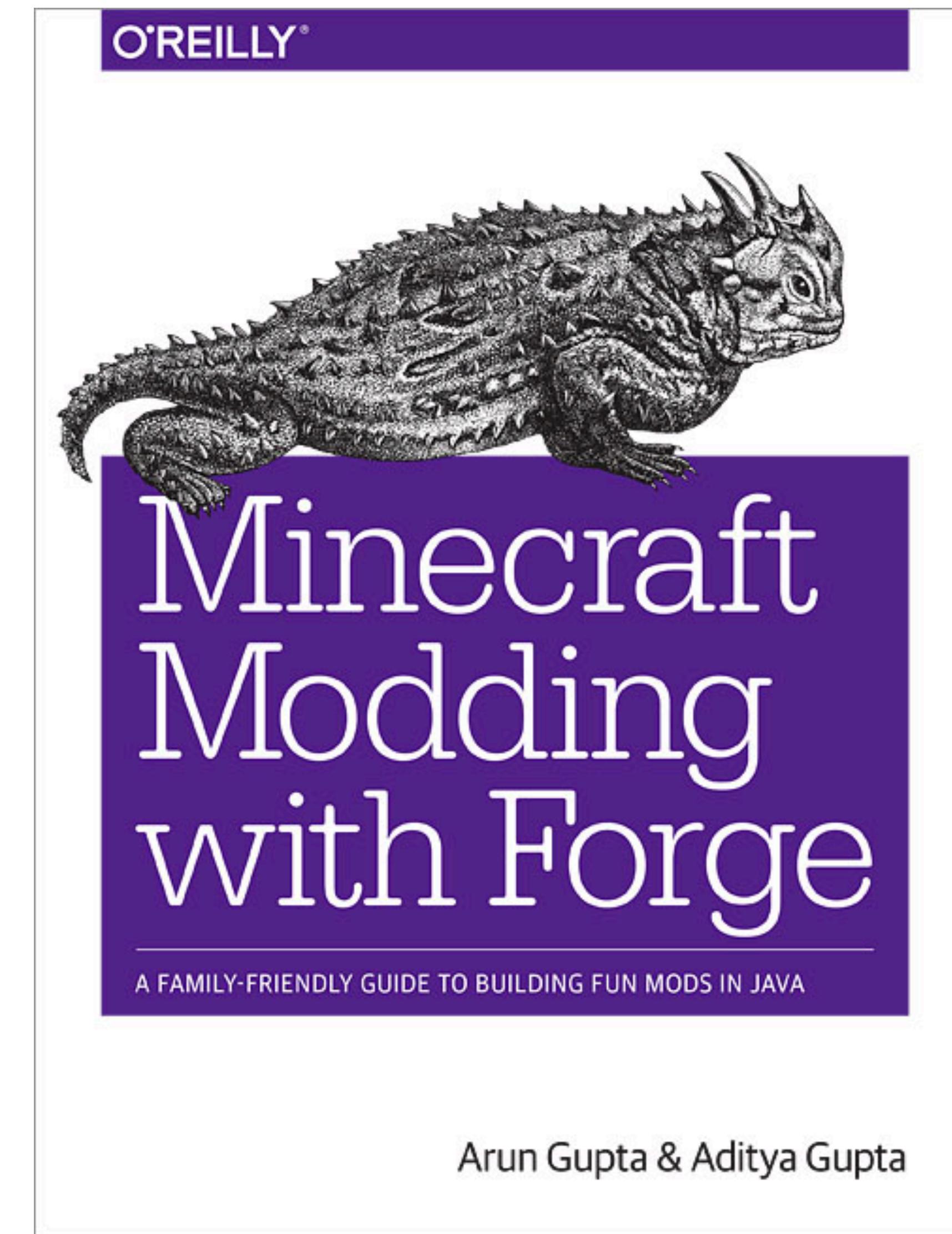




# Getting Started with Kubernetes

Arun Gupta, @arungupta  
VP Developer Advocacy, Couchbase



**CONTENTS**

- » What Is Kubernetes?
- » Kubernetes Architecture
- » Starting With Kubernetes
- » Run Your First Container
- » Scale Applications...and more!

Get More Refcardz! Visit [DZone.com/Refcardz](http://DZone.com/Refcardz)

**WHAT IS KUBERNETES?**

Kubernetes ([kubernetes.io](https://kubernetes.io)) is an open-source orchestration system for managing containerized applications across multiple hosts, providing basic mechanisms for the deployment, maintenance, and scaling of applications.

Kubernetes, or “k8s” or “kube” for short, allows the user to declaratively specify the desired state of a cluster using high-level primitives. For example, the user may specify that they want three instances of the Couchbase server container running. Kubernetes’ self-healing mechanisms, such as auto-restarting, re-scheduling, and replicating containers then converge the actual state towards the desired state.

Kubernetes supports Docker and Rocket containers. An abstraction around the containerization layer will allow for other container image formats and runtimes to be supported in the future.

**KEY CONCEPTS OF KUBERNETES****POD**

A Pod is the smallest deployable unit that can be created, scheduled, and managed. It’s a logical collection of containers that belong to an application.

Each resource in Kubernetes is defined using a configuration file. For example, a Couchbase pod can be defined with the following .yaml file:

A Replication Controller creating two instances of a Couchbase pod can be defined as:

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: couchbase-controller
spec:
  # Two replicas of the Pod to be created
  replicas: 2
  # Identifies the label key and value on the Pod that
  # this Replication Controller is responsible for
  # managing
  selector:
    app: couchbase-rc-pod
  # ‘cookie cutter’ used for creating new pods when
  # necessary
  template:
    metadata:
      labels:
        # label key and value on the pod.
        # These must match the selector above.
        app: couchbase-rc-pod
    spec:
      containers:
        - name: couchbase
          image: couchbase
          ports:
            - containerPort: 8091
```

**SERVICE**

Each Pod is assigned a unique IP address. If the Pod is inside a Replication Controller, then the pod is recreated but may be given

# Kubernetes

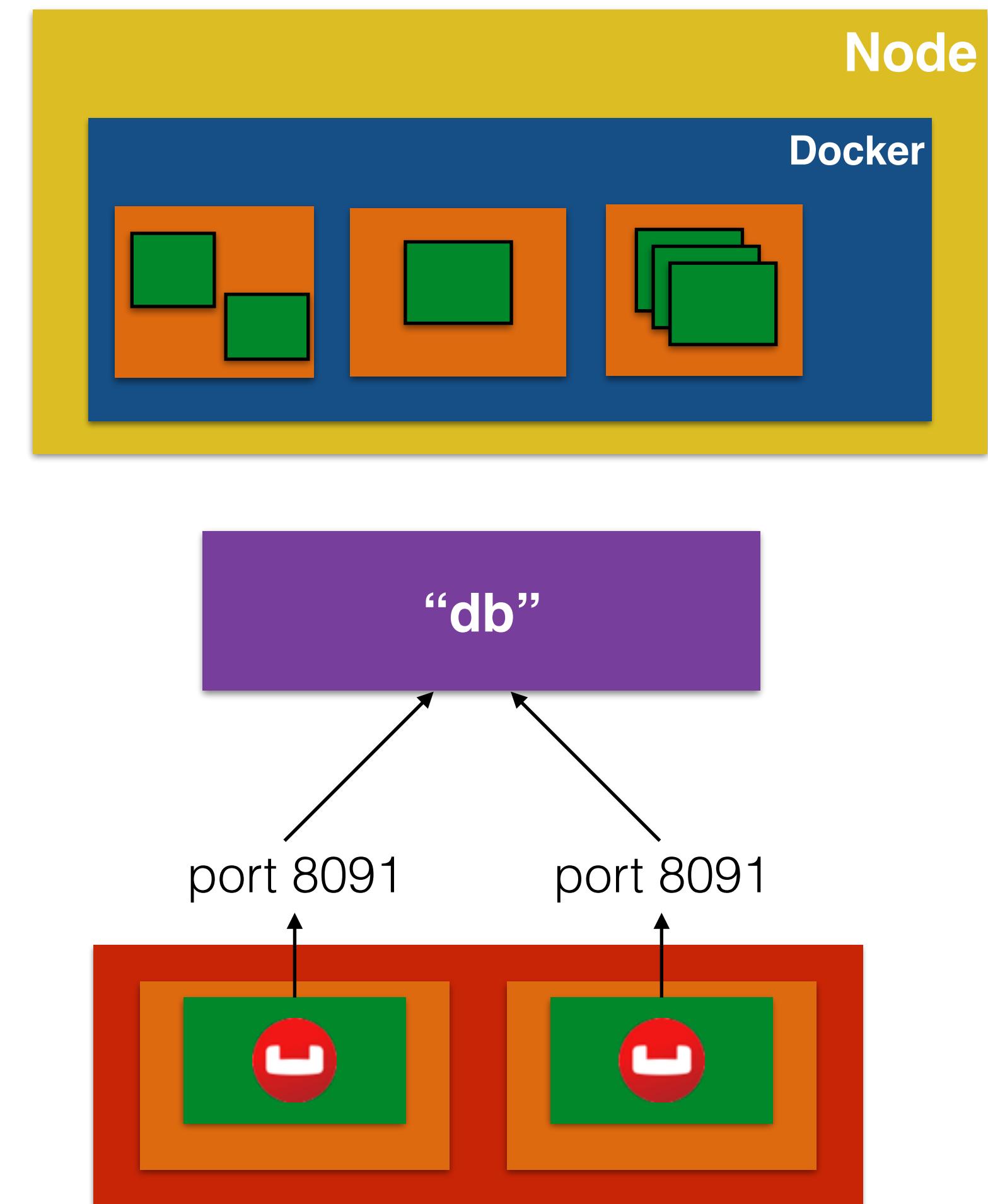
BY ARUN GUPTA

# Kubernetes

- Open source orchestration system for Docker containers
- Provide declarative primitives for the “desired state”
  - Self-healing
  - Auto-restarting
  - Schedule across hosts
  - Replicating

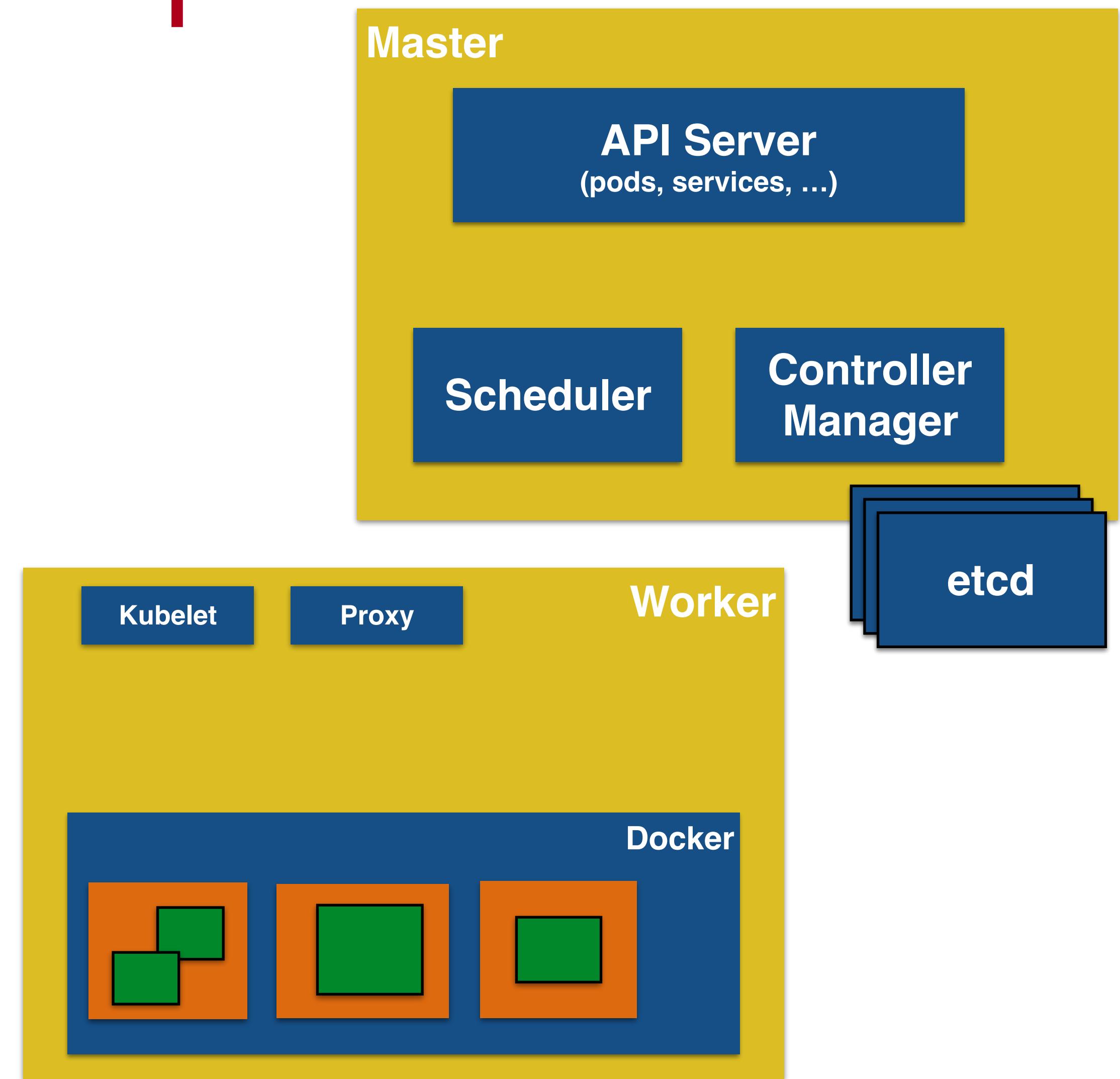
# Kubernetes Concepts

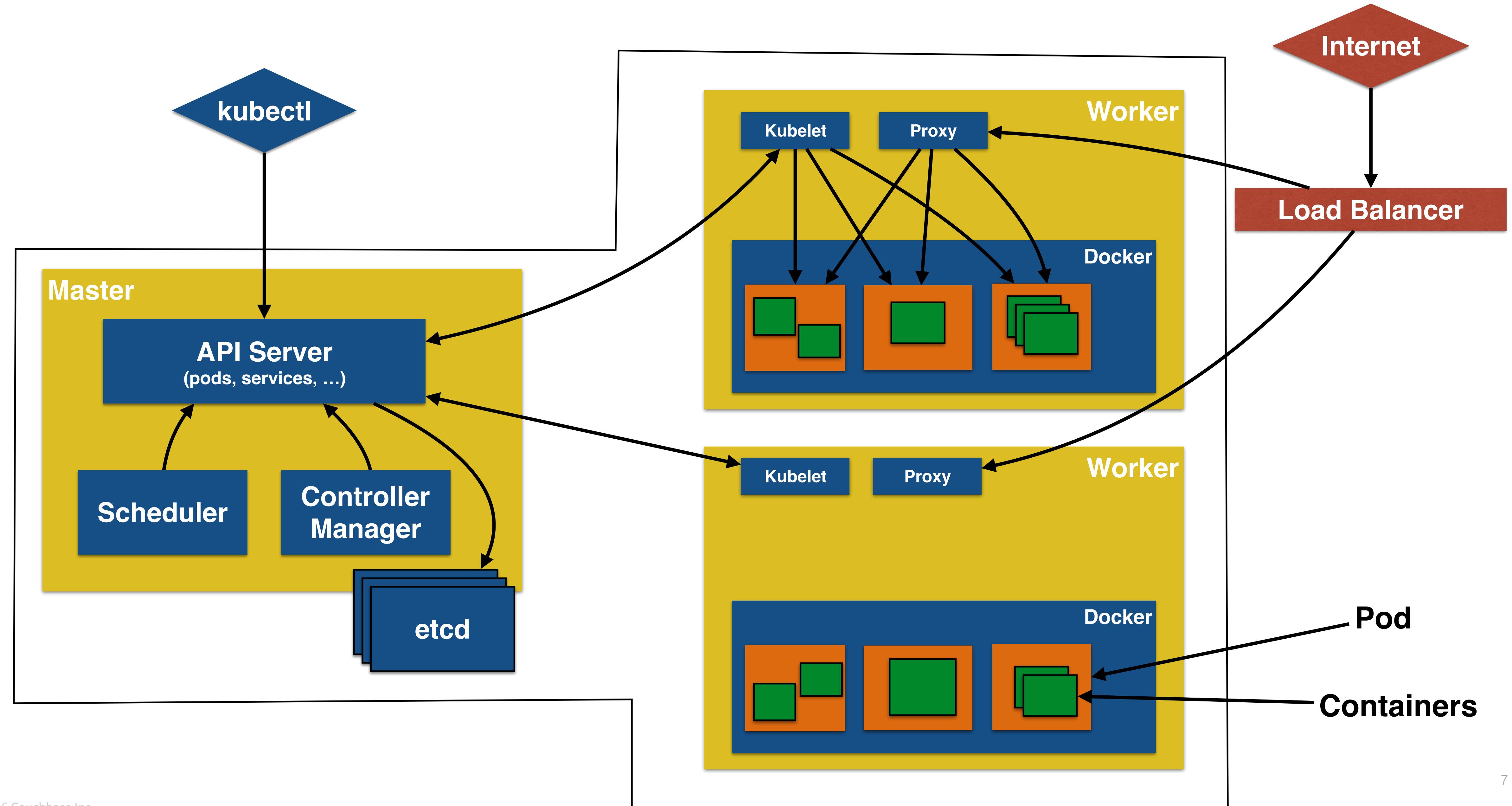
- **Pods**: colocated group of containers that share an IP, namespace, storage volume
- **Replica Set**: manages the lifecycle of pods and ensures specified number are running (next gen Replication Controller)
- **Service**: Single, stable name for a set of pods, also acts as LB
- **Label**: used to organize and select group of objects



# Kubernetes Components

- **Node**: Machine or VM in the cluster
- **Master**: Central control plane, provides unified view of the cluster
  - **etcd**: distributed key-value store used to persist Kubernetes system state
- **Worker**: Docker host running *kubelet* (node agent) and *proxy* services
  - Runs pods and containers
  - Monitored by *systemd* (CentOS) or *monit* (Debian)







# kubectl

- Controls the Kubernetes cluster manager
- `kubectl get pods or minions`
- `kubectl create -f <filename>`
- `kubectl update or delete`
- `kubectl scale --replicas=3 rc/<name>`

# Kubernetes Pod Configuration

```
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: wildfly-pod
5    labels:
6      name: wildfly-pod
7  spec:
8    containers:
9      - name: wildfly
10     image: jboss/wildfly
11     ports:
12       - containerPort: 8080
```

# Replication Controller

- Ensures that a specified number of pod "replicas" are running
  - Pod templates are cookie cutters
  - Rescheduling
  - Manual or auto-scale replicas
  - Rolling updates
- Generally wrap a pod in a RC
- Only appropriate for pods with `Restart=Always` policy (default)

# Kubernetes Replication Controller Configuration

