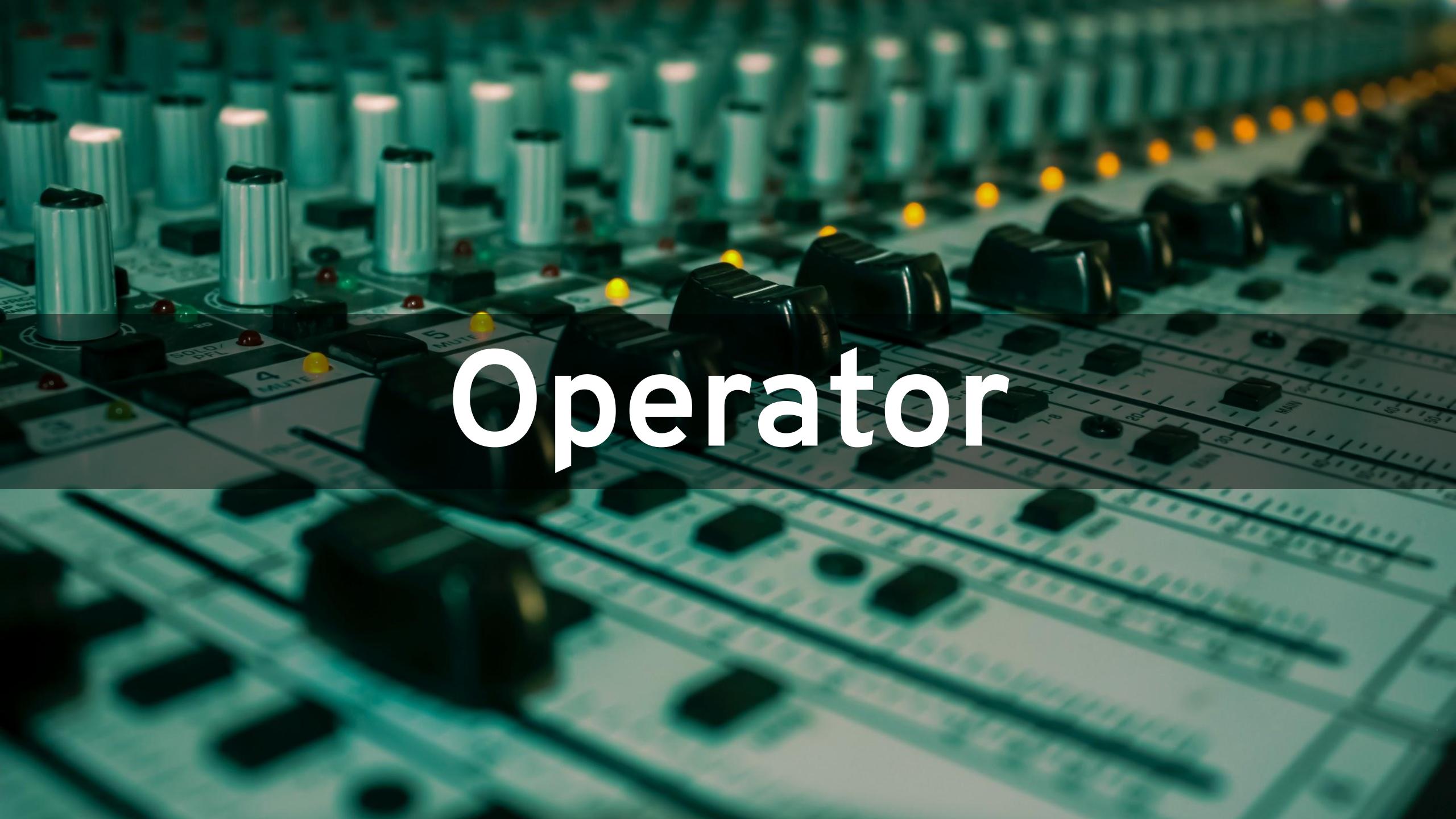
- Labels
  - Indexed by backend
  - Suitable for selector-like functionality
  - Limitation on charset for names and values
- Annotations
  - No syntax restrictions
  - Not indexed
- ConfigMaps
  - Good for complex structured state declarations
  - Simple alternative to CustomResourceDefinitions

