Kubernetes for Java Developers

Arun Gupta, @arungupta
Docker Captain, Java Champion

Docker Captain

Java Champion

JavaOne Rock Star (4 years)

NetBeans Dream Team

Silicon Valley JUG Leader

Author

Runner

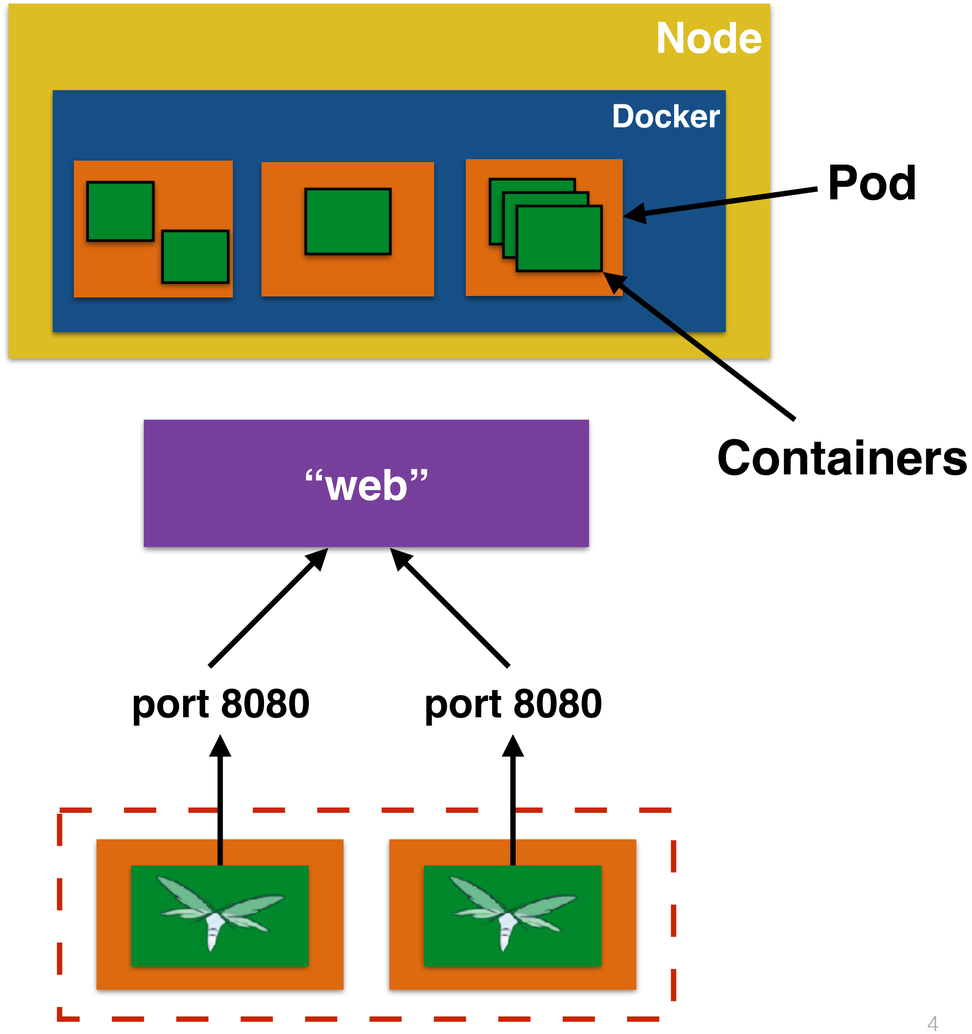
Lifelong learner

Kubernetes

- Open source orchestration system for containers
  - Docker, rkt, OCI, …
- Started by Google, donated to CNCF
- Provide declarative primitives for the “desired state”
  - Self-healing
  - Auto-restarting
  - Schedule across hosts
  - Replicating

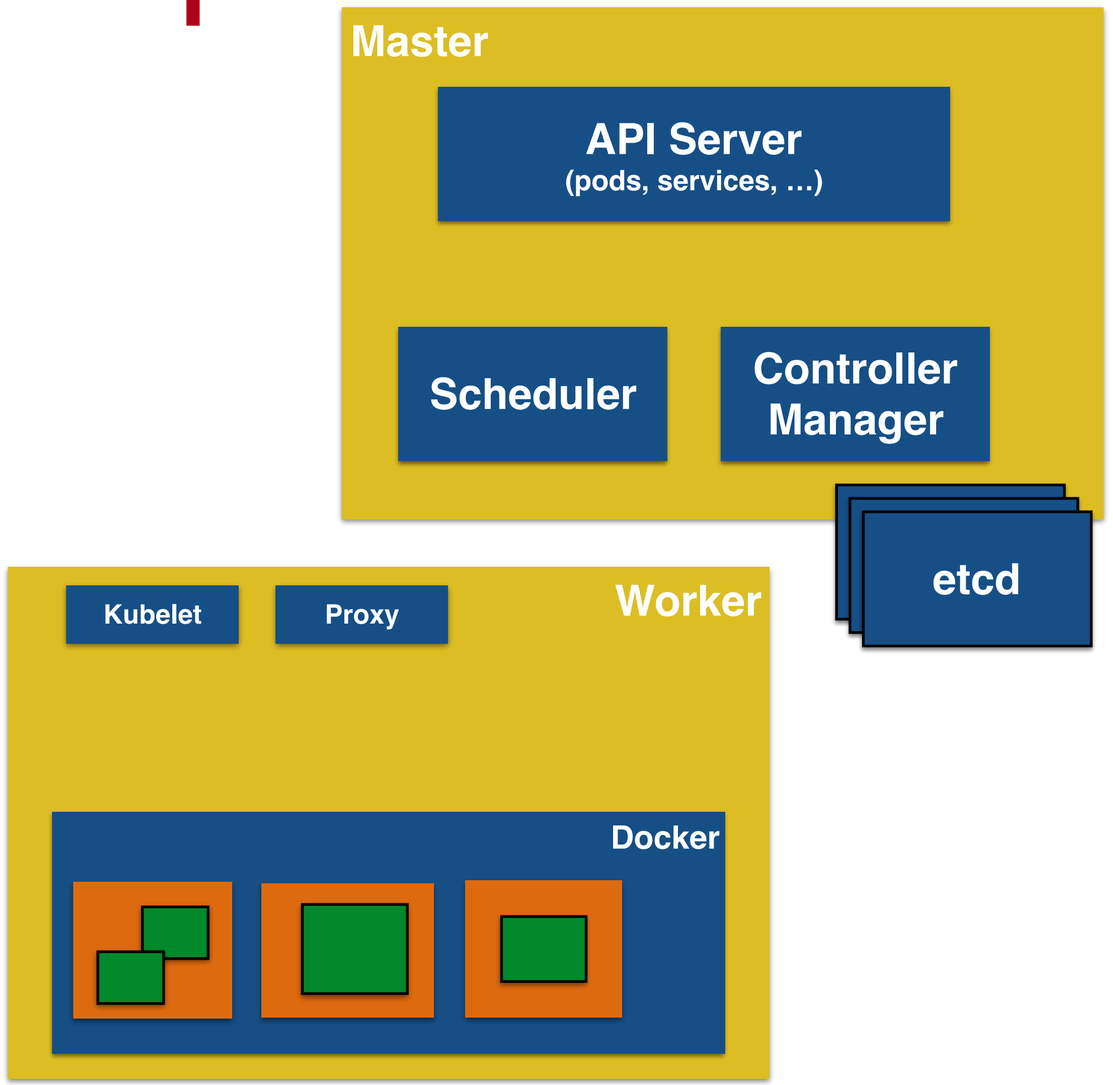
Kubernetes Concepts

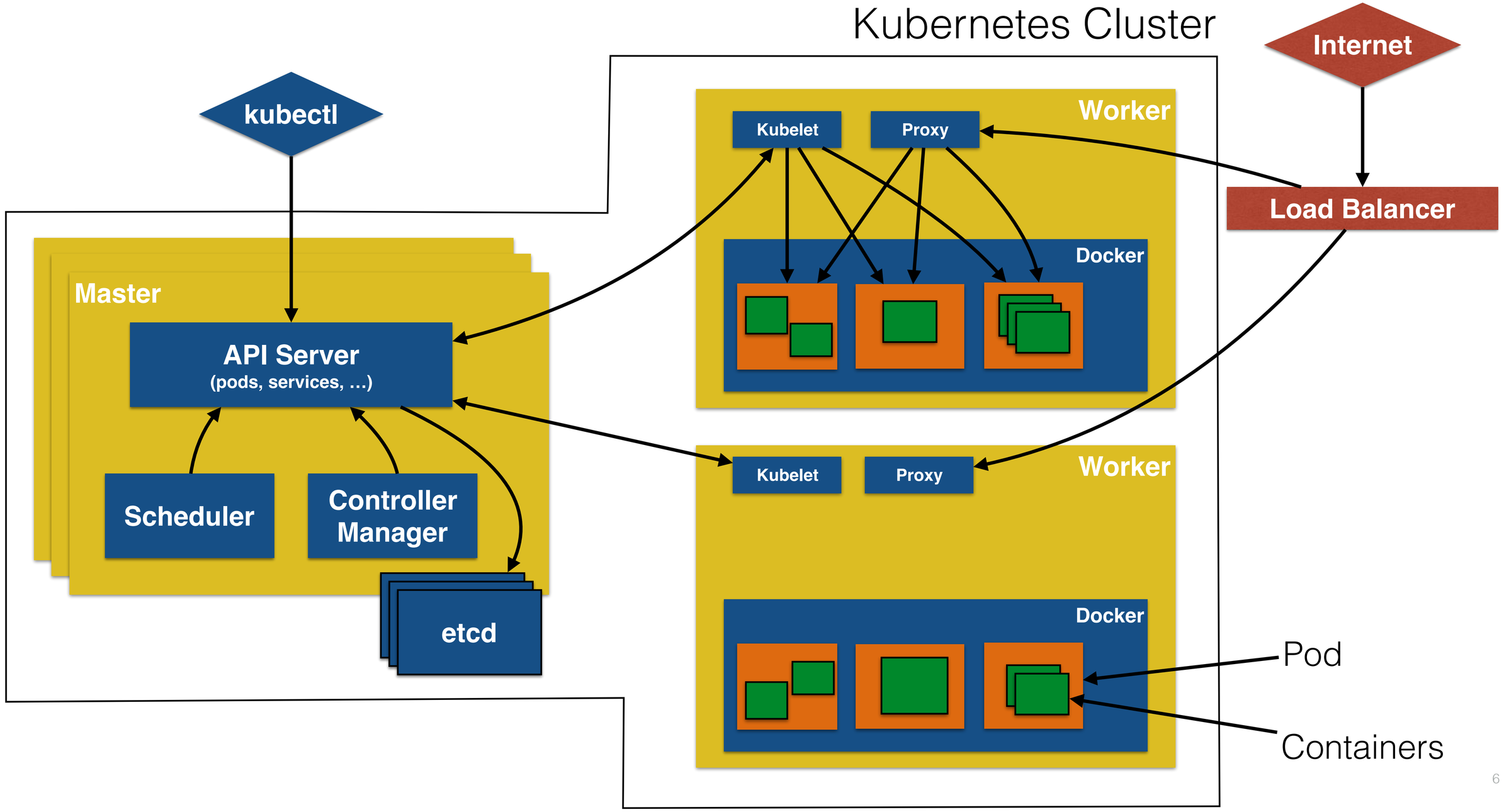
- **Pods**: colocated group of containers that share an IP, namespace, storage volume
- **Replica Set**: manages the lifecycle of pods and ensures specified number are running (next gen Replication Controller)
- **Service**: Single, stable name for a set of pods, also acts as LB
- **Label**: used to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




Kubernetes Cluster
Internet
kubectl
Worker
Kubelet
Proxy
Load Balancer
Docker
Master
API Server
(pods, services, …)
Scheduler
Controller
Manager
etcd
Kubelet
Proxy
Worker
Docker
Pod
Containers

