

Docker & Certified Kubernetes Administrator (CKA)

Concepts & Architecture | Storage | Networking | HA & Clusters | Scheduling
Administration | Docker Compose | Security | Troubleshooting

16 Module | 60+ Lessons | 33 Hands-On Labs | Exam Preparation | On-Job Support





docker

+



kubernetes

ReplicaSet

Introduction

- Purpose is to maintain a stable set of replica Pods running at any given time. (Self Healing)
- Used to guarantee the availability of a specified number of identical Pods.
- If there is increased load, you can easily add more of the same pod to deal with the load (Scaling)

How a ReplicaSet works?

- A ReplicaSet is defined with:
 - Selector that specifies how to identify Pods it can acquire
 - A number of replicas indicating how many Pods it should be maintaining
 - A pod template specifying the data of new Pods.
- A ReplicaSet then fulfills its purpose by creating/deleting Pods as needed to reach the desired number.

How a ReplicaSet works?..

- It's all about the state
 - Desired state
 - Current state (also known as actual state or observed state)
 - Declarative model
- Fundamental to desired state is the concept of background reconciliation loops (a.k.a. control loops/watch loops).
- Kubernetes is constantly making sure that current state matches desired state

Example:

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      app: web-app
  template:
    metadata:
      labels:
        app: web-app
    spec:
      containers:
        - name: web
          image: nginx
```

Pod Selector

- The .spec.selector field is a label selector.
- These are the labels used to identify potential Pods to acquire.
- The selector was:
 - matchLabels:
 - app: web-app

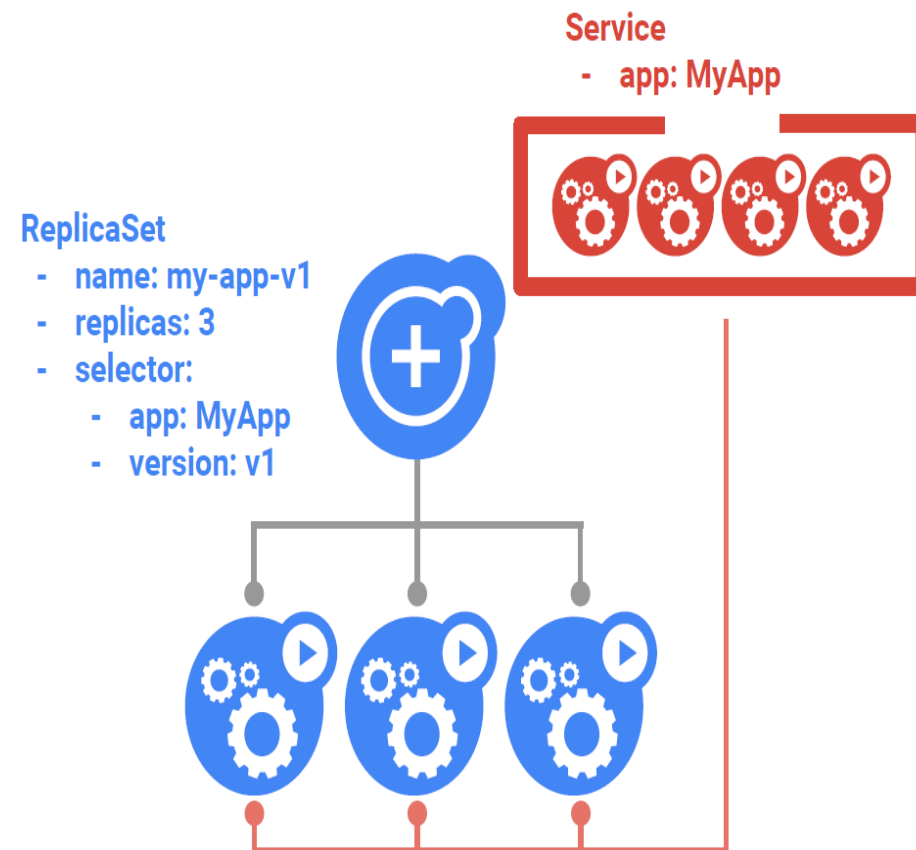
Replicas

- Can specify how many Pods should run concurrently by setting `.spec.replicas`.
- The ReplicaSet will create/delete its Pods to match this number.
- If you do not specify `.spec.replicas`, then it defaults to 1.

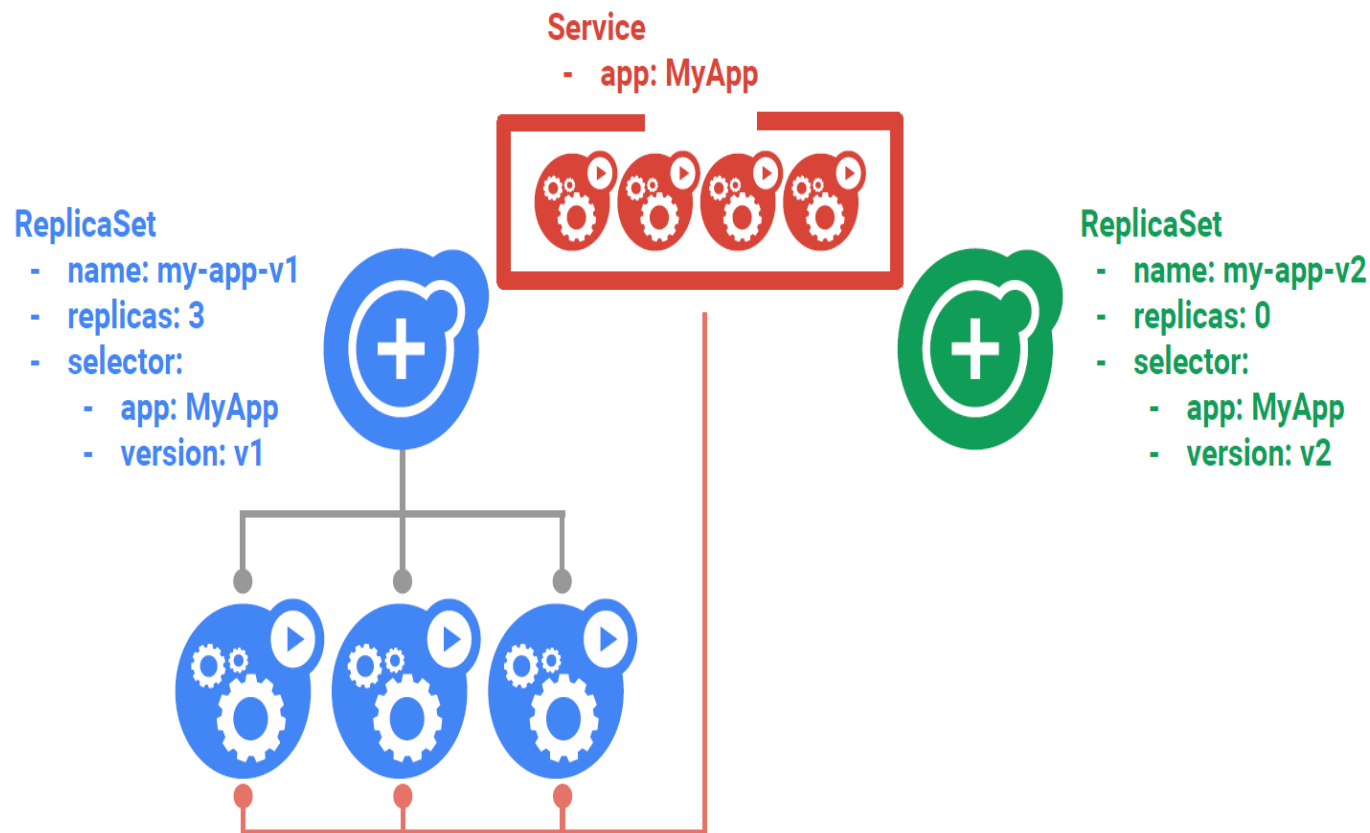
