RED HAT®
DEVELOPER
PROGRAM

Kubernetes Introduction

 @rafabene

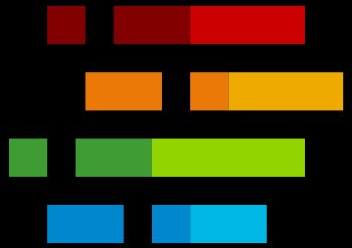
 benevides@redhat.com

Link  http://bit.ly/kubernetes-intro

Rafael Benevides
Director of Developer Experience at Red Hat
Apache DeltaSpike P.M.C

benevides@redhat.com
@rafabene

Java Certifications:
SCJA / SCJP / SCWCD / SCBCD / SCEA
JBoss Certifications:
JBCD / JBCAA
Red Hat Certifications:
OpenShift / Containers / Ansible
Other Certifications:
SAP Netweaver / ITIL / IBM Software Quality

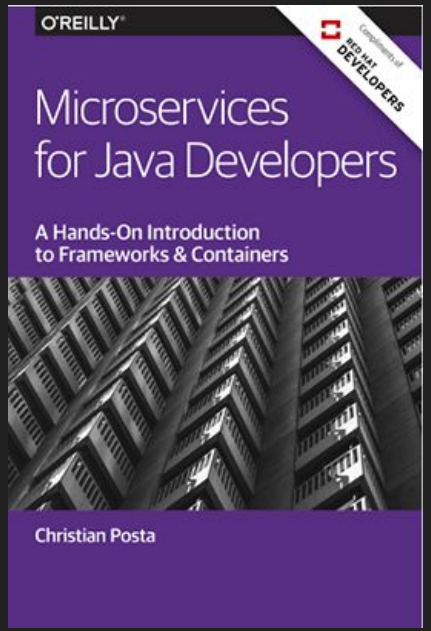


RED HAT® DEVELOPER

Get software and know-how.
Get started with Red Hat technologies.

Join at **developers.redhat.com**.

bit.ly/javamicroservicesbook

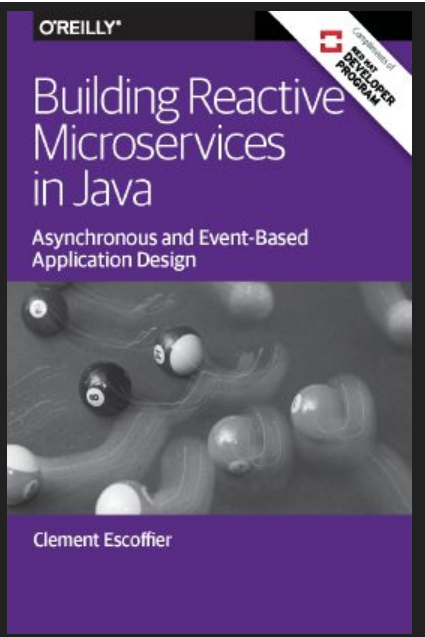


Free eBooks from developers.redhat.com

Microservices Introductory Materials

Demo: bit.ly/msa-instructions
Slides: bit.ly/microservicesdeepdive
Video Training: bit.ly/microservicesvideo
Kubernetes for Java Developers

bit.ly/reactivemicroservicesbook



Advanced Materials

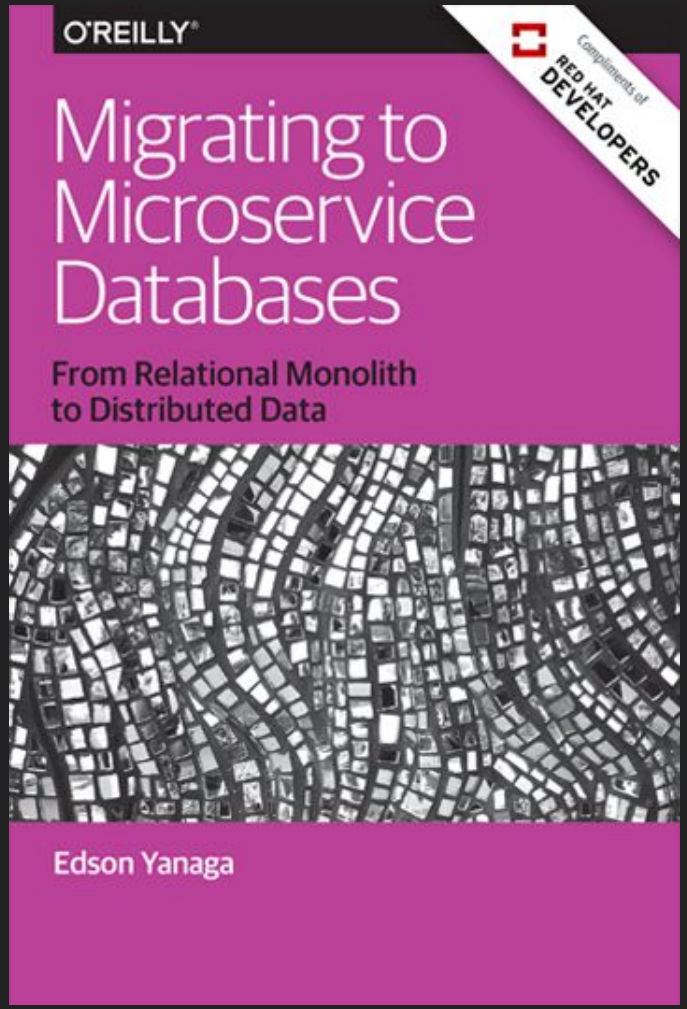
bit.ly/istio-tutorial
learn.openshift.com/servicemesh

bit.ly/faas-tutorial
learn.openshift.com/serverless

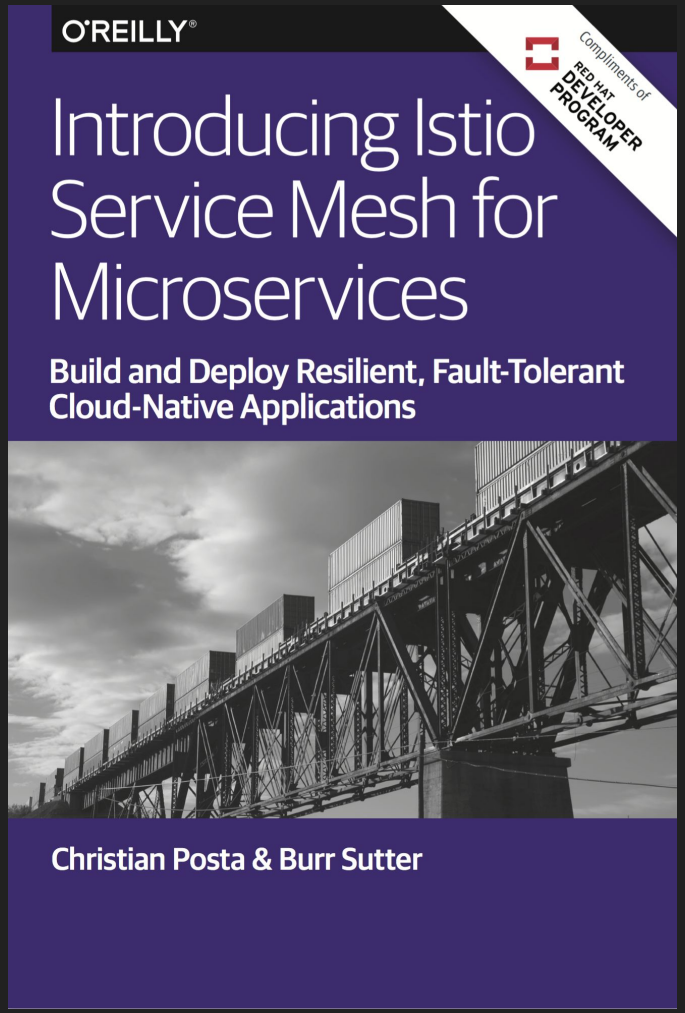
http://bit.ly/kubernetes-intro

@rafabene

RED HAT DEVELOPER PROGRAM



bit.ly/mono2microdb



bit.ly/istio-book

Why do you want to run your application inside containers?

HelloWorld (~/javastarter) - gedit

Open Save Undo

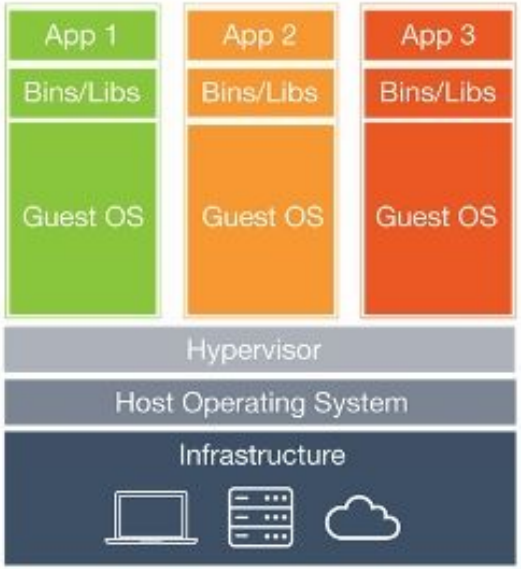
HelloWorld

```
// A small Java program
public class HelloWorld {
        public static void main(String[] args) {
                System.out.println("Hello World");
        }
}
```