```
1  apiVersion: v1
2  kind: ReplicationController
3  metadata:
4    name: wildfly-rc
5  spec:
6    replicas: 2
7    selector:
8      app: wildfly-rc-pod
9    template:
10      metadata:
11        labels:
12          app: wildfly-rc-pod
13      spec:
14        containers:
15          - name: wildfly
16            image: jboss/wildfly
17        ports:
18          - containerPort: 8080
```

# Services

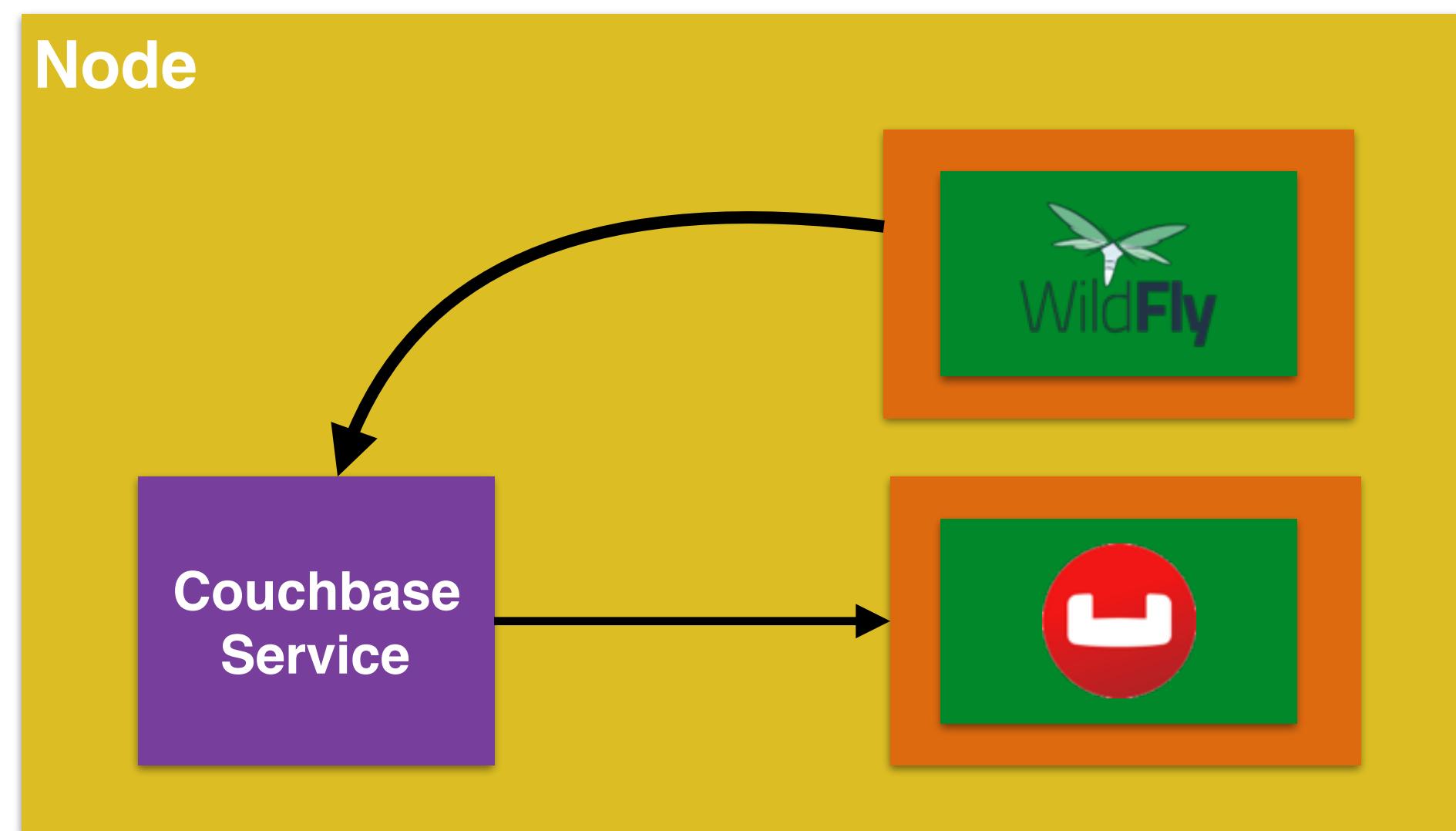
- Abstract a set of pods as a single IP and port
  - Simple TCP/UDP load balancing
- Creates environment variables in other pods or DNS resolution
- Stable endpoint for pods to reference
  - Allows list of pods to change dynamically

# Kubernetes Service Configuration

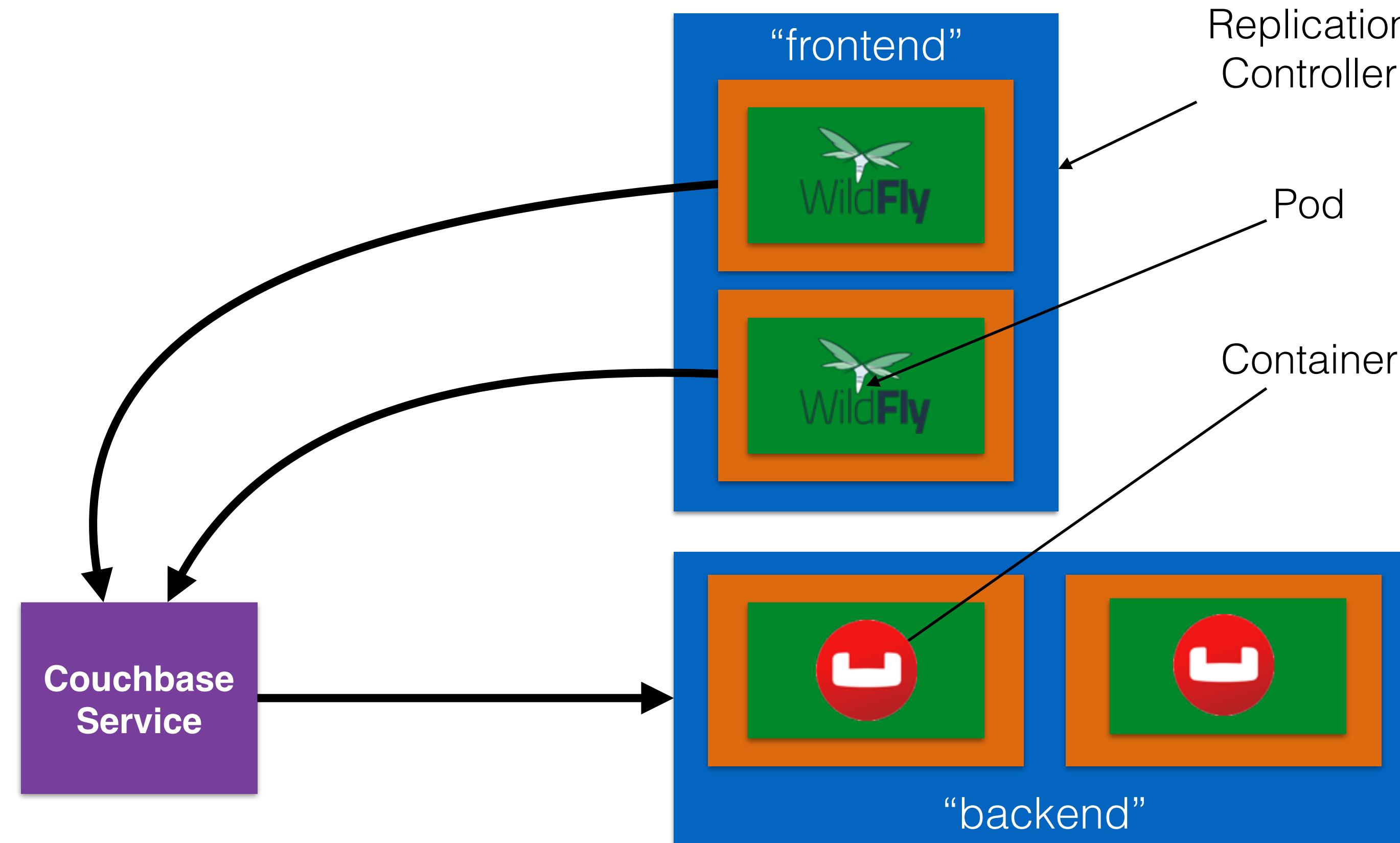
```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: couchbase-service
5 spec:
6   selector:
7     app: couchbase-rc-pod
8   ports:
9     - name: admin
10    port: 8091
11    - name: query
12      port: 8093
13 ---
```

