Intro to Kubernetes

Marek Sadowski
Developer Advocate | IBM San Francisco

@blumareks

IBM **Code**

http://ibm.biz/kubernetes070318

CALL FOR CODE
GLOBAL INITIATIVE 2018

Commit to the cause. Push for change.

Call for Code inspires developers to solve **pressing global problems** with **sustainable software solutions**, delivering on their vast potential to do good.

Bringing together NGOs, academic institutions, enterprises, and startup developers to compete build effective **disaster mitigation solutions**, with a focus on health and well-being.

International Federation of Red Cross/Red Crescent, **The American Red Cross**, and the **United Nations Office of Human Rights** combine for the ***Call for Code Award*** *to* elevate the profile of developers.

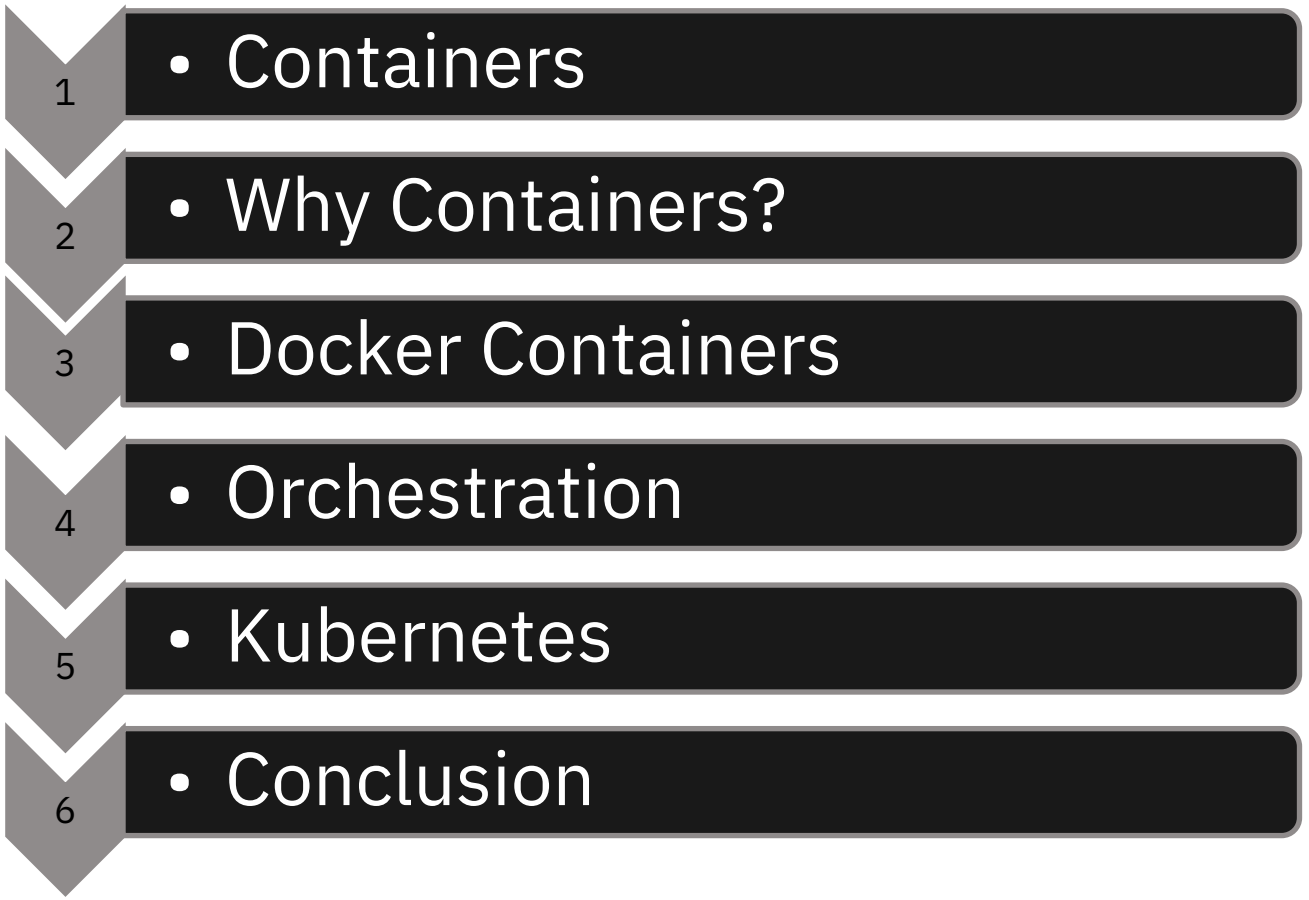
Award winners will receive **long-term support** through **open source foundations**, **financial prizes**, the **opportunity to present their solution to leading VCs**, and will deploy their solution through **IBM's Corporate Service Corps**.

Developers will jump-start their project with dedicated **IBM Code Patterns**, combined with **optional enterprise technology** to build projects over the course of three months.

Judged by the world's most **renowned technologists**, the **grand prize** will be presented in **October** at an Award Event.

developer.ibm.com/callforcode

Agenda


1
• Containers
2
• Why Containers?
3
• Docker Containers
4
• Orchestration
5
• Kubernetes
6
• Conclusion

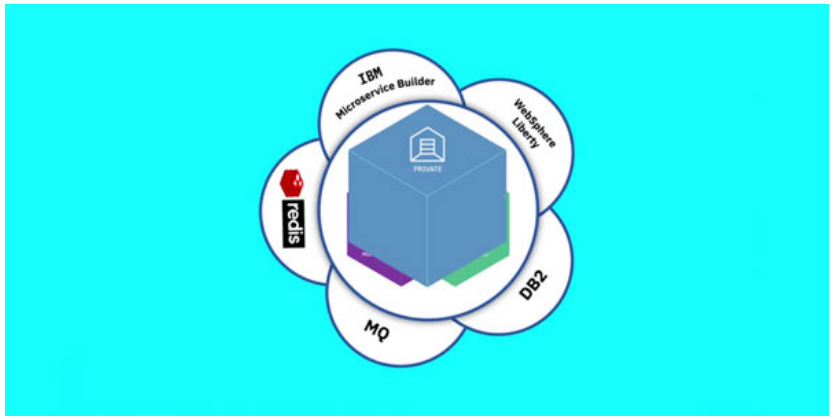
Containers

A standard way to package an application and all its dependencies so that it can be moved between environments and run without changes.

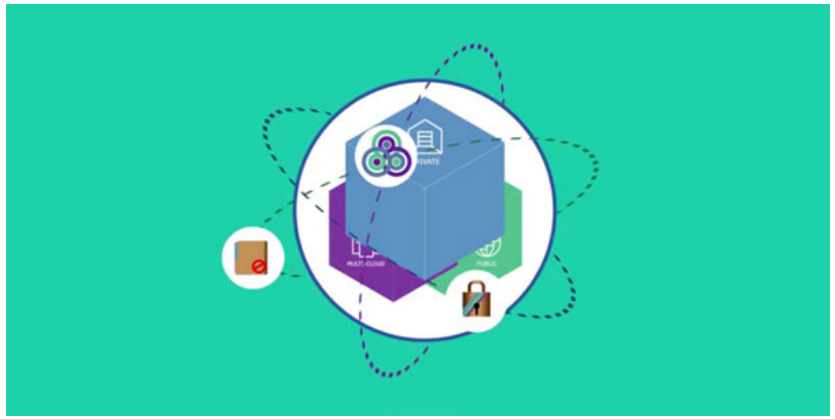
Containers work by isolating the differences between applications inside the container so that everything outside the container can be standardized.

Why Containers?