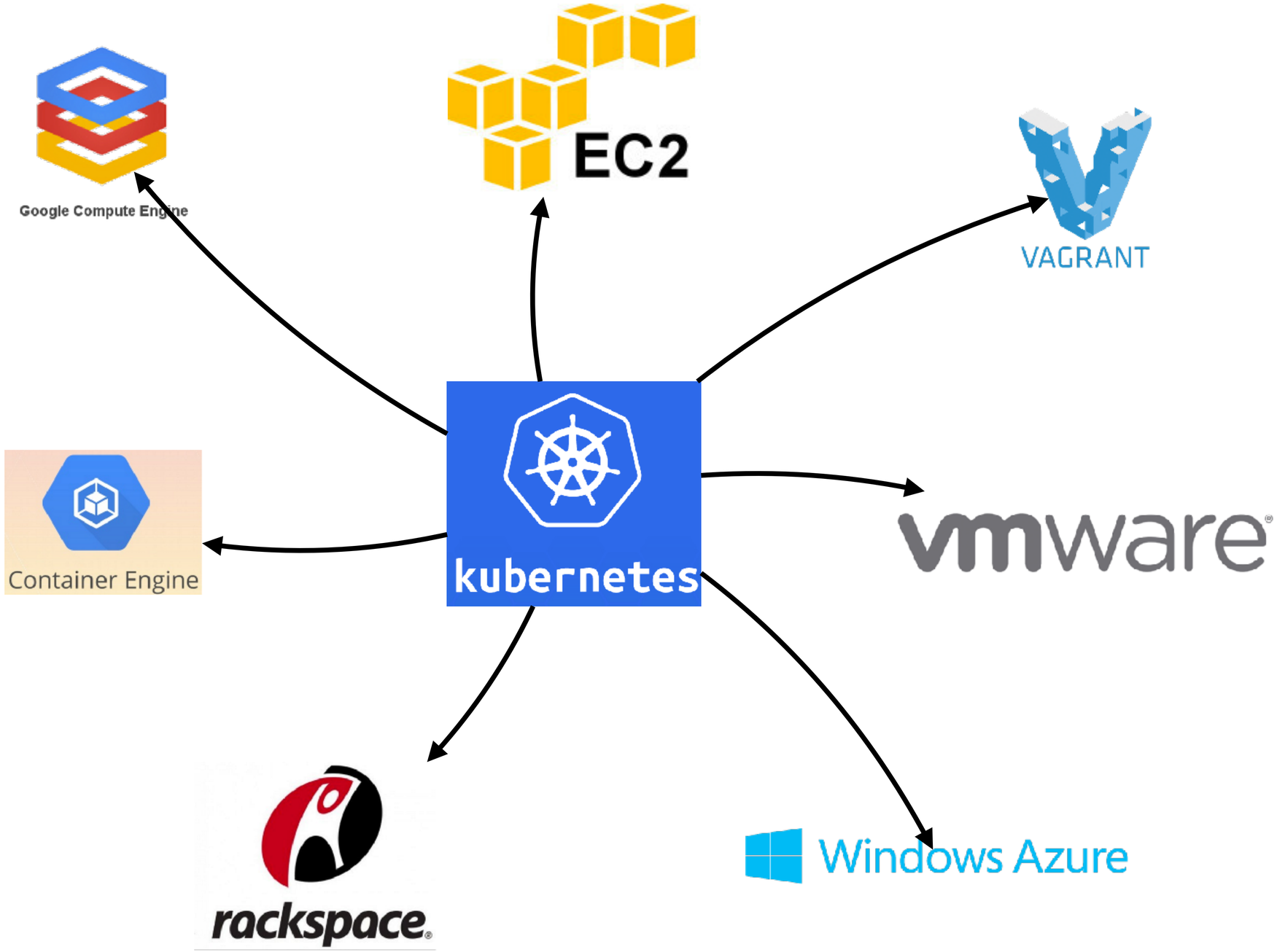
kubectl

- Controls the Kubernetes cluster manager
- CRUD Kubernetes resources
  - `create`, `get`, `describe`, `delete`, …
  - `kubectl create -f <filename>`
- `kubectl get nodes` or `pods`
- `kubectl scale –replicas=3 rc/<name>`

minikube

Minikube

- Runs a single node cluster in a VM
- Targeted for local development
- **`minikube start`**, **`stop`**, **`docker-env`**, …
- Requires **`kubectl`** CLI
- github.com/kubernetes/minikube/releases



https://get.k8s.io/

Kubernetes on AWS

- Clocker: http://www.clocker.io/tutorials/kubernetes-cluster.html
- Heptio: https://github.com/aws-quickstart/quickstart-heptio
- Juju Charms: https://jujucharms.com/canonical-kubernetes/
- Kargo: https://github.com/kubernetes-incubator/kargo
- Kismatic Enterprise Toolkit: https://github.com/apprenda/kismatic
- Kraken 2: https://github.com/samsung-cnct/k2
- kube-aws: https://github.com/kubernetes-incubator/kube-aws
- Kubeadm Quickstart: https://github.com/upmc-enterprises/kubeadm-aws
- Kubernetes Operations (kops): https://github.com/kubernetes/kops
- OpenShift: https://access.redhat.com/articles/2623521
- Stackpoint.io: https://stackpoint.io
- Tack: https://github.com/kz8s/tack
- Tectonic: http://github.com/coreos/tectonic-installer
- Weaveworks AMI: https://github.com/weaveworks/kubernetes-ami

https://github.com/kubernetes/community/blob/master/sig-aws/kubernetes-on-aws.md

Start Kubernetes Cluster

Kubernetes Pod Configuration

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: wildfly-pod
  labels:
    name: wildfly-pod
spec:
  containers:
  - name: wildfly
    image: jboss/wildfly
    ports:
    - containerPort: 8080
```

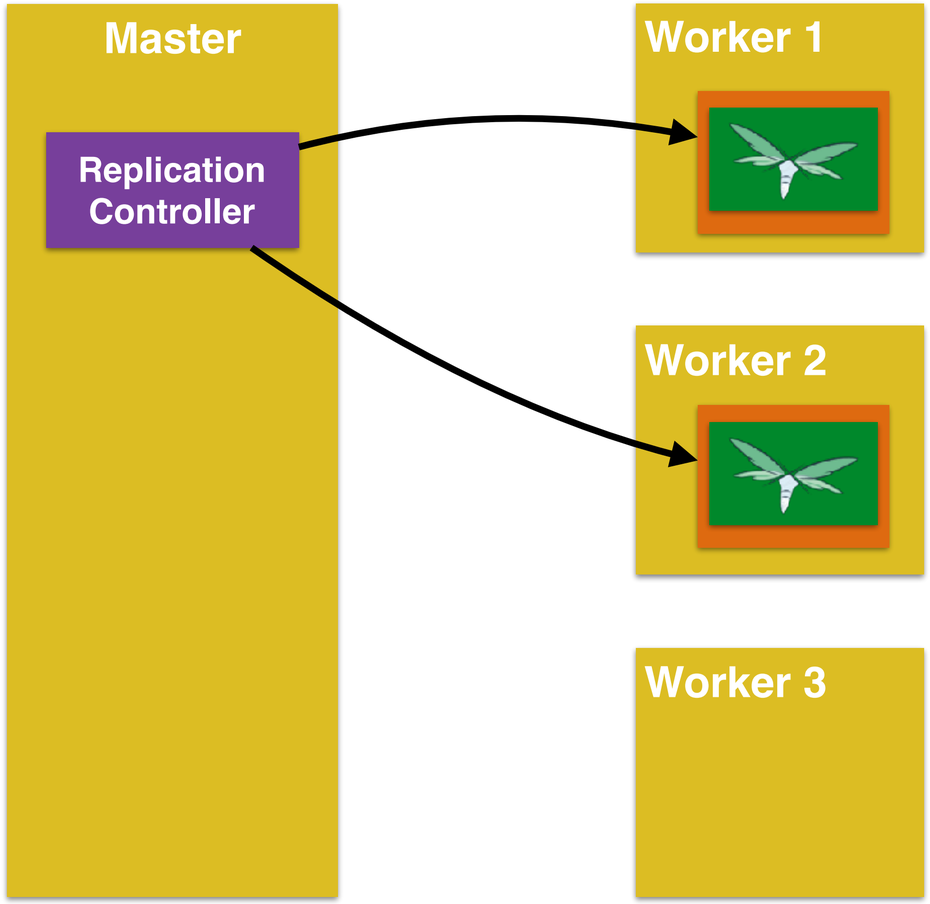
Replication Controller

- 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)

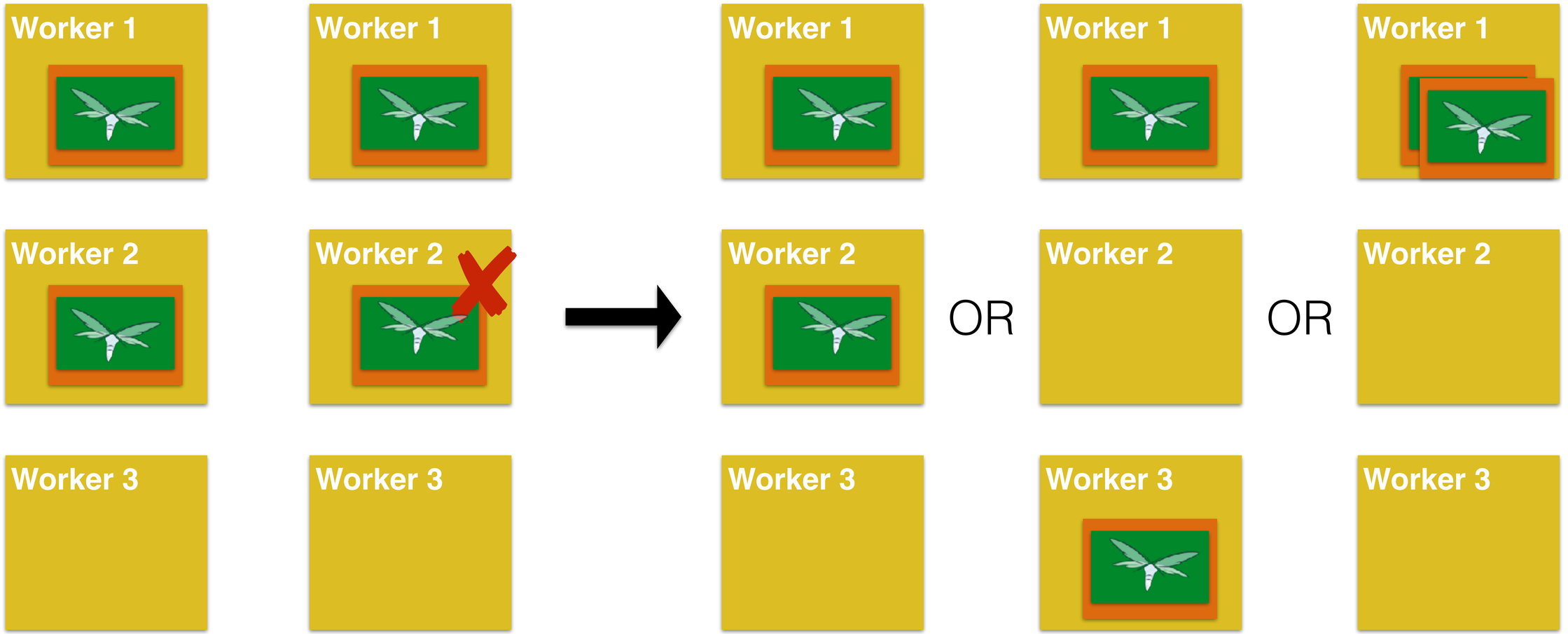
Kubernetes Replication Controller Configuration