# ConfigMap Watch Controller

```
namespace=${WATCH_NAMESPACE:-default}
base=http://localhost:8001
ns=namespaces/$namespace

curl -N -s $base/api/v1/$ns/configmaps?watch=true | \
while read -r event
do
    type=$(echo "$event" | jq -r '.type')
    config_map=$(echo "$event" | jq -r '.object.metadata.name' )
    annotations=$(echo "$event" | jq -r '.object.metadata.annotations' )

    if [ $type = "MODIFIED" ]; then
        # Restart Pods using this ConfigMap
        # ...
    fi
done
```



# Operator

---

# How to encapsulate operational knowledge into executable software

# Definition

“” An **operator** is a Kubernetes **controller** that understands two domains: Kubernetes and *something else*. By combining knowledge of both areas, it can **automate** tasks that usually require a human operator that understands both domains.

---

Jimmy Zelinskie

<http://bit.ly/2Fjlx1h>

Technical:

**Operator = Controller + CustomResourceDefinition**

# CustomResourceDefinition

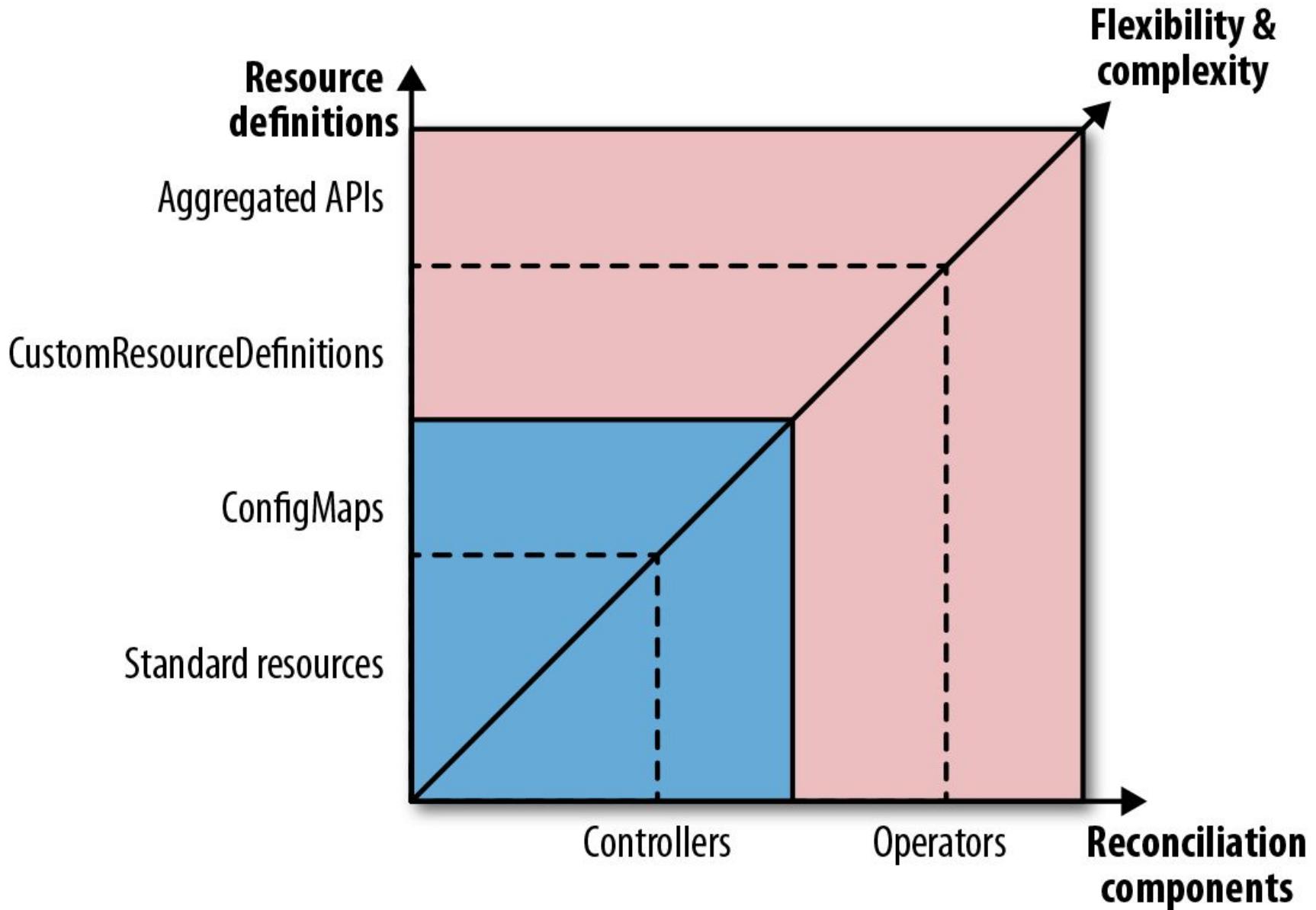
Custom resource is modelling a custom domain and managed through the Kubernetes API

```
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: configwatchers.k8spatterns.io
spec:
  scope: Namespaced
  group: k8spatterns.io
  version: v1
  names:
    kind: ConfigWatcher
    plural: configwatchers
  validation:
    openAPIV3Schema:
      ...