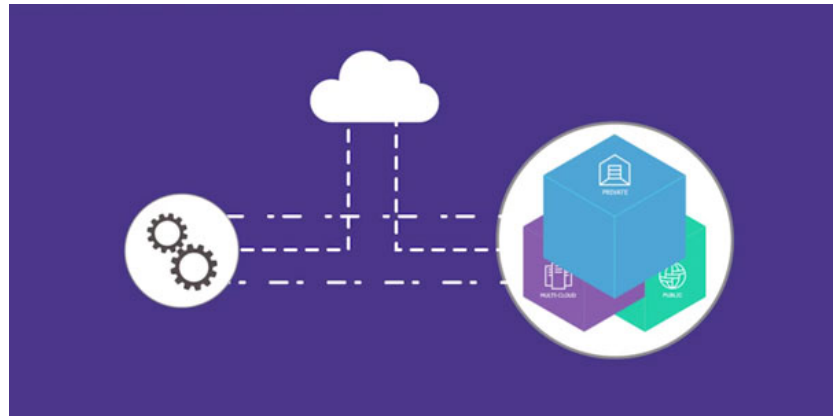
Use Cases for Containers



Create new cloud-native apps
Streamline development
with built-in microservices, runtimes, containers and Kubernetes orchestration plus integrated management.



Modernize your heritage apps on cloud
Move your apps as-is to the cloud or re-factor an app and use it in new development and application workload models.



Open your data center to work with cloud services
Protect and leverage your in-house data and pull in external

Dev vs. Ops

Dev	Ops

- Code
- Libraries
- Configuration
- Server runtime
- OS

- Logging
- Remote access
- Network configuration
- Monitoring

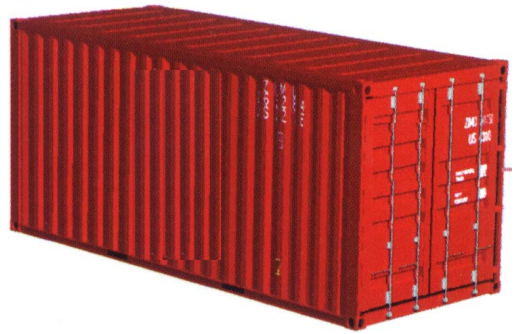
Separation of concerns

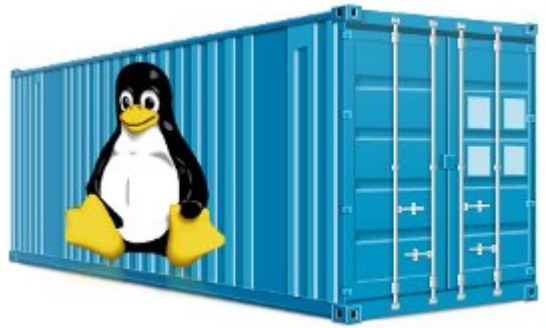
A container separates and bridges the Dev and Ops in DevOps

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

Container Advantages

- Containers are portable
- Containers are easy to manage
- Containers provide “just enough” isolation
- Containers use hardware more efficiently
- Containers are immutable





Docker Containers

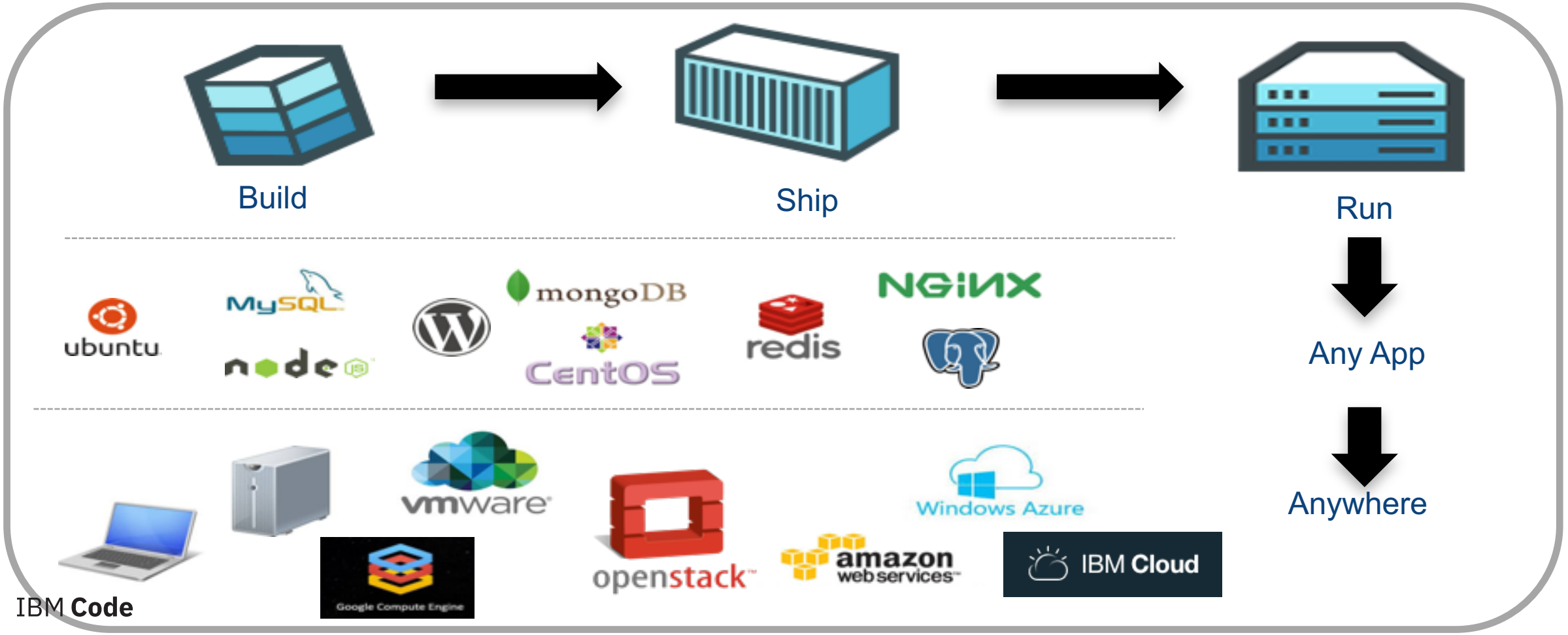
Docker Adoption

Docker enables application development **efficiency**, making deployment more **efficient**, eliminating vendor ‘lock-in’ with true **portability**