```yaml
apiVersion: v1
kind: ReplicationController
metadata:
  name: wildfly-rc
spec:
  replicas: 2
  selector:
    app: wildfly-rc-pod
  template:
    metadata:
      labels:
        app: wildfly-rc-pod
    spec:
      containers:
      - name: wildfly
        image: jboss/wildfly
        ports:
        - containerPort: 8080
```

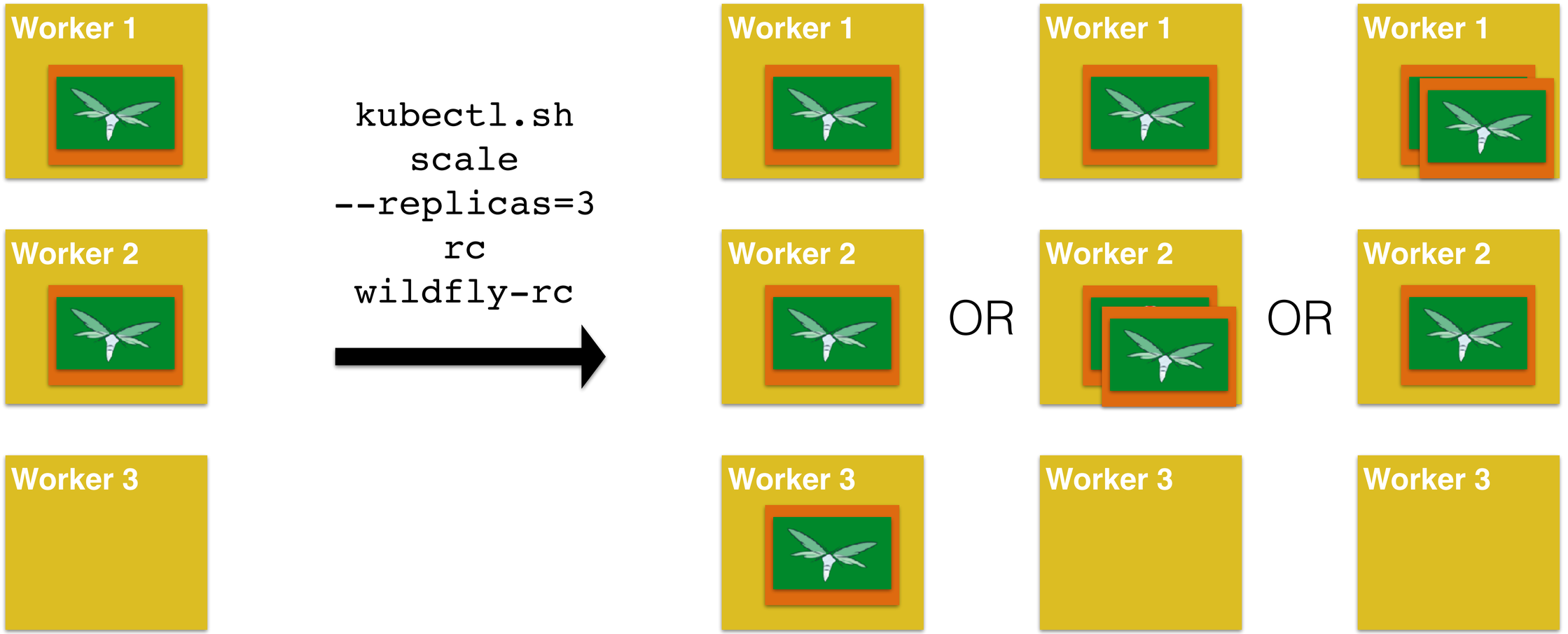
Replication Controller



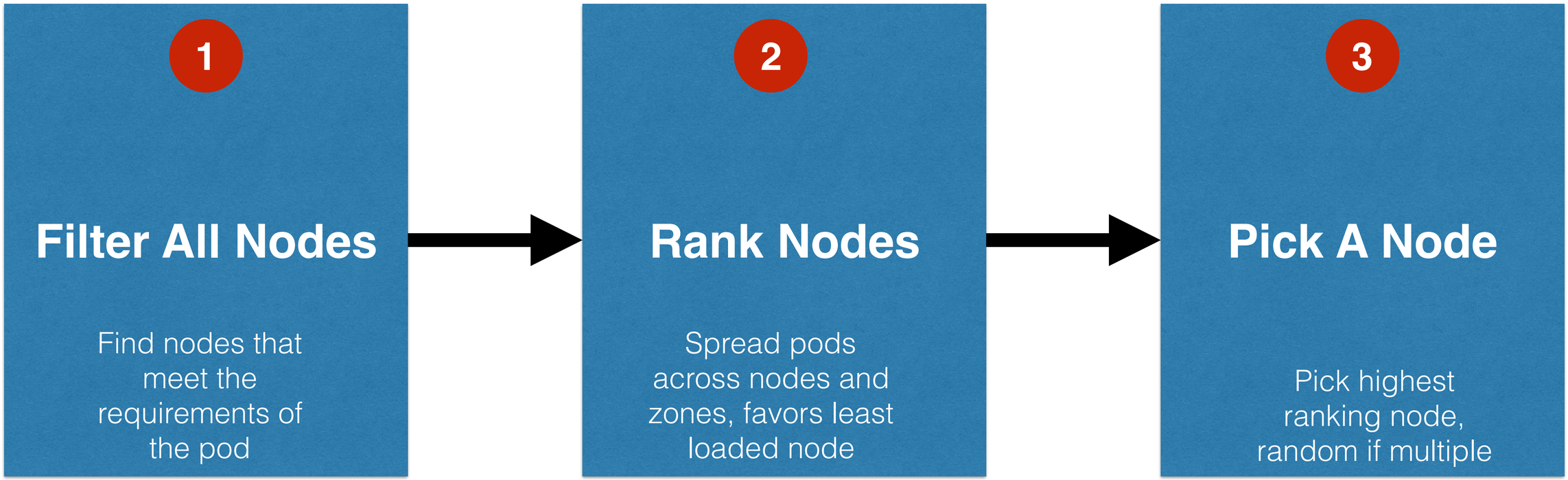
RC: “Actual” vs “Desired” State



RC: Scale Pods



Kubernetes Scheduling Algorithm



Replica Set

- Next generation Replication Controller
- Set-based selector requirement
  - Expression: **`key`**, **`operator`**, **`value`**
  - Operators: **`In`**, **`NotIn`**, **`Exists`**, **`DoesNotExist`**
- Generally created with **`Deployment`**
- Enables **`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
```

Services

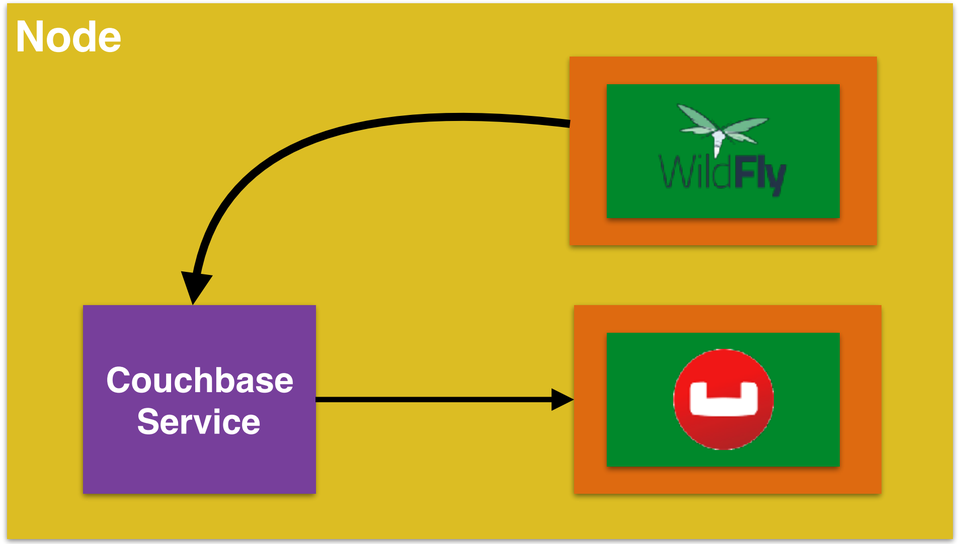
- Abstract a set of pods as a single IP and port
  - Simple TCP/UDP load balancing
- Creates environment variables in other pods or DNS resolution
- Stable endpoint for pods to reference
  - Allows list of pods to change dynamically

Kubernetes Service Configuration

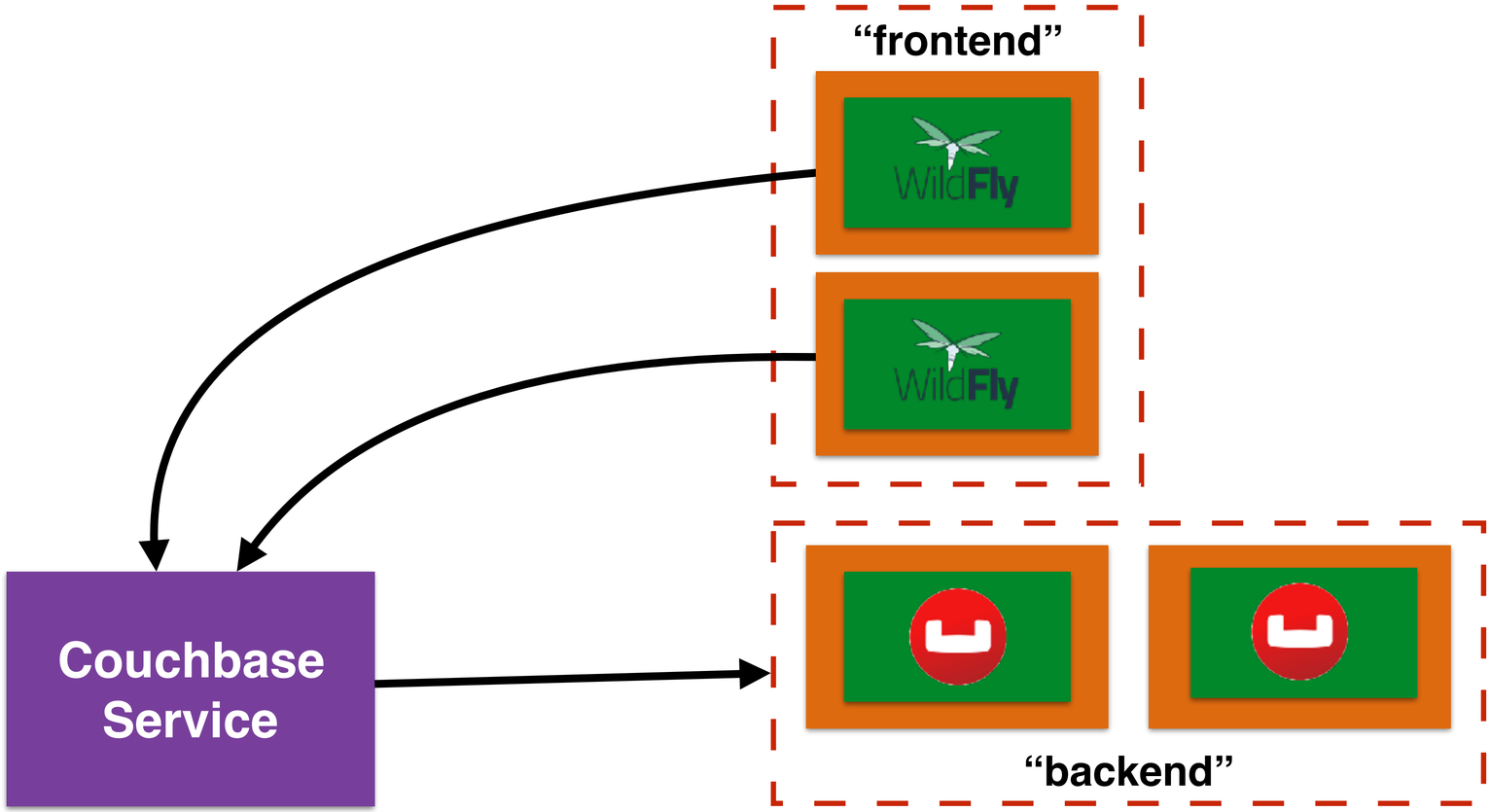
```yaml
apiVersion: v1
kind: Service
metadata:
  name: couchbase-service
spec:
  selector:
    app: couchbase-rc-pod
  ports:
    - name: admin
      port: 8091
    - name: query
      port: 8093
---
```

```yaml
apiVersion: v1
kind: ReplicationController
metadata:
  name: couchbase-rc
spec:
  replicas: 2
  selector:
    app: couchbase-rc-pod
  template:
    metadata:
      labels:
        app: couchbase-rc-pod
    spec:
      containers:
      - name: couchbase
        image: couchbase
        ports:
        - containerPort: 8091
```