```
14 apiVersion: v1
15 kind: ReplicationController
16 metadata:
17   name: couchbase-rc
18 spec:
19   replicas: 2
20   selector:
21     app: couchbase-rc-pod
22   template:
23     metadata:
24       labels:
25         app: couchbase-rc-pod
26   spec:
27     containers:
28       - name: couchbase
29         image: couchbase
30     ports:
31       - containerPort: 8091
```

# Couchbase Service



# Service and Replication Controller

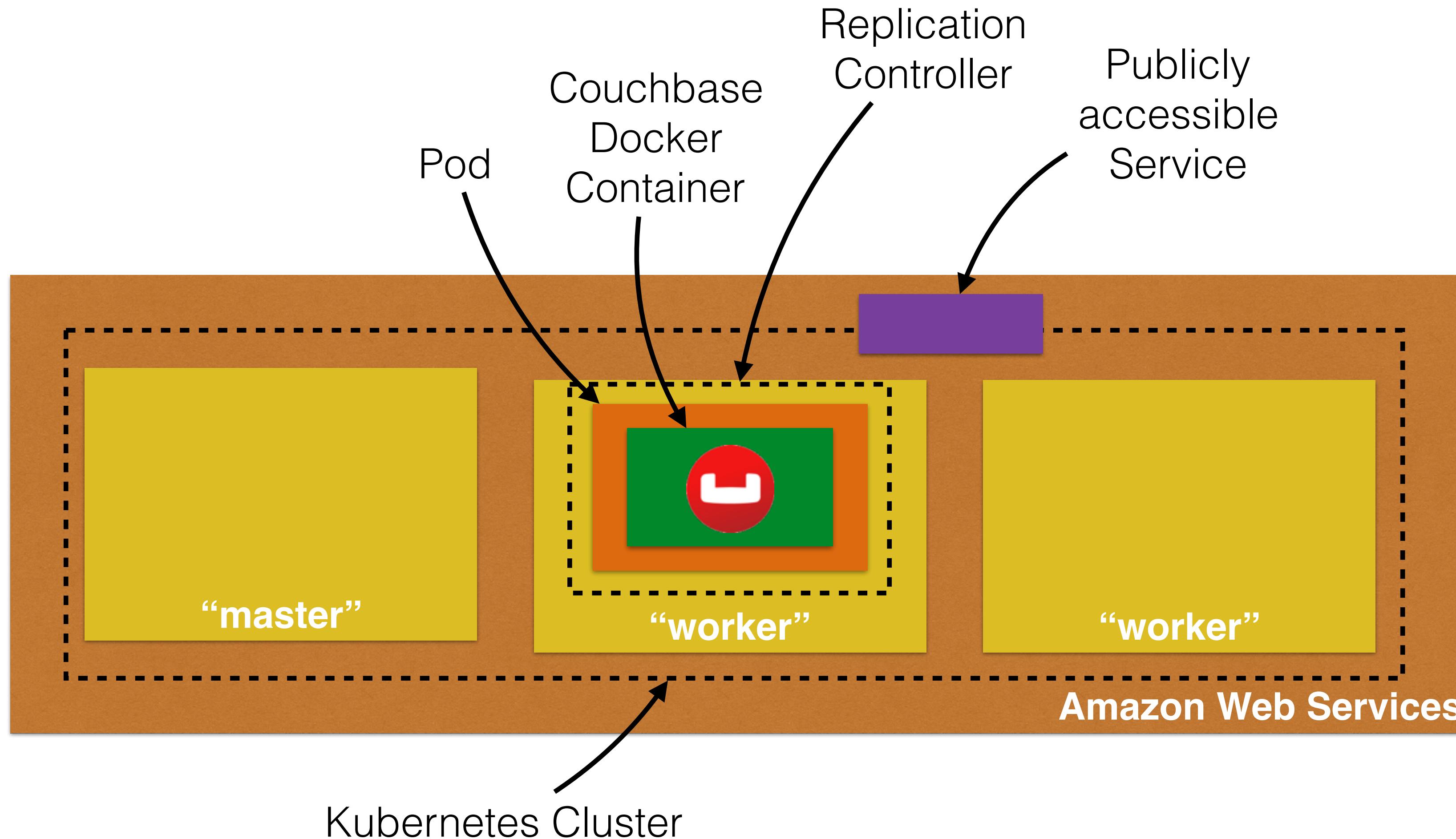


# Exposing Service

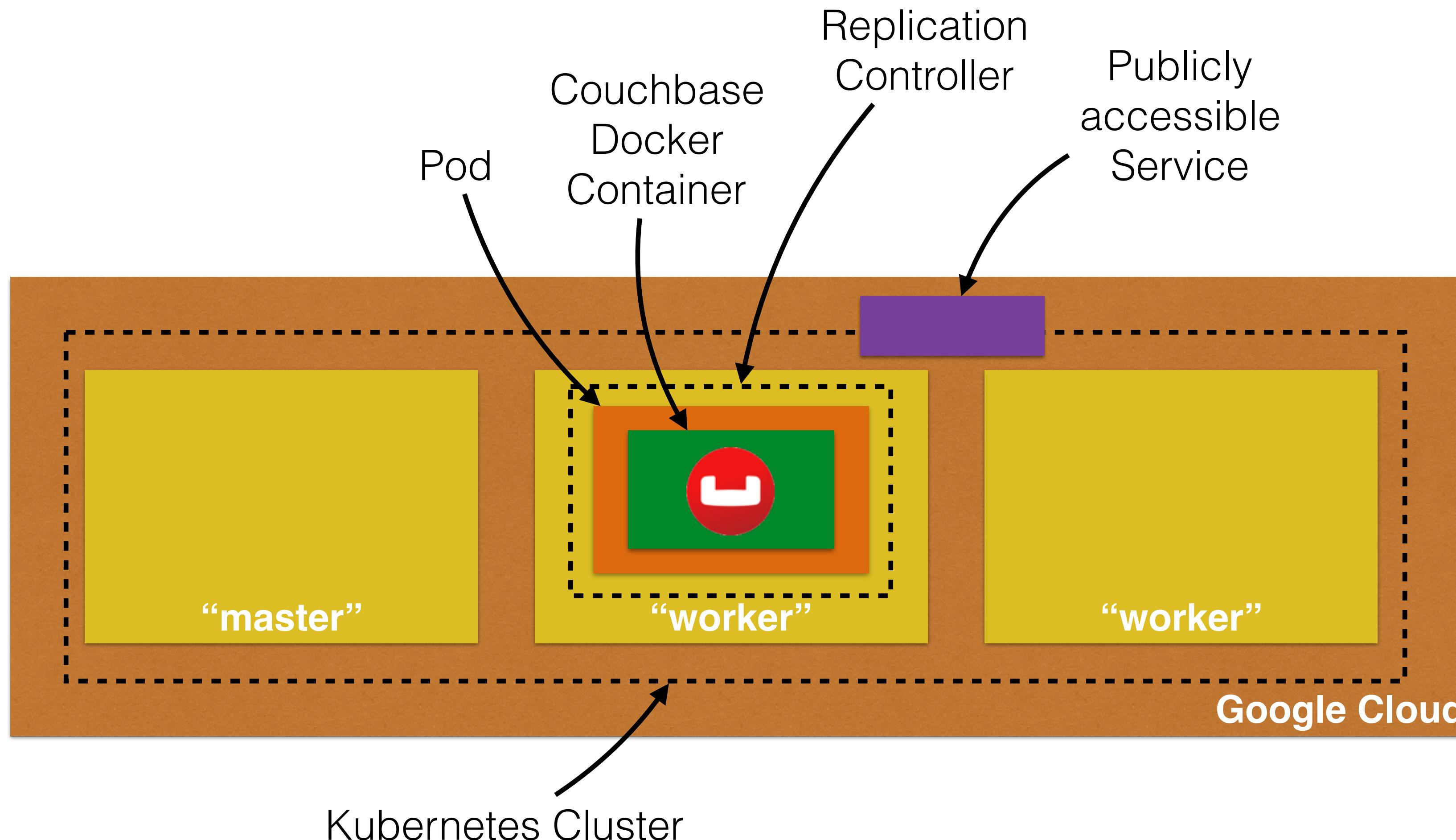
- Service may be exposed outside cluster or on Internet using **type**
  - **ClusterIP** (default)
  - **NodePort**: A port on each node

```
spec:  
  type: NodePort  
  ports:  
    - name: admin  
      port: 8091  
      nodePort: 30001  
    - name: query  
      port: 8093  
      nodePort: 30002
```
  - **LoadBalancer**: On cloud providers that support external LB

# Exposing Service on Amazon



# Exposing Service on GCE



# Replica Set

- Next generation Replication Controller
- Set-based selector requirement
- Useful for Horizontal Pod Autoscaling

# Replica Set Configuration

```
1  apiVersion: extensions/v1beta1
2  kind: ReplicaSet
3  metadata:
4    name: wildfly-rs
5  spec:
6    replicas: 2
7    selector:
8      matchLabels:
9        app: wildfly-rs-pod
10     matchExpressions:
11       - {key: tier, operator: In, values: ["backend"]}
12       - {key: environment, operator: NotIn, values: ["dev"]}
13   template:
14     metadata:
15       labels:
16         app: wildfly-rs-pod
17         tier: backend
18         environment: dev
19     spec:
20       containers:
21         - name: wildfly
22           image: jboss/wildfly
23         ports:
24           - containerPort: 8080
```

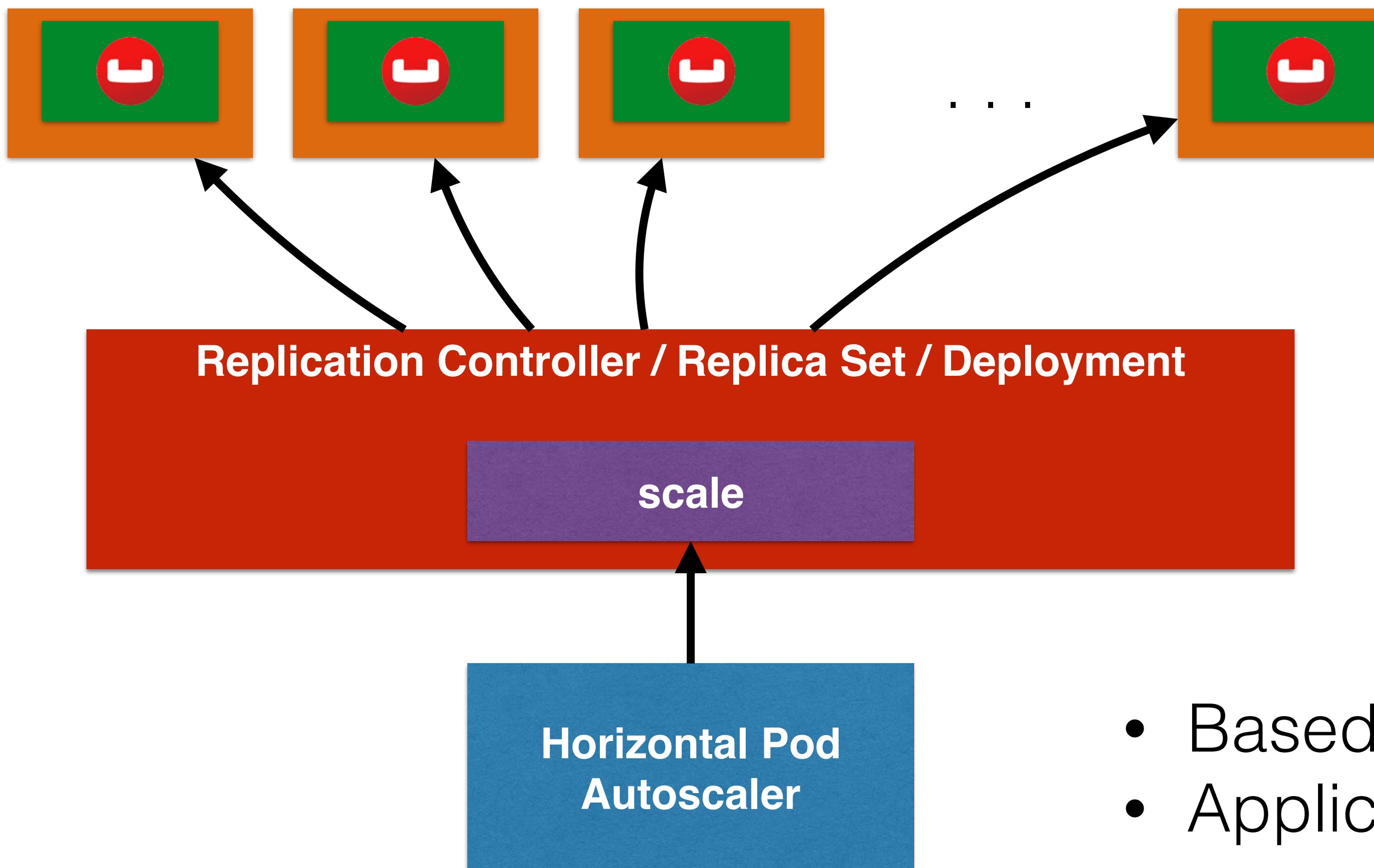
# Deployment

- Declarative updates for pods and replica sets
  - For example: rolling updates
- Differences from `kubectl`
  - Declarative instead of imperative
  - Server-side and so is faster
  - More features, e.g. rollback to previous version

# Deployment Configuration