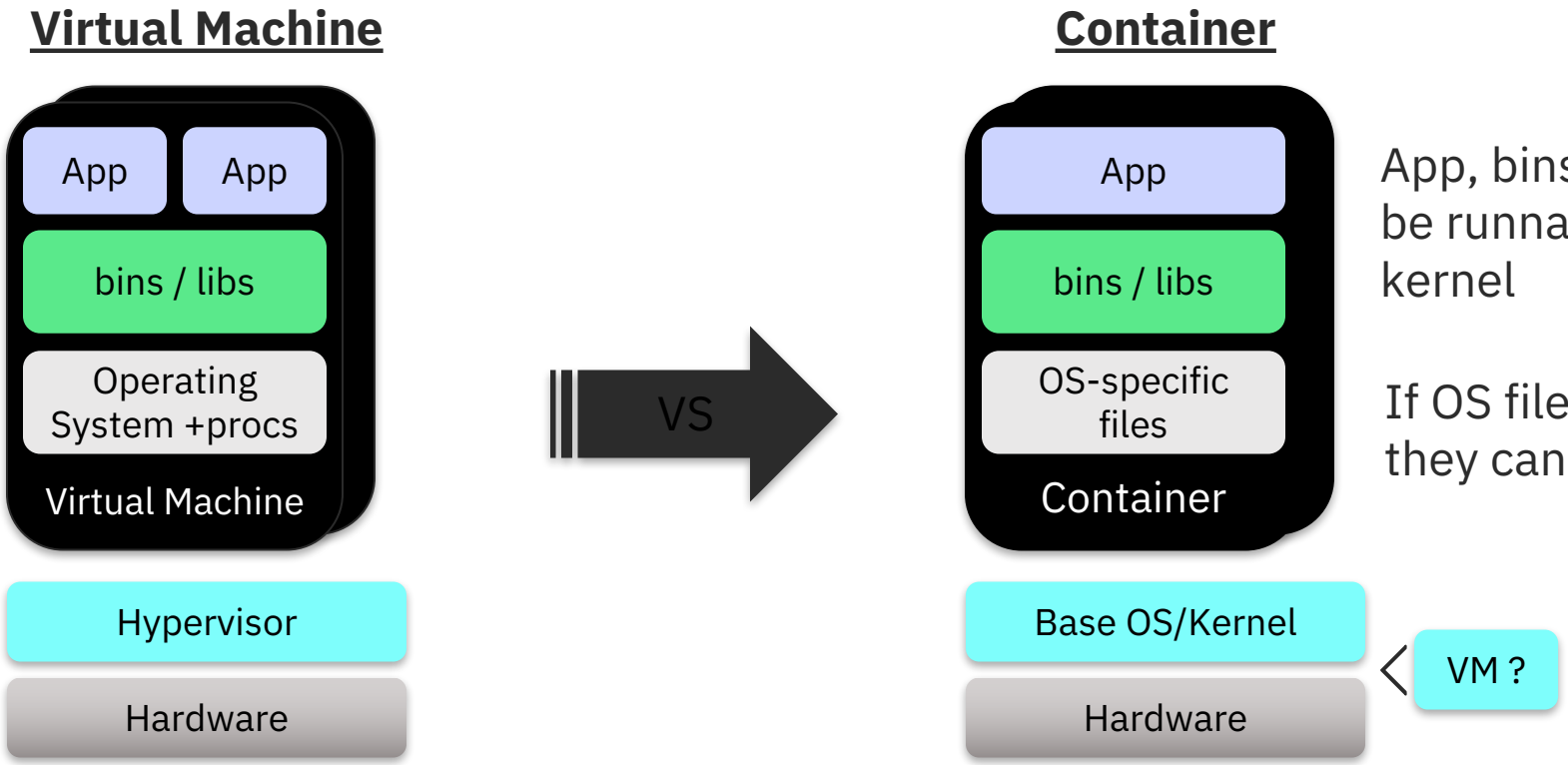
- Open software
- Open contribution
- Open design
- Open governance

Docker Mission

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



VM vs Container



Each VM has its own OS

Containers share the same base Kernel

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

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

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

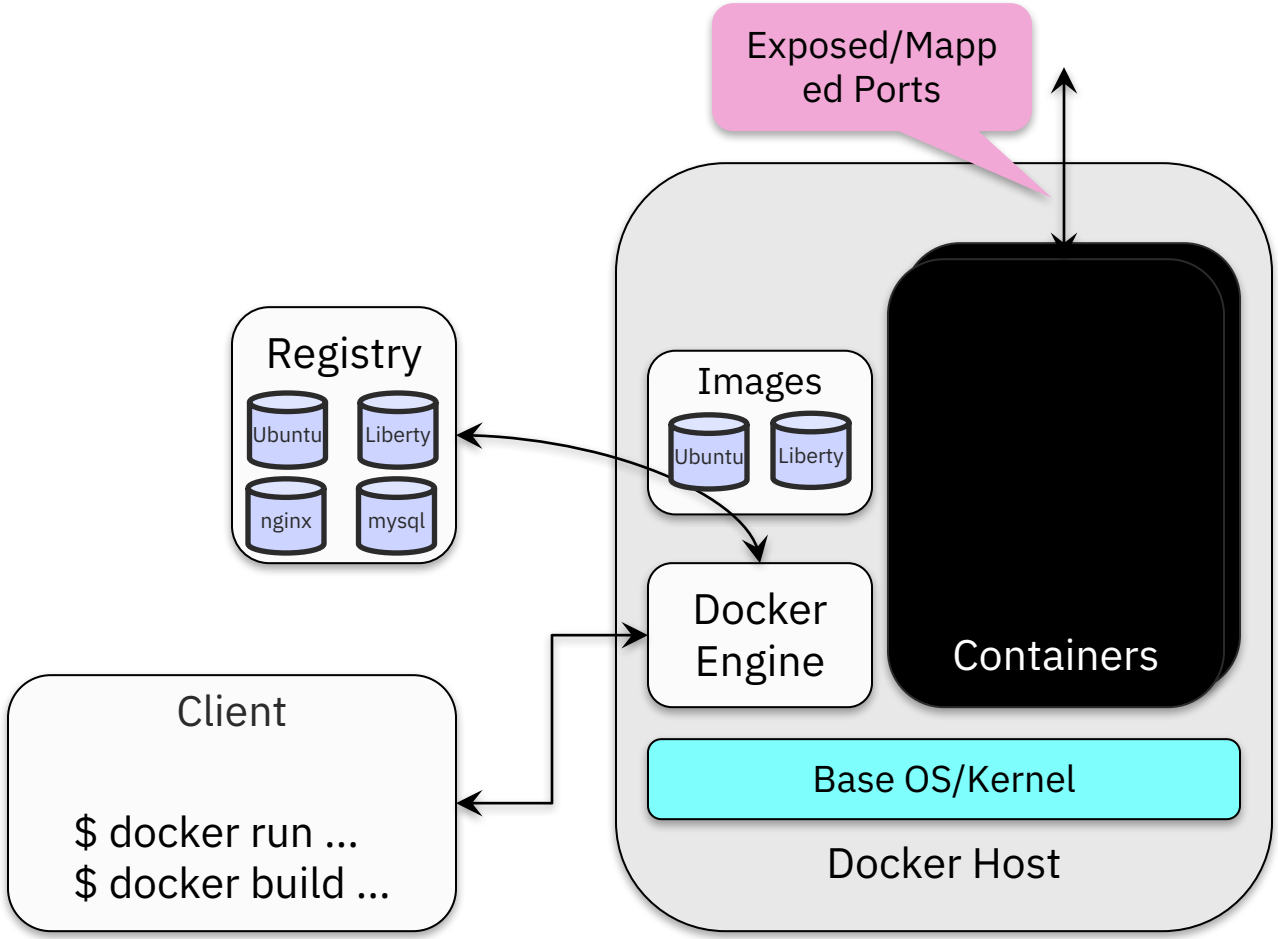
Images

Docker Client

- Drives engine
- Drives "builder" of Images

Docker Registry

- Image DB



What is Docker?