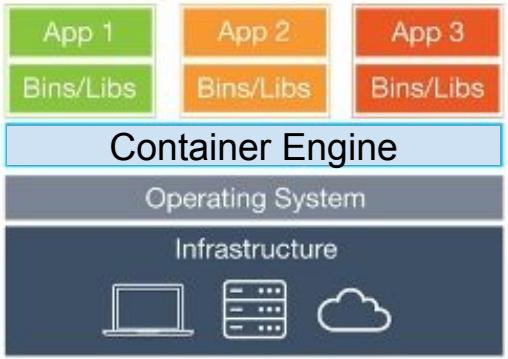
C Tab Width: 8 Ln 6, Col 2

Container Advantages

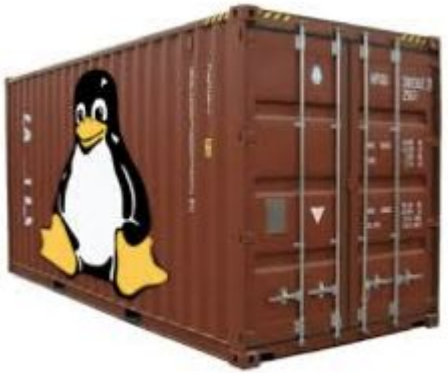
- Lightweight footprint and minimal overhead,
- Portability across machines,
- Simplify DevOps practices,
- Speeds up Continuous Integration,
- Empower Microservices Architectures.
- Isolation



Virtual Machines

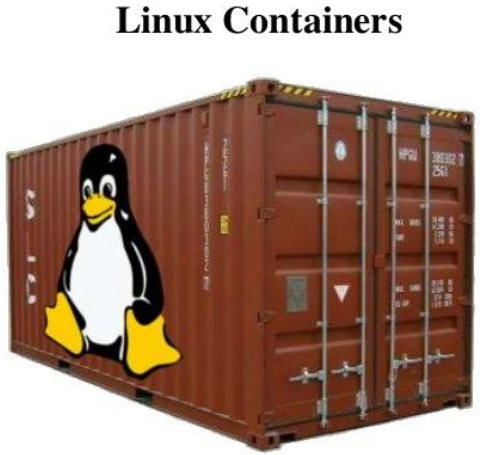


Containers



A way to run a Linux container:

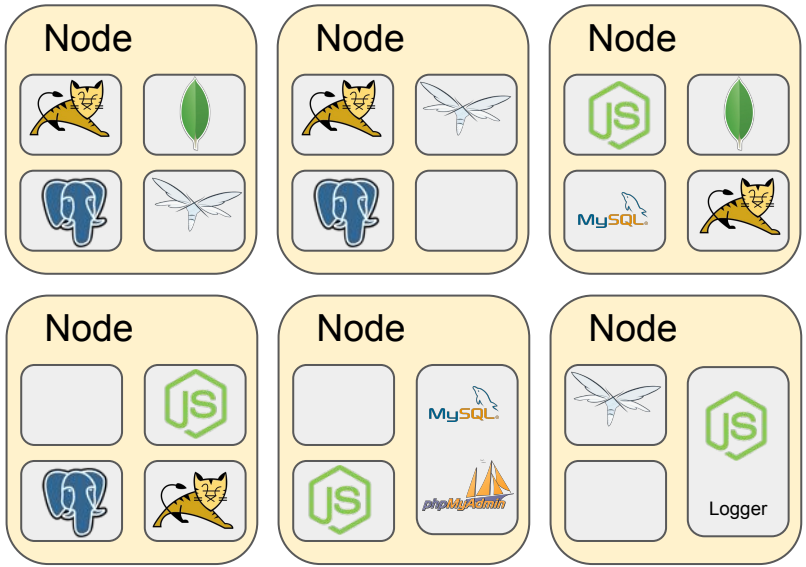
```
$ docker run -d <image-name>
```



A single and isolated Linux process running in a single machine

DevOps challenges for multiple containers

- How to scale?
- How to avoid port conflicts?
- How to manage them in multiple hosts?
- What happens if a host has a trouble?
- How to keep them running?
- How to update them?
- Where are my containers?



Meet Kubernetes

Greek for *“Helmsman”*; also the root of the word *“Governor”* *(from latin: gubernator)*

- Container **orchestrator**
- Supports **multiple cloud** and **bare-metal** environments
- Inspired by Google’s experience with containers
- **Open source**, written in **Go**

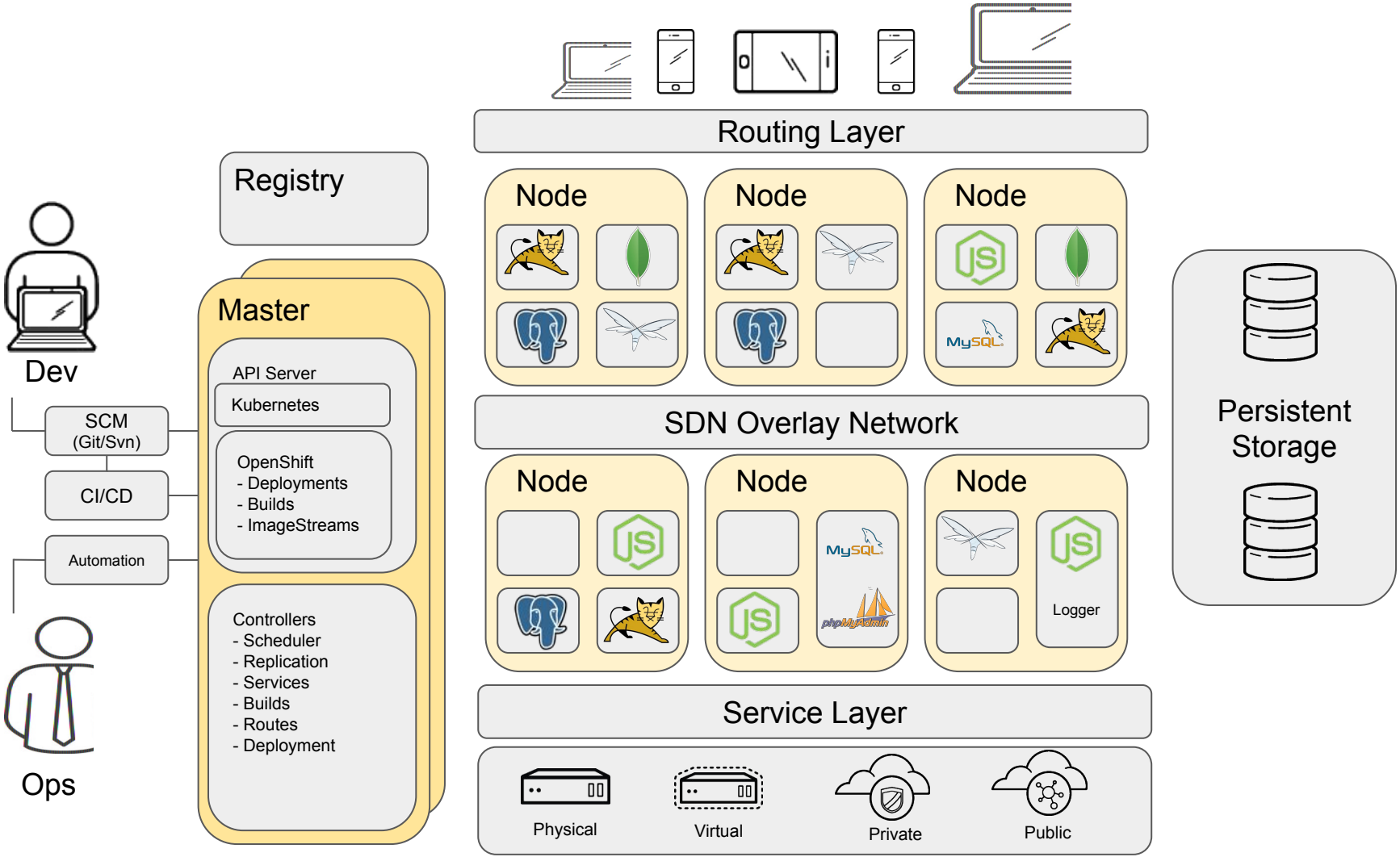
Manage **applications**, not machines

Open Source community

Project	Partners	
Version 1.11	**Red Hat**	CoreOS
Hosted on GitHub	HP	Pivotal
1700+ contributors	IBM	SaltStack
67,000+ commits	Mesosphere	VMWare
38,000+ GitHub stars	Microsoft	

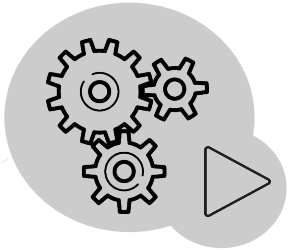
http://kubernetes.io/

https://github.com/kubernetes/kubernetes


Routing Layer
Registry
Node
Node
Node
Dev
Master
API Server
Kubernetes
SCM
(Git/Svn)
CI/CD
Automation
OpenShift
- Deployments
- Builds
- ImageStreams
Controllers
- Scheduler
- Replication
- Services
- Builds
- Routes
- Deployment
SDN Overlay Network
Node
Node
Node
Logger
Persistent
Storage
Service Layer
Physical
Virtual
Private
Public
Ops

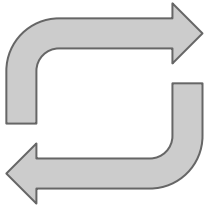
Kubernetes Concepts

Pod



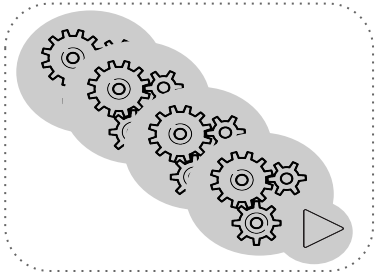
One or More Containers
Shared IP
Shared Storage Volume
Shared Resources
Shared Lifecycle