```

# Custom Resource

```
kind: ConfigWatcher
apiVersion: k8spatterns.io/v1
metadata:
  name: webapp-config-watcher
spec:
  configMap: webapp-config
  podSelector:
    app: webapp
```



# CRD Classification

- Installation CRDs
  - Installing and operating applications
  - Backup and Restore
  - Monitoring and self-healing
  - Example: Prometheus for installing Prometheus & components
- Application CRDs
  - Application specific domain concepts
  - Example: ServiceMonitor for registering Kubernetes service to be scraped by Prometheus

# Operator Hub

The screenshot shows the OperatorHub.io homepage with a search bar and a provider filter set to "Red+Hat". The results page displays five operator cards:

- Jaeger Tracing** (provided by Jaeger): Provides tracing, monitoring and troubleshooting for microservices-based applications.
- Kubernetes Federation** (provided by Red Hat): Gain Hybrid Cloud capabilities between your clusters with Kubernetes Federation.
- MongoDB** (provided by MongoDB, Inc): The MongoDB Enterprise Kubernetes Operator enables easy deploys of MongoDB.
- Prometheus Operator** (provided by Red Hat): The Prometheus Operator for Kubernetes provides easy monitoring definitions for your application metrics.
- Strimzi Kafka** (provided by Red Hat): Run an Apache Kafka cluster, including Kafka Connect, ZooKeeper and more.

# Operator Development

- Operator can be implemented in any language
- Frameworks:
  - Operator Framework (Golang, Helm, Ansible)  
<https://github.com/operator-framework>
  - Kubebuilder (Golang)  
<https://github.com/kubernetes-sigs/kubebuilder>
  - Metacontroller (Language agnostic)  
<https://metacontroller.app/>
  - jvm-operators (Java, Groovy, Kotlin, ...)  
<https://github.com/jvm-operators>

# Thank you



<https://k8spatterns.io>



@ro14nd



@bibryam



@k8spatterns

# Picture Credits

<https://www.pexels.com/photo/brown-and-black-pattern-2158386/>

<https://pixabay.com/photos/ship-helm-sunset-cutter-coast-guard-759954/>

<https://unsplash.com/photos/y001Z-9HQAw>

<https://www.pexels.com/photo/turned-on-light-crane-1117211/>

<https://www.pexels.com/photo/shallow-focus-photography-of-black-ship-1095814/>

<https://pixabay.com/photos/containers-storage-rusted-rusty-1209079/>

<https://pixabay.com/photos/motocross-sidecar-race-motorsport-1045661/>

<https://www.pexels.com/photo/golden-gate-bridge-san-francisco-california-1141853/>

[https://unsplash.com/photos/M\\_I-crkDO-k](https://unsplash.com/photos/M_I-crkDO-k)

<https://www.pexels.com/photo/reflection-playstation-pad-gaming-18174/>

[https://unsplash.com/photos/UJP\\_QpCKj7M](https://unsplash.com/photos/UJP_QpCKj7M)