Tooling to manage containers

– Containers are not new

– Docker just made them easy to use

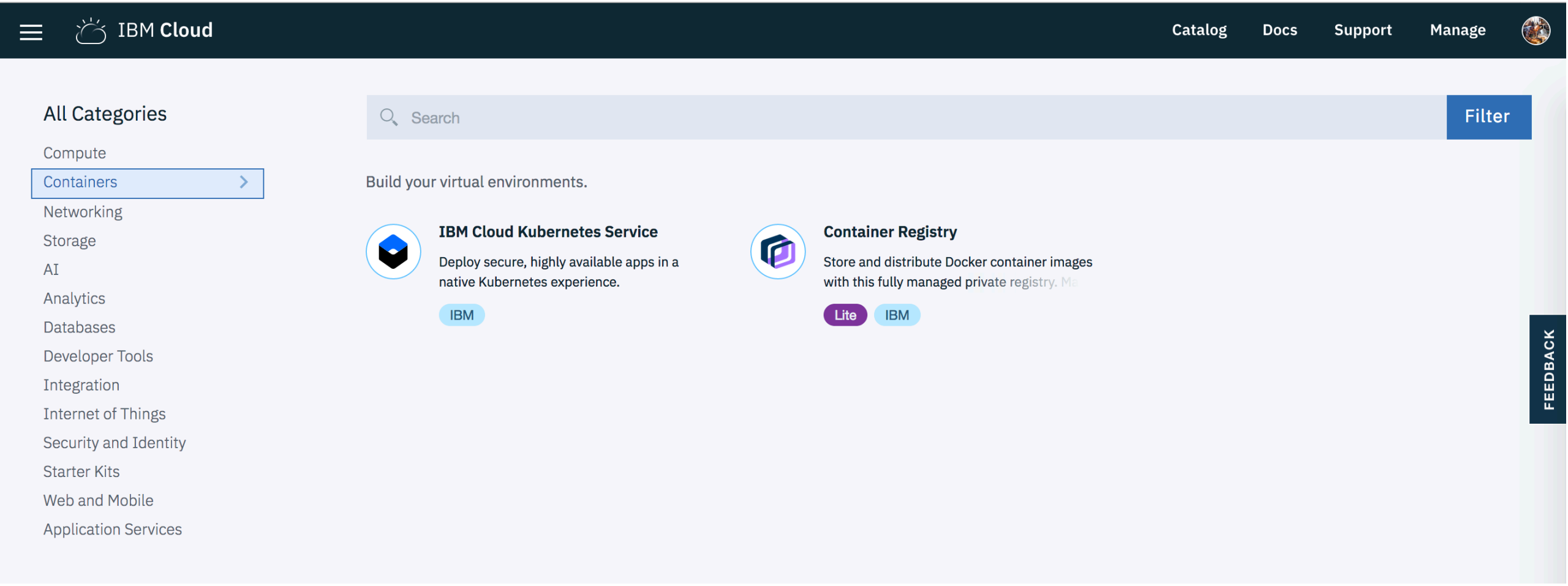
Docker creates and manages the lifecycle of containers

– Setup filesystem

– CRUD container

- Setup networks
- Setup volumes / mounts
- Create: start new process telling OS to run it in isolation

IBM Cloud – Enterprise grade registry

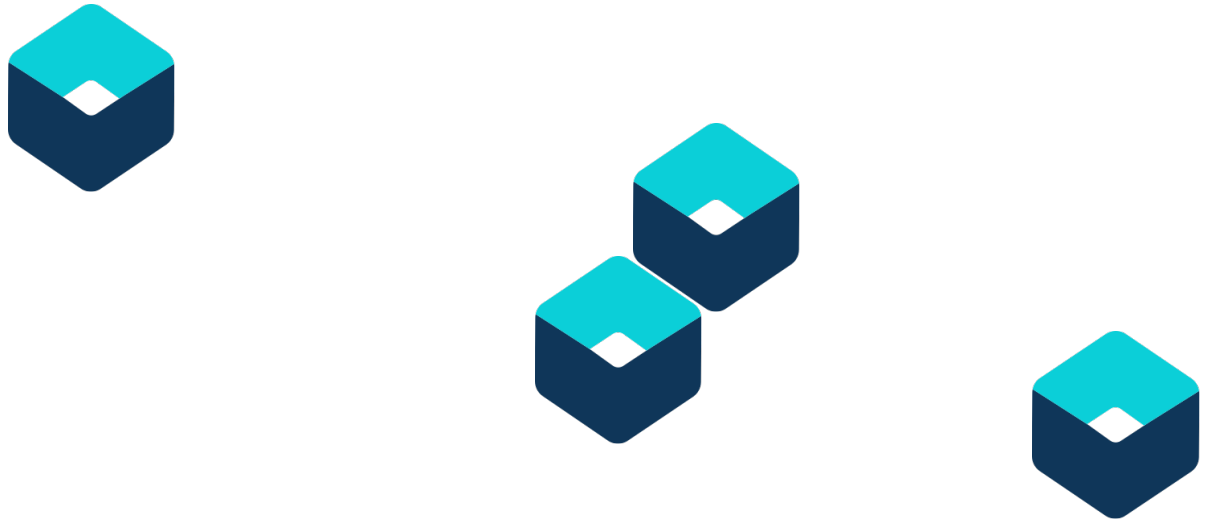


http://ibm.biz/kubernetes070318

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

Orchestration

Container Orchestration

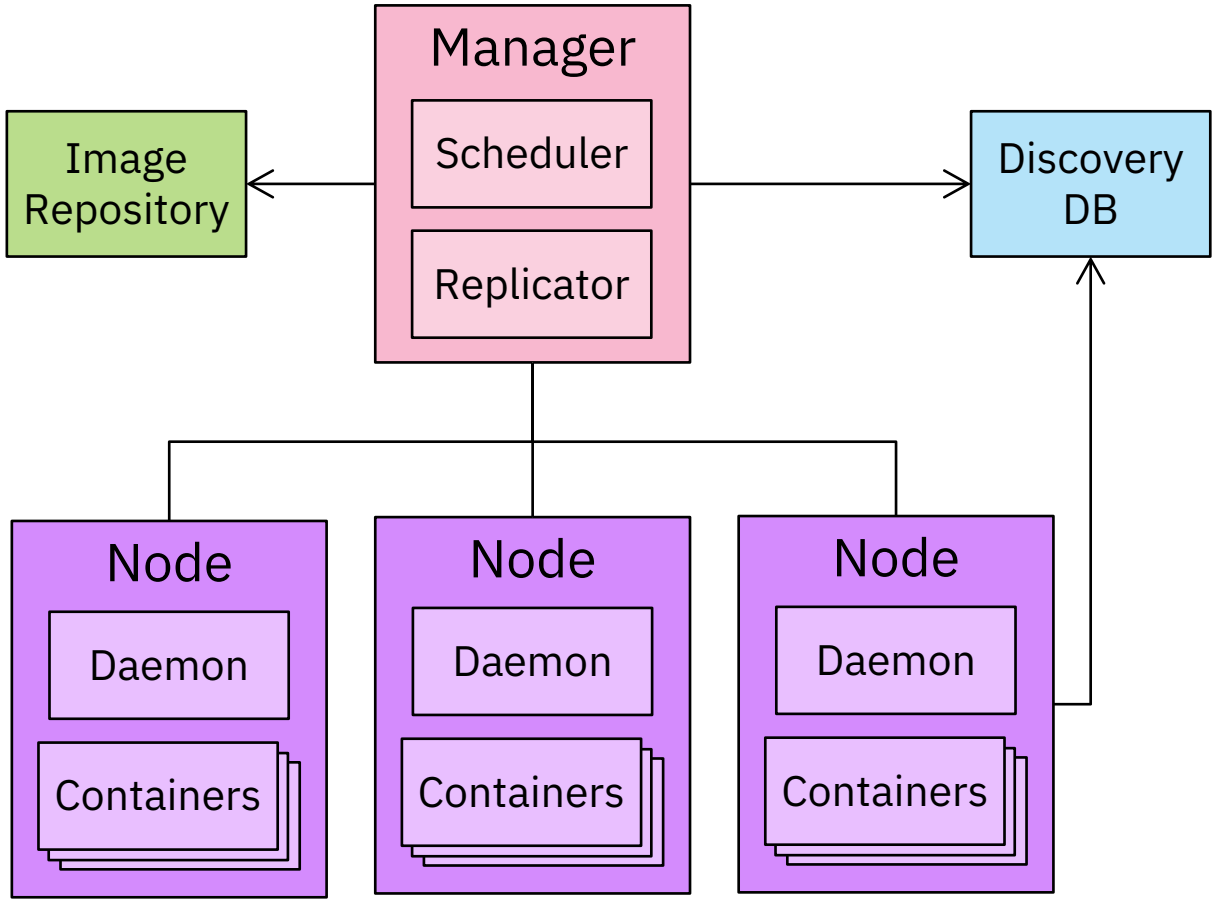
Allows users to define how to coordinate the containers in the cloud when the multi-container packaged application is deployed.