Couchbase Service



Service and Replication Controller



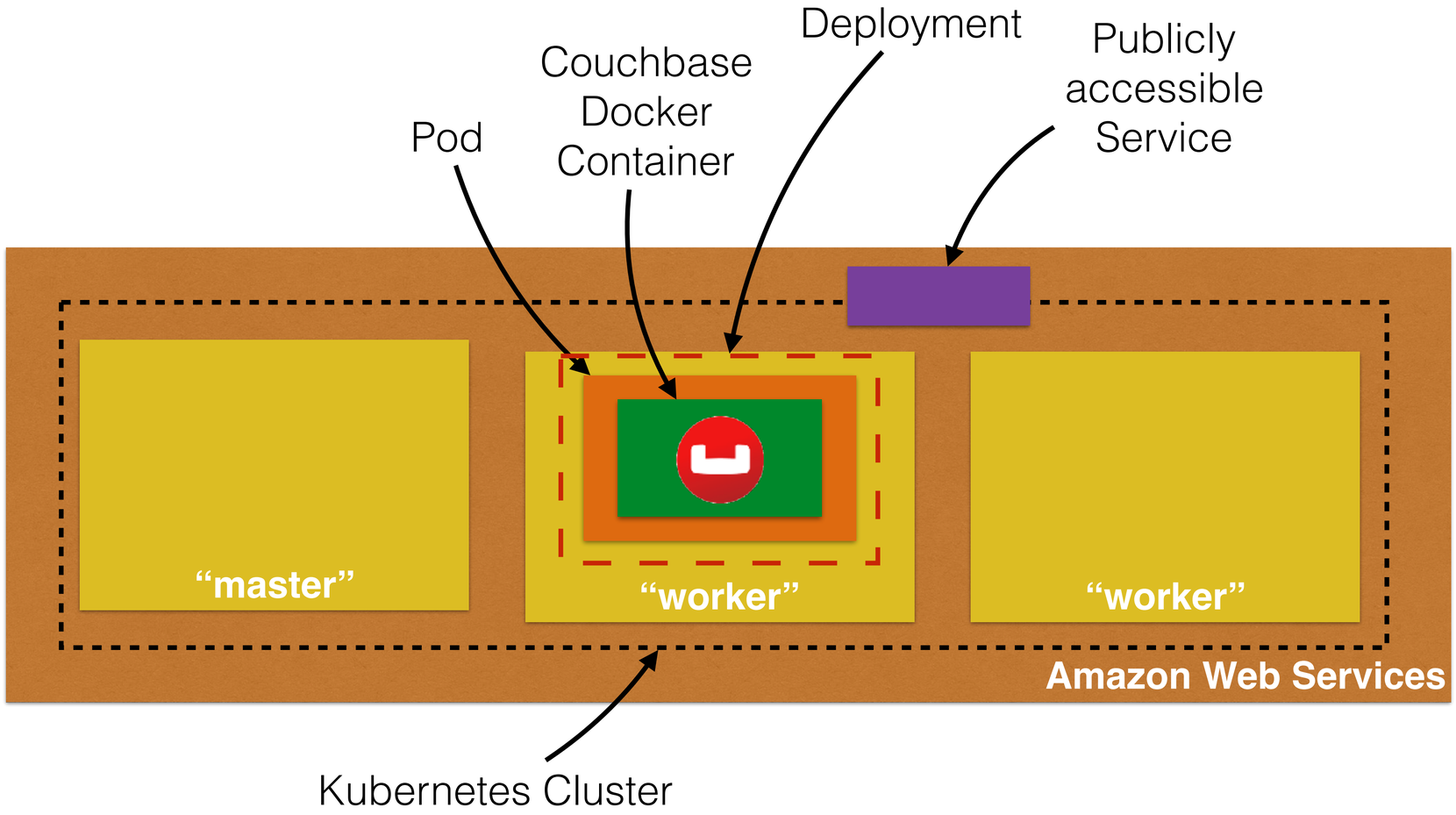
Exposing Service

- 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



Deployment

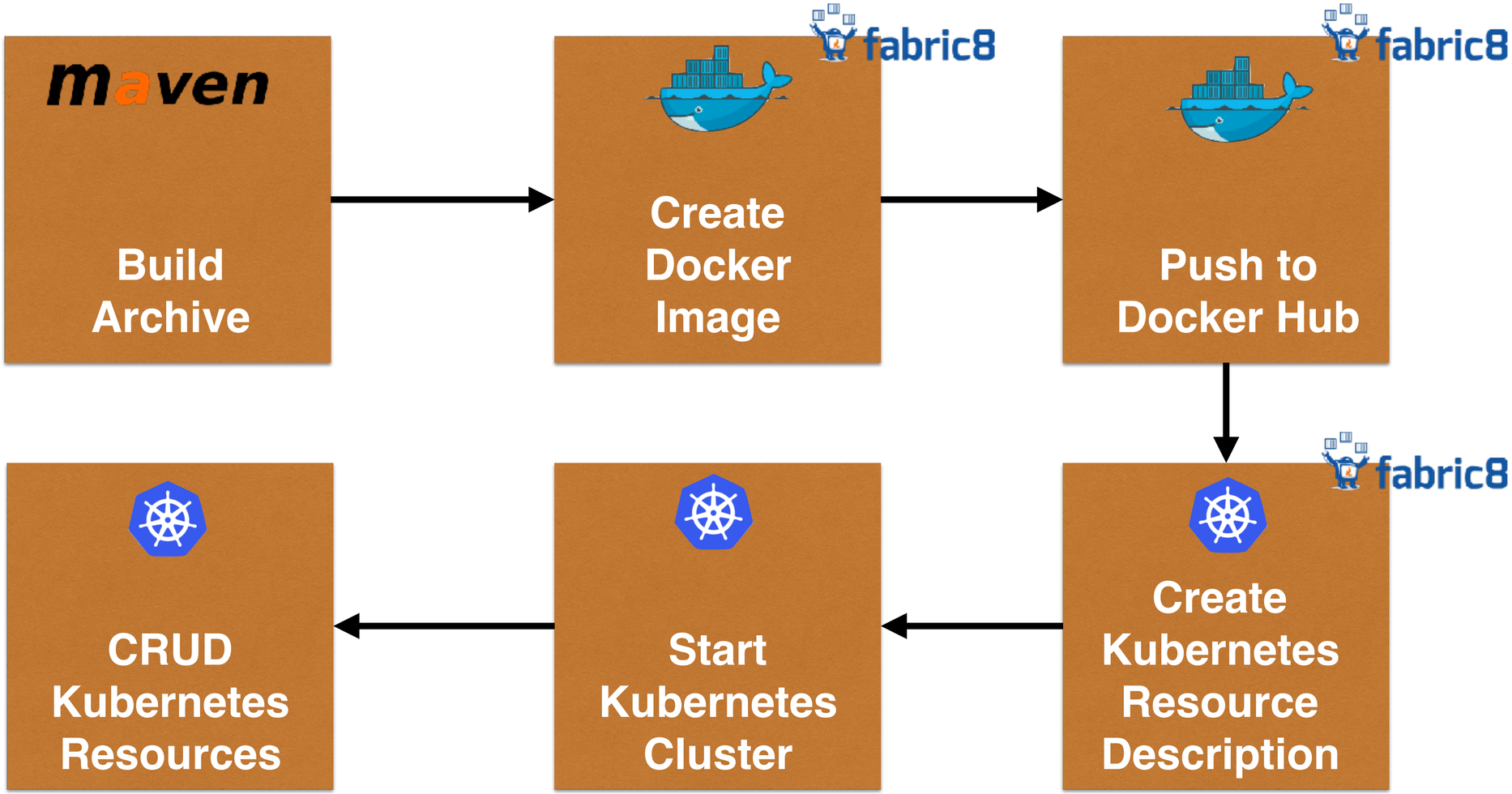
- Declarative updates for pods and replica sets
  - For example: rolling updates
- Differences from **`kubectl`**
  - Declarative instead of imperative
  - Server-side, and so is faster
  - More features, e.g. rollback to previous version

Deployment Configuration

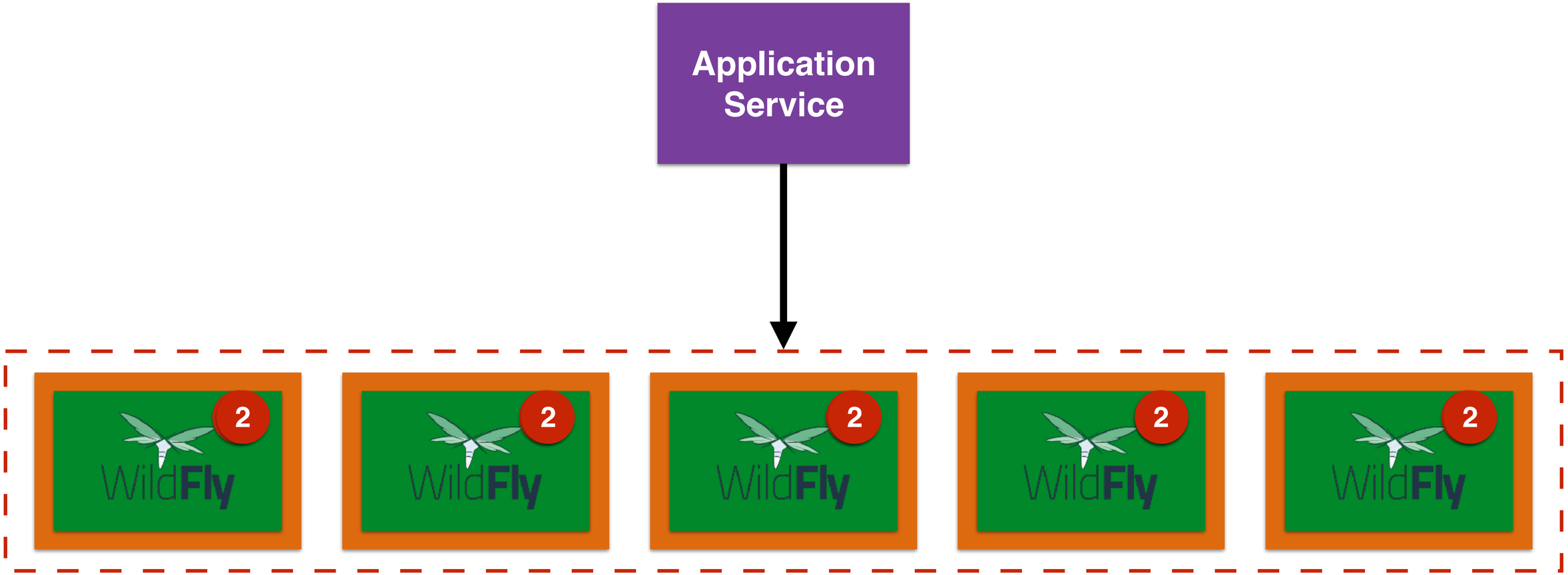
```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: wildfly-deployment
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: wildfly
    spec:
      containers:
      - name: wildfly
        image: jboss/wildfly
        ports:
        - containerPort: 8080
```