```
1 apiVersion: extensions/v1beta1
2 kind: Deployment
3 metadata:
4   name: wildfly-deployment
5 spec:
6   replicas: 3
7   template:
8     metadata:
9       labels:
10      app: wildfly
11   spec:
12     containers:
13       - name: wildfly
14         image: jboss/wildfly
15     ports:
16       - containerPort: 8080
```

# Horizontal Pod Autoscaling



- Based on observed CPU utilization
- Application provided metrics (health)

# Horizontal Pod Autoscaling

- Typical usage
  - `kubectl autoscale deployment | rc | rs --min=<PODS> --max=<PODS> --cpu-percent=<CPU>`
- Autoscale a deployment with number of pods between 2 and 10
  - `kubectl autoscale deployment db --min=2 --max=10`
- Autoscale a Replication with maximum number of pods 5, target CPU utilization of 80%
  - `kubectl autoscale rc --max=5 --cpu-percent=80`

# HPA Configuration

# Volumes

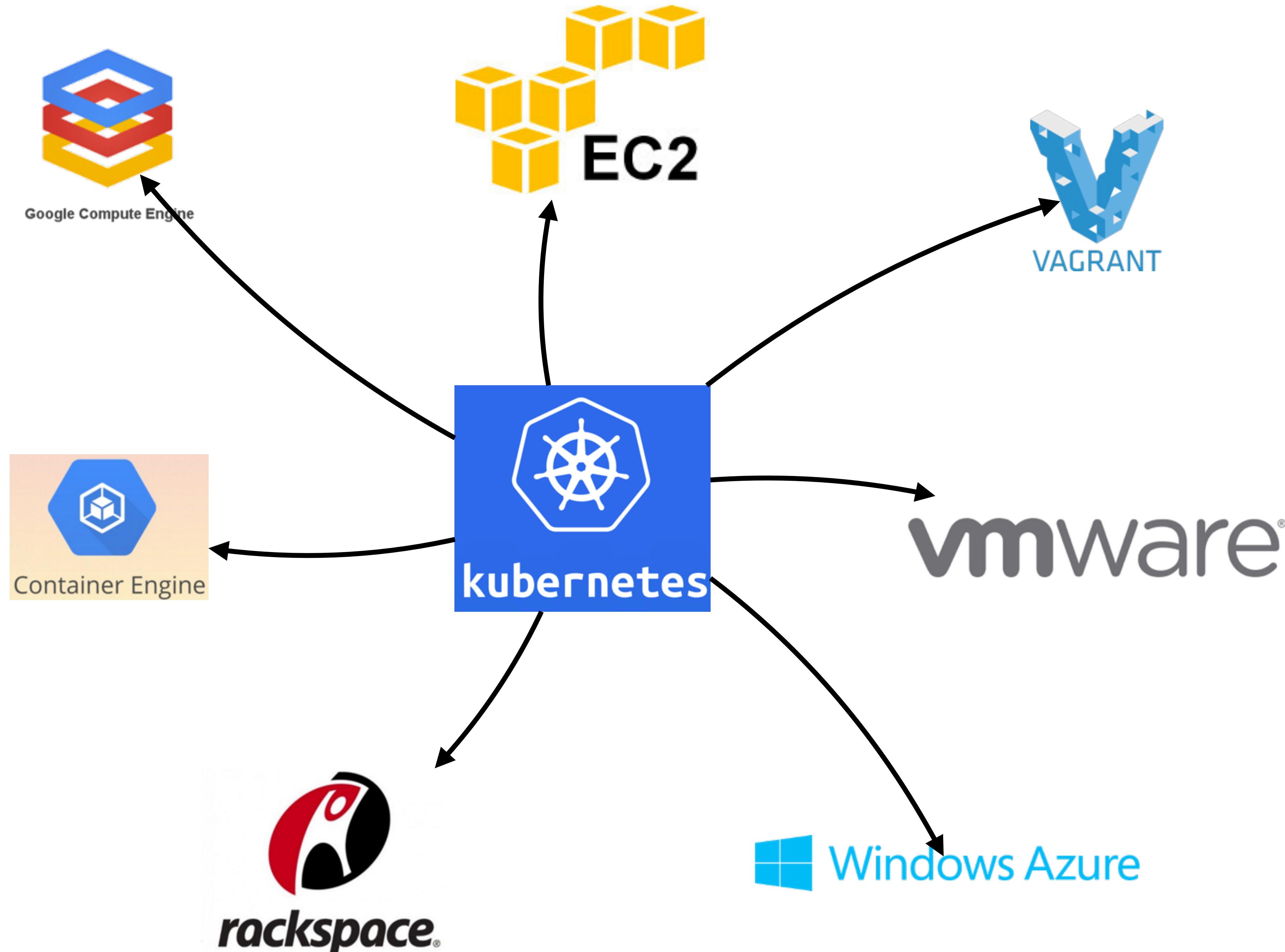
- Directory accessible to the containers in a pod
- Volume outlives any containers in a pod
- Common types
  - hostPath
  - nfs
  - awsElasticBlockStore
  - gcePersistentDisk

# Volume Configuration

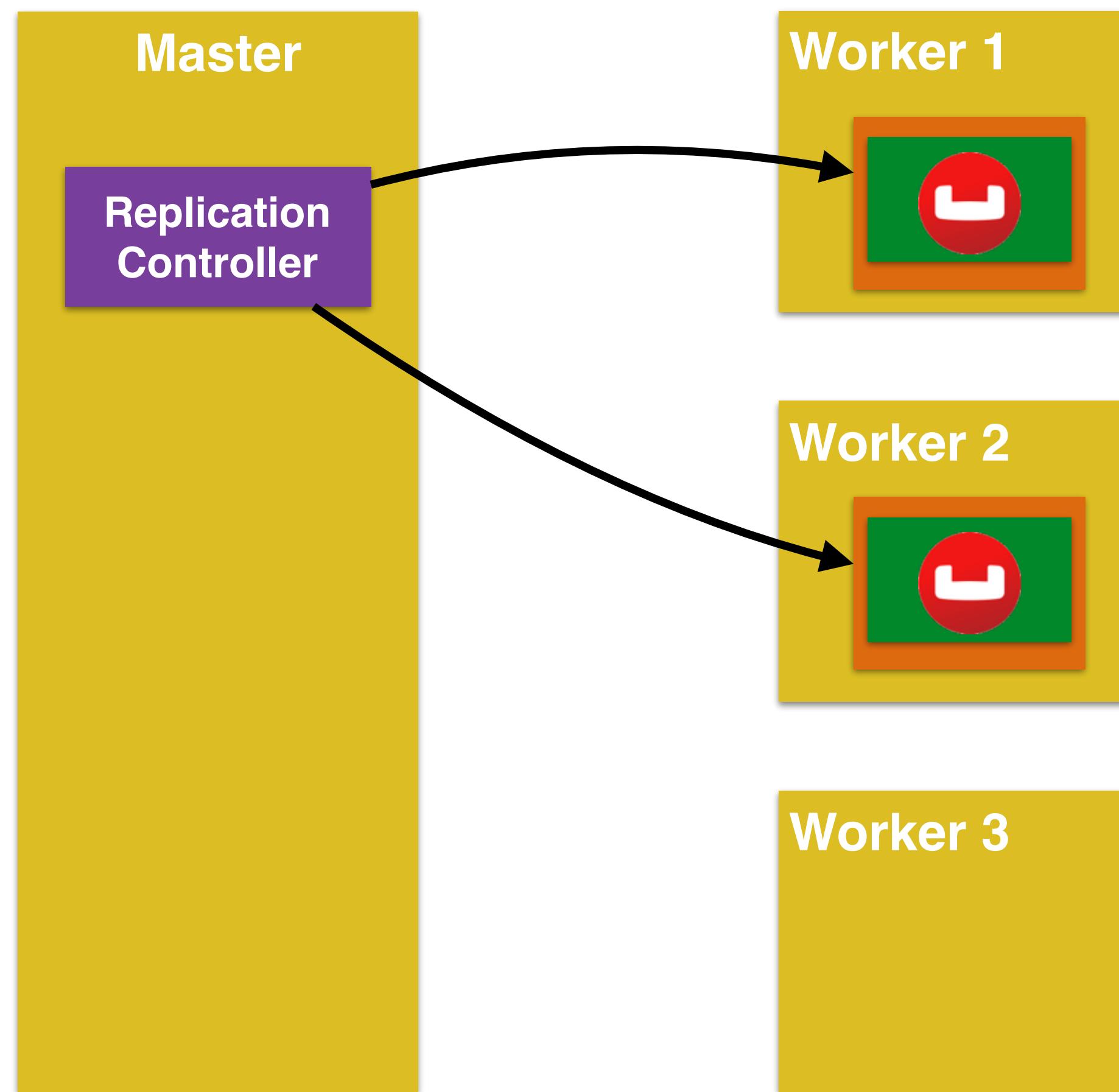
```
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: couchbase-pod
5    labels:
6      name: couchbase-pod
7  spec:
8    containers:
9      - name: couchbase
10        image: arungupta/couchbase-oreilly:k8s
11    ports:
12      - containerPort: 8091
13    volumeMounts:
14      - mountPath: /var/couchbase/lib
15        name: couchbase-data
16    volumes:
17      - name: couchbase-data
18        hostPath:
19          path: /opt/data
```

# Namespace

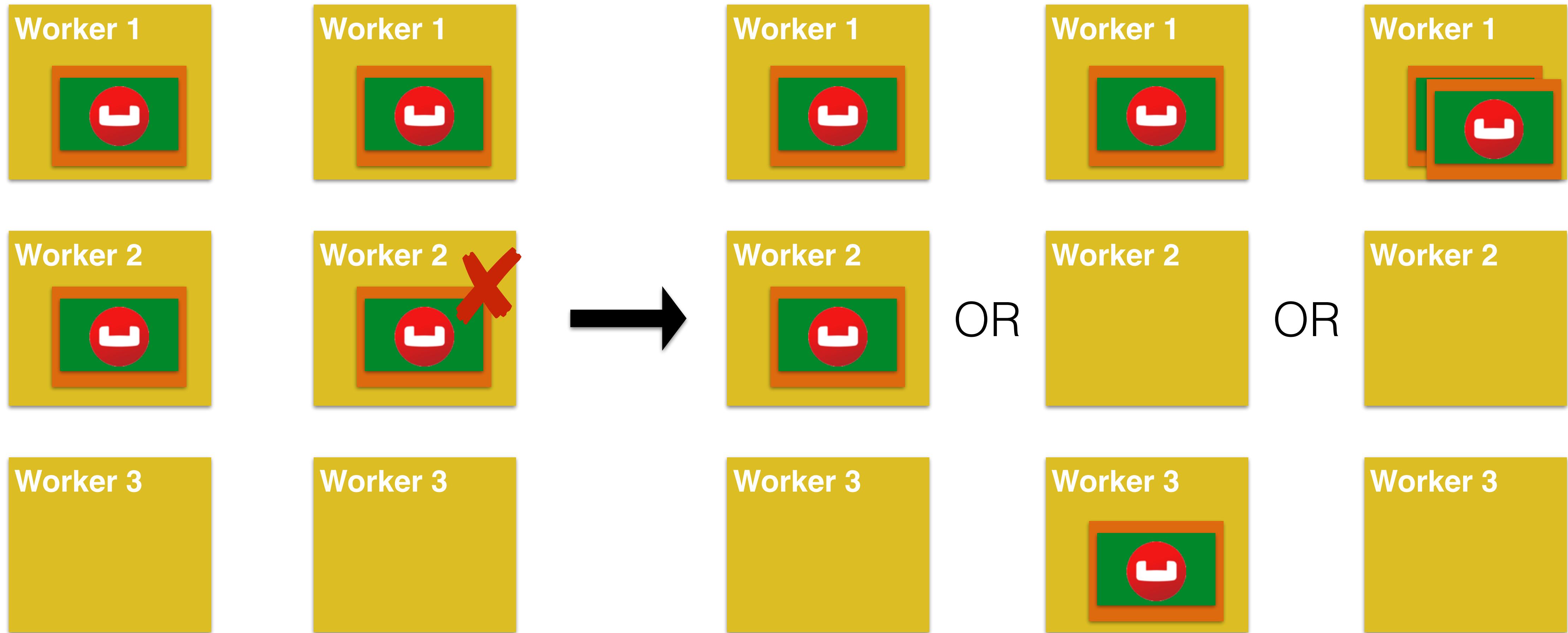
- Namespace allows to partition resources into a logical group