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

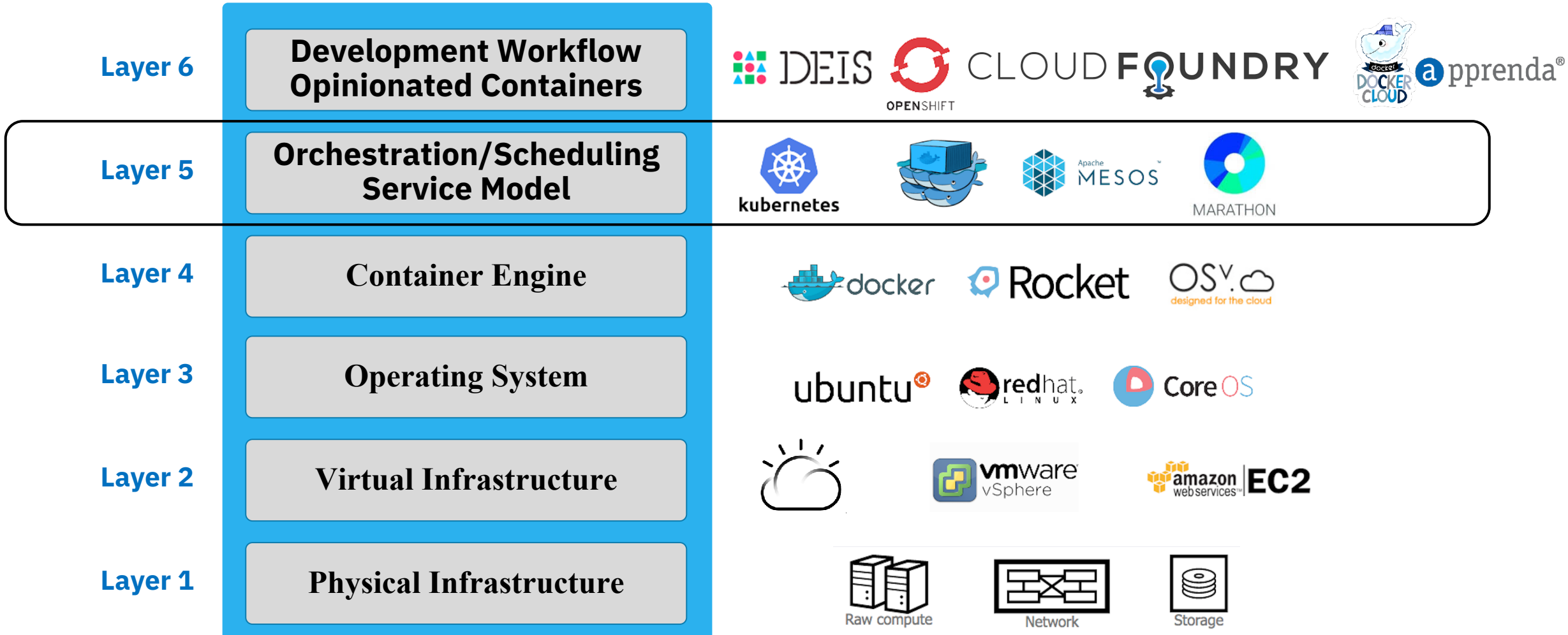
What is Container Orchestration?

- Container orchestration
  - Cluster management
  - Scheduling
  - Service discovery
  - Replication
  - Health management

Container Orchestrator



Container Ecosystem Layers



Kubernetes – Mesos – Marathon – Docker Swarm

Kubernetes is a cluster manager for containers (large deployments)

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 build to be run on mesos.

(Kubernetes can be run on Mesos)

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)

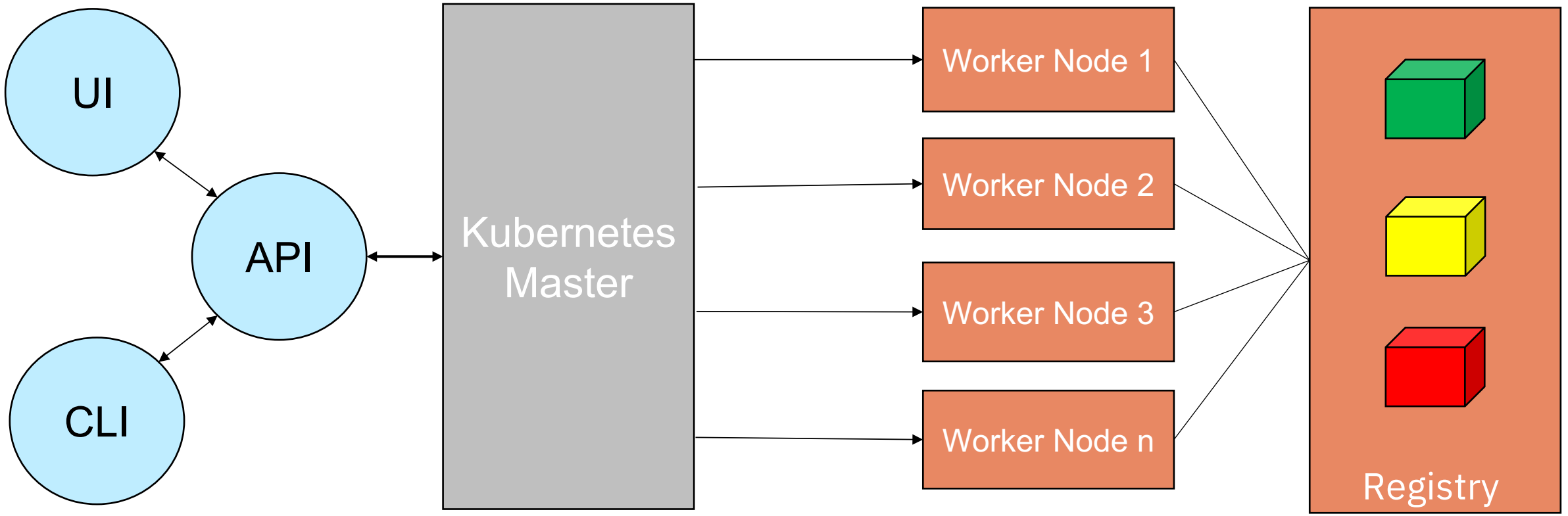
Marathon is a cluster-wide init and control system for running Linux services in cgroups and Docker containers, and it is run on top of Mesos

Kubernetes

What is Kubernetes?

