

Kafka and Kubernetes OpenShift World Tour

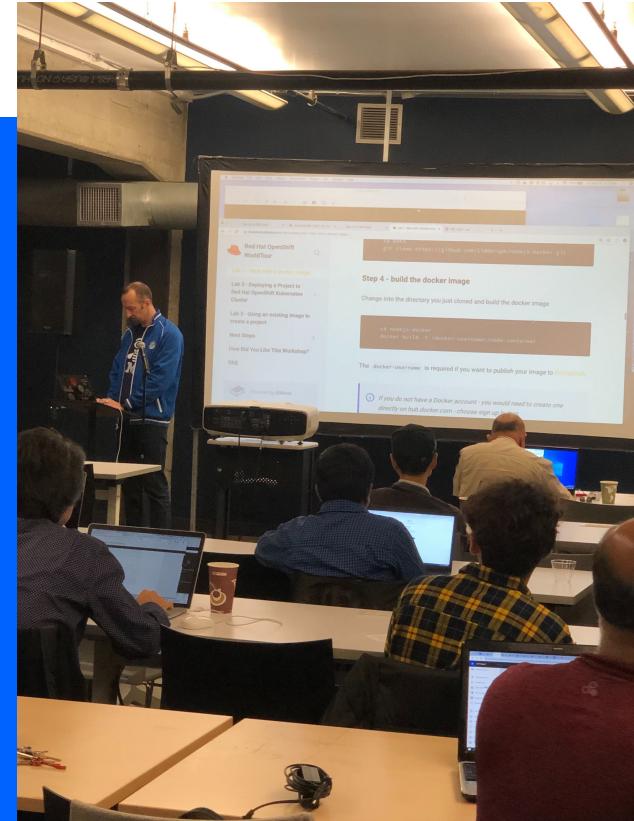
Marek Sadowski

Developer Advocate | IBM San Francisco City Team

ibm.biz/202010-kafka

V1 2020 10 15

IBM Developer



IBM

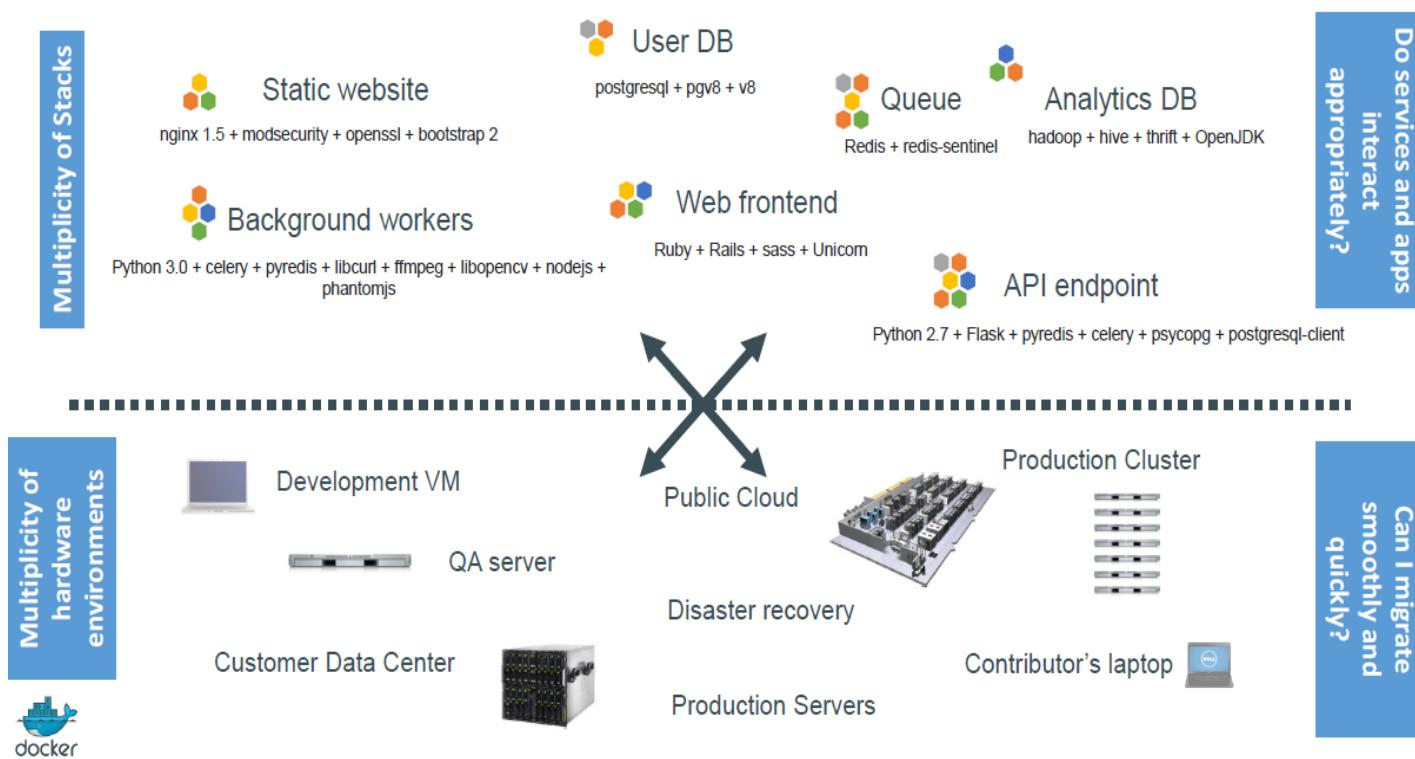
Agenda:

1. Intro to containers
2. containers to host microservices
3. **LAB 0:** Docker build-ship-run
4. orchestration of many containers
5. Kubernetes
6. OKD project - OpenShift
9. Red Hat OpenShift vs Kubernetes
10. **LAB 1:** Kubernetes on OpenShift hands-on
11. **LAB 2:** Kafka on Kubernetes