```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      app: web-app
  template:
    metadata:
      labels:
        app: web-app
    spec:
      containers:
        - name: web
          image: nginx
```


ReplicaSet

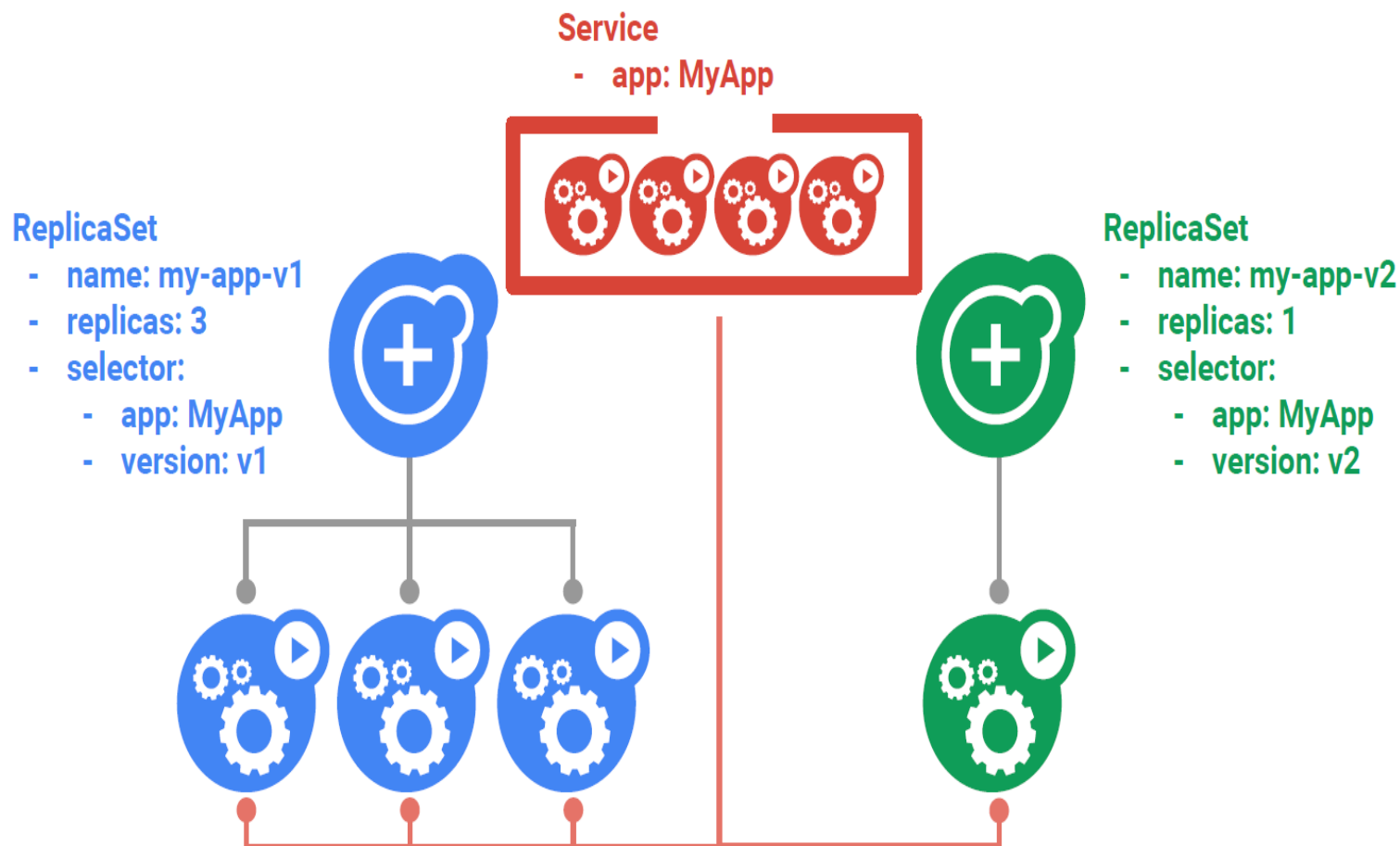
- Features of ReplicaSet:
 - Self-healing
 - Manual Scalability
 - Manual Zero downtime Rolling updates
 - Manual Zero downtime Roll Backs



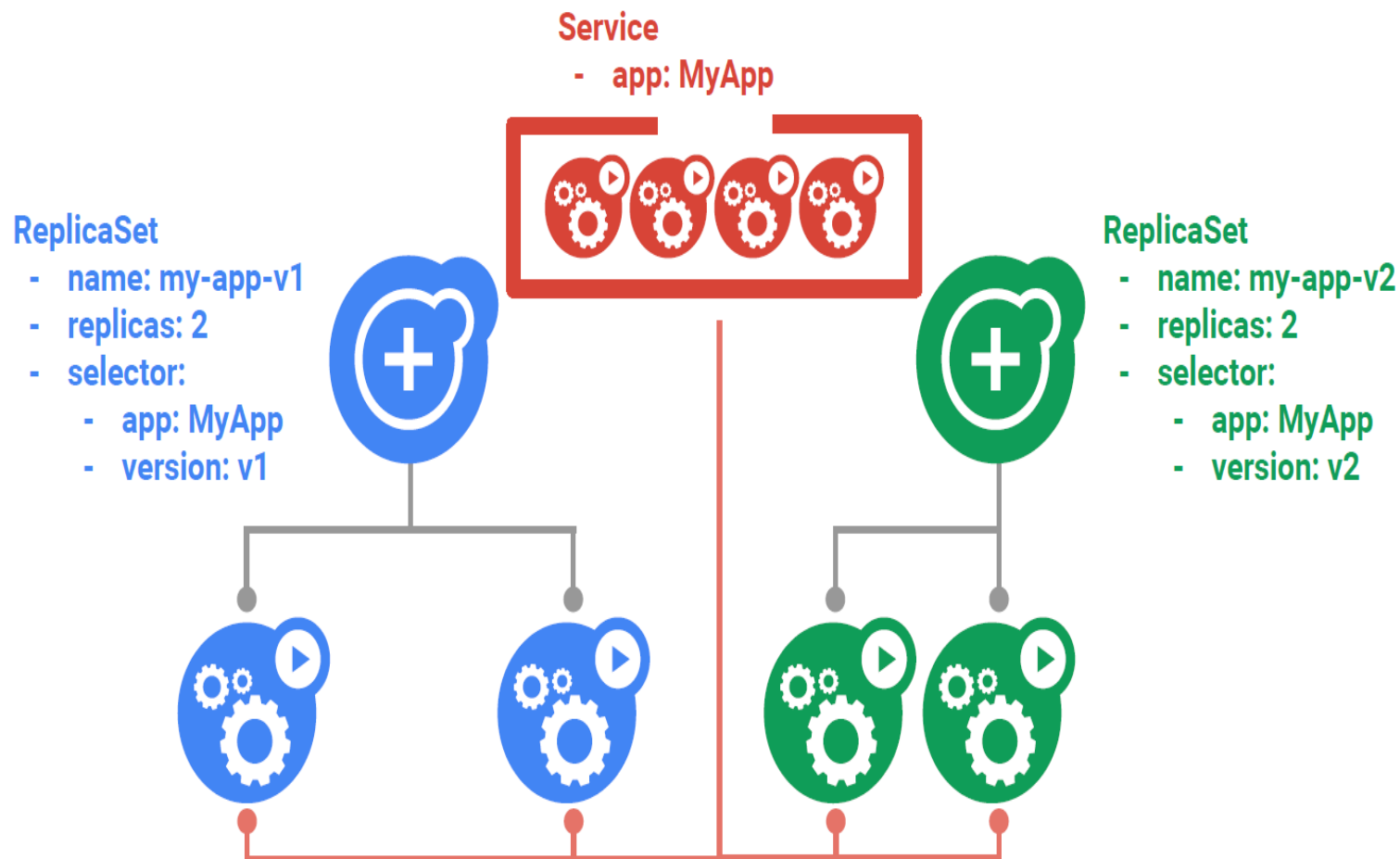
Replica Set – Keep Track of Pod Replicas



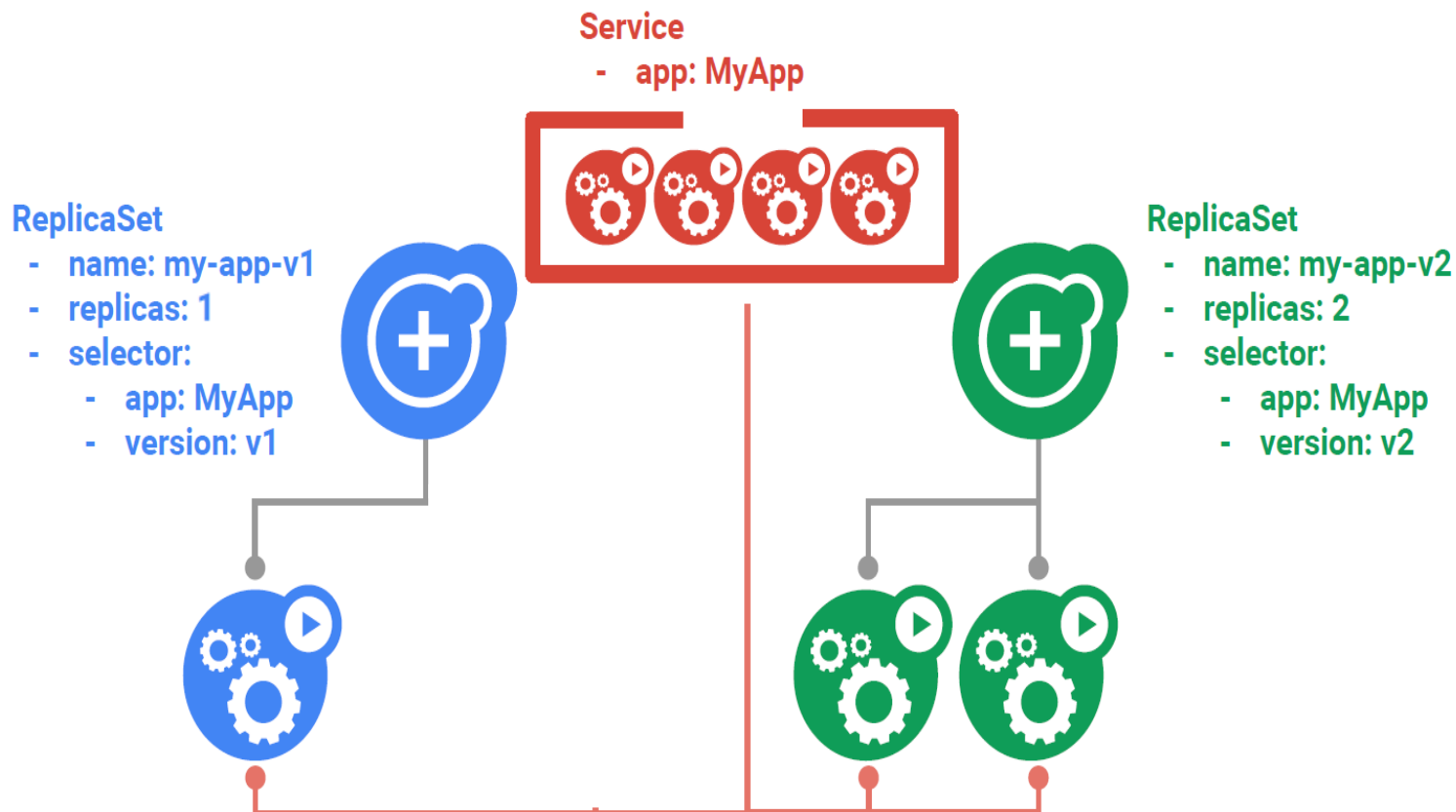
Replica Set – Keep Track of Pod Replicas..



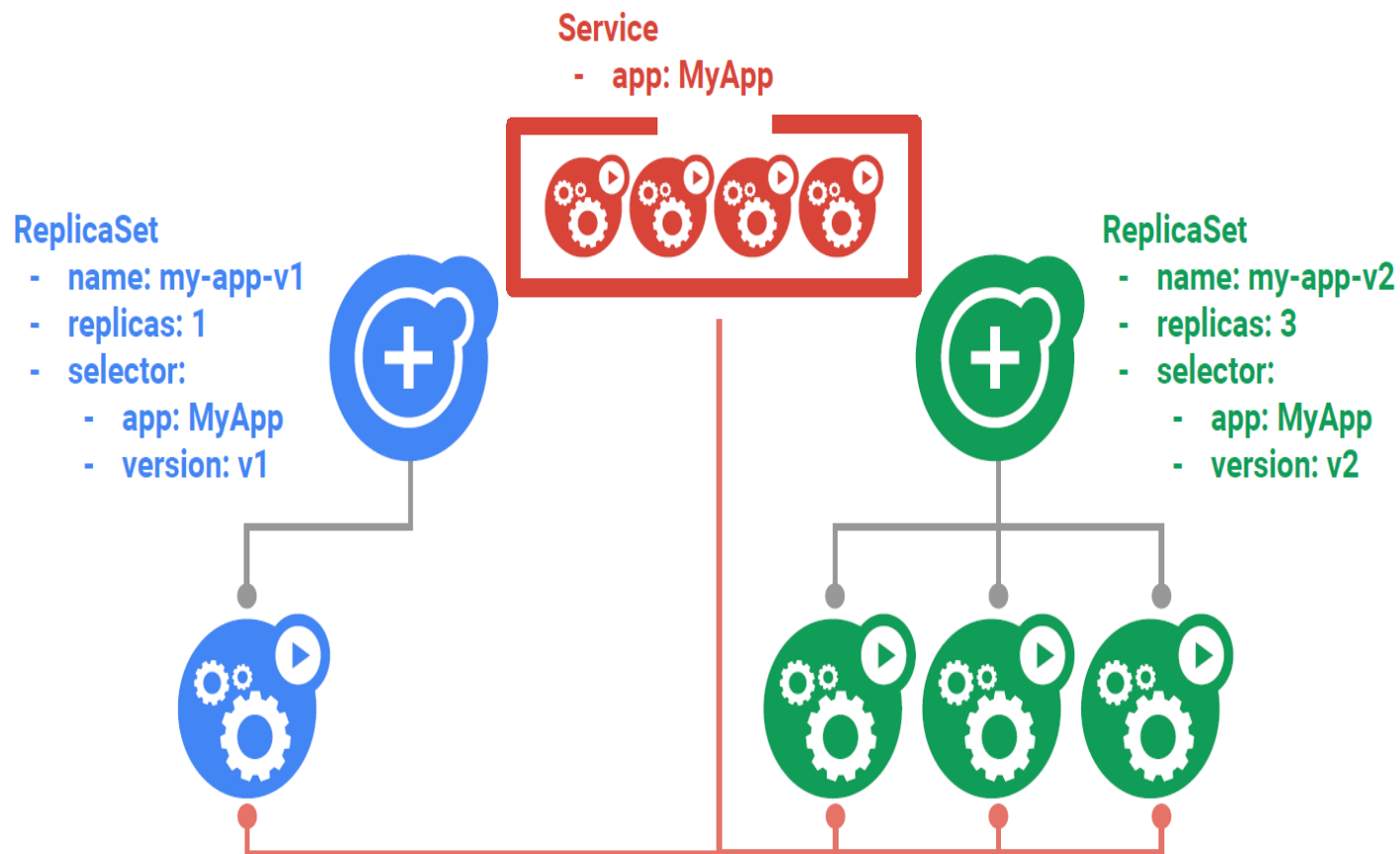
Replica Set – Keep Track of Pod Replicas..



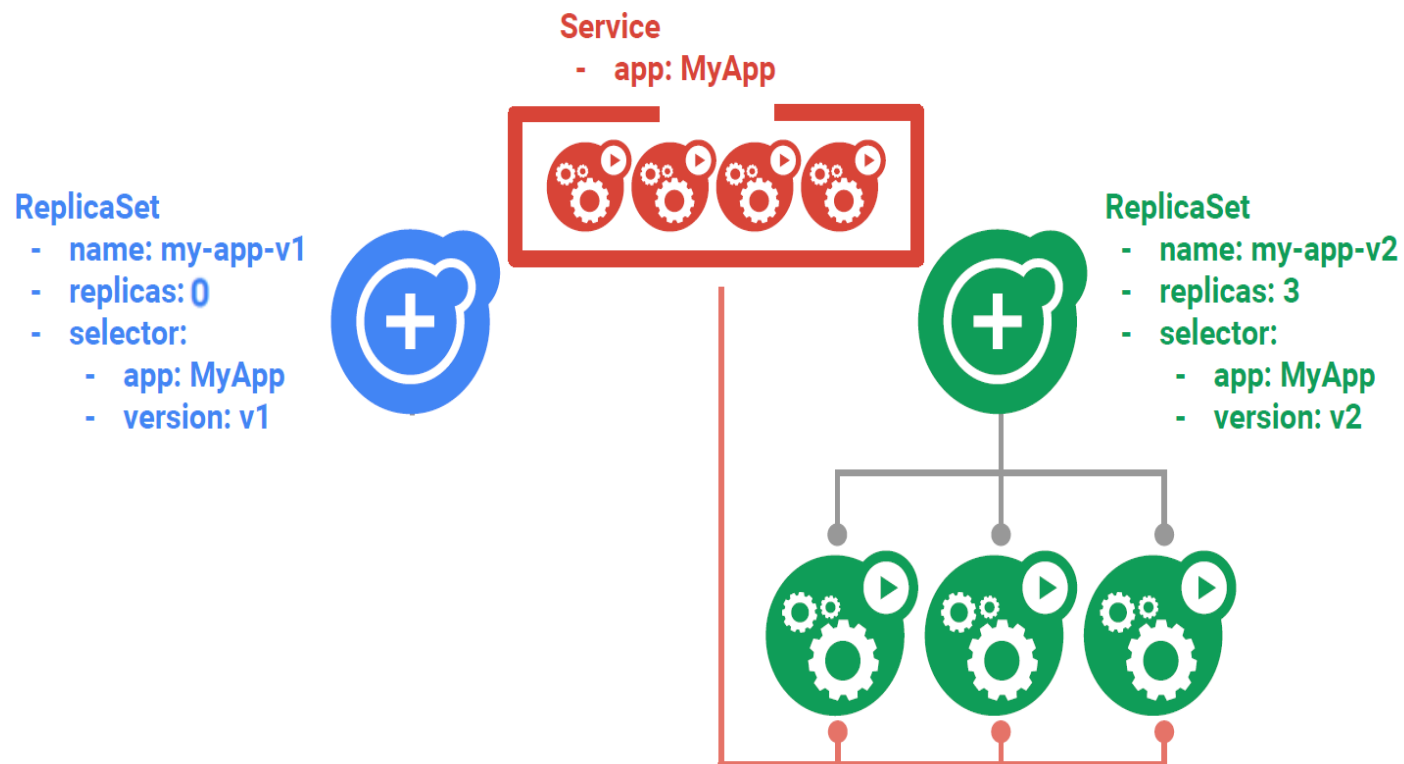
Replica Set – Keep Track of Pod Replicas..



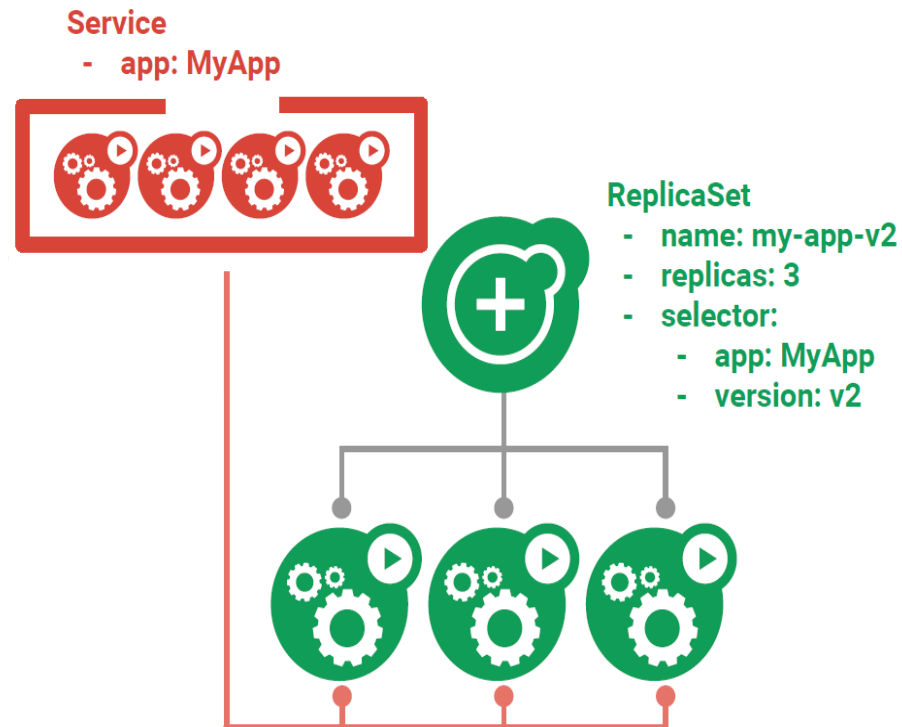
Replica Set – Keep Track of Pod Replicas..



Replica Set – Keep Track of Pod Replicas..



Replica Set – Keep Track of Pod Replicas..



Non-Template Pod Acquisitions

- ReplicaSet is not limited to owning Pods specified by its template-- it can acquire other Pods.