- Each namespace provides:
  - **scope** for resources to avoid collisions
  - **policies** to ensure appropriate authority to trusted users
  - **constraints** for resource consumption



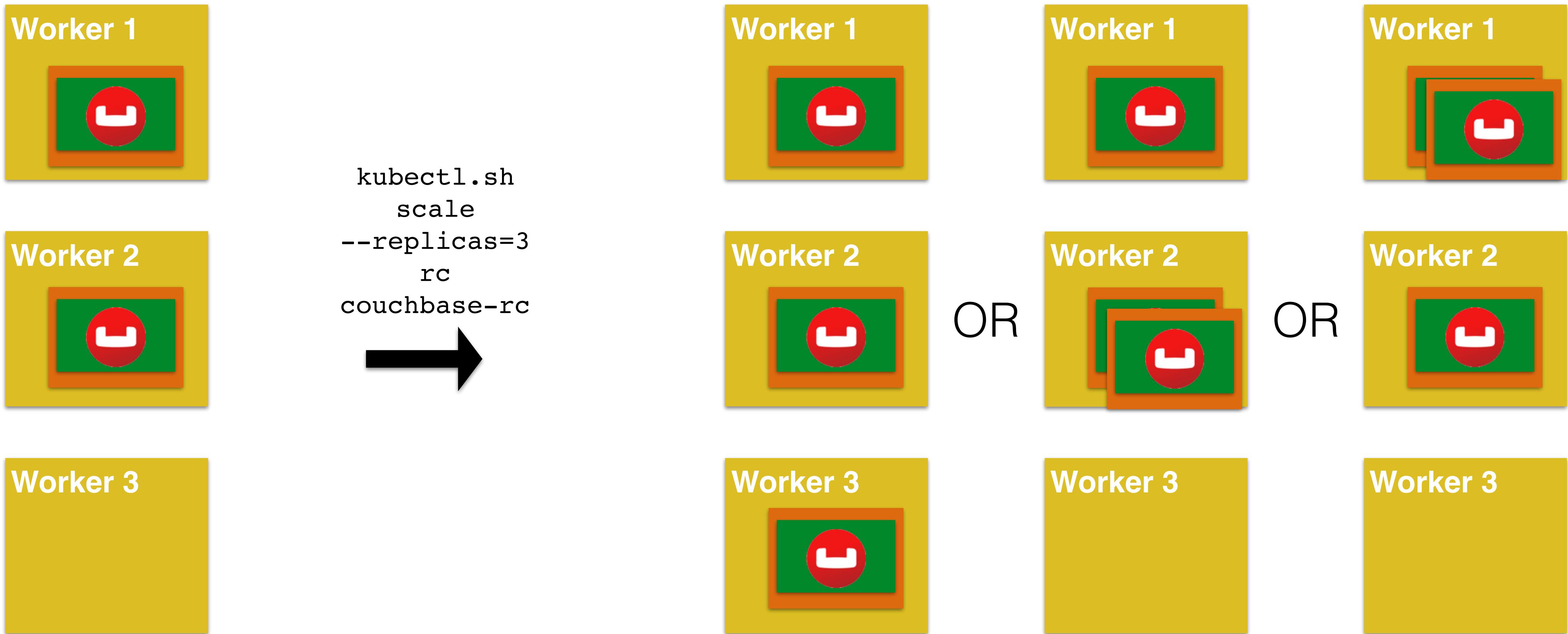
# Replication Controller



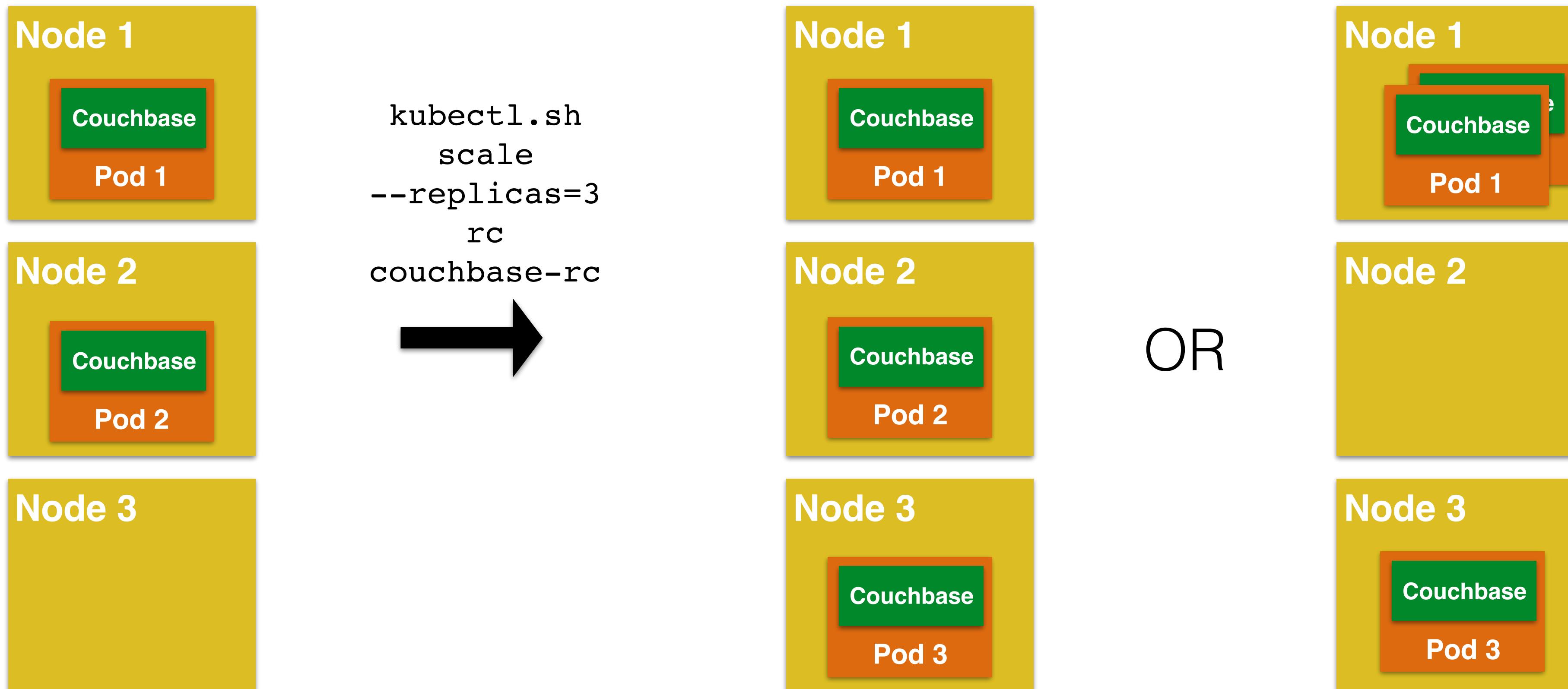
# RC: “Actual” vs “Desired” State



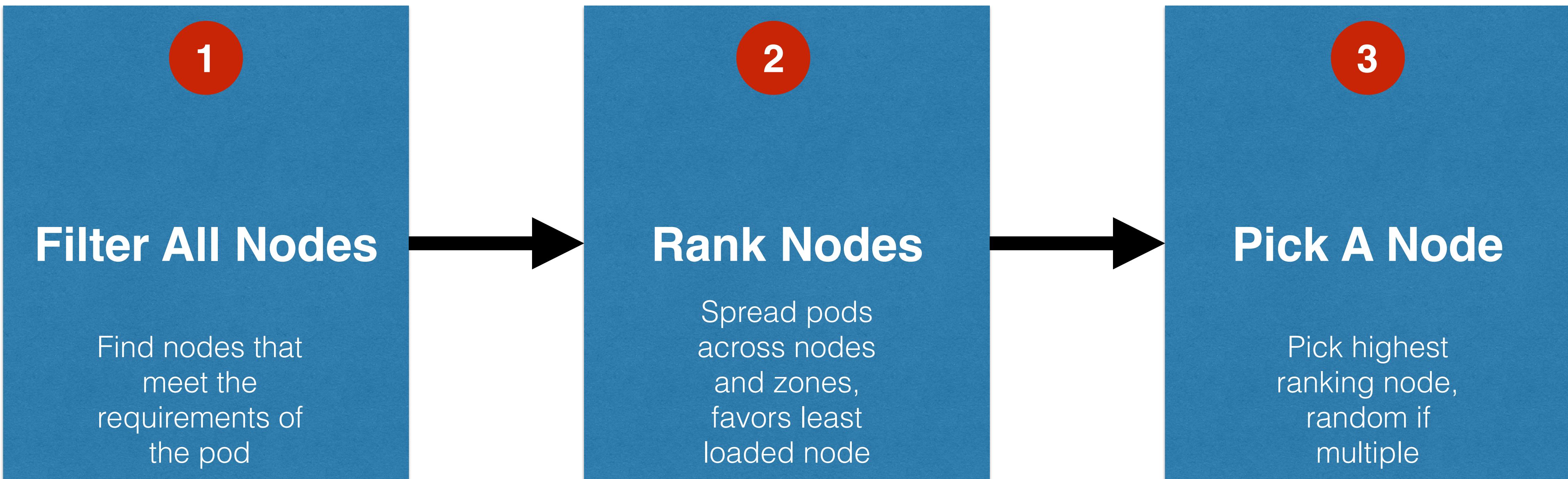
# RC: Scale Pods



# Replication Controller: Scaling

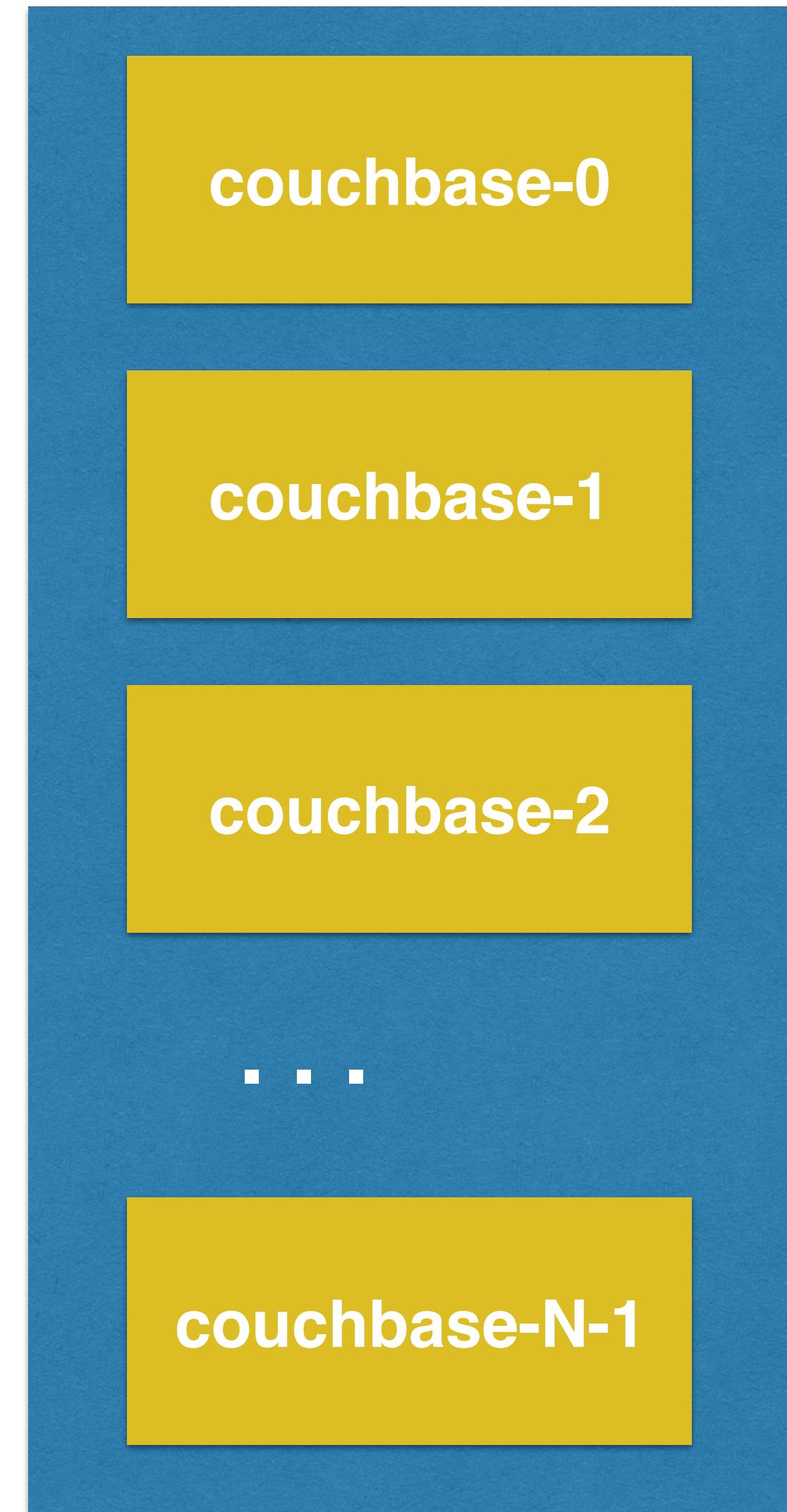


# Kubernetes Scheduling Algorithm



# Pet Set

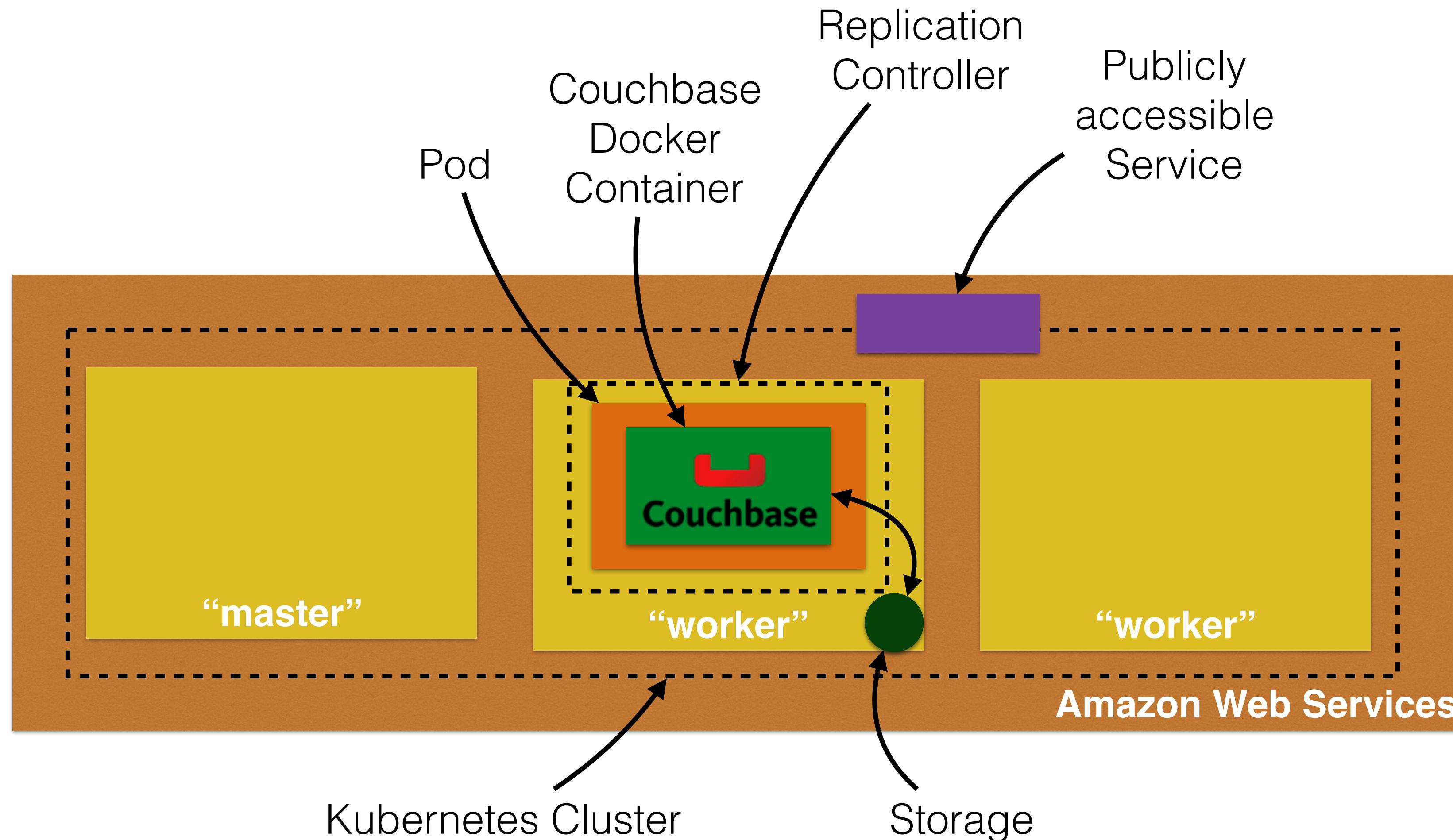
- Alpha resource introduced in 1.3
- Stateful pods
- PetSet has 0..N-1 Pets
- Each Pet has a deterministic name, and a unique identity
  - Identity = stable hostname, ordinal index, stable storage linked to ordinal & hostname
- Each Pet has at most one pod
- Each Pet Set has at most one Pet with a given identity