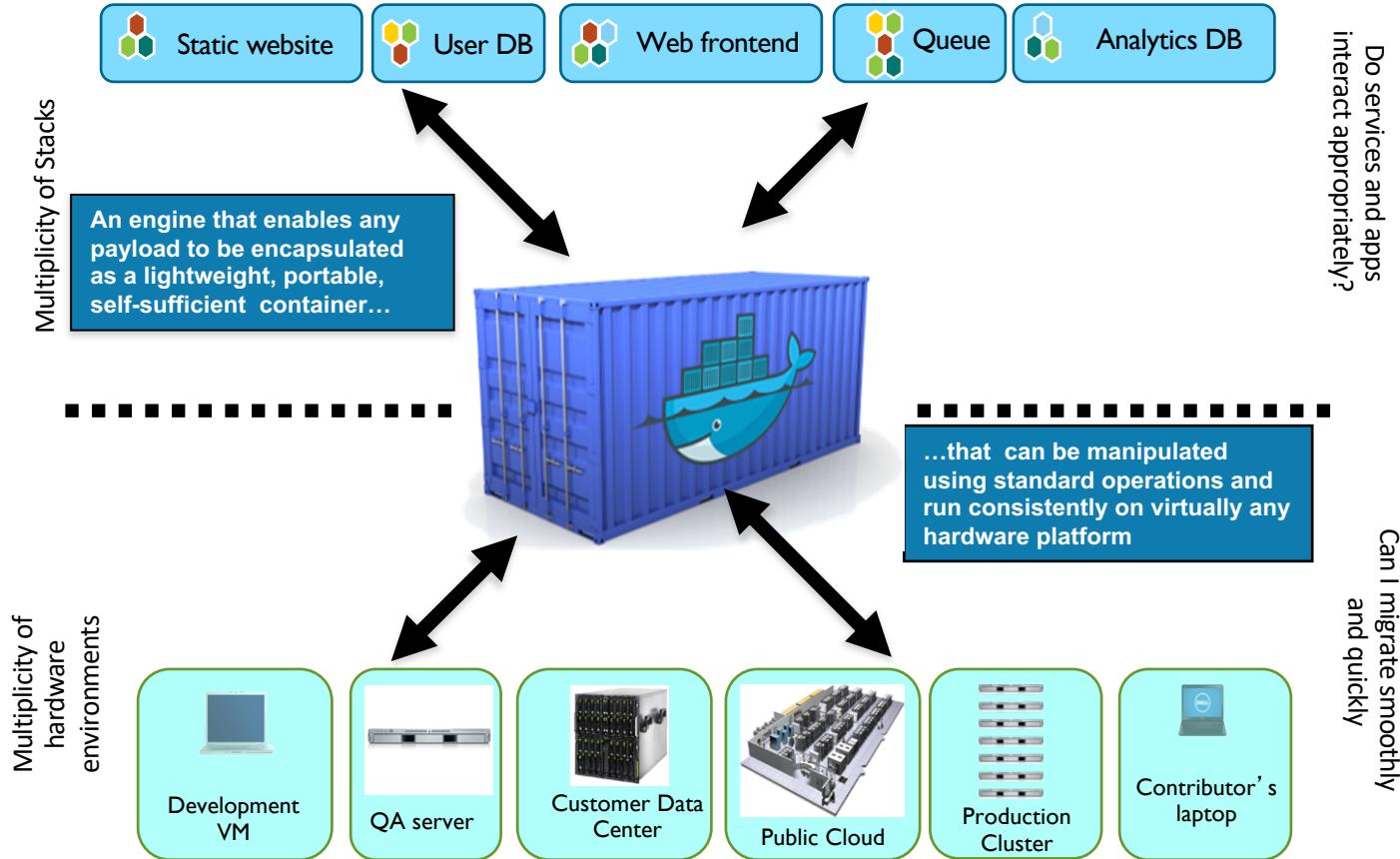
THE POLL

Why Containers?