Replication Controller / Deployment



Ensures that a specified number of pod replicas are running at any one time

Service



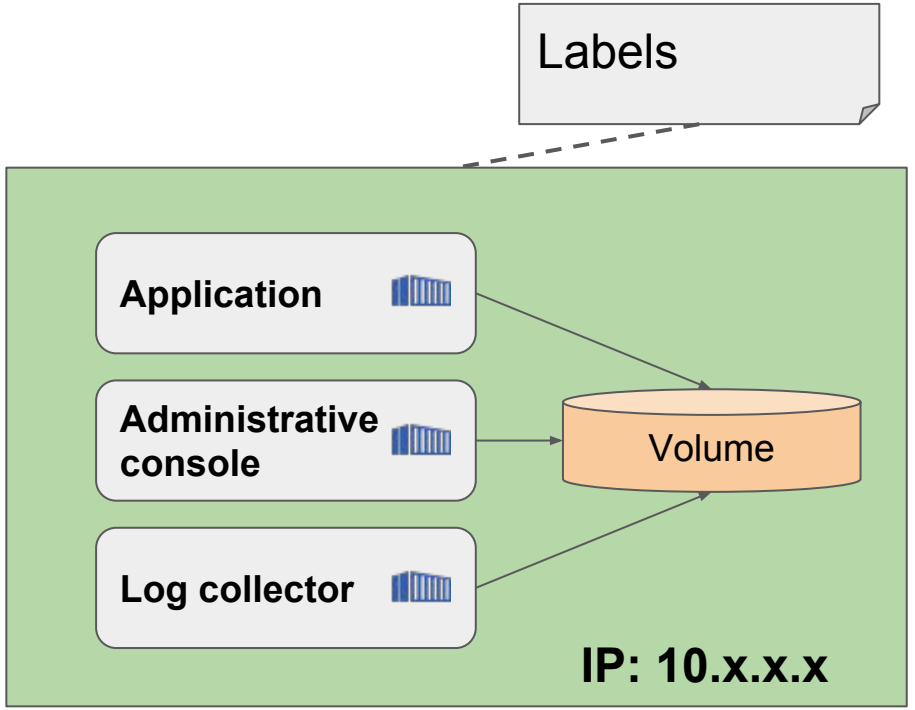
Grouping of pods, act as one, has stable virtual IP and DNS name

Label

Key/Value pairs associated with Kubernetes objects (e.g. env=production)

Concept: Pod

- Group of containers
- Live and die together
- Share:
  - IP
  - Secrets
  - Labels *
  - Volumes *

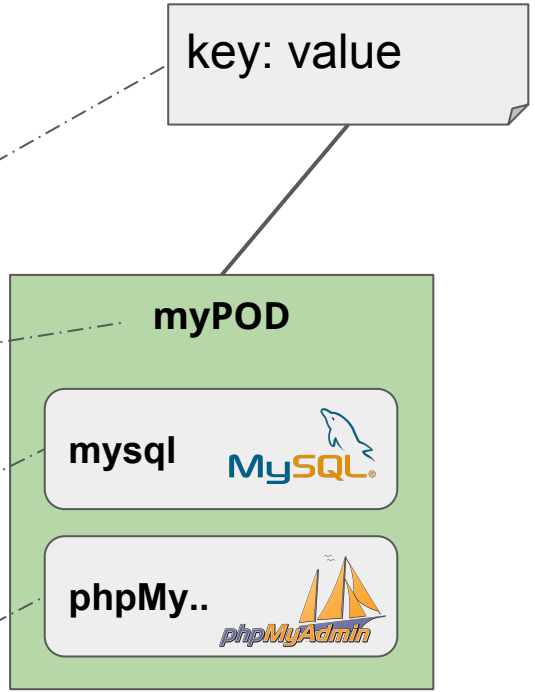


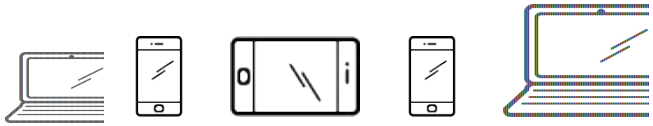
** we will talk about these concepts later*

Concept: POD

Defining a POD as YAML:

```
apiVersion: v1
kind: Pod
metadata:
  name: myPod
  labels:
    key: value
spec:
  containers:
    - name: mysql
      image: username/image
    - name: phpMyAdmin
      image: username/image2
```





Routing Layer

Registry

Dev

Ops

SCM (Git/Svn)

CI/CD

Automation

Master

API Server

Kubernetes

OpenShift
- Deployments
- Builds
- ImageStreams

Controllers
- Scheduler
- Replication
- Services
- Builds
- Routes
- Deployment

Node

Node

Node

SDN Overlay Network

Node

Node

Node

Logger

Service Layer

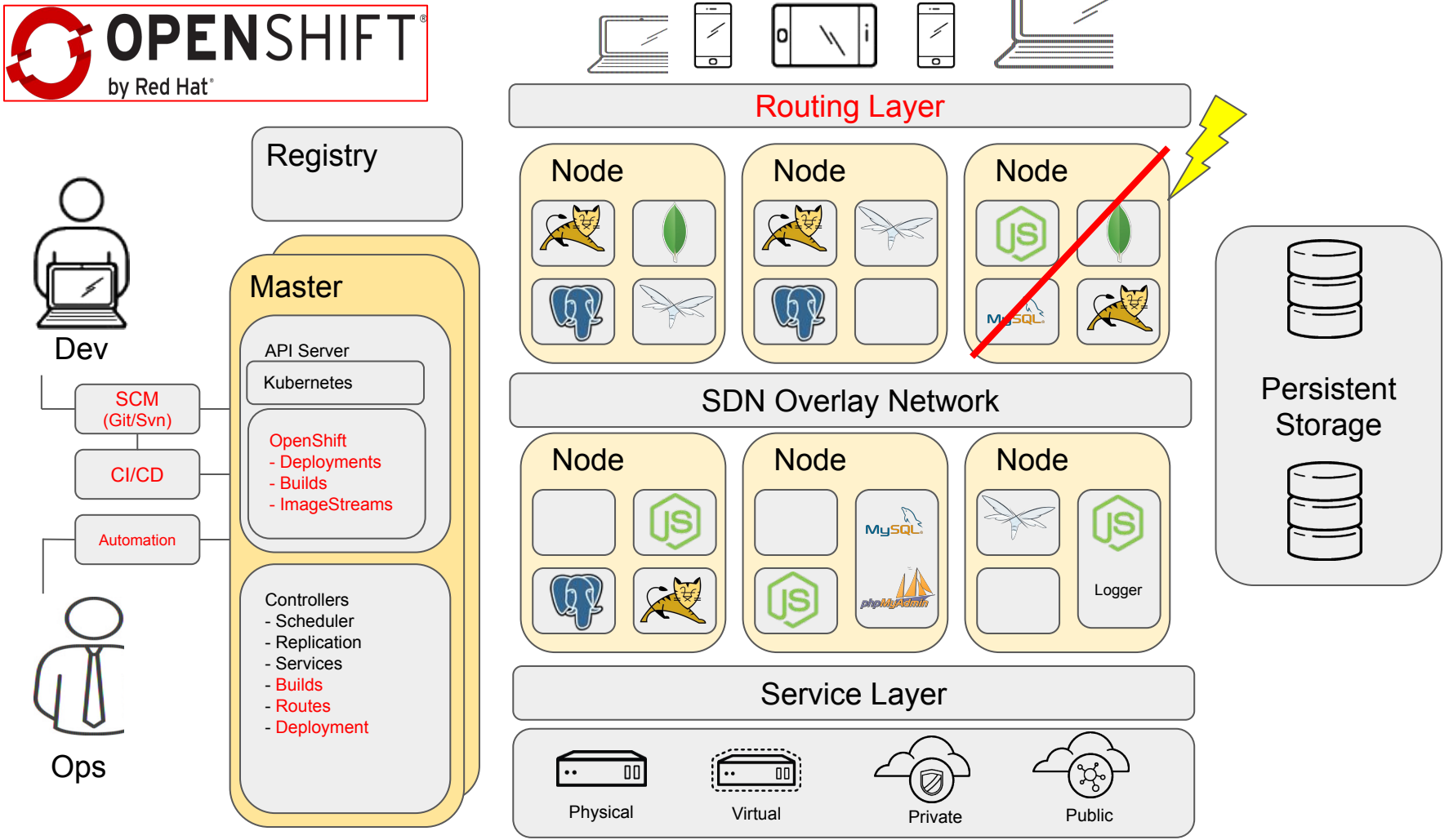
Physical

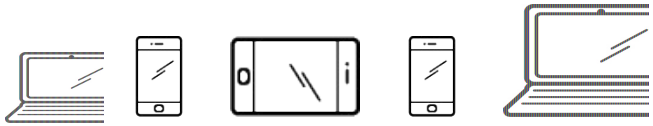
Virtual

Private

Public

Persistent Storage


OPENSHIFT
by Red Hat
Registry
Dev
Ops
SCM
(Git/Svn)
CI/CD
Automation
Master
API Server
Kubernetes
OpenShift
- Deployments
- Builds
- ImageStreams
Controllers
- Scheduler
- Replication
- Services
- Builds
- Routes
- Deployment
Routing Layer
Node
Node
Node
SDN Overlay Network
Node
Node
Node
MySQL
phpMyAdmin
Logger
Service Layer
Physical
Virtual
Private
Public
Persistent
Storage



Routing Layer

Registry

Dev

Ops

SCM (Git/Svn)