```
apiVersion: v1
kind: Pod
metadata:
  name: pod1
  labels:
    app: web-app
spec:
  containers:
  - name: hello1
    image: gcr.io/google-samples/hello-app:1.0
---
apiVersion: v1
kind: Pod
metadata:
  name: pod2
  labels:
    app: web-app
spec:
  containers:
  - name: hello2
    image: gcr.io/google-samples/hello-app:2.0
```



docker

+

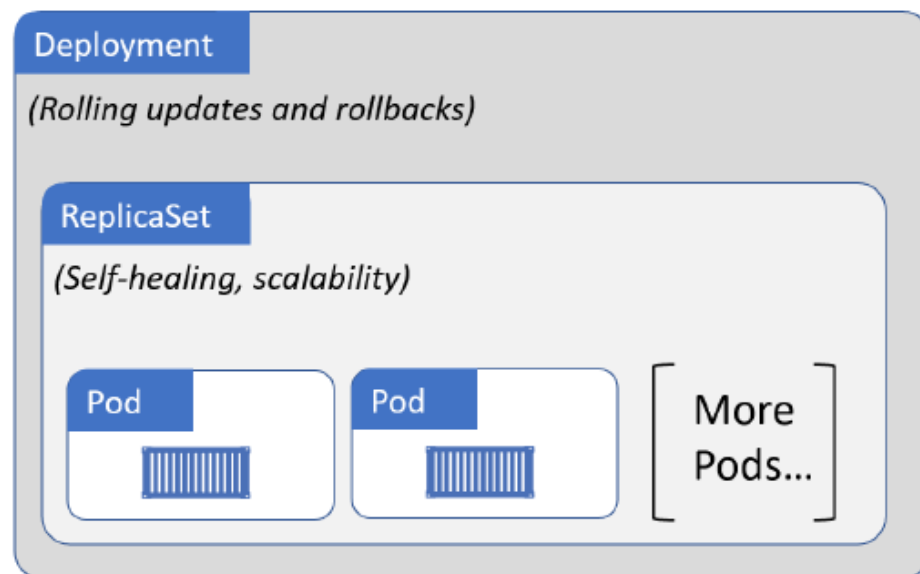


kubernetes

Deployments

Introduction

- A deployment allows you to describe an application's life cycle, such as which images to use for the app, the number of pods there should be, and the way in which they should be updated.
- A Deployment provides declarative updates for Pods and ReplicaSets.
- Deployment uses the ReplicaSet to control the desired state.



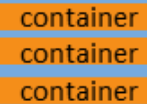
Creating a Deployment

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx
```