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

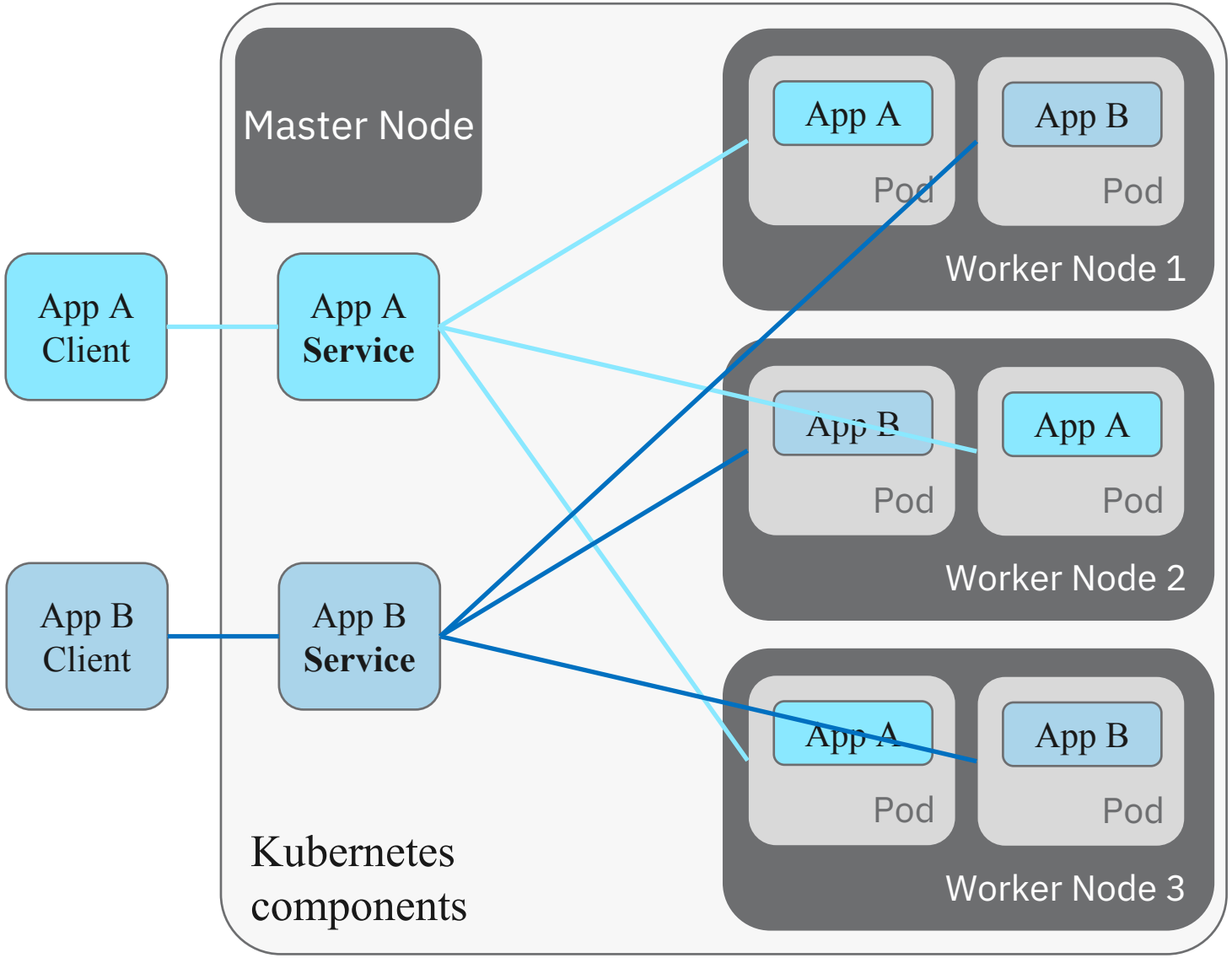
Kubernetes Architecture



- Etcd
- API Server
- Controller Manager Server
- Scheduler

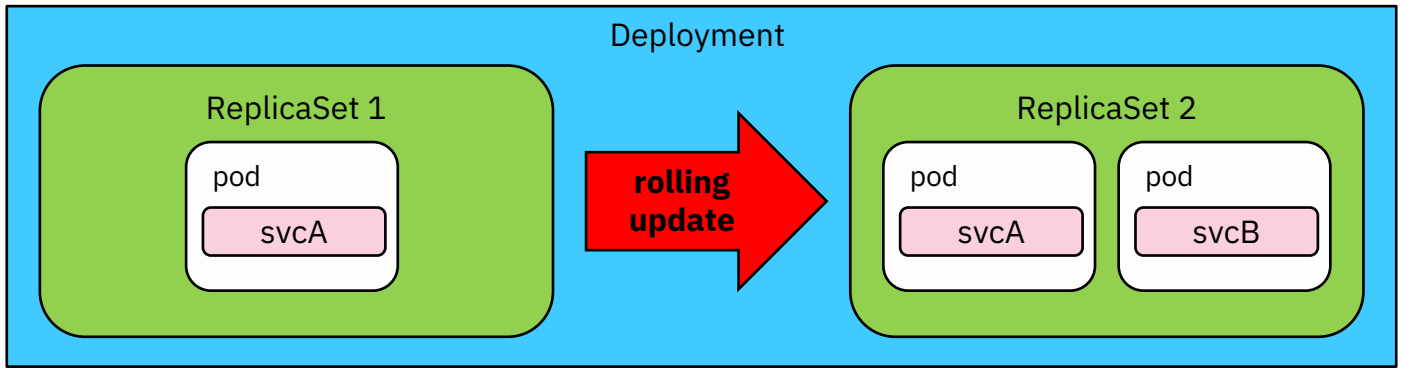
Kubernetes Architecture: Workloads

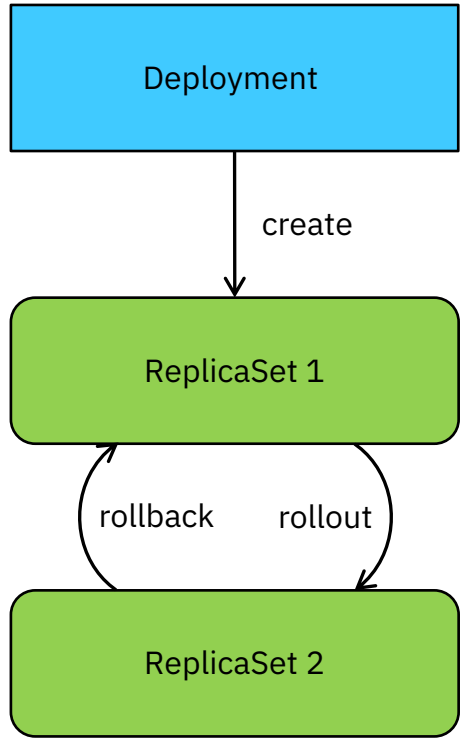
- Container
  - Packaging of an app
- Pod
  - Unit of deployment
- Service
  - Fixed endpoint for 1+ pods



Kubernetes Terminology: Deployment

- Deployment
  - A set of pods to be deployed together, such as an application
  - Declarative: Revising a Deployment creates a ReplicaSet describing the desired state
  - Rollout: Deployment controller changes the actual state to the desired state at a controlled rate
  - Rollback: Each Deployment revision can be rolled back
  - Scale and auto scale: A Deployment can be scaled
- ReplicaSet
  - The next-generation Replication Controller
  - A set of pod templates that describe a set of pod replicas
  - Uses a template that describes specifically what each pod should contain
  - Ensures that a specified number of pod replicas are running at any given time