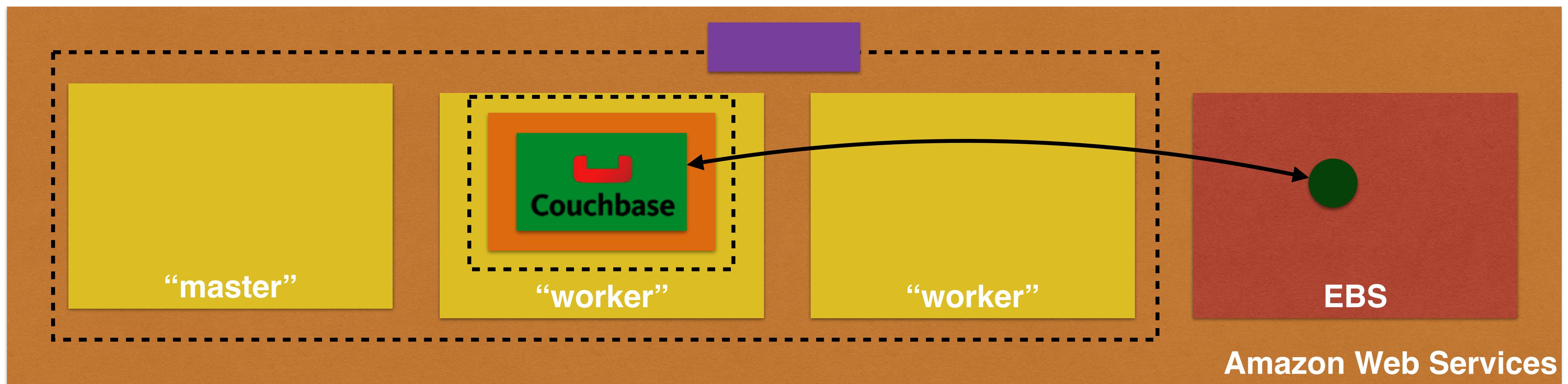
# Health Checks

- Restarts Pod, if wrapped in RC
- Application-level health checks
  - HTTP
  - Container Exec
  - TCP Socket
- Health checks performed by Kubelet

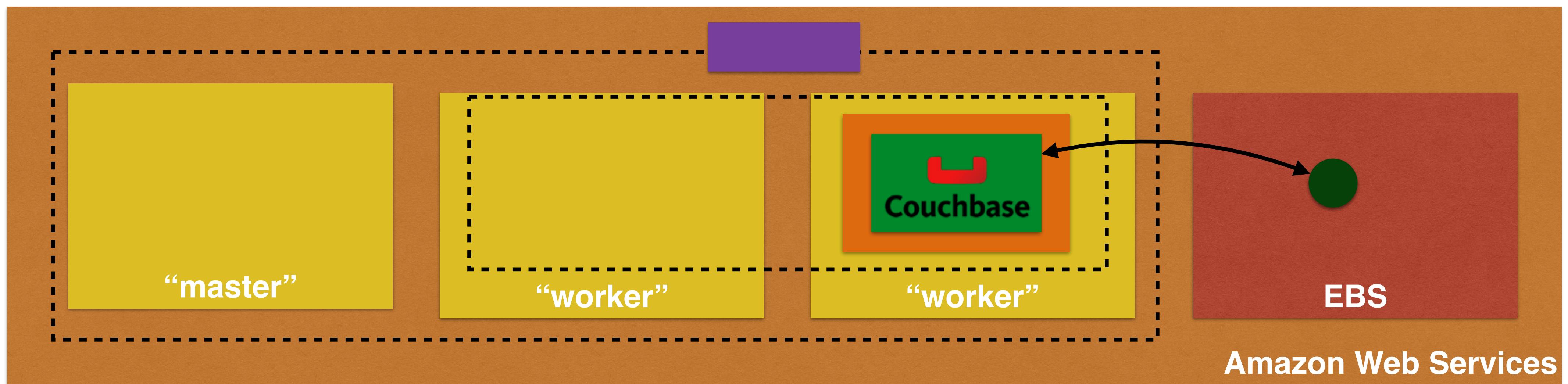
# Stateful Containers



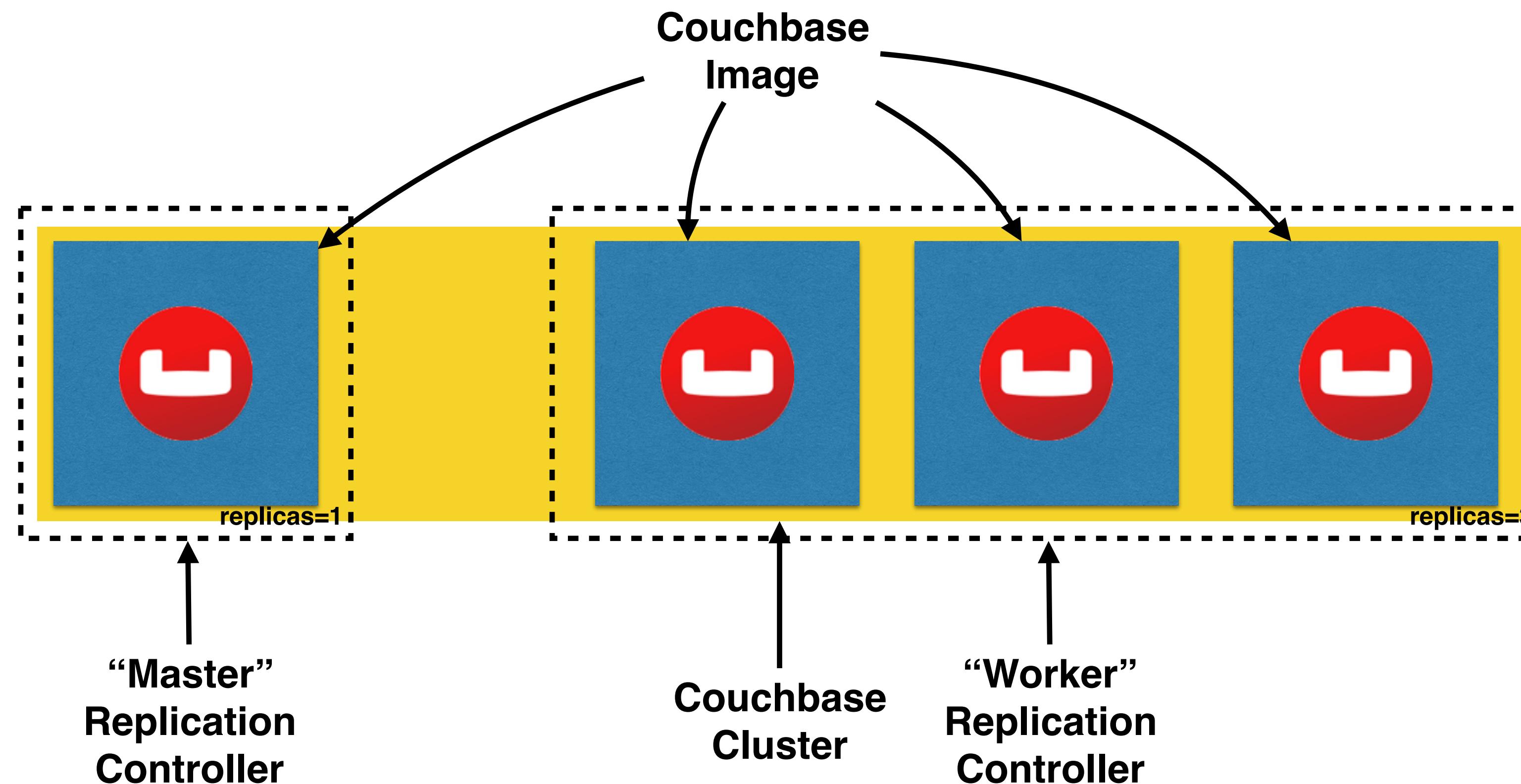
# Stateful Containers



# Stateful Containers



# Couchbase Cluster on Kubernetes

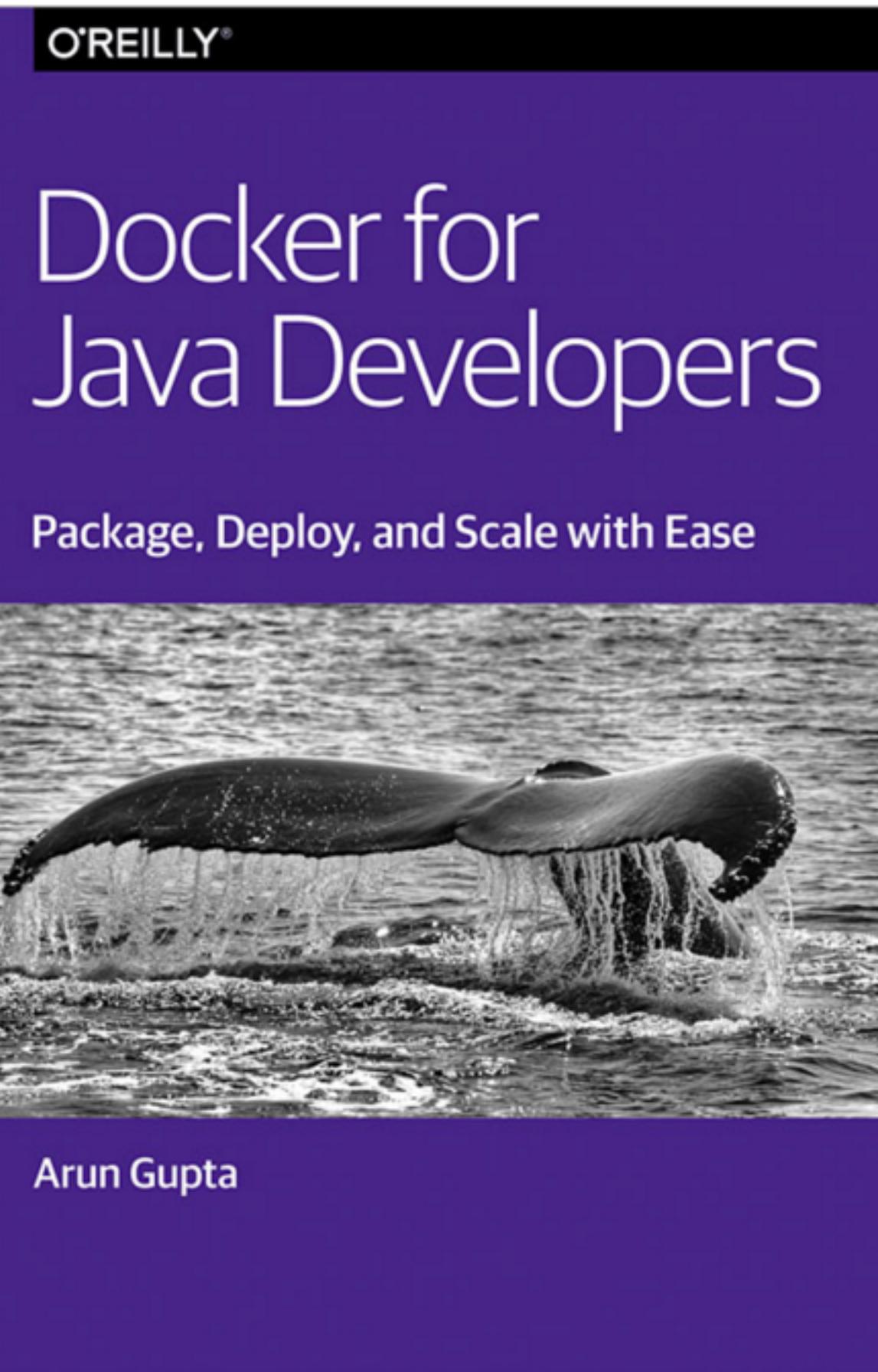


# Rolling Update

# Kubernetes and Maven

- fabric8-maven-plugin

[bit.ly/dockerjava](http://bit.ly/dockerjava)



[bit.ly/kubejava](http://bit.ly/kubejava)



# References

- [github.com/javaee-samples/docker-java](https://github.com/javaee-samples/docker-java)
- [kubernetes.io](https://kubernetes.io)
- Containers recipe: [couchbase.com/containers](https://couchbase.com/containers)