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx
```

Deployment

```
replicas: 3  
template:Pod
```

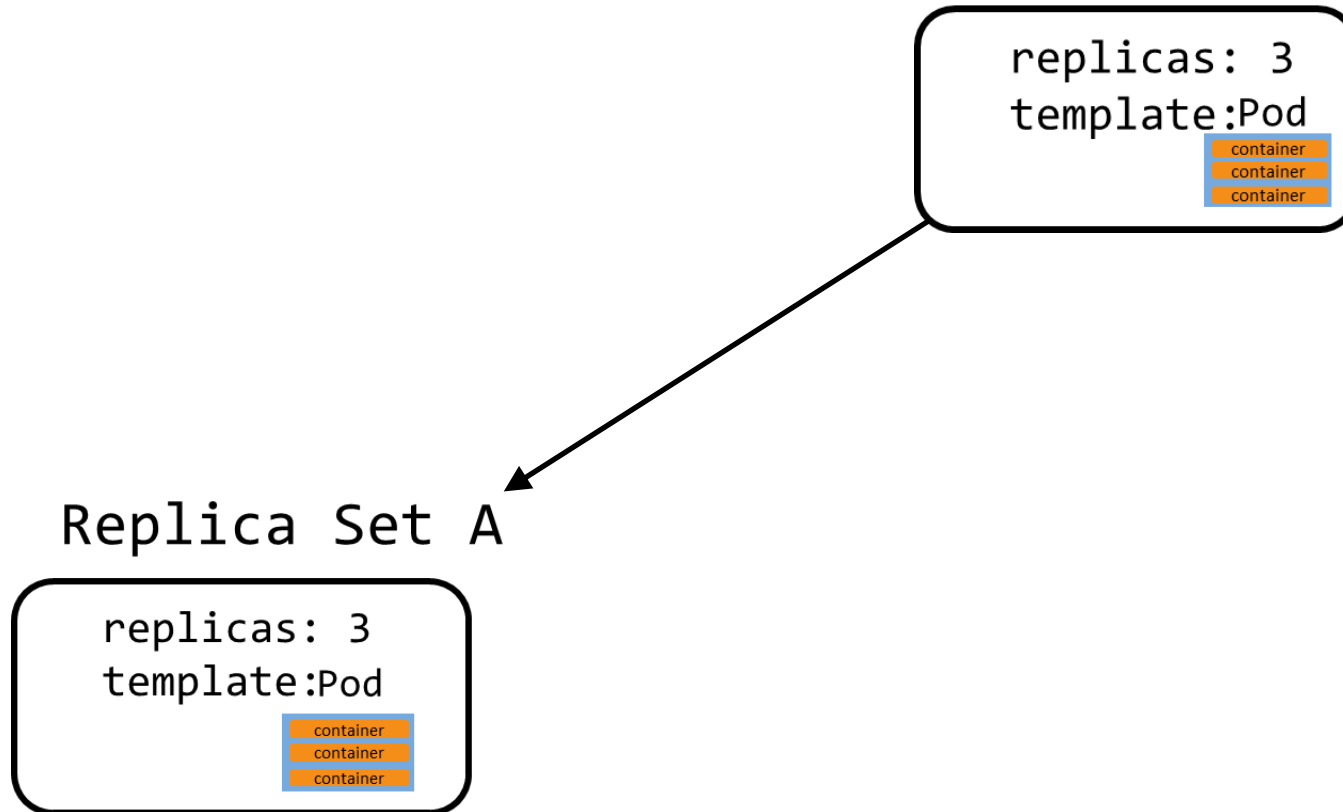


A diagram showing a Pod, represented by a blue-outlined rounded