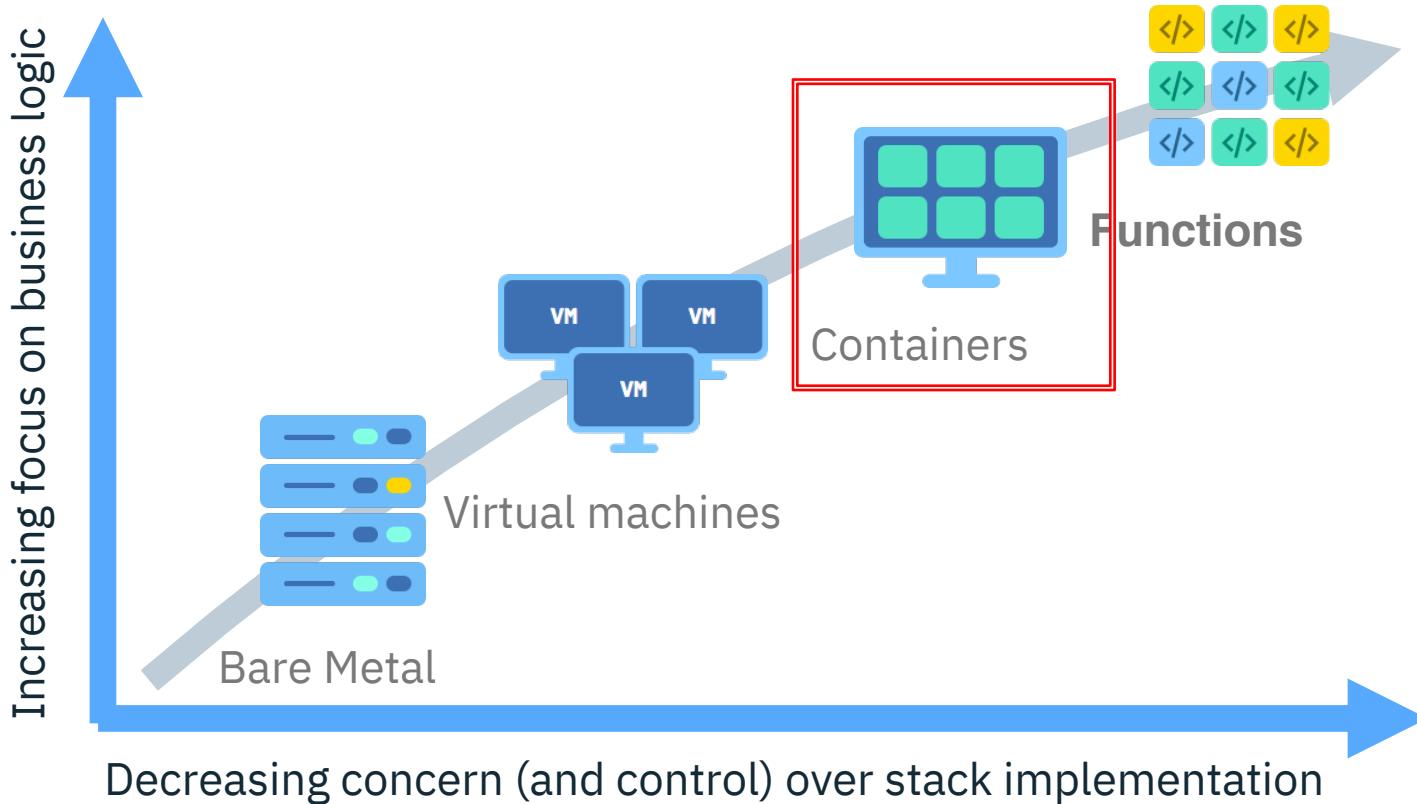
The Challenge



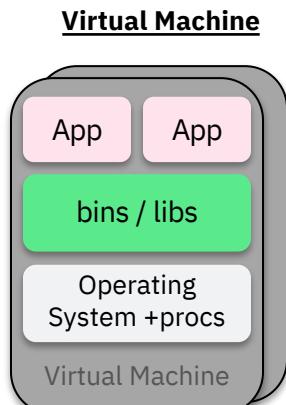
The Solution - A Shipping Container for Code



How we got here



VM vs Container



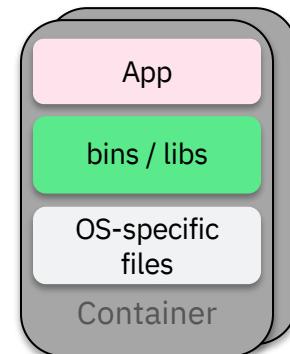
Hypervisor

Hardware

Each VM has its own
OS



Container



Base OS/Kernel

Hardware

App, bins/libs/OS must all be
runnable on the shared kernel

If OS files aren't needed they can be
excluded.

VM ?

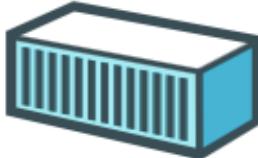
Containers share
the same base
Kernel

Docker Mission

Docker is an **open platform** for building distributed applications for **developers** and **system administrators**



Build



Ship



Run



IBM Code



Any App

Anywhere

Dev vs. Ops

Dev

Ops



- Code
- Libraries
- Configuration
- Server runtime
- OS

- Logging
- Remote access
- Network configuration
- Monitoring

Separation of concerns

A container separates and bridges the Dev and Ops in DevOps

- Dev focuses on the application environment
- Ops focuses on the deployment environment

Docker Component Overview

Docker Engine

- Manages containers on a host
- Accepts requests from clients
 - REST API
- Maps container ports to host ports
 - E.g. 80 → 3582