CI/CD

Automation

Master

API Server

Kubernetes

OpenShift
- Deployments
- Builds
- ImageStreams

Controllers
- Scheduler
- Replication
- Services
- Builds
- Routes
- Deployment

Node

Node

Node

SDN Overlay Network

Node

Node

Node

Logger

Persistent Storage

Service Layer

Physical

Virtual

Private

Public

Concept: Replication Controllers / Deployment

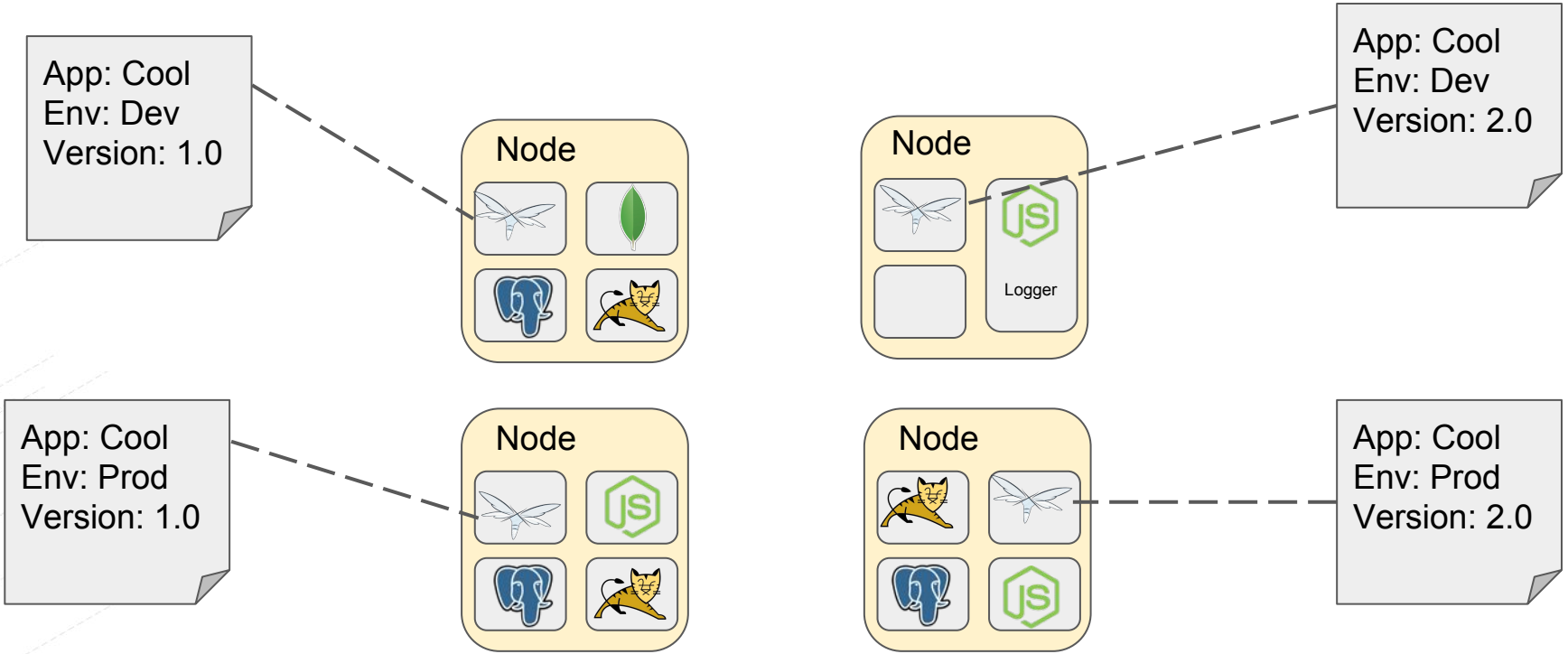
Defining a Deployment as YAML:

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: myDeployment
spec:
  replicas: 4
  template:
    metadata:
    spec:
```

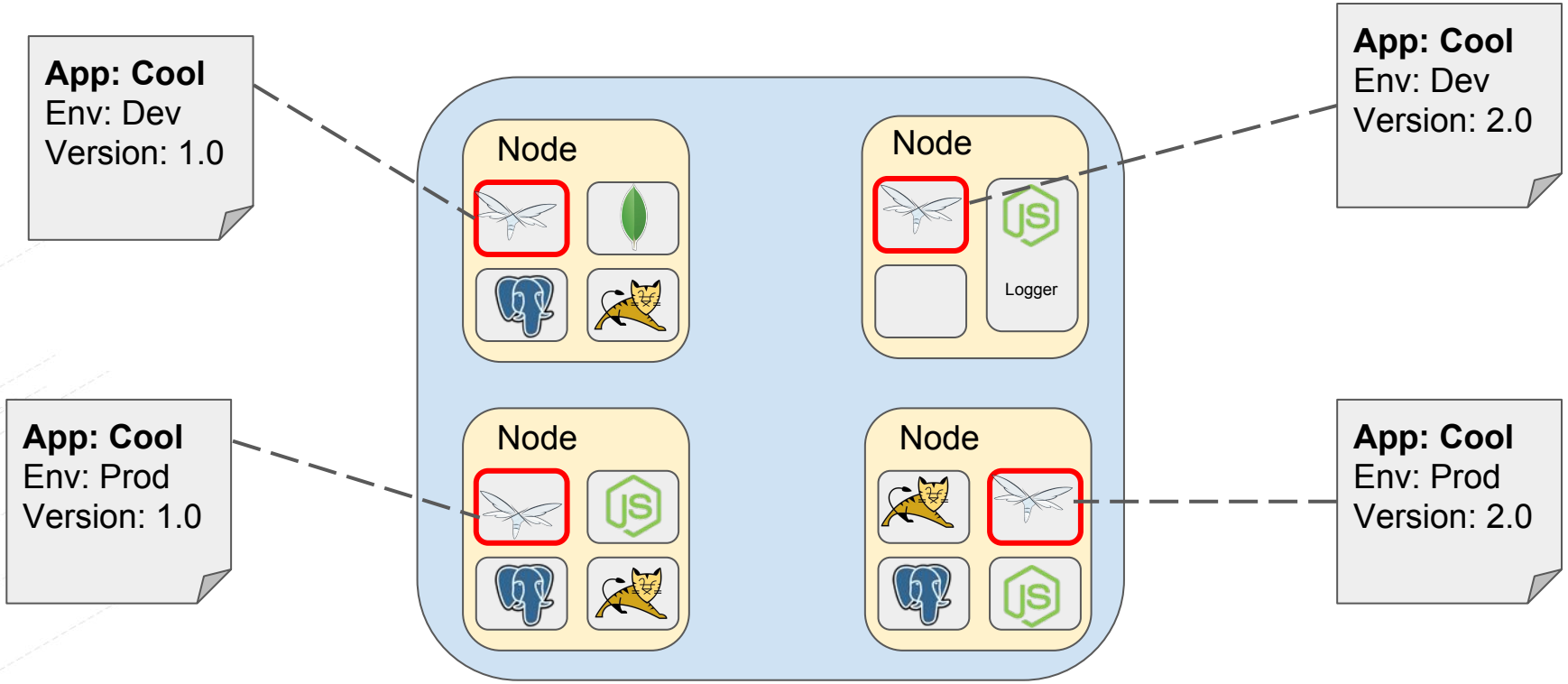
```
apiVersion: v1
kind: Pod
metadata:
  name: myPod
  labels:
    key: value
spec:
  containers:
    - name: myPod
      image: username/image
  ports:
      - name: http
        containerPort: 8080
```

Concept: Labels

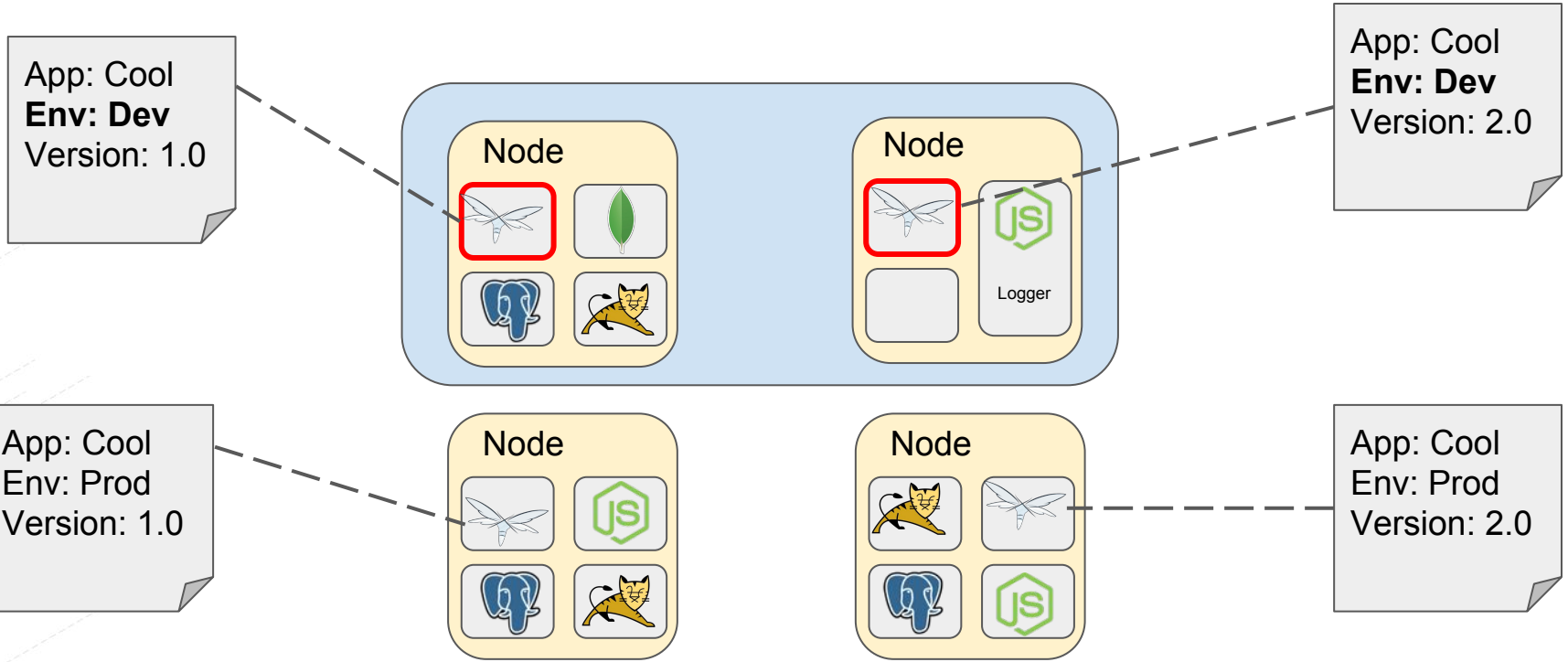
Everything in Kubernetes can have a label



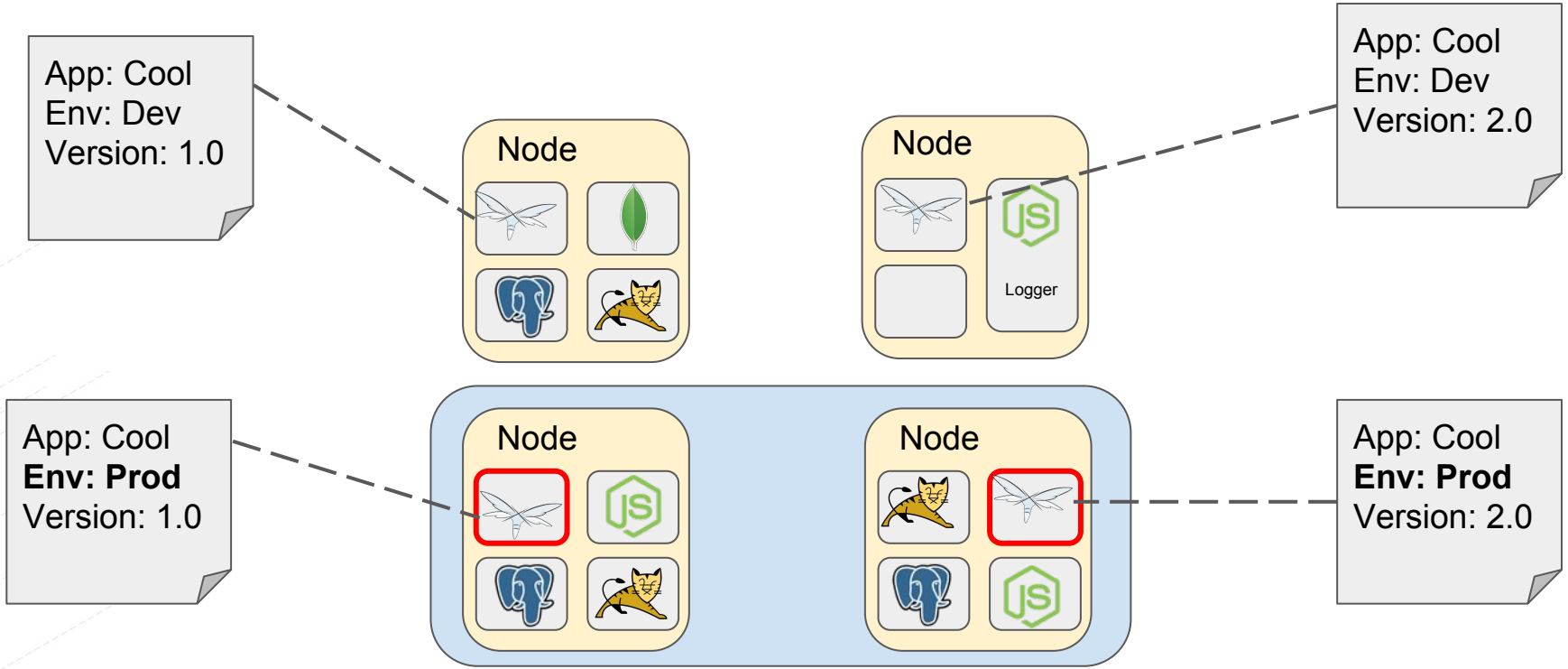
Concept: Labels



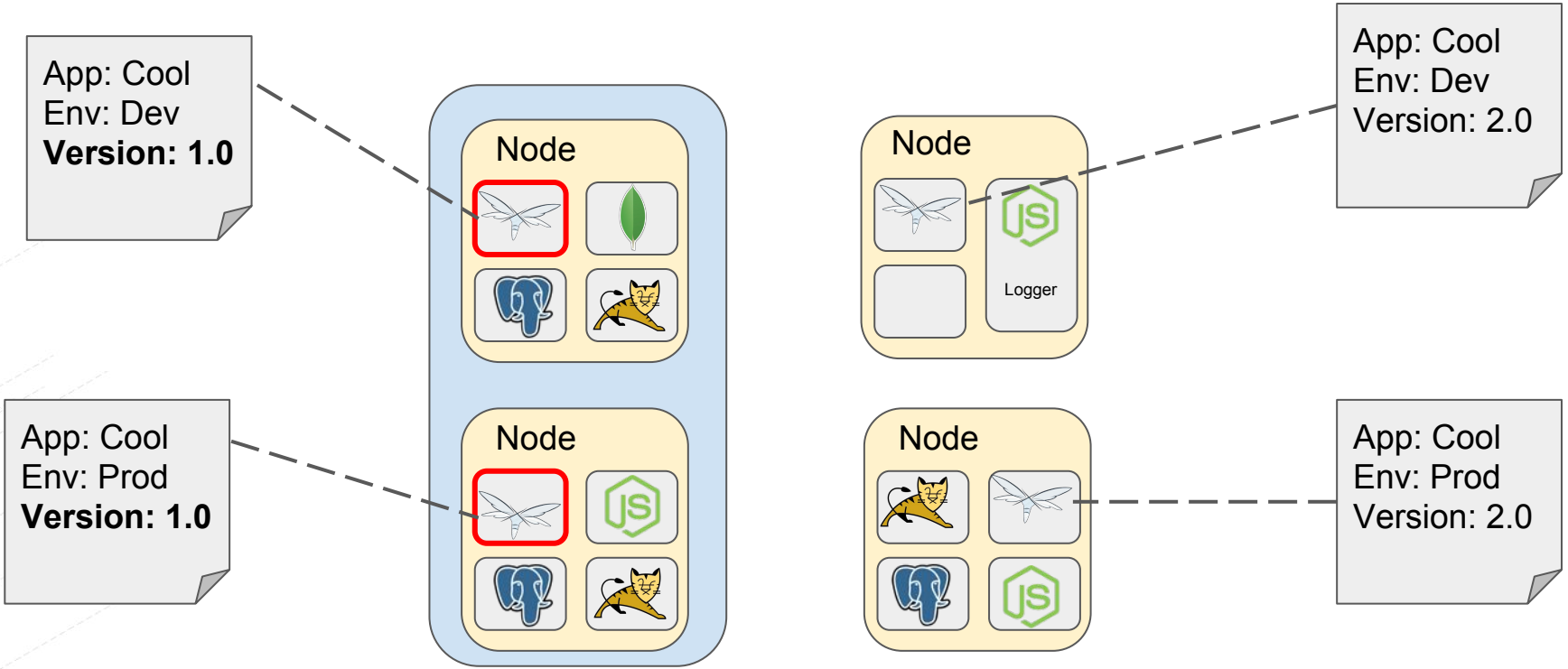
Concept: Labels



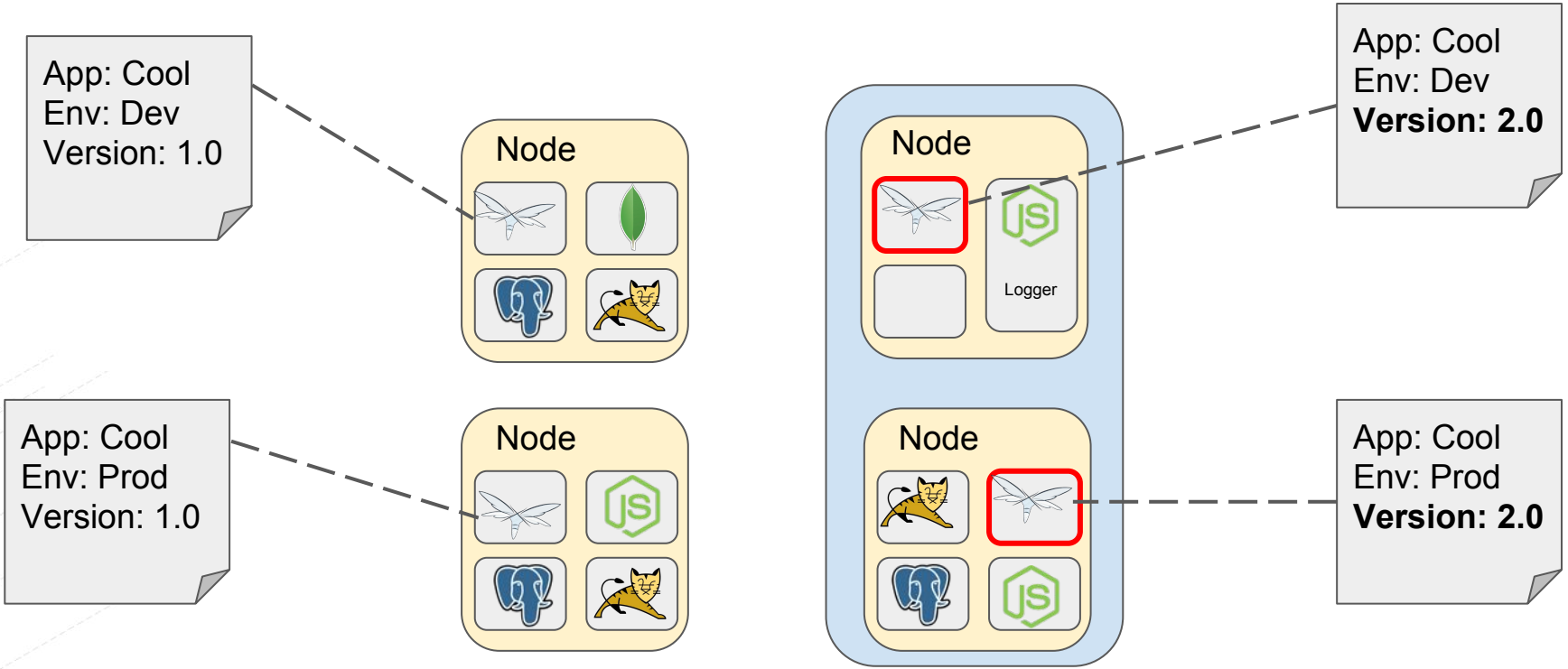
Concept: Labels



Concept: Labels



Concept: Labels



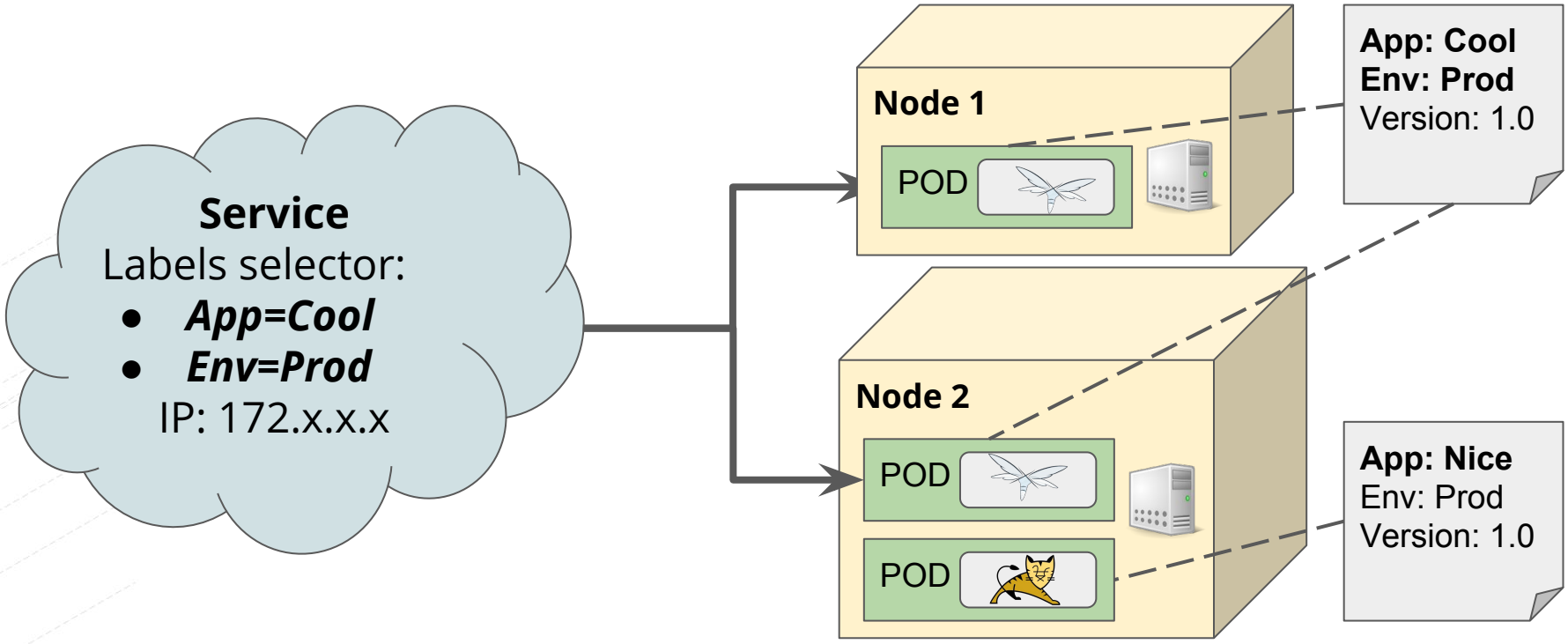
Concept: Labels

Defining Labels as YAML:
(can be placed in any object metadata)