<https://www.pexels.com/photo/grayscale-photo-of-person-holding-chess-piece-1498958/>

<https://pixabay.com/photos/cans-manufacturing-business-2888650/>

<https://unsplash.com/photos/lRoX0shwjUQ>

<https://www.freeimages.com/photo/poppy-in-wheat-1344010>

<https://pixabay.com/photos/telescope-field-glass-spyglass-1966366/>

<https://unsplash.com/photos/UHDx3BHIFvY>

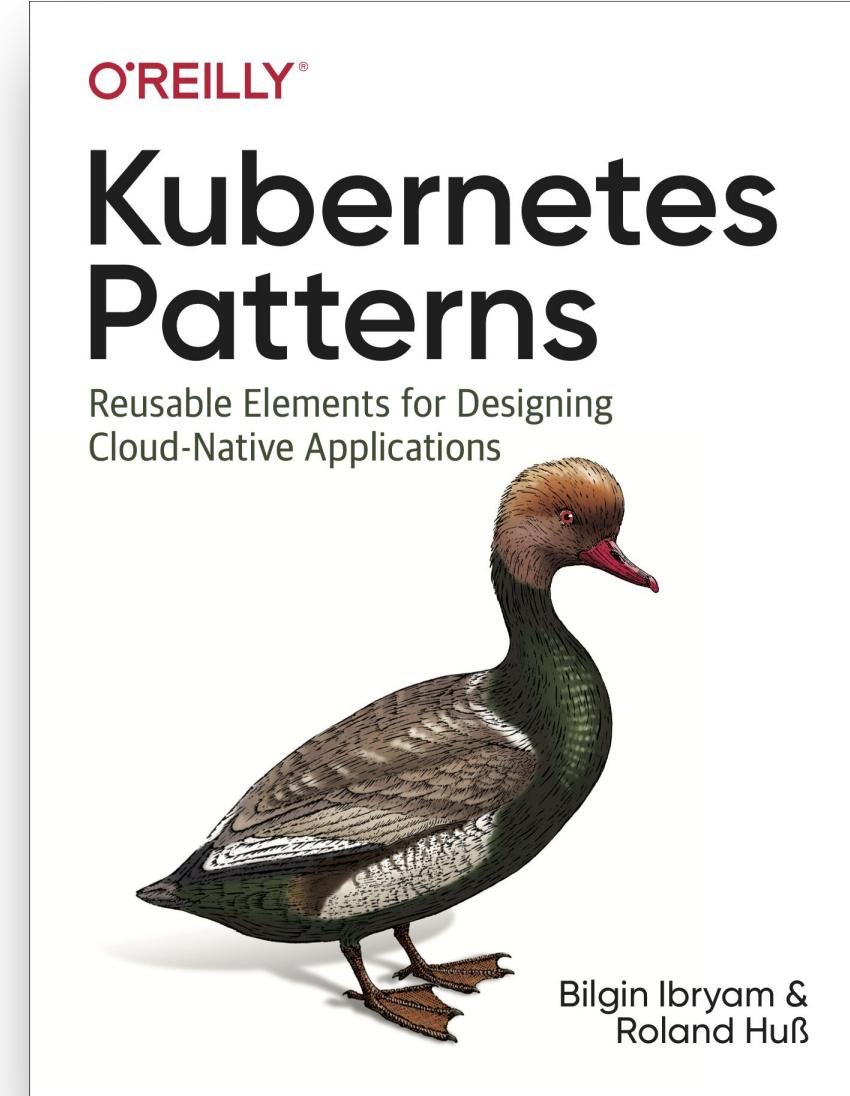
<https://pixabay.com/photos/bake-advent-christmas-cookie-1786926/>

<https://pixabay.com/photos/balloons-colors-party-celebration-1869790/>

[https://unsplash.com/photos/ymf4\\_9Y9S\\_A](https://unsplash.com/photos/ymf4_9Y9S_A)

<https://unsplash.com/photos/ZUabNmumOcA>

<https://pixabay.com/photos/lost-places-machines-old-factory-3991951/>



<https://k8spatterns.io>