rectangle, containing three stacked orange rectangles, each labeled 'container'.

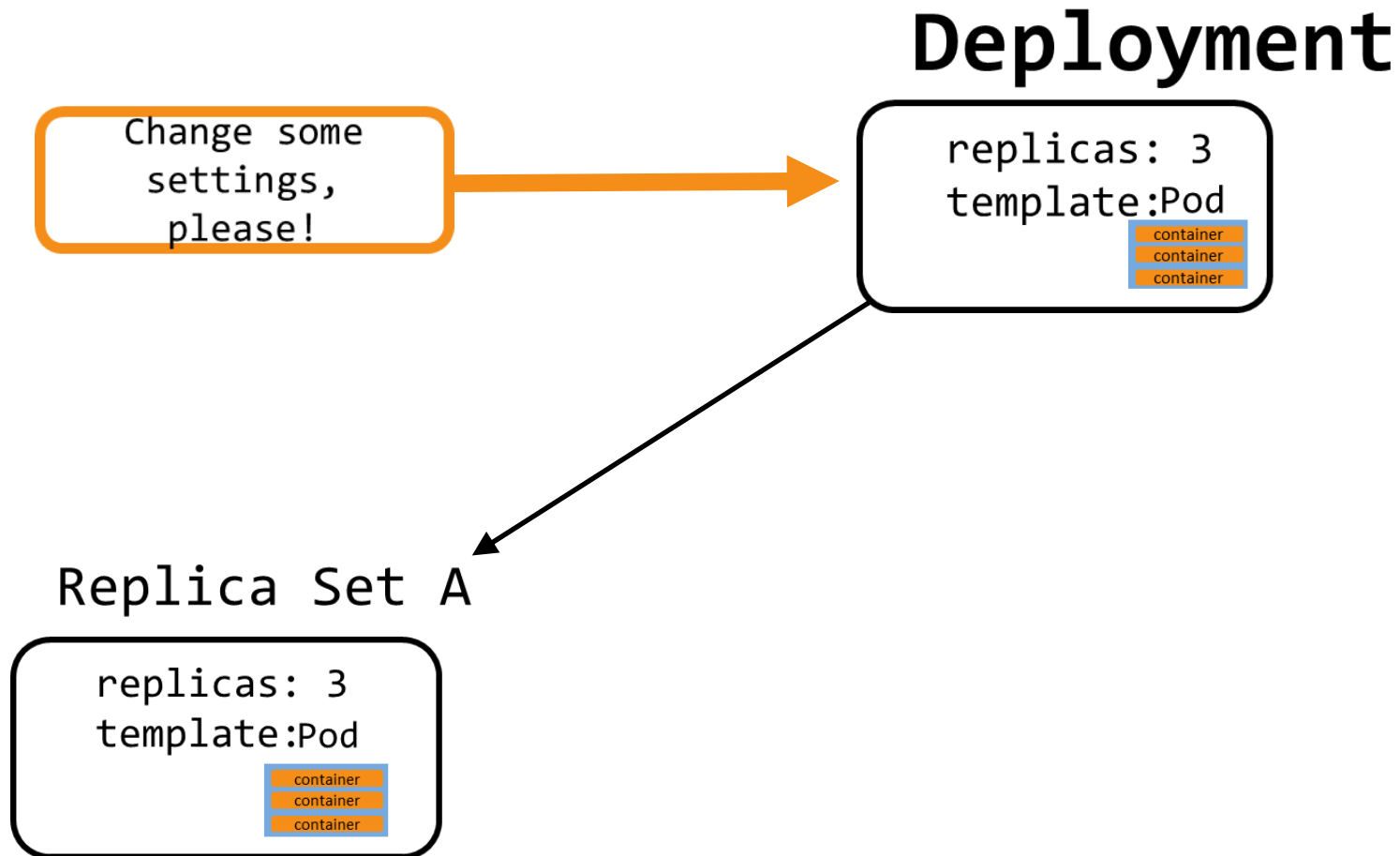
container
container
container

Deployment..

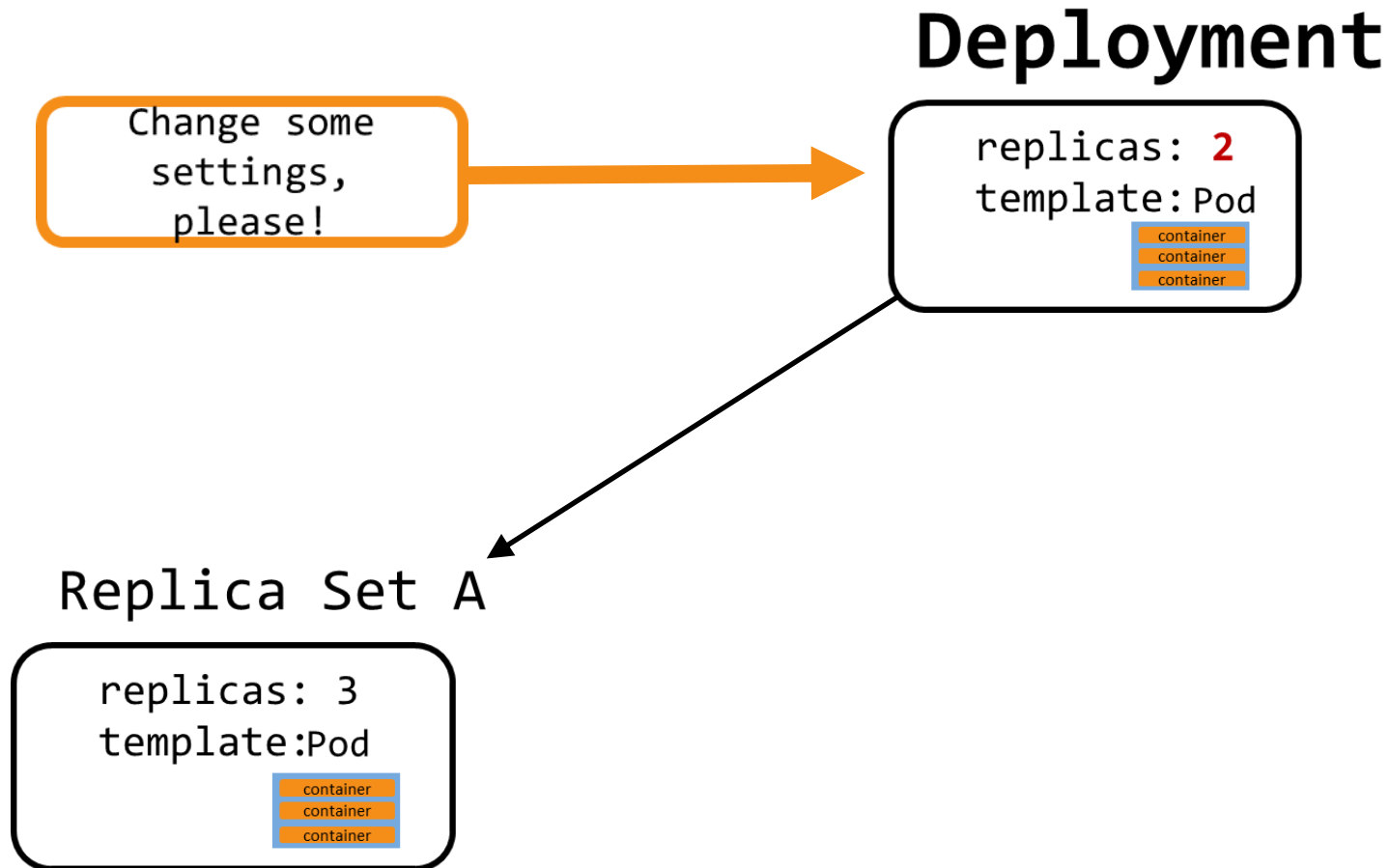
Deployment