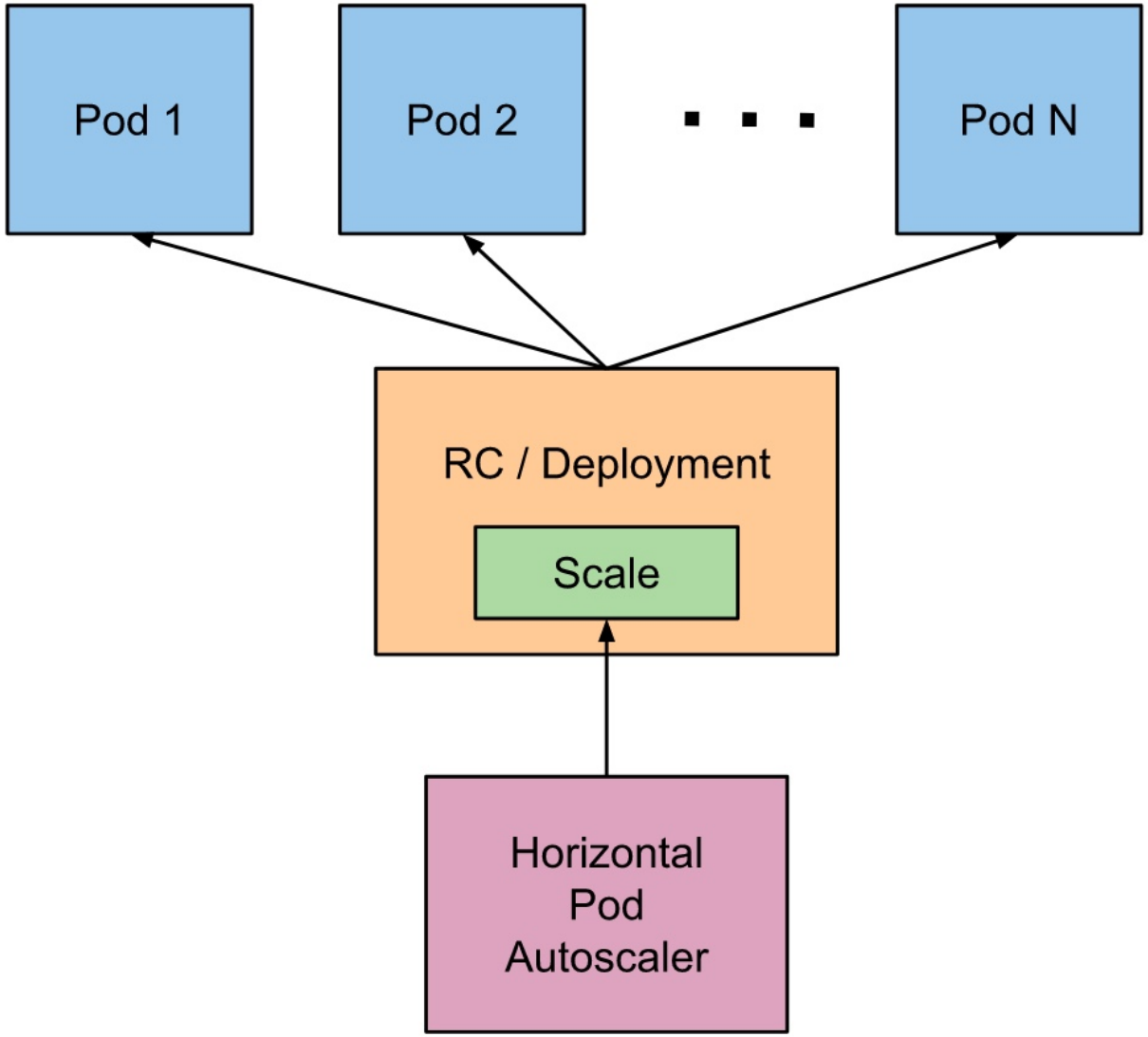




Kubernetes Terminology: Autoscaling

- Horizontal Pod Autoscaling (HPA)

```
$ kubectl autoscale deployment
<deployment-name> --cpu-percent=50
  --min=1 --max=10 deployment
"<hpa-name>" autoscaled
```



Common Kubernetes Commands

- Get the state of your cluster
  ```
  $ kubectl cluster-info
  ```
- Get all the nodes of your cluster
  ```
  $ kubectl get nodes -o wide
  ```
- Get info about the pods of your cluster
  ```
  $ kubectl get pods -o wide
  ```
- Get info about the replication controllers of your cluster
  ```
  $ kubectl get rc -o wide
  ```
- Get info about the services of your cluster
  ```
  $ kubectl get services
  ```
- Get full config info about a Service
  ```
  $ kubectl get service
  NAME_OF_SERVICE -o json
  ```
- Get the IP of a Pod
  ```
  $ kubectl get pod NAME_OF_POD -
  template={{.status.podIP}}
  ```
- Delete a Pod
  ```
  $ kubectl delete pod NAME
  ```
- Delete a Service
  ```
  $ kubectl delete service
  NAME_OF_SERVICE
  ```

Helm & Helm Chart

- Helm
  - Package manager for Kubernetes
  - Used to manage Kubernetes applications
- Helm Chart
  - Used to define, install, and upgrade complex Kubernetes applications
  - Easy to create, version, share and publish
  - Expressed in “Yet Another Markup Language” (YAML) files

Kubernetes in the IBM Cloud

http://ibm.biz/kubernetes070318

Kubernetes in the IBM Cloud

LEVERAGING KUBERNETES IN IBM CLOUD PUBLIC AND PRIVATE

Combine the power of open source with the scale of the IBM Cloud

Accelerate Innovation and improve time-to-market by empowering developers to rapidly build cloud-native and cloud compatible applications for both new and existing workloads

Integrate cloud and on-prem applications to a host of enterprise-scale capabilities and services with the applications you are developing such as analytics, Machine Learning, AI (Watson), middleware, and IoT services.

- Leverage operational services to help you manage your environment, including monitoring, log management, and security

http://ibm.biz/kubernetes070318

Kubernetes in the IBM Cloud

TWO PRIMARY OPTIONS

IBM Cloud (Public)

IBM Kubernetes Container Service available as a fully managed service, as well as private Docker registry

Pods and Containers can leverage other IBM Cloud services, such as Watson AI, the Watson Data Platform, and many others

Pods and Containers can access your services either on-prem or from other cloud providers through secure means

Full DevOps services available to help manage application development

IBM Cloud Private

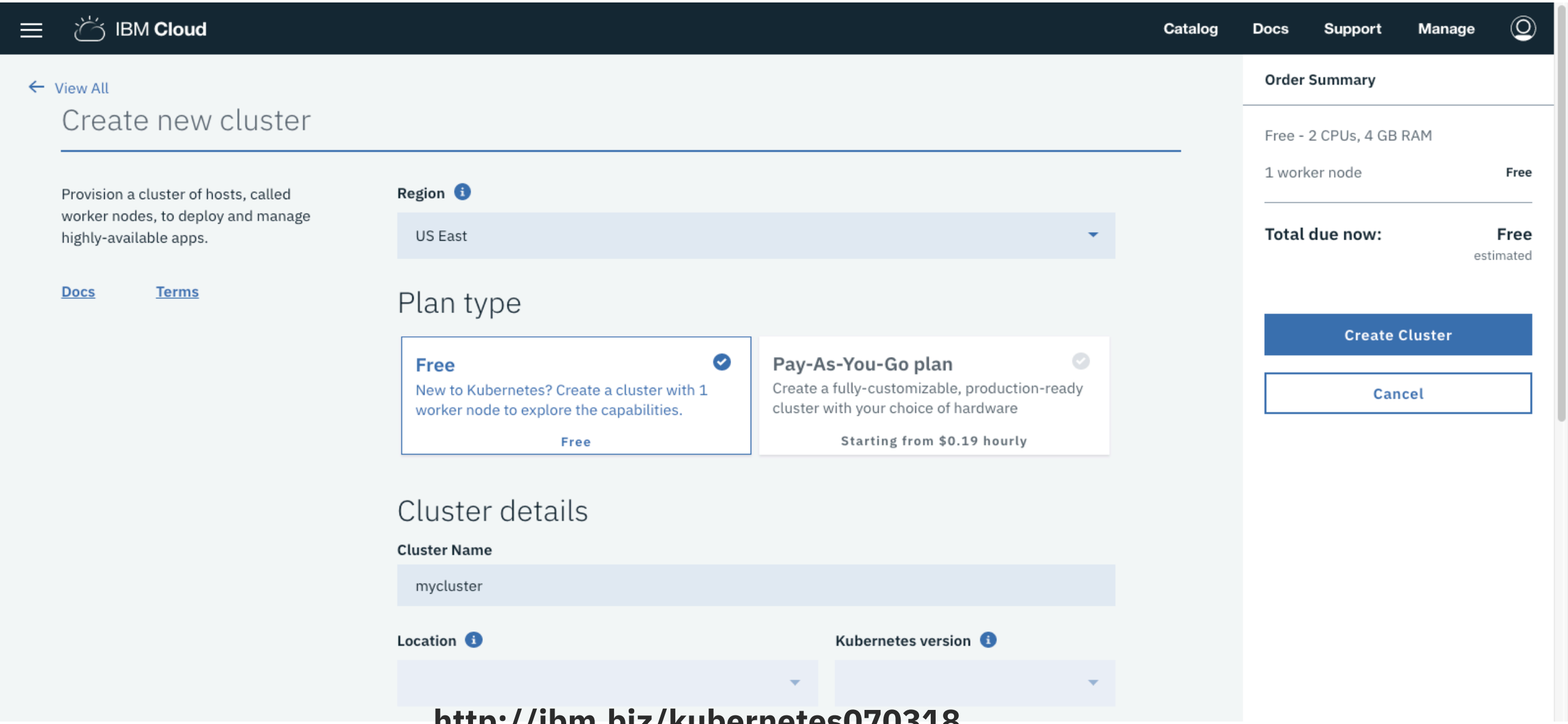
An operational Kubernetes-focused development and production version of IBM Cloud for deployment in your on-prem or cloud environment