CRUD Kubernetes Resources

Kubernetes and Java Developers



Rolling Updates



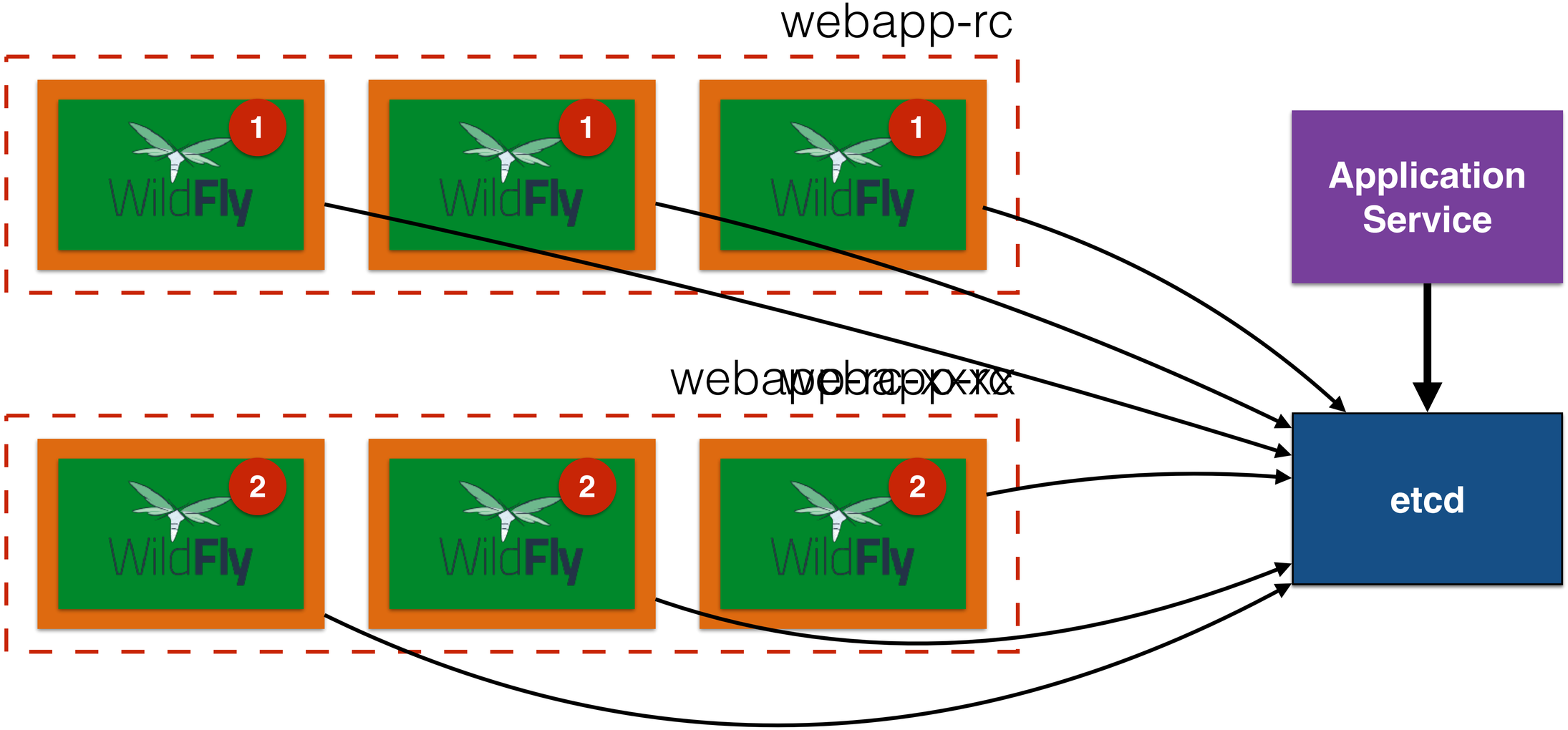
Rolling Updates - Replication Controller

- `kubectl rolling-update webapp-rc -f webapp-rc2.json`
- `kubectl rolling-update webapp-rc --image=arungupta/wildfly-app:2`

```
Created webapp-rc-5a11f15230716f6026c407eb9c1a60ca
Scaling up webapp-rc-5a11f15230716f6026c407eb9c1a60ca from 0 to 2, scaling down
webapp-rc from 2 to 0 (keep 2 pods available, don't exceed 3 pods)
Scaling webapp-rc-5a11f15230716f6026c407eb9c1a60ca up to 1
Scaling webapp-rc down to 1
Scaling webapp-rc-5a11f15230716f6026c407eb9c1a60ca up to 2
Scaling webapp-rc down to 0
Update succeeded. Deleting old controller: webapp-rc
Renaming webapp-rc-5a11f15230716f6026c407eb9c1a60ca to webapp-rc
replicationcontroller "webapp-rc" rolling updated
```

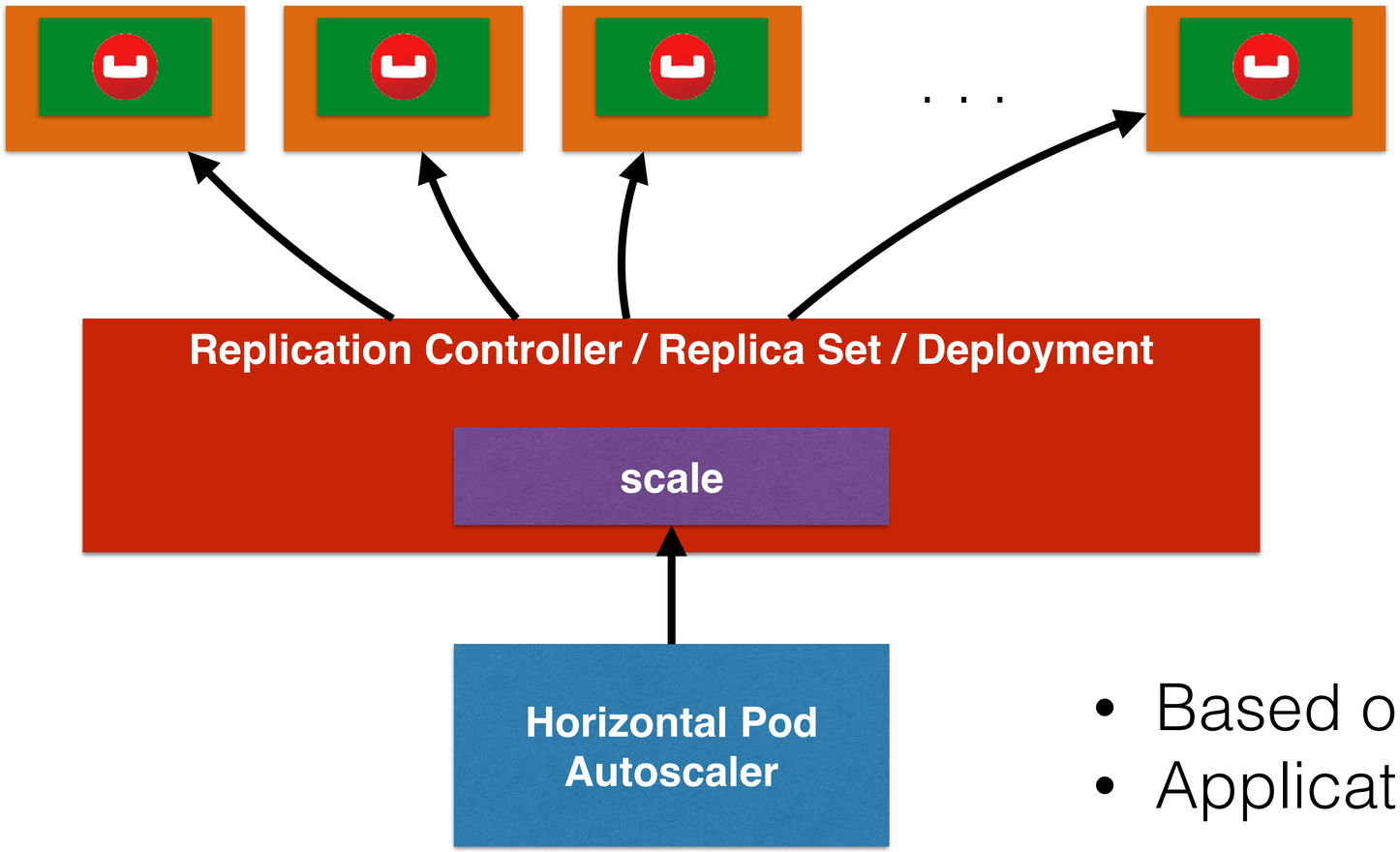

https://github.com/arun-gupta/kubernetes-java-sample/tree/master/rolling-update

Rolling Updates



https://github.com/arun-gupta/kubernetes-java-sample/tree/master/rolling-update

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

```yaml
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: wildfly-scaler
spec:
  scaleTargetRef:
    kind: ReplicaSet
    name: wildfly-rs
  minReplicas: 3
  maxReplicas: 10
  targetCPUUtilizationPercentage: 50
```