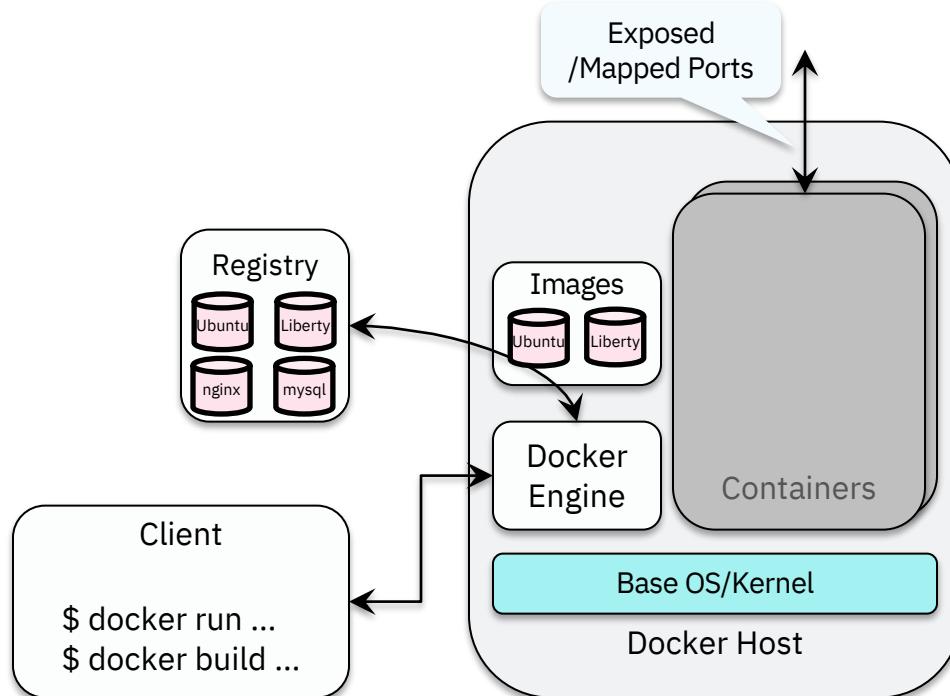
Images

Docker Client

- Drives engine
- Drives "builder" of Images

Docker Registry

- Image DB



Shared / Layered / Union Filesystems

Docker uses a copy-on-write (union) filesystem

New files(& edits) are only visible to current/above layers

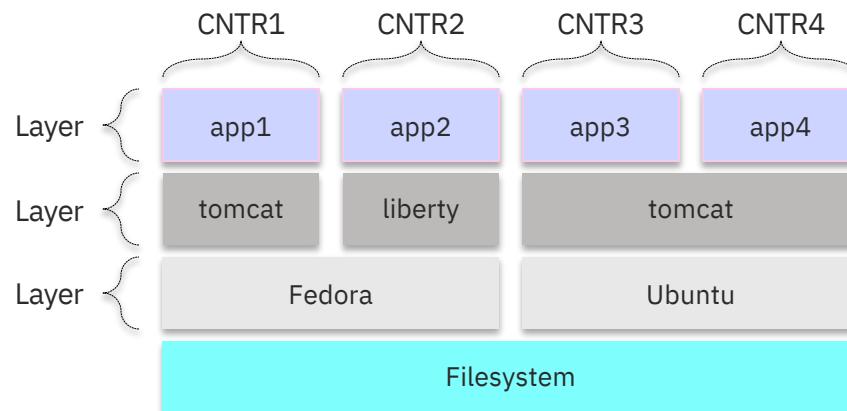
Layers allow for reuse

- More containers per host
- Faster start-up/download time

Images

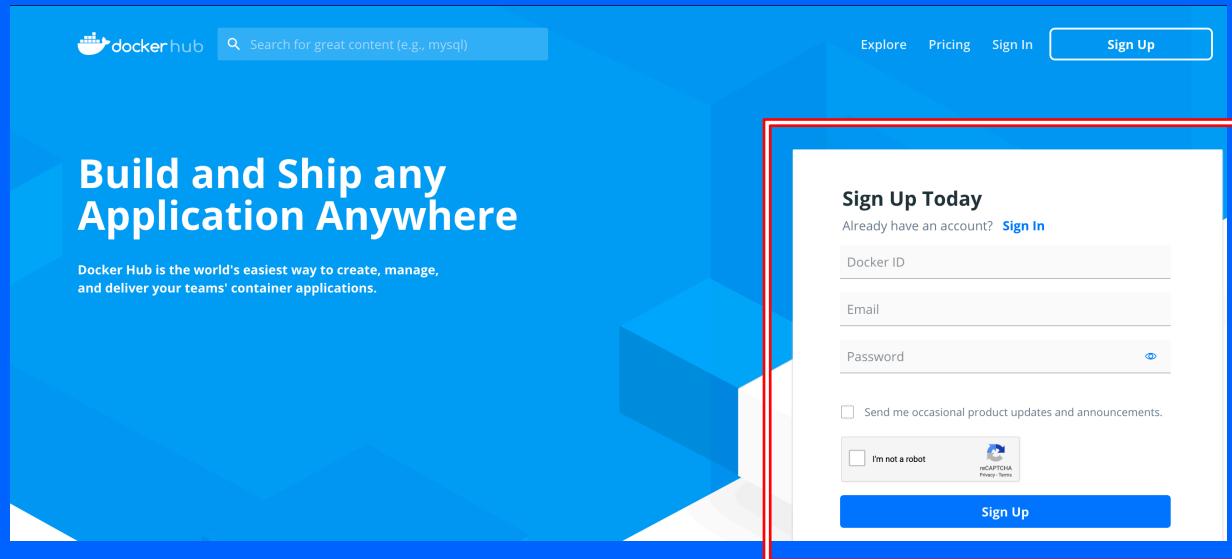
- Tarball of layers

Think: Transparencies on projector



Hands-on

Download FREE Docker Desktop and use your terminal on your computer or just sit back and watch



Our First Container

```
$ docker run ubuntu echo Hello World
```

```
Hello World
```

What happened?

Docker created a directory with a "ubuntu" filesystem (image)

Docker created a new set of namespaces

Ran a new process: echo Hello World

Using those namespaces to isolate it from other processes

Using that new directory as the "root" of the filesystem (chroot)

That's it!

Notice as a user I never installed "ubuntu"

Run it again - notice how quickly it ran

```
[Mareks-MBP:~ mareksadowski$ docker run ubuntu echo Hello World
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
6b98dfc16071: Pull complete
4001a1209541: Pull complete
6319fc68c576: Pull complete
b24603670dc3: Pull complete
97f170c87c6f: Pull complete
Digest: sha256:5f4bdc3467537cbbe563e80db2c3ec95d548a9145d64453b06
Status: Downloaded newer image for ubuntu:latest
Hello World
```

```
[Mareks-MBP:~ mareksadowski$ docker run ubuntu echo Hello World
Hello World
Mareks-MBP:~ mareksadowski$ █
```

LAB 0

building a translation service with Watson:

<http://ibm.biz/202010-kafka-lab0>

IBM Cloud Catalog

<https://cloud.ibm.com/catalog>

The screenshot shows the IBM Cloud Catalog interface. At the top, there's a navigation bar with links for Catalog, Docs, Log in, and Sign up. Below the navigation is a search bar with the placeholder "Search the catalog...". A horizontal menu bar has two tabs: "Services" (which is highlighted with a blue underline) and "Software". To the left, a sidebar lists various service categories: All Categories, VPC Infrastructure, Compute, Containers, Networking, Storage, AI, Analytics, Databases, Developer Tools, Integration, Internet of Things, and Security and Identity. On the right, the main content area is titled "Services" and describes the broad portfolio of managed services. A "Featured" section highlights the "Kubernetes Service" with its icon, name, category (IBM • Compute • Containers), a brief description ("Deploy secure, highly available apps in a native Kubernetes experience."), and status indicators ("Free • IAM-enabled • Service Endpoint Supported"). A "FEEDBACK" button is located on the far right edge.

Second part –
the orchestration
of successful API
deployment

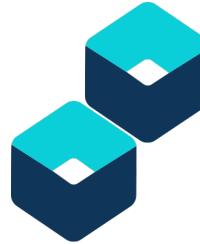


Containers



Everyone's container journey starts with one container....

Containers



At first the growth is easy to handle....