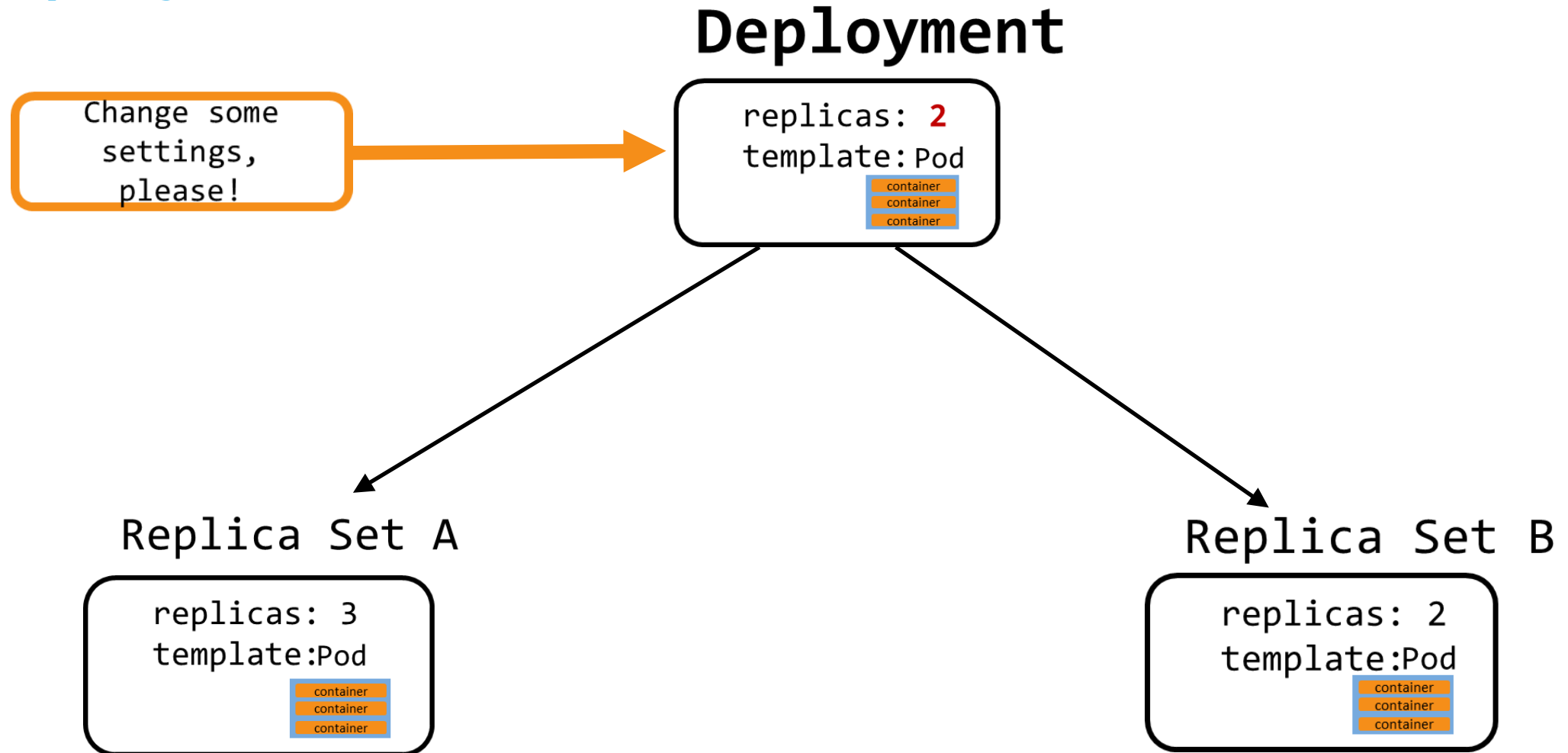
Deployment..



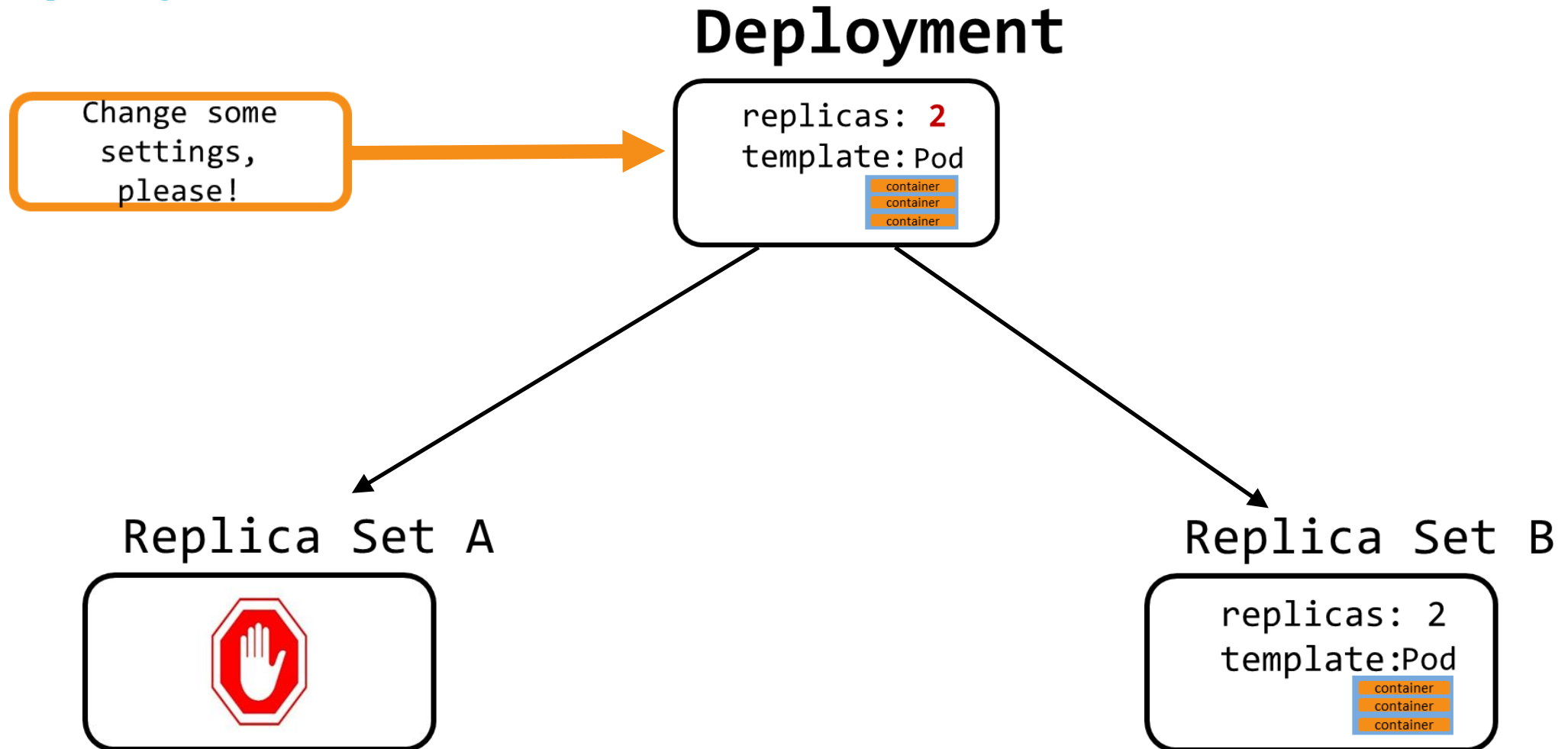
Deployment..



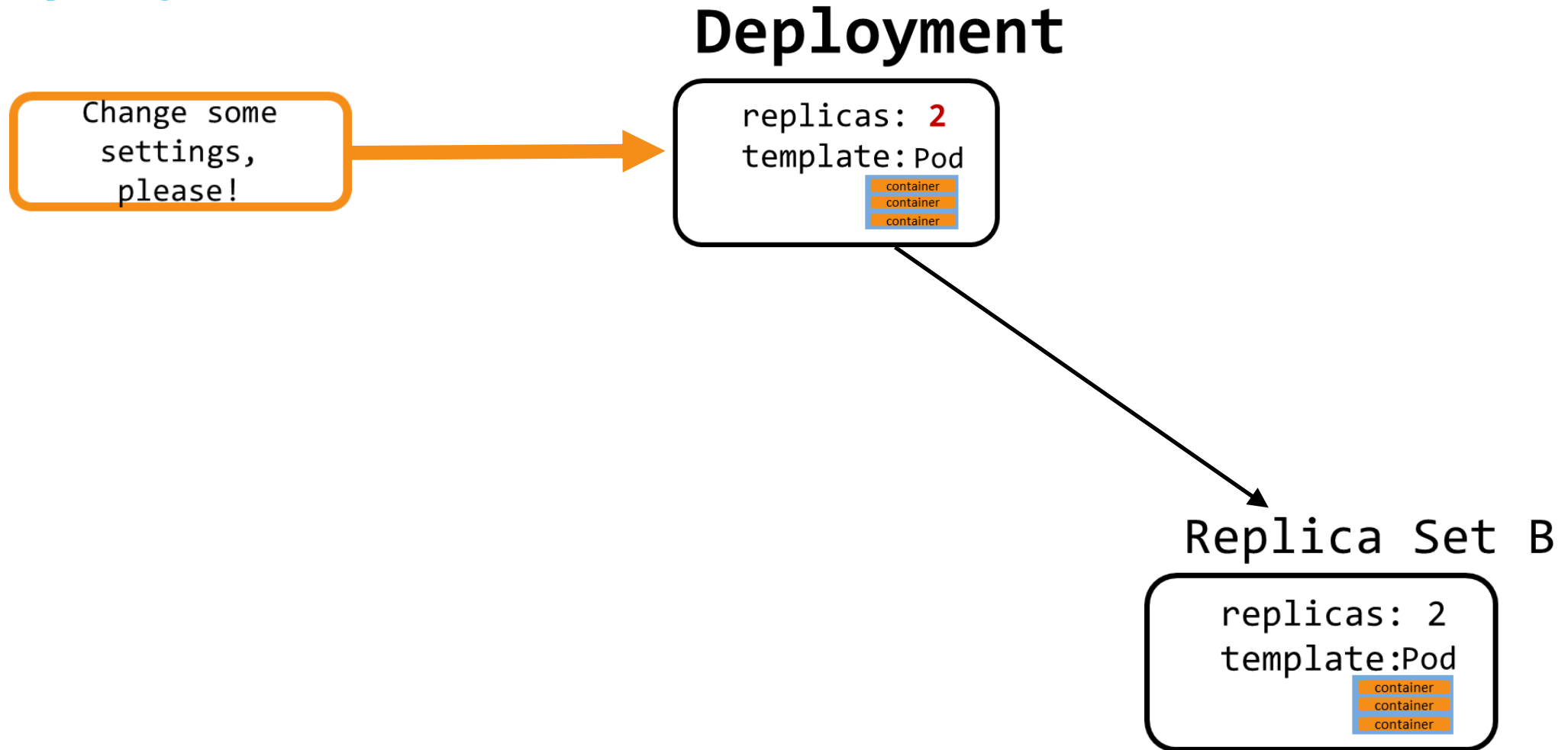
Deployment..



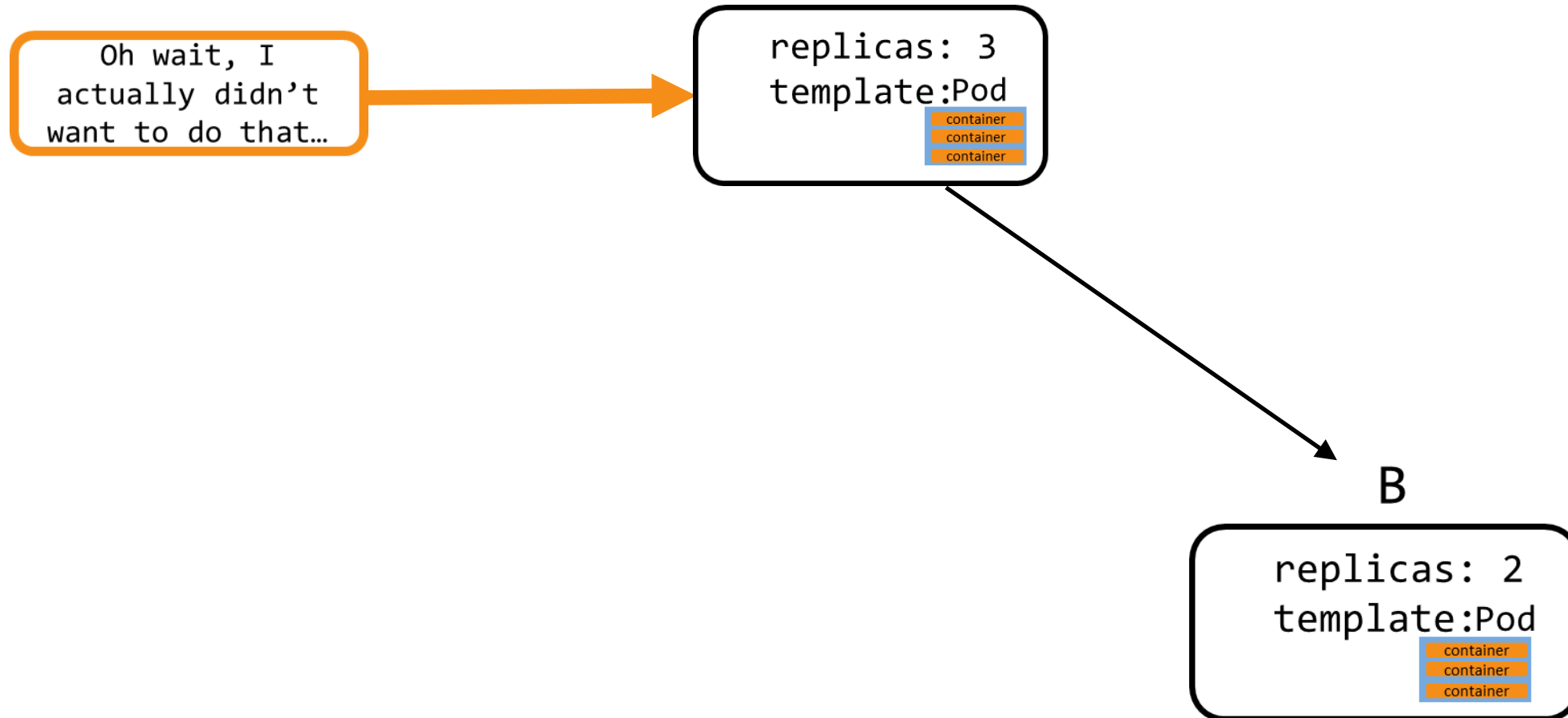
Deployment..