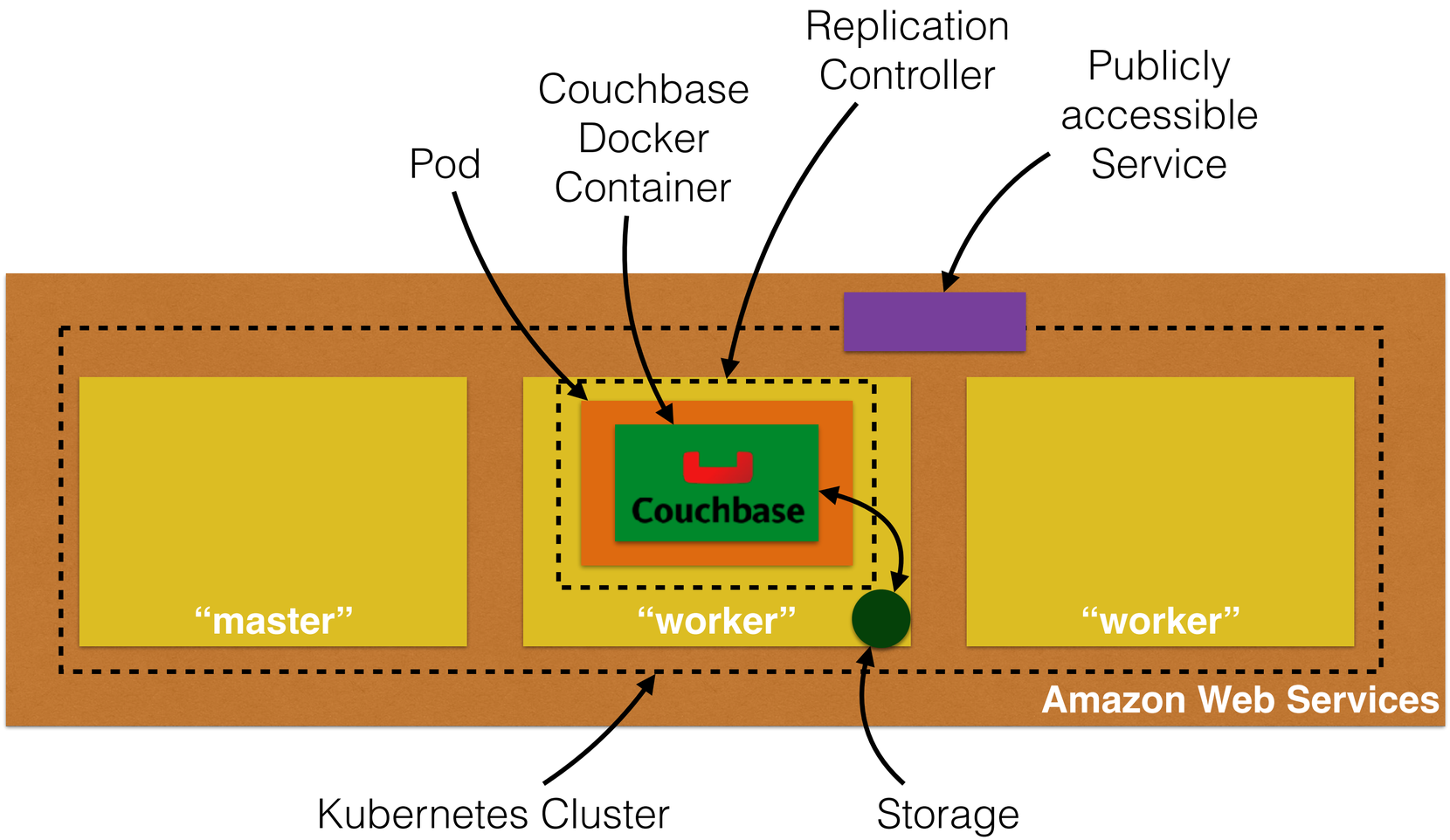
Volumes

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

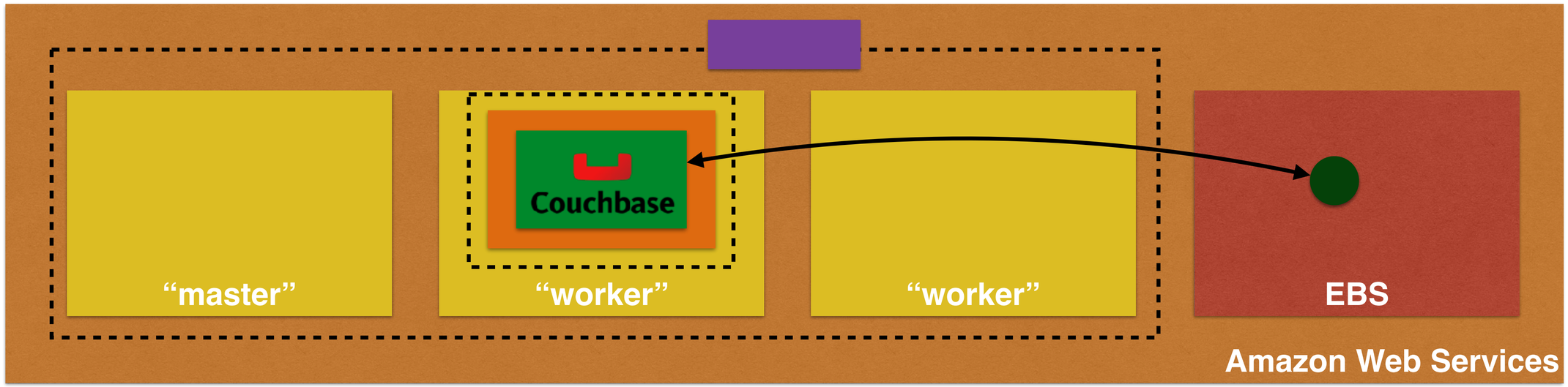
Volume Configuration

```
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
```