```
metadata:
  name: objectName
  labels:
    App: Cool
    Env: Dev
    Version: 1.0
```

App: Cool
Env: Dev
Version: 1.0

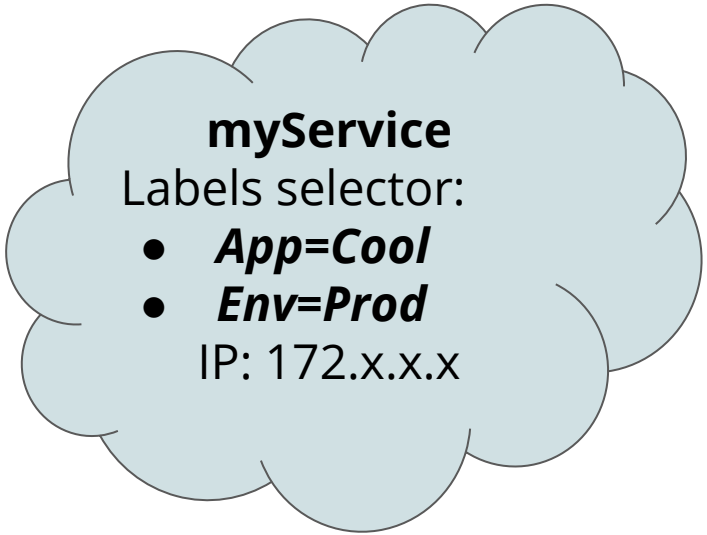
Concept: Services



Concept: Services

Defining a Service as YAML:

```
apiVersion: v1
kind: Service
metadata:
  name: myService
  labels:
        ...
spec:
  ports:
    - port: 80
      targetPort: 80
  selector:
    App: Cool
    Env: Prod
```



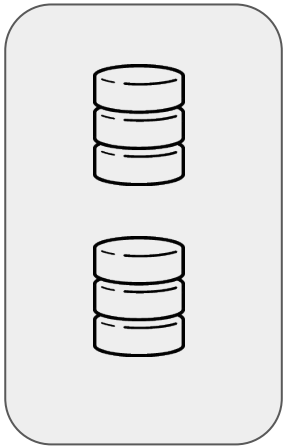
Service discovery inside Kubernetes

Using Environment variables:

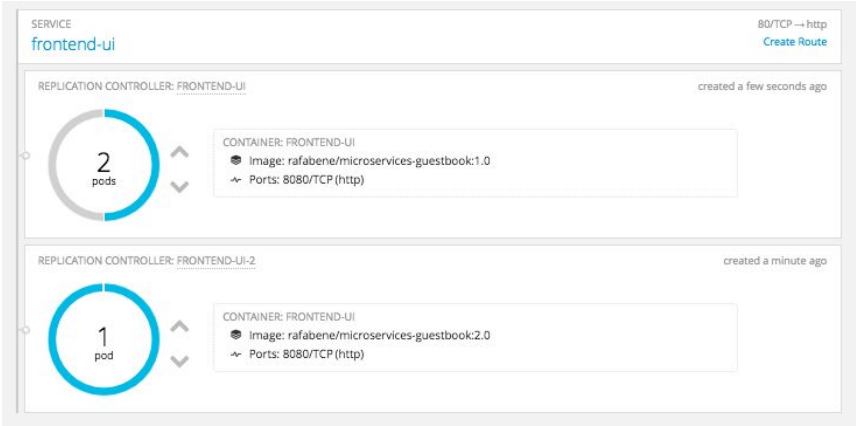
```
sh-4.2$ set|grep MYSQL
MYSQL_PORT=tcp://172.30.154.164:3306
MYSQL_PORT_3306_TCP=tcp://172.30.154.164:3306
MYSQL_PORT_3306_TCP_ADDR=172.30.154.164
MYSQL_PORT_3306_TCP_PORT=3306
MYSQL_PORT_3306_TCP_PROTO=tcp
MYSQL_SERVICE_HOST=172.30.154.164
MYSQL_SERVICE_PORT=3306
```

Using internal DNS: $ *ping mysql*

Other concepts



Persistent Volumes

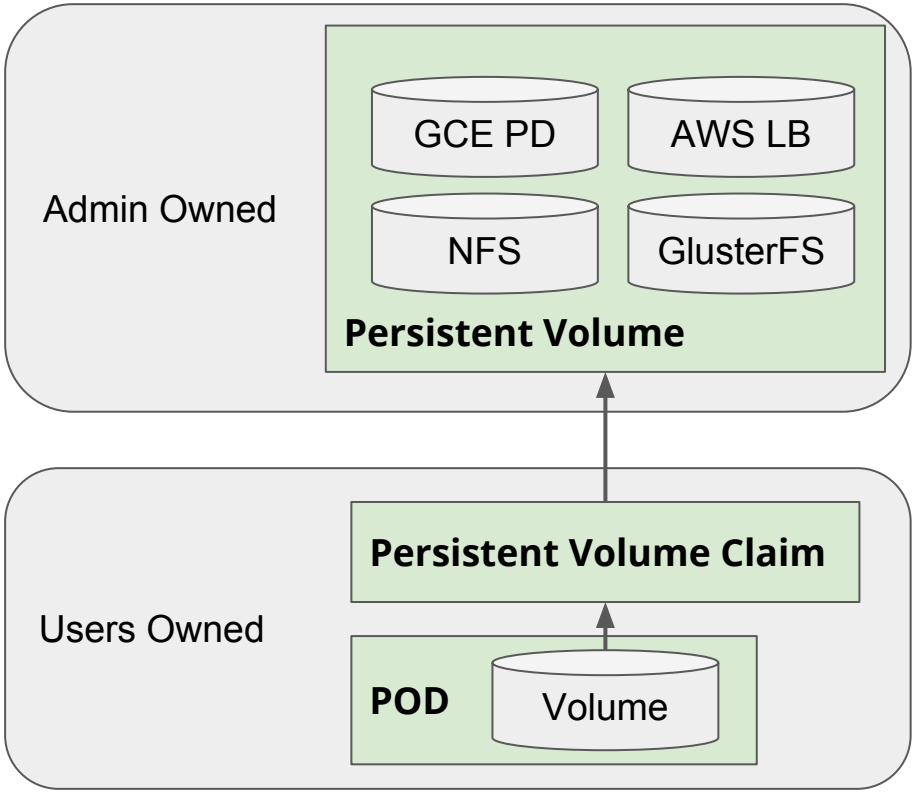


Rolling updates

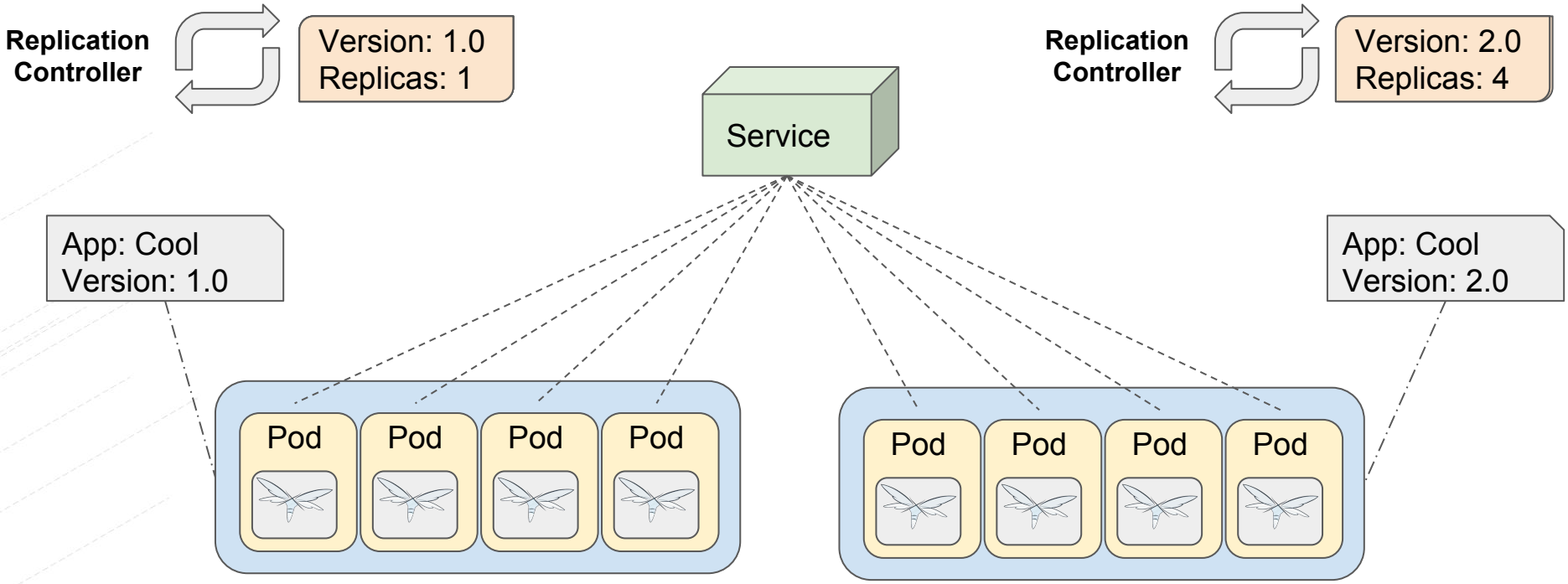
Concept: Persistent Volumes

- Admin provisions them, Users claim them
- High-level abstraction
- Pods can mount PVCs as Volumes