Containers



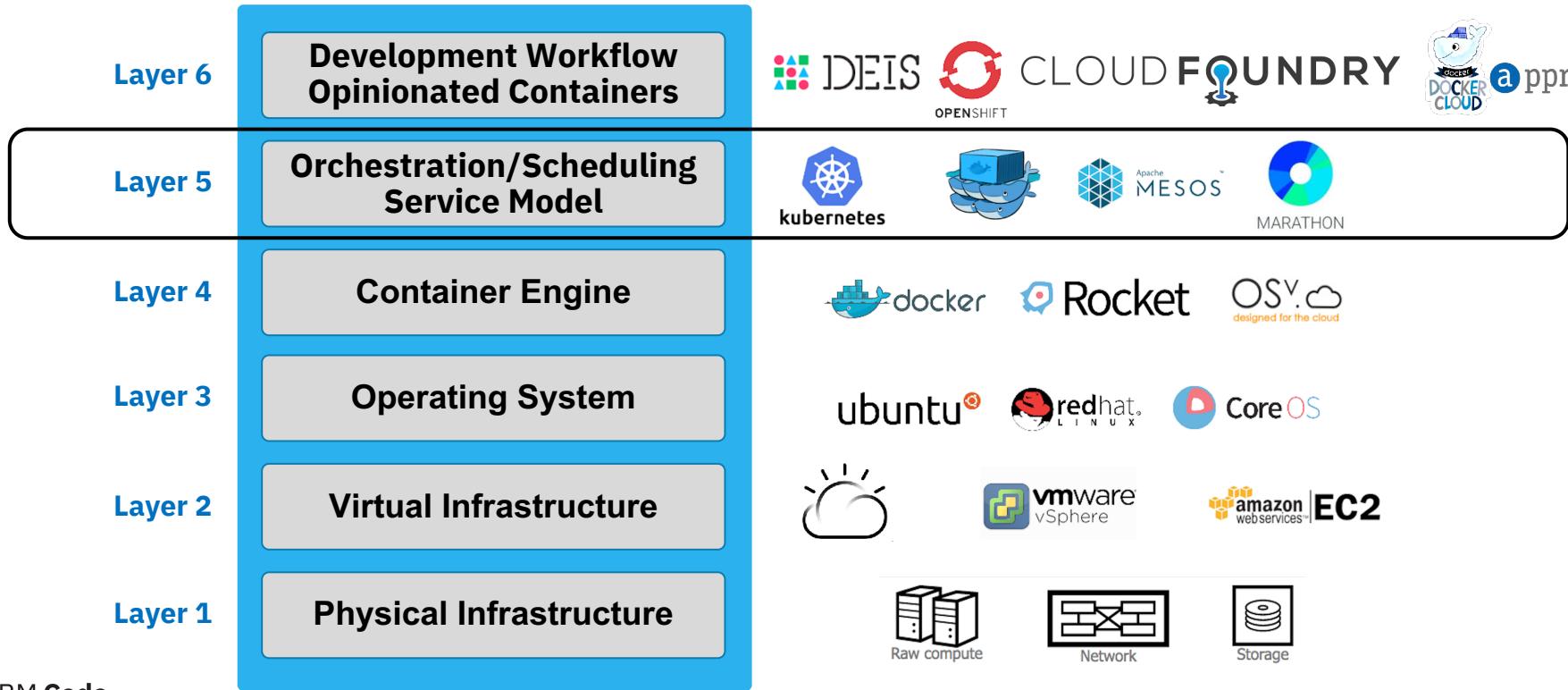
But soon it is overwhelming... chaos reigns

Container Orchestration

Allows users to define how to coordinate the containers in the cloud when the multi-container packaged application is deployed.

- Scheduling
- Cluster management
- Service discovery
- Provisioning
- Monitoring
- Configuration management

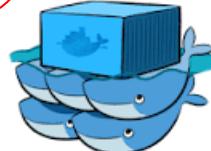
Container Ecosystem Layers



Kubernetes – Mesos – Marathon – Docker Swarm



Kubernetes is a cluster manager for containers (large deployments)



Swarm has a YAML-based deployment model, auto-healing of clusters, overlay networks with DNS, high-availability through the use of multiple masters, and network security using TLS with a Certificate Authority (1-10 containers)



MESOS is a distributed system kernel that will make your cluster look like one giant computer system to all supported frameworks and apps that are built to be run on mesos.

(Kubernetes can be run on Mesos)



Marathon is a cluster-wide init and control system for running Linux services in cgroups and Docker containers, and it is run on top of Mesos

What is Kubernetes?