Stateful Containers

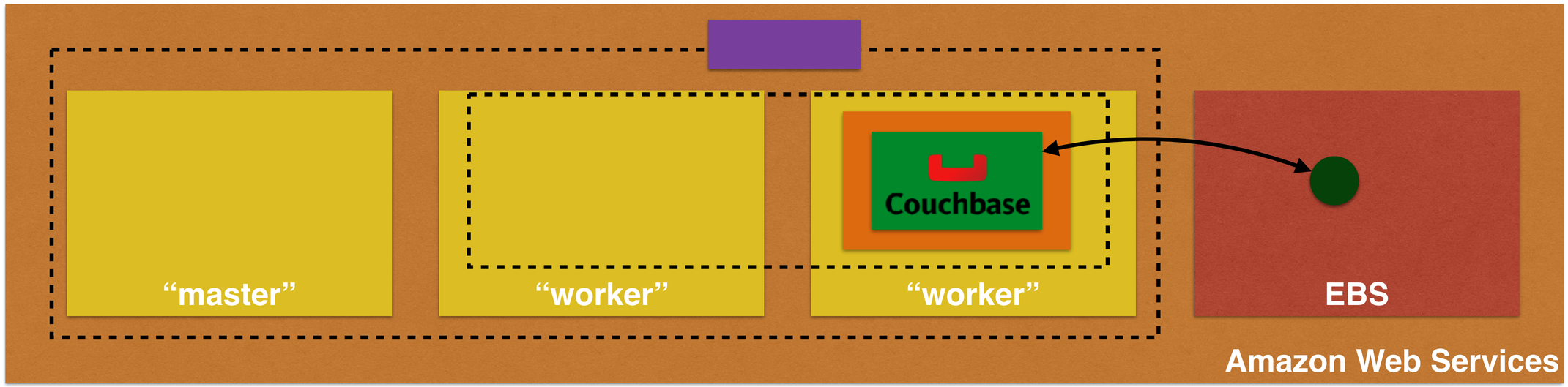


Stateful Containers

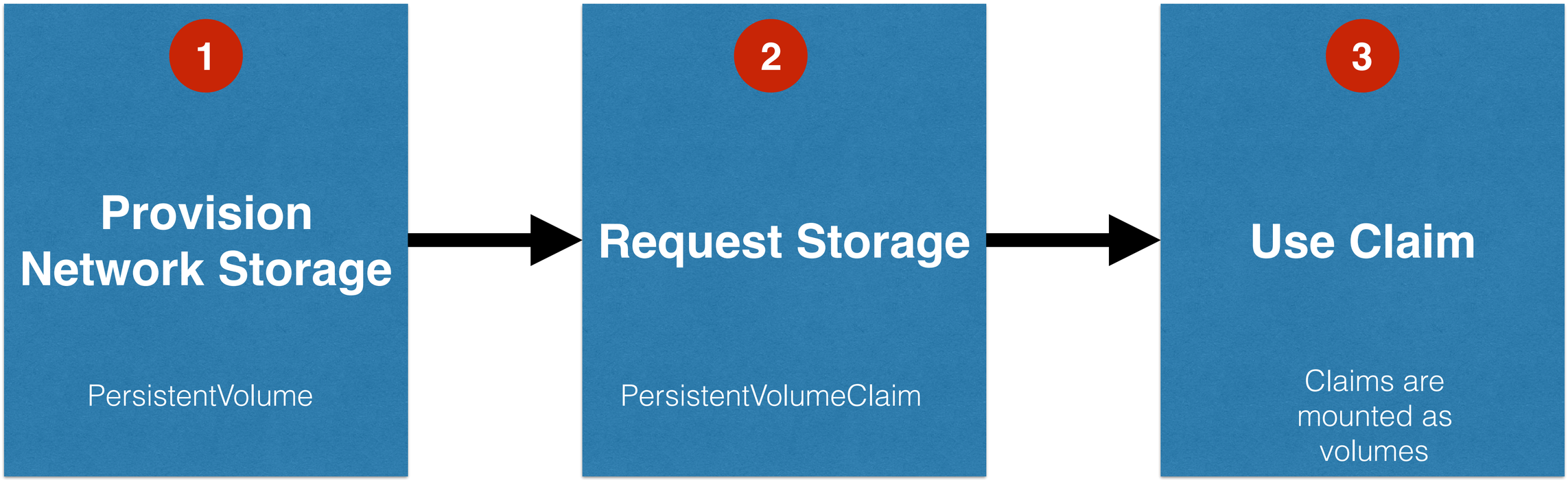


http://blog.couchbase.com/2016/july/stateful-containers-kubernetes-amazon-ebs

Stateful Containers

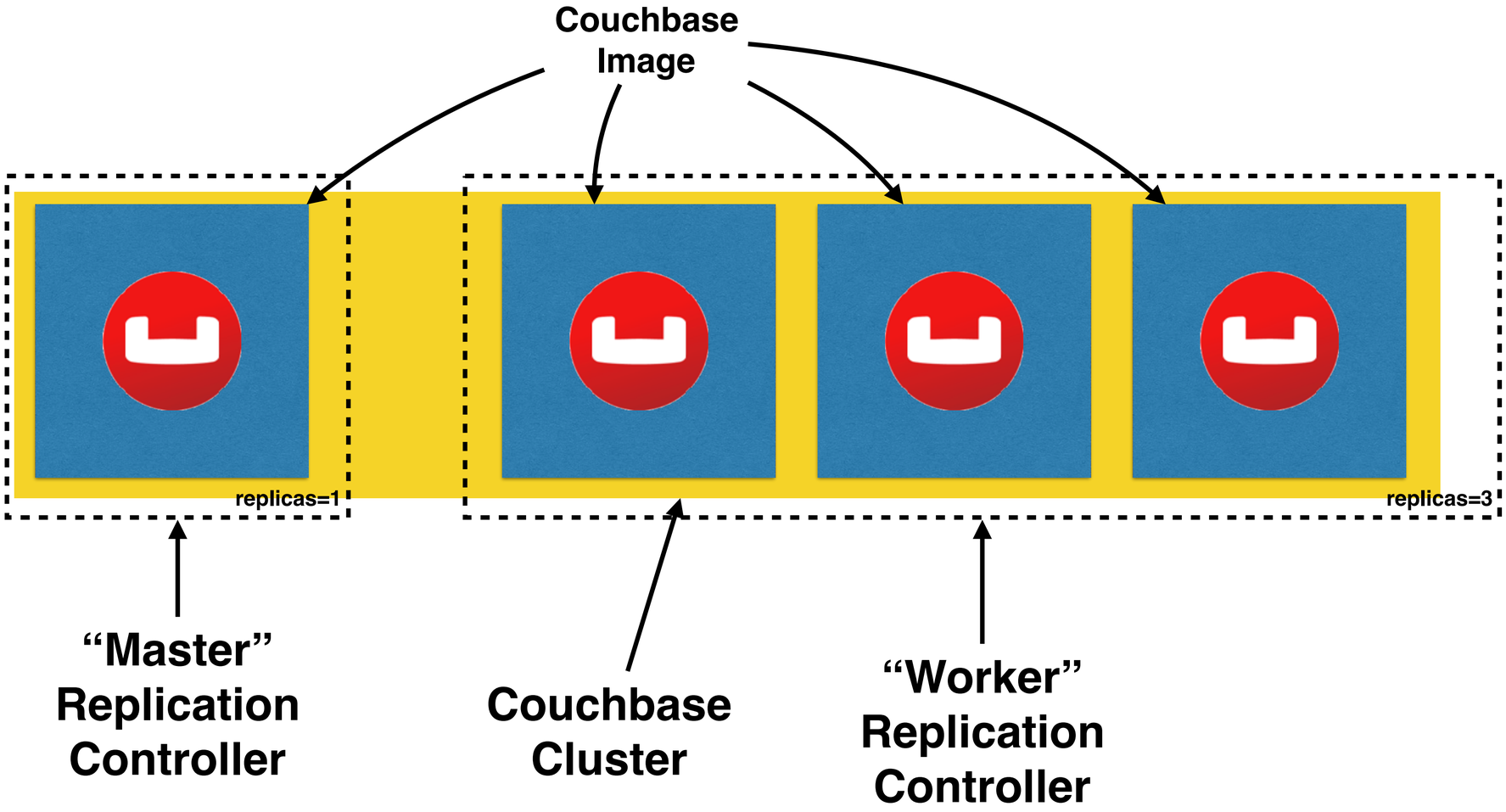


Persistent Volume



http://blog.couchbase.com/2016/july/stateful-containers-kubernetes-amazon-ebs

Couchbase Cluster on Kubernetes

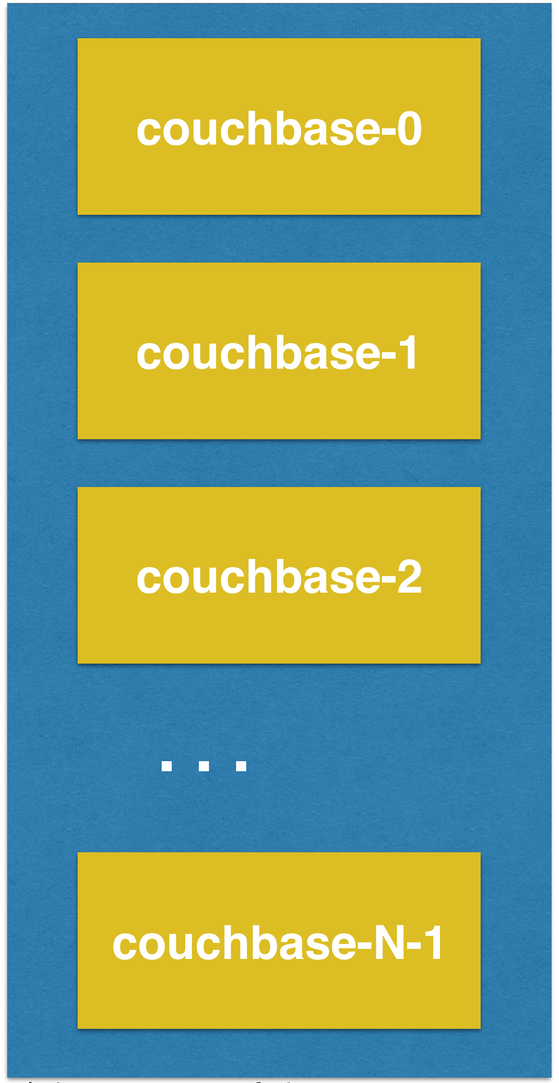


https://github.com/arun-gupta/couchbase-kubernetes/tree/master/cluster

http://blog.kubernetes.io/2016/08/create-couchbase-cluster-using-kubernetes.html

Stateful Set

- `PetSet` introduced as Alpha resource in 1.3
- Renamed to `StatefulSet` in 1.5, upgraded to Beta
- Stateful pods
- StatefulSet has 0..N-1 Pods
- Each Pod has a deterministic name, and a unique identity
  - stable hostname
  - ordinal index
  - stable storage linked to ordinal & hostname
- Each StatefulSet has at most one Pod with a given identity



https://github.com/arun-gupta/couchbase-kubernetes/tree/master/cluster-statefulset

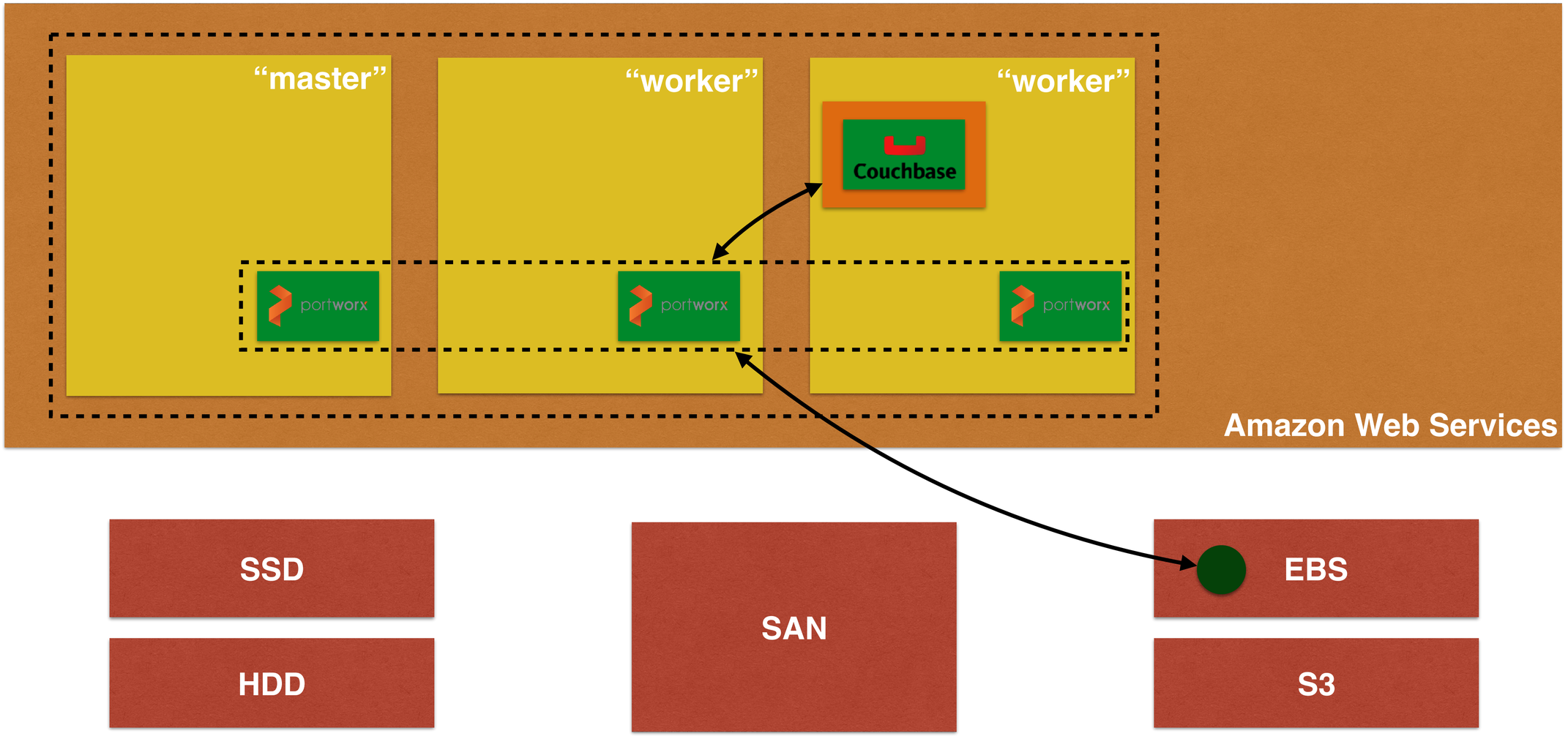
```yaml
apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
  name: couchbase
spec:
  serviceName: "couchbase"
  replicas: 2
  template:
    metadata:
      labels:
        app: couchbase
    spec:
      terminationGracePeriodSeconds: 0
      containers:
      - name: couchbase
        image: arungupta/couchbase:k8s-statefulset
        ports:
        - containerPort: 8091
        volumeMounts:
        - name: couchbase-data
          mountPath: /opt/couchbase/var
        env:
          - name: COUCHBASE_MASTER
            value: "couchbase-0.couchbase.default.svc.cluster.local"
          - name: AUTO_REBALANCE
            value: "false"
  volumeClaimTemplates:
  - metadata:
      name: couchbase-data
      annotations:
        volume.alpha.kubernetes.io/storage-class: anything
    spec:
      accessModes: [ "ReadWriteOnce" ]
      resources:
        requests:
          storage: 1Gi

```

Portworx

- Data services for containers
- Deploys as a container
- Automated with scheduler to pool capacity
  - Kubernetes, Swarm and DC/OS
- Multi-cluster visibility and management
- Replication and snapshots
- REST API and CLI

Stateful Containers with Portworx



https://github.com/arun-gupta/couchbase-kubernetes/tree/master/cluster-petset-portworx

Multitenancy - 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
- Anti-pattern: Separate slightly different resources, e.g. different version
  - Use labels instead

Batch Jobs

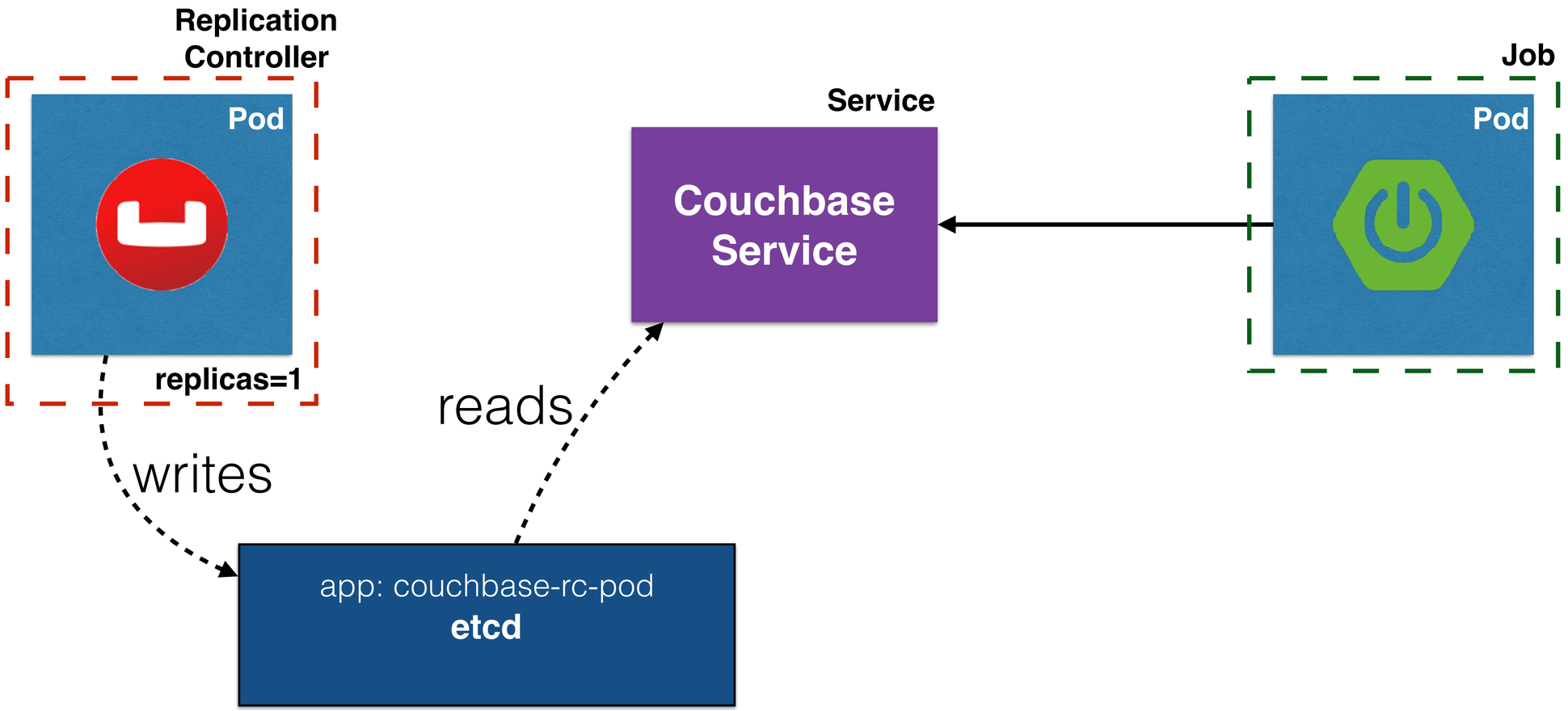
- Run-once jobs
  - Replication Controller, Replica Set or Deployments not suitable
- Three types
  - Non-parallel: only one pod is started
  - Parallel:
    - With a fixed completion count
    - With a work queue

http://kubernetes.io/docs/user-guide/jobs/

Run-Once Job Specification

```
1  apiVersion: batch/v1
2  kind: Job
3  metadata:
4    name: wait
5  spec:
6    template:
7      metadata:
8        name: wait
9      spec:
10       containers:
11       - name: wait
12         image: ubuntu
13         command: ["sleep",  "20"]
14       restartPolicy: Never
```