```
  volumeMounts:
    # name must match the volume name below
  - name: mysql-persistent-volume
    # mount path within the container
    mountPath: /var/lib/mysql/data
volumes:
- name: mysql-persistent-volume
  persistentVolumeClaim:
    claimName: mysql-pvc
```

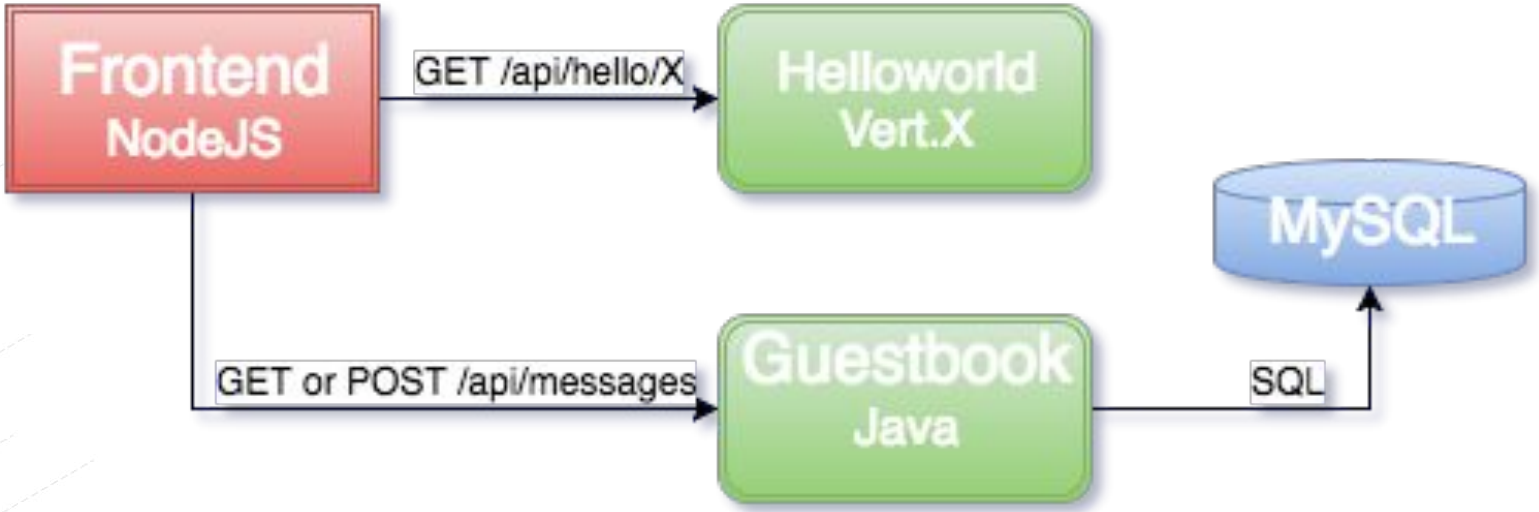


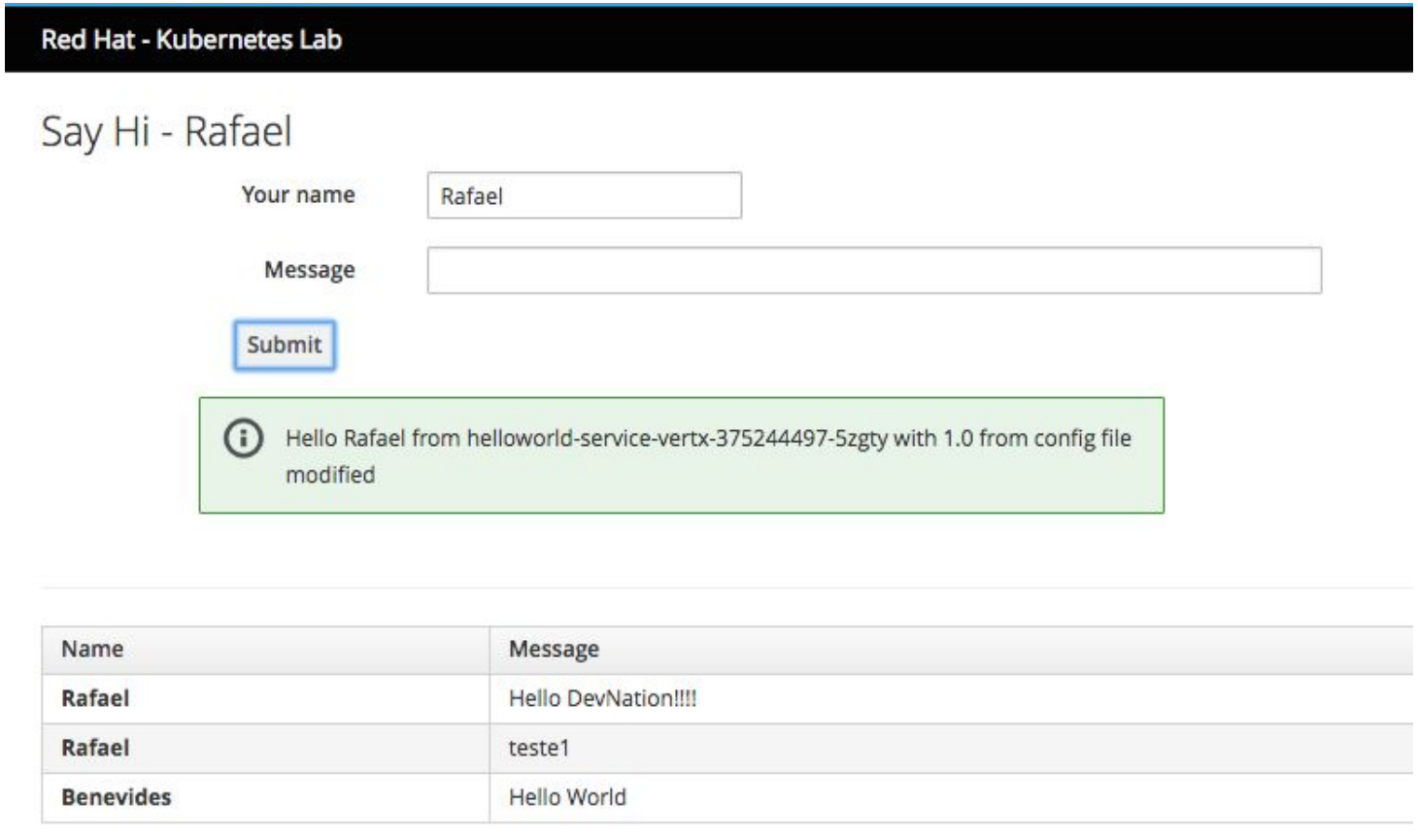
Deployment Concept: Rolling Updates

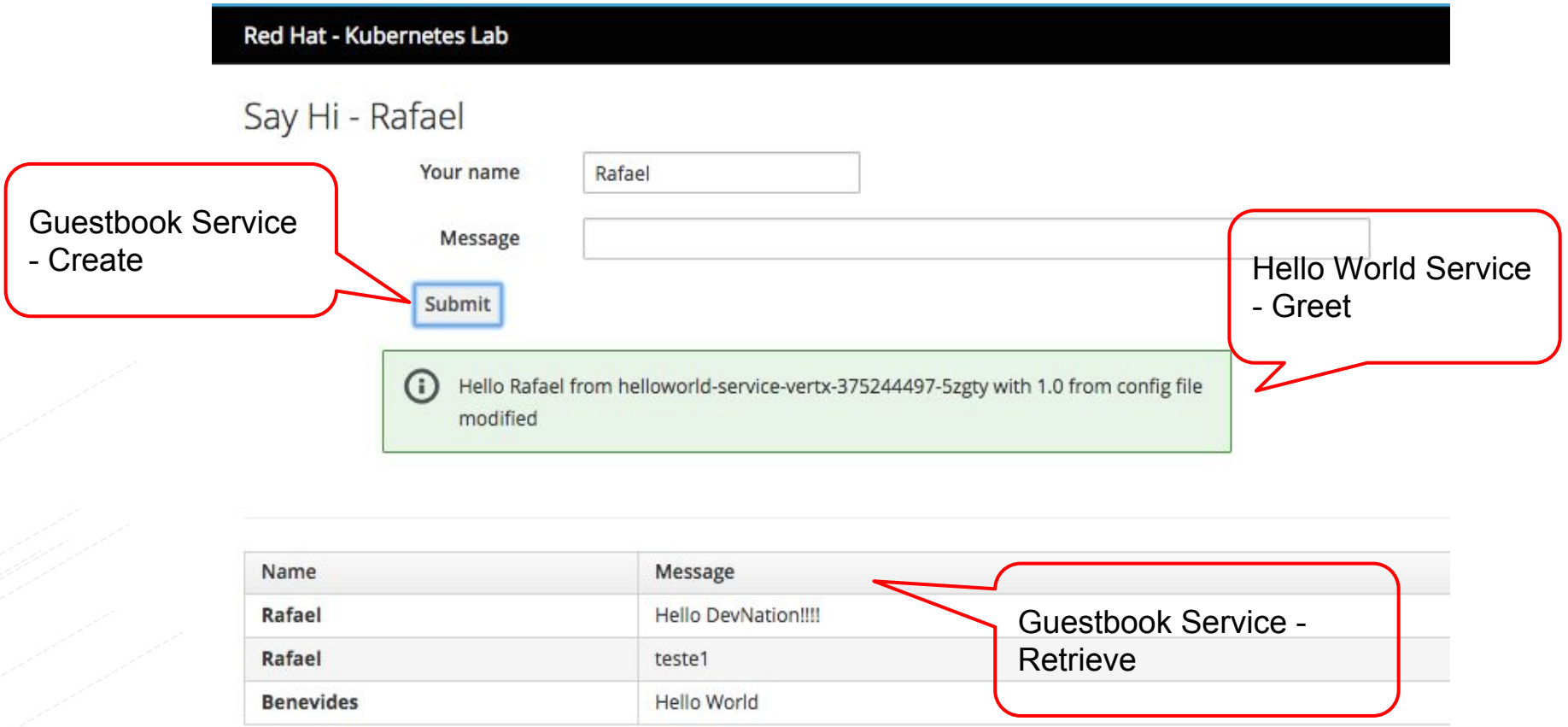


Kubernetes Example

Application Overview

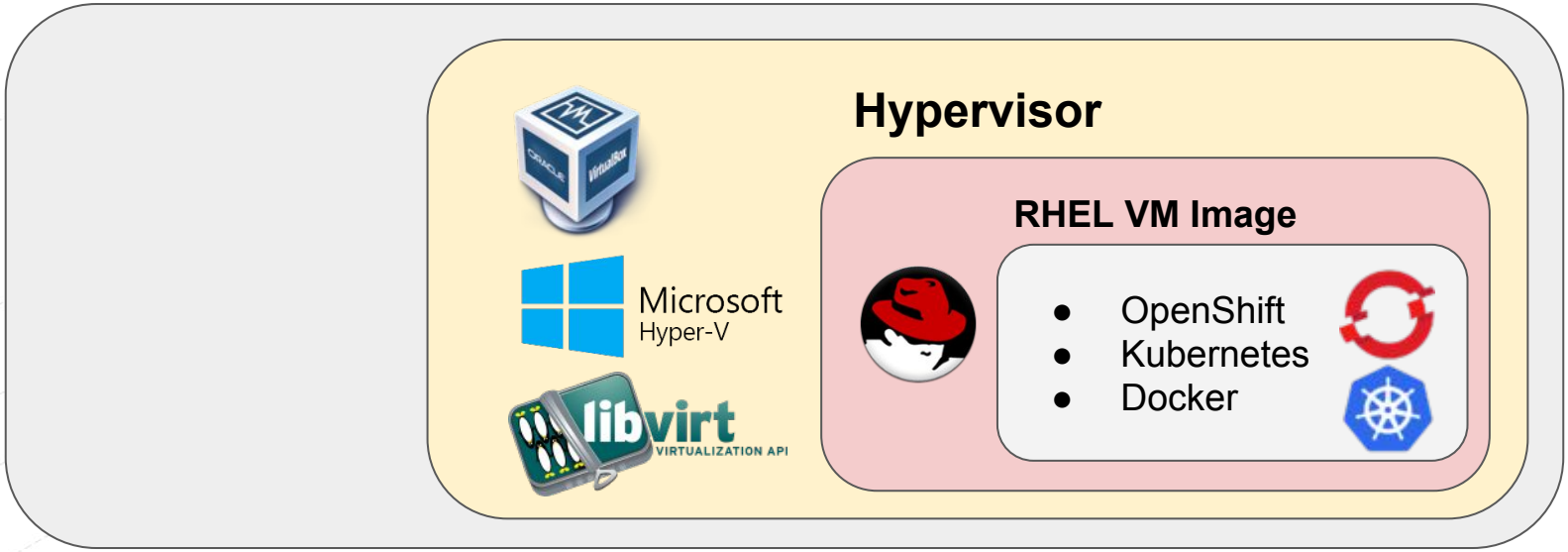



Red Hat - Kubernetes Lab
Say Hi - Rafael
Your name
Rafael
Message
Submit
Hello Rafael from helloworld-service-vertx-375244497-5zgty with 1.0 from config file modified
Name
Message
Rafael
Hello DevNation!!!!
Rafael
teste1
Benevides
Hello World


Red Hat - Kubernetes Lab
Say Hi - Rafael
Your name
Rafael
Message
Submit
Hello Rafael from helloworld-service-vertx-375244497-5zgty with 1.0 from config file modified
Name
Message
Rafael
Hello DevNation!!!!
Rafael
teste1
Benevides
Hello World
Guestbook Service - Create
Hello World Service - Greet
Guestbook Service - Retrieve

Lab infrastructure

minishift or CDK



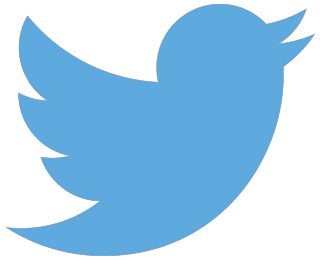
http://developers.redhat.com/products/cdk/download/

Kubernetes lab

VERY IMPORTANT

http://bit.ly/kubernetes-lab

Follow me on the Setup environment section!



@RAFABENE

RED HAT®
DEVELOPER
PROGRAM

Kubernetes Introduction

 @rafabene

 benevides@redhat.com

Link → http://bit.ly/kubernetes-intro