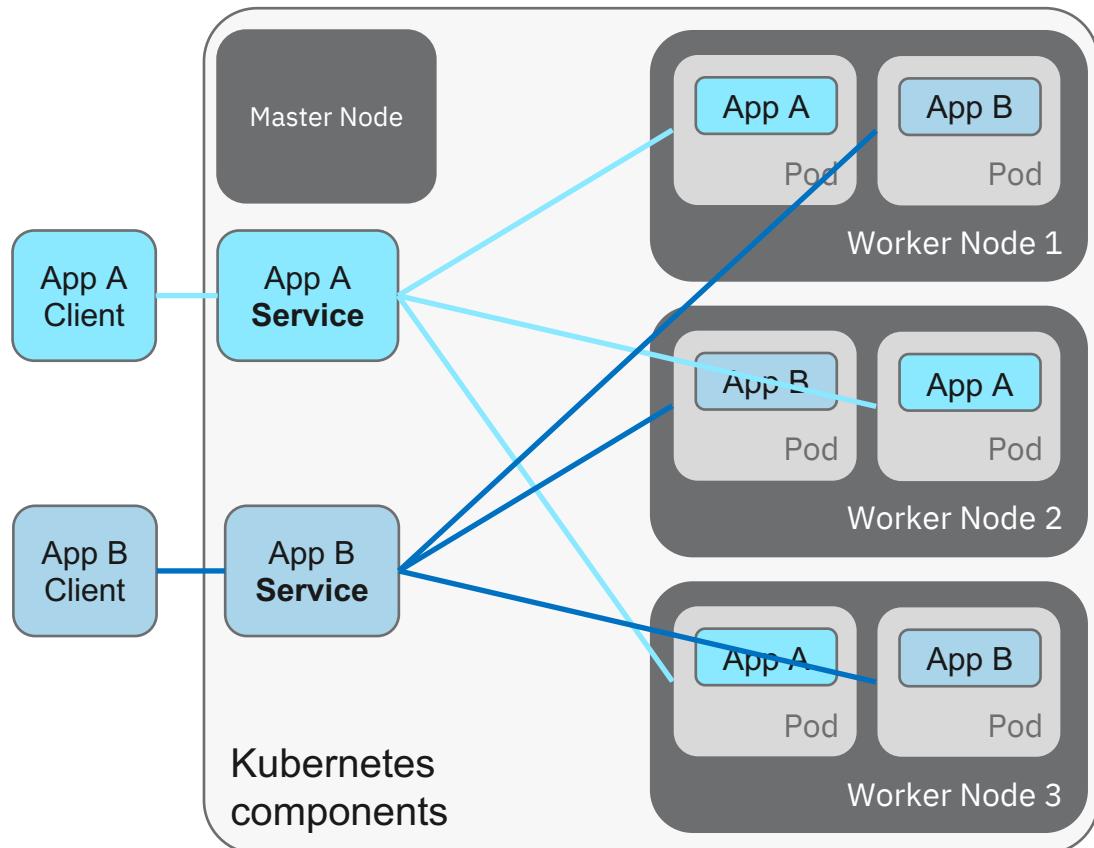


- Container orchestrator
- Manage applications, **not machines**
- Designed for extensibility
- Open source project managed by the Linux Foundation



Kubernetes Architecture: Workloads

- Container
 - Packaging of an app
- Pod
 - Unit of deployment
- Service
 - Fixed endpoint for 1+ pods



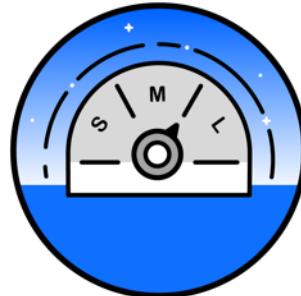
Kubernetes



Intelligent Scheduling



Self-healing



Horizontal scaling



Service discovery & load balancing



Automated rollouts and rollbacks



Secret and configuration management

ORIGIN COMMUNITY DISTRIBUTION OF KUBERNETES



The Origin Community Distribution of Kubernetes that powers Red Hat OpenShift.

Built around a core of OCI container packaging and Kubernetes container cluster management, OKD is also augmented by application lifecycle management functionality and DevOps tooling. OKD provides a complete open source container application platform.

Standardization through Containerization

Standards are powerful forces in the software industry. They can drive technology forward by bringing together the combined efforts of multiple developers, different communities, and even competing vendors.



kubernetes
Google

Open source container orchestration and cluster management at scale.

[Learn more](#)



docker

Standardized Linux container packaging for applications and their dependencies.

[Learn more](#)



Core OS

A container-focused OS that's designed for painless management in large clusters.

[Learn more](#)



OPERATOR
FRAMEWORK

An open source project that provides developer and runtime Kubernetes tools, enabling you to accelerate the development of an Operator.

[Learn more](#)



cri-o

A lightweight container runtime for Kubernetes.

[Learn more](#)

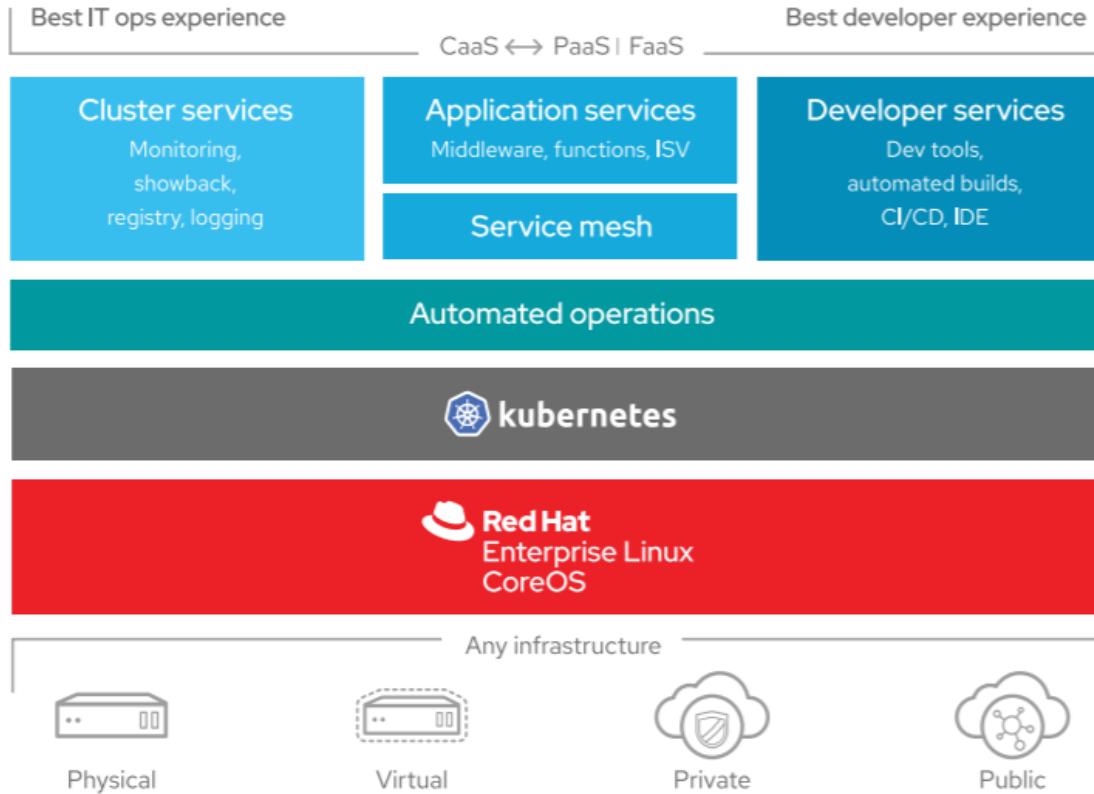


Prometheus

Prometheus is a systems and service monitoring toolkit that collects metrics from configured targets at given intervals, evaluates rule expressions, displays the results, and can trigger alerts if some condition is observed to be true.