Java Application in Kubernetes

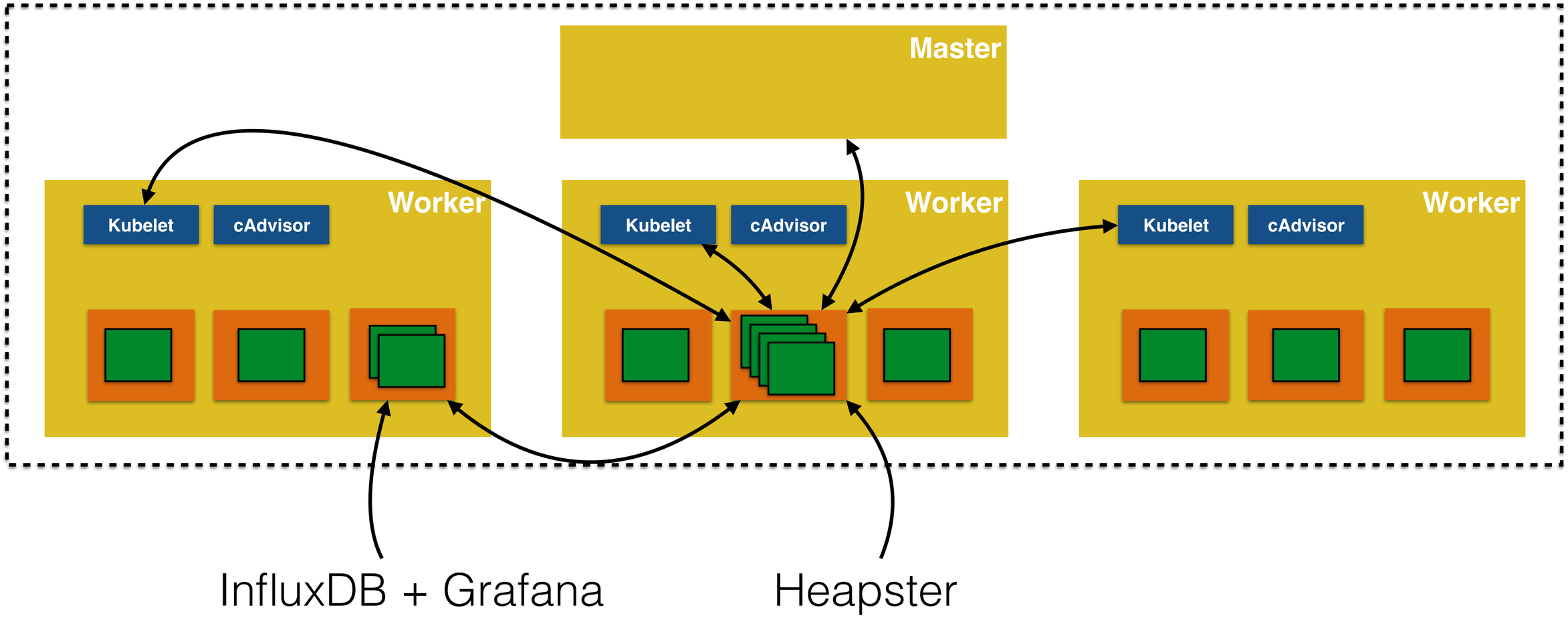


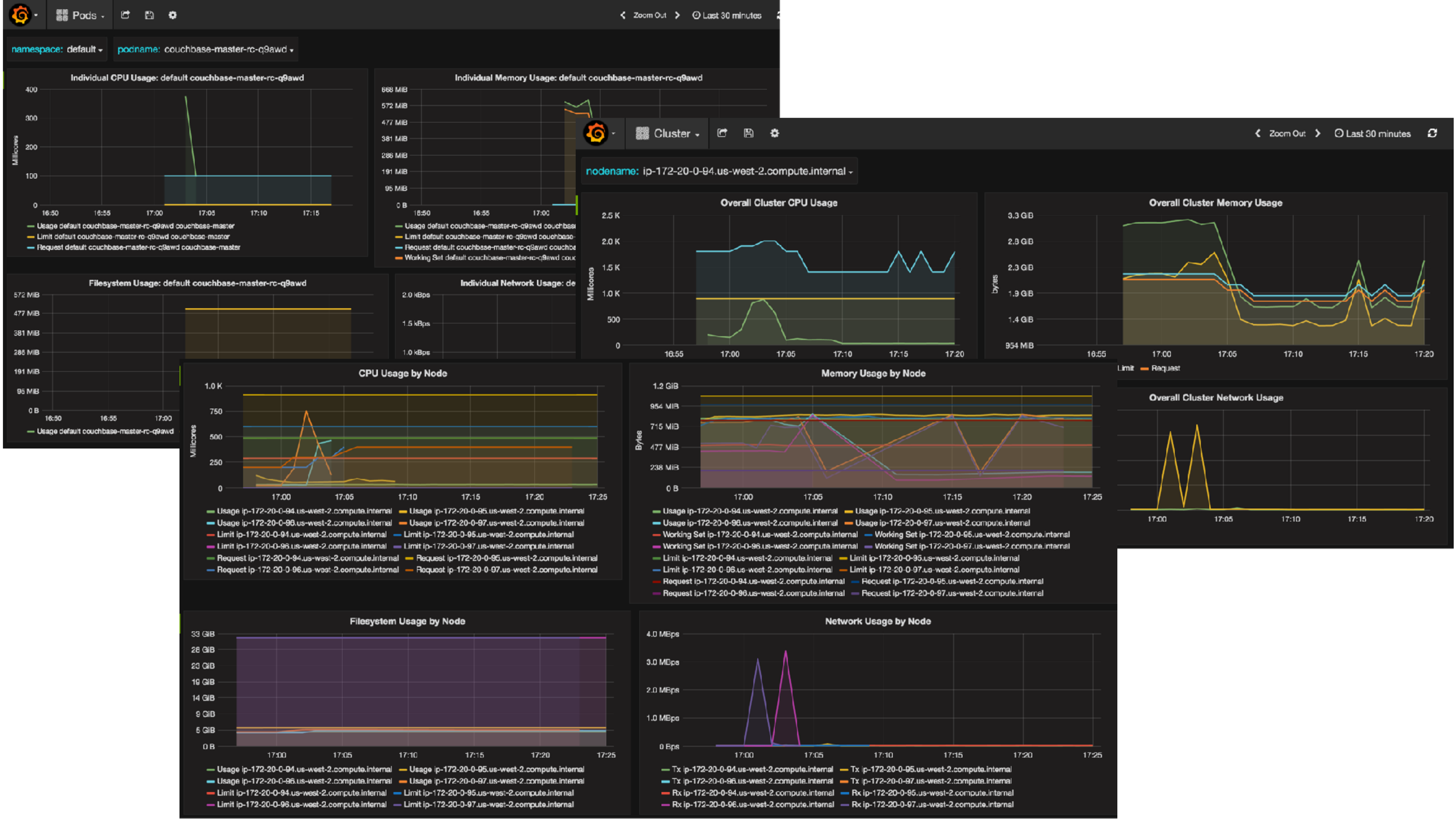
Health Checks

- Health checks performed by Kubelet
- Probes - Liveness and Readiness
- Diagnostics
  - Container Exec
  - TCP Socket
  - HTTP

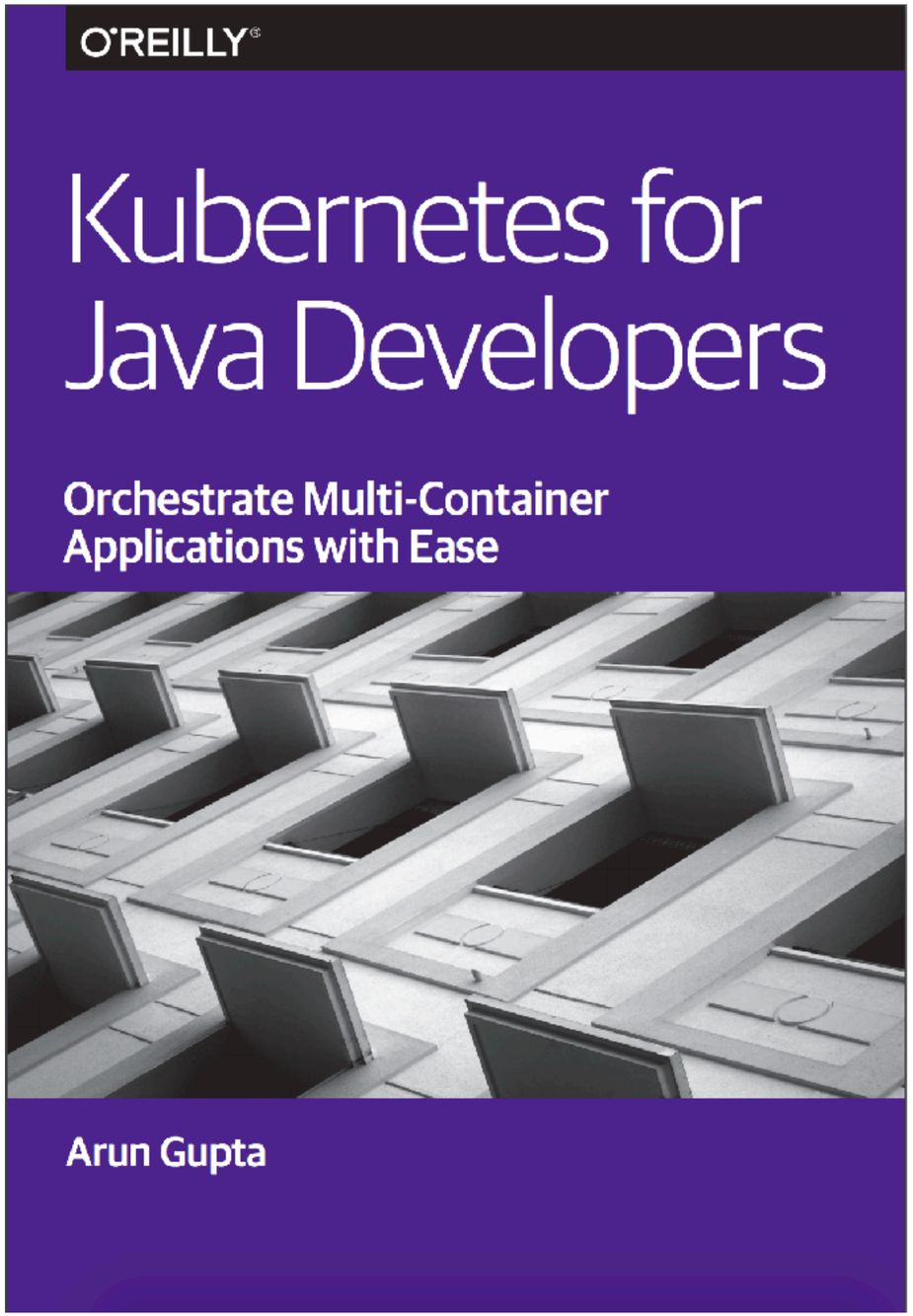
```yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: couchbase
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: couchbase
    spec:
      containers:
      - name: couchbase
        image: arungupta/couchbase
        ports:
        - containerPort: 8091
        livenessProbe:
          httpGet:
            path: /pools
            port: 8091
          initialDelaySeconds: 30
          timeoutSeconds: 1
```

Monitoring Kubernetes Resources




Pods
Zoom Out
Last 30 minutes
namespace: default
podname: couchbase-master-rc-q9awd
Individual CPU Usage: default couchbase-master-rc-q9awd
Millicores
Usage default couchbase-master-rc-q9awd couchbase-master
Limit default couchbase-master-rc-q9awd couchbase-master
Request default couchbase-master-rc-q9awd couchbase-master
Individual Memory Usage: default couchbase-master-rc-q9awd
Filesystem Usage: default couchbase-master-rc-q9awd
Individual Network Usage: de
Cluster
nodename: ip-172-20-0-94.us-west-2.compute.internal
Overall Cluster CPU Usage
Overall Cluster Memory Usage
bytes
Limit
Request
CPU Usage by Node
Memory Usage by Node
Overall Cluster Network Usage
Filesystem Usage by Node
Network Usage by Node

Kubernetes Monitoring


O'REILLY
Kubernetes for Java Developers
Orchestrate Multi-Container Applications with Ease
Arun Gupta

References

- kubernetes.io
- github.com/arun-gupta/kubernetes-java-sample