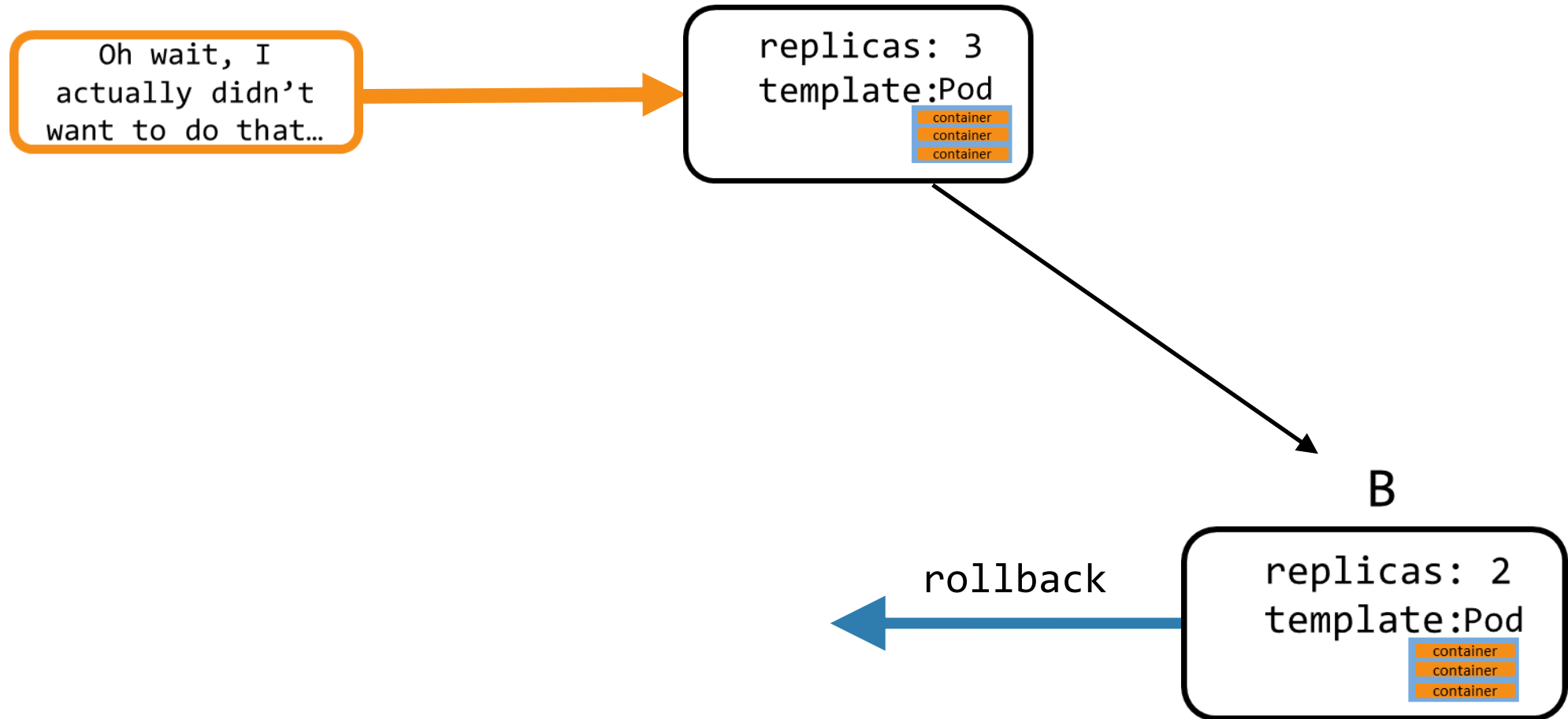
Deployment..



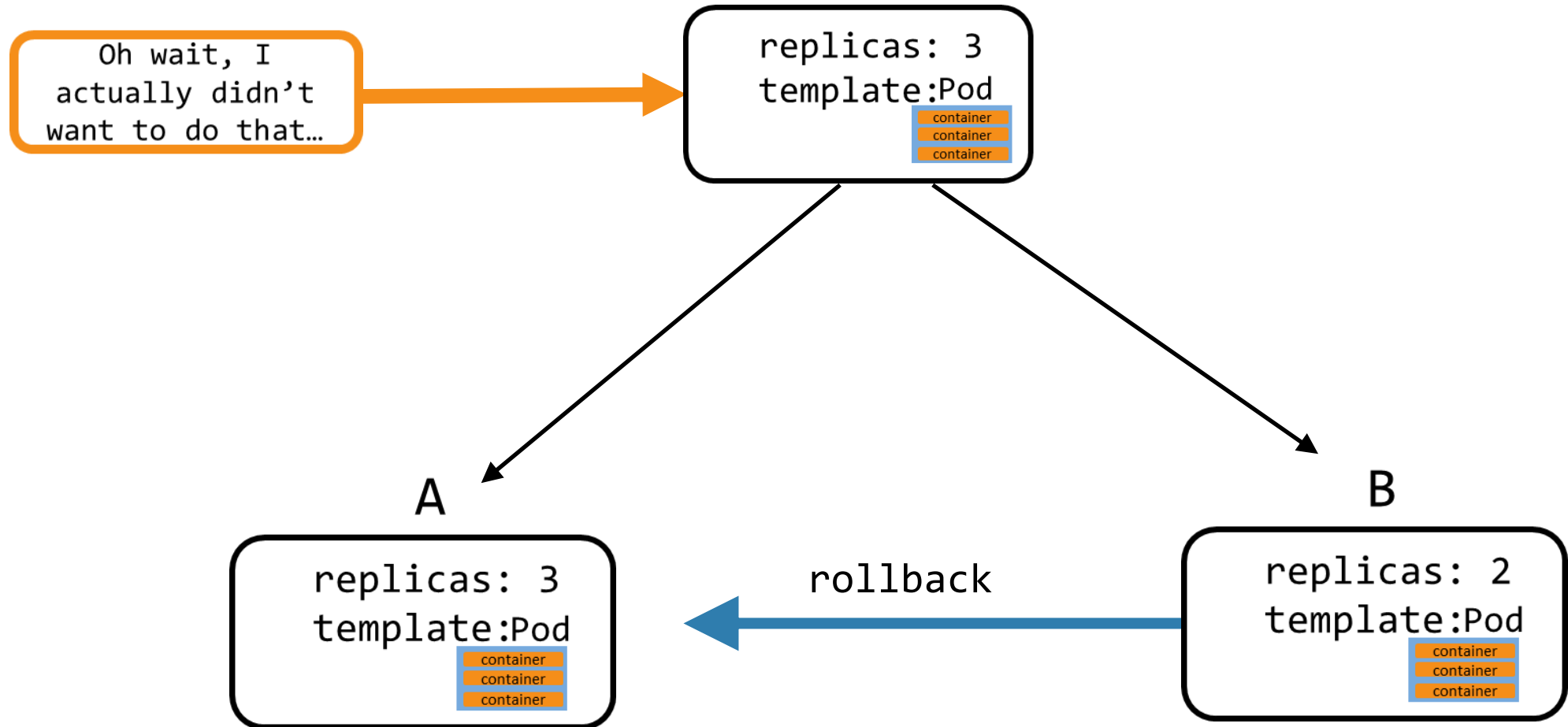
Deployment..



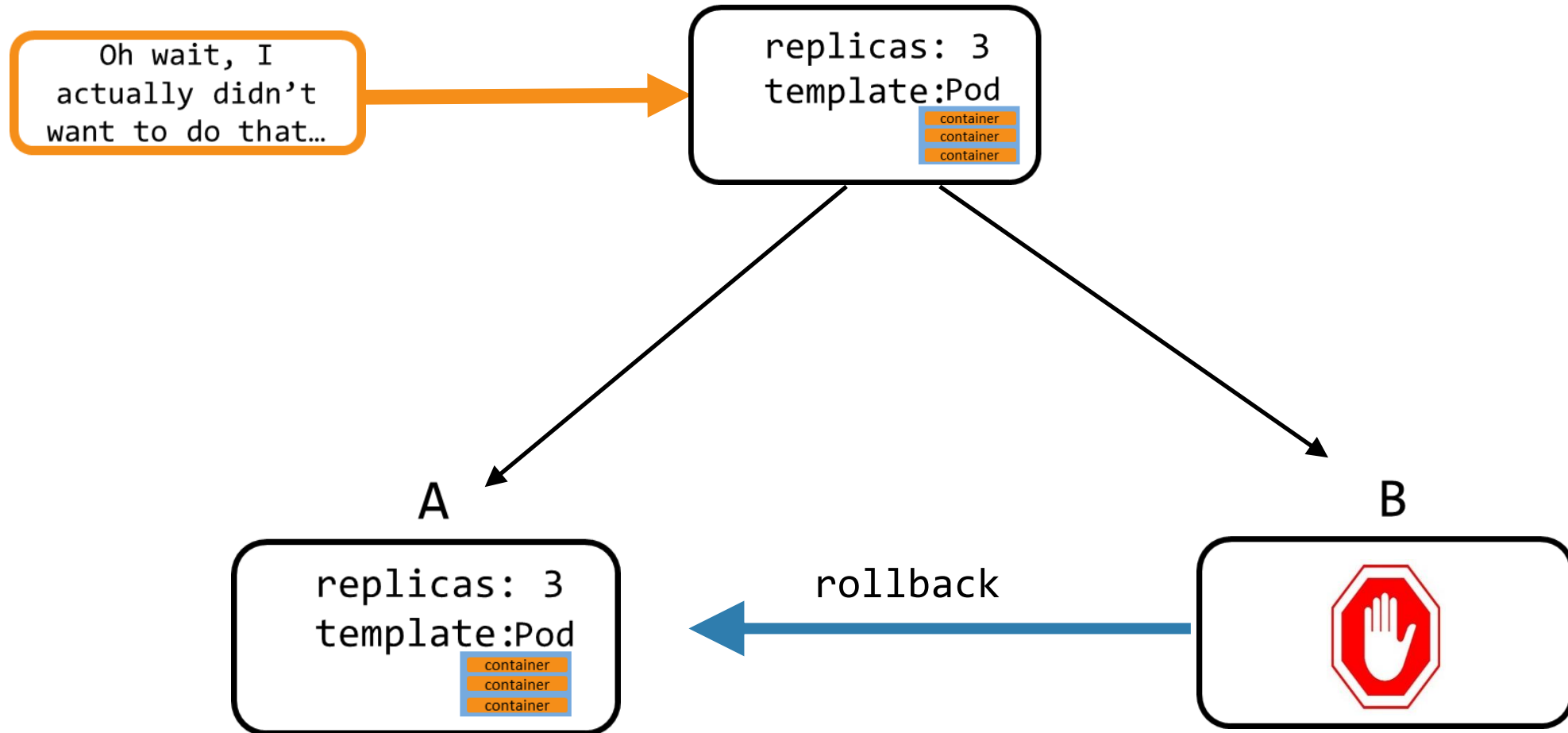
Deployment..



Deployment..



Deployment..



Deployment..

Deployment