[Learn more](#)

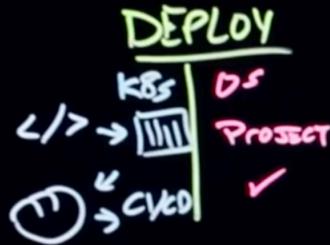
Open Source | Secure | Fully Certified | Portable



#IBMDveloper



KUBERNETES + OPENSHIFT



Lab 1: follow steps here: <http://ibm.biz/202010-kafka-lab1>

URL: <https://oskafka.mybluemix.net>

Key: oslab



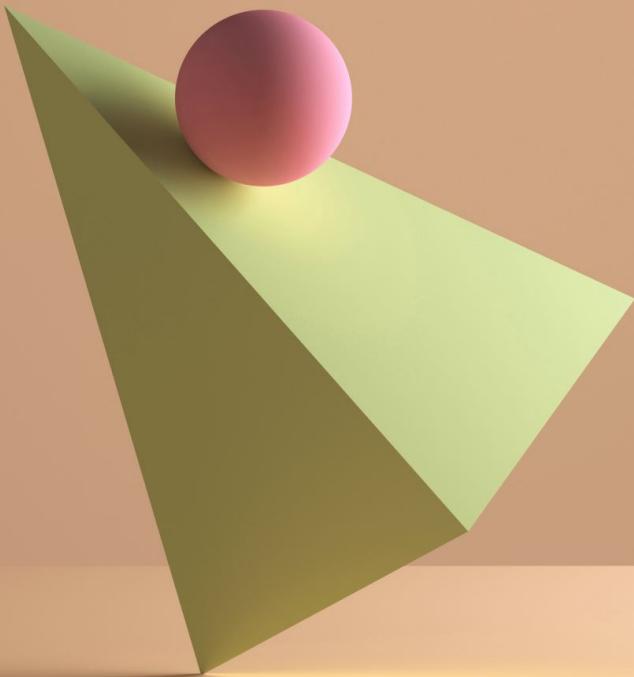
Stretch LAB with Red Hat

- <https://medium.com/@blumareks/running-a-microservice-containers-on-kubernetes-straight-from-a-github-ff73047e877b>

LAB 2: Kafka and Kubernetes

ibm.biz/202010-kafka-lab2

<http://ibm.biz/202010-kafka-lab2-env>



LAB2:

The screenshot shows the IBM Skills Network Labs landing page. At the top, there's a navigation bar with the URL "labs.cognitiveclass.ai". Below the URL is the "IBM Developer Skills Network" logo, which includes a colorful book icon and the word "Labs". A main text block explains that IBM Skills Network Labs are a place to practice data science, machine learning, and AI skills learned in online courses. It mentions preinstalled tools like JupyterLab, Zeppelin, and RStudio, along with Apache Spark and Python/R/Scala packages. Below this text is a "Log in with social" section with a "Click button to log in." instruction and social media icons for LinkedIn, GitHub, Google, and Facebook. A note below states that by logging in, the user acknowledges the use of their personal data and is at least sixteen years old, linking to a Privacy Notice.

IBM Skills Network Labs are a place for you to practice the data science, machine learning, and AI skills you're learning in your online courses. You have access to JupyterLab, Zeppelin, and RStudio preinstalled with Apache Spark and the necessary packages to learn new skills in Python, R, and Scala.

Log in with social

Click button to log in.

Cognitive Class

By logging in, I acknowledge that I understand how IBM Developer Skills Network is using my basic personal data, and that I am at least sixteen years of age. Our [Privacy Notice](#) provides more details.

[Don't have an account? Sign up!](#)

Terms of Use Privacy Notice
Crafted with ❤ and support from IBM
© Copyright IBM Developer Skills Network 2020

The screenshot shows the Cognitive Class sign-in page. At the top, there's a navigation bar with links for Learning Paths, Courses, Badges, Badges Program, and Business. Below the navigation is a message for first-time users to "Create an Account". The main area has a "Sign In" form with fields for "Email" (containing "username@domain.com") and "Password". There are "Forgot password?" and "Remember me" checkboxes, and a "Sign in" button. Below the sign-in form is a "or sign in with" section featuring social media and developer identity provider icons: Facebook, GitHub, Google, IBMId, and LinkedIn.

First time here? [Create an Account.](#)

Sign In

Email
username@domain.com

The email address you used to register with Cognitive Class

Password

Forgot password?

Remember me

Sign in

or sign in with

Facebook GitHub Google
 IBMId LinkedIn

Skills Network Labs

Step 1 of 5
Creating reactive microservices using MicroProfile Reactive Messaging

Learn how to write reactive Java microservices using MicroProfile Reactive Messaging.

What you'll learn

You will learn how to build reactive microservices that can send requests to other microservices, and asynchronously receive and process the responses. You will use an external messaging system to handle the asynchronous messages that are sent and received between the microservices as streams of events. MicroProfile Reactive Messaging makes it easier to write and configure your application to send, receive, and process the events efficiently.

Aynchronous messaging between microservices

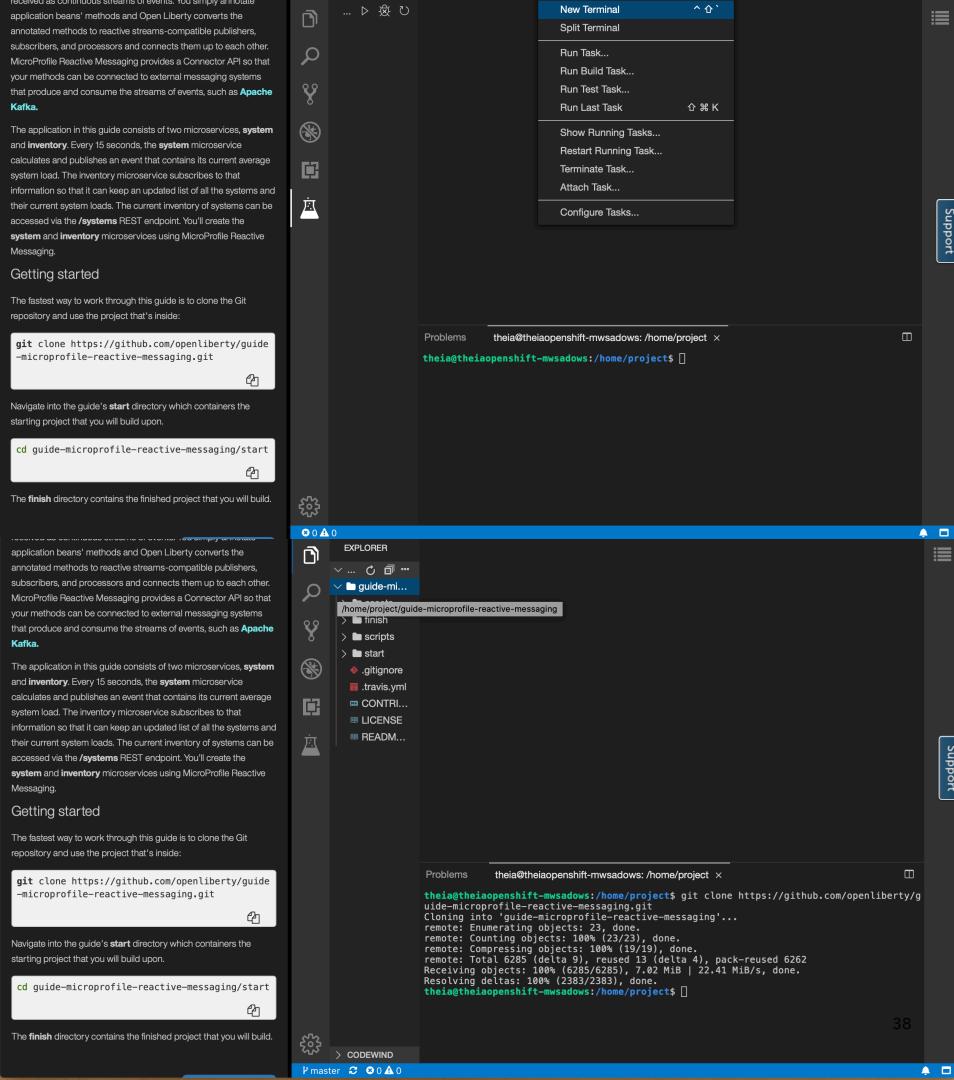
Asynchronous communication between microservices can be used to build reactive and responsive applications. By decoupling the components out to a message bus from the application that it processes

Your server is starting up.
You will be redirected automatically when it's ready for you.
2020-10-15 15:38:23+00:00 [Normal] Started container theia

Event log

LAB2: doing the lab

The new terminal – scroll for instructions



LAB 2: tips (use OpenShift console to find out the route to the app)

```
Problems theia@theiaopenshift-mwsadows: /home/project/guide-micropoint-reactive-messaging/start × □  
theia@theiaopenshift-mwsadows: /home/project/guide-micropoint-reactive-messaging/start$ sed -i 's=$NAME=$NAMESPACE_NAME=' "$NAME=g" openshift.yaml  
theia@theiaopenshift-mwsadows: /home/project/guide-micropoint-reactive-messaging/start$ oc apply -f openshift.yaml  
deployment.apps/system-deployment created  
deployment.apps/inventory-deployment created  
deployment.apps/kafka-deployment created  
deployment.apps/zookeeper-deployment created  
service/system-service created  
service/kafka-service created  
service/inventory-service created  
service/zookeeper-service created  
route.route.openshift.io/inventory-route created  
theia@theiaopenshift-mwsadows: /home/project/guide-micropoint-reactive-messaging/start$ oc get routes  
NAME          HOST/PORT          PATH      SERVICES      PORT      TERMINATION      WILDCARD  
RD  
inventory-route  inventory-route-sn-labs-mwsadows.labs-user-sandbox-830083-a45631dc5778dc6371  
theia@theiaopenshift-mwsadows: /home/project/guide-micropoint-reactive-messaging/start$ █  
theia@theiaopenshift-mwsadows: /home/project/guide-micropoint-reactive-messaging/start$ █
```

The image shows two screenshots of web browsers. The top screenshot displays the JSON response from the OpenShift API endpoint for a deployment named 'system-deployment'. The response includes fields like 'hostname' (set to 'system-deployment-5d4bf94f94-pz7jb') and 'systemLoad' (set to '0.38'). The bottom screenshot shows a browser window with a red box highlighting the URL '8dc6371c67d206ba9ae5c-0000.tor01.containers.appdomain.cloud/inventory/systems...'. An arrow points from this URL to another browser window below it, which shows the AWS Lambda function URL 'Search with Amazon.com'.

inventory-route-sn-labs-mwsadows.labs-user-sandbox-830083-a45631dc5778dc6371

Raw Data Headers

hostname: "system-deployment-5d4bf94f94-pz7jb"
systemLoad: 0.38

8dc6371c67d206ba9ae5c-0000.tor01.containers.appdomain.cloud/inventory/systems...
Search with Amazon.com
IBM Cloud — cloud.ibm.com
w3 — w3.ibm.com

Thank you!

Marek Sadowski

Join me for My
office hours - use
slack:

Developer Advocate

@blumareks

[https://join.slack.com
/t/biz-on-
cloud/shared_invite/zt
-hop799ua-
MmK32III9VS6xxqMfkzrlg](https://join.slack.com/t/biz-on-cloud/shared_invite/zt-hop799ua-MmK32III9VS6xxqMfkzrlg)