Enterprise-quality management, scalability, security, and resiliency features to support your Kubernetes Cluster deployment

Built-in access to enterprise services such as analytics, middleware, data storage, and data science

Access and integrate to your other on-prem and/or cloud services

IBM Cloud Kubernetes Options



http://ibm.biz/kubernetes070318

Istio

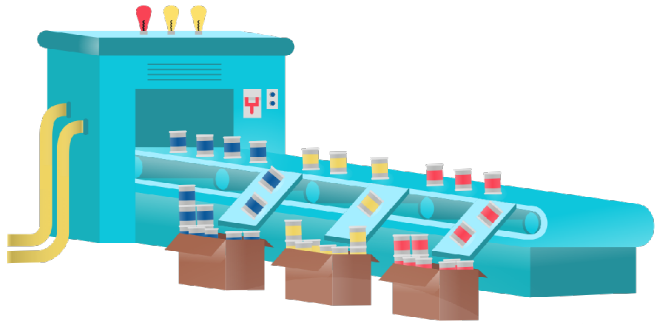
An open platform to connect, manage, and secure microservices

http://istio.io

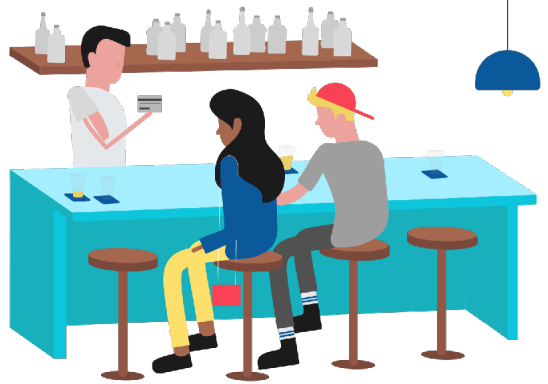
Istio

WHERE ISTIO TYPE OF SOFTWARE FINDS ITS PLACE

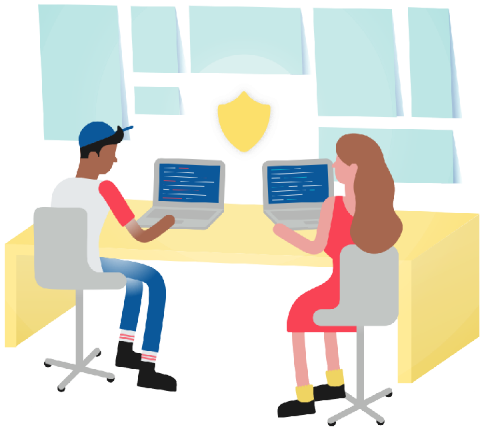
Intelligent Routing and Load Balancing



Fleet Wide Policy Enforcement

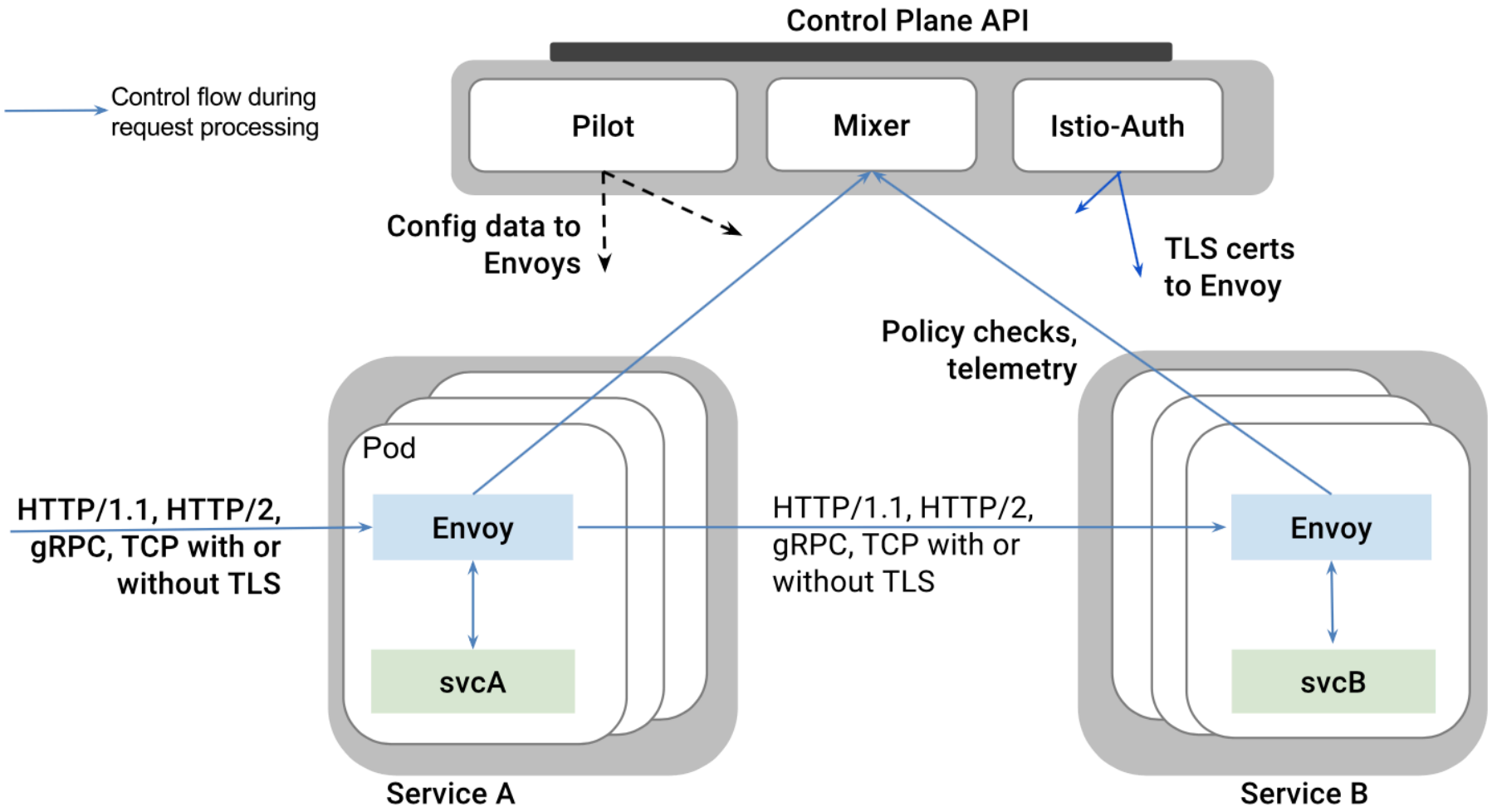


Resiliency across Languages and Platforms



In-Depth Telemetry and Reporting

Istio Architecture



https://istio.io/docs/concepts/what-is-istio/overview.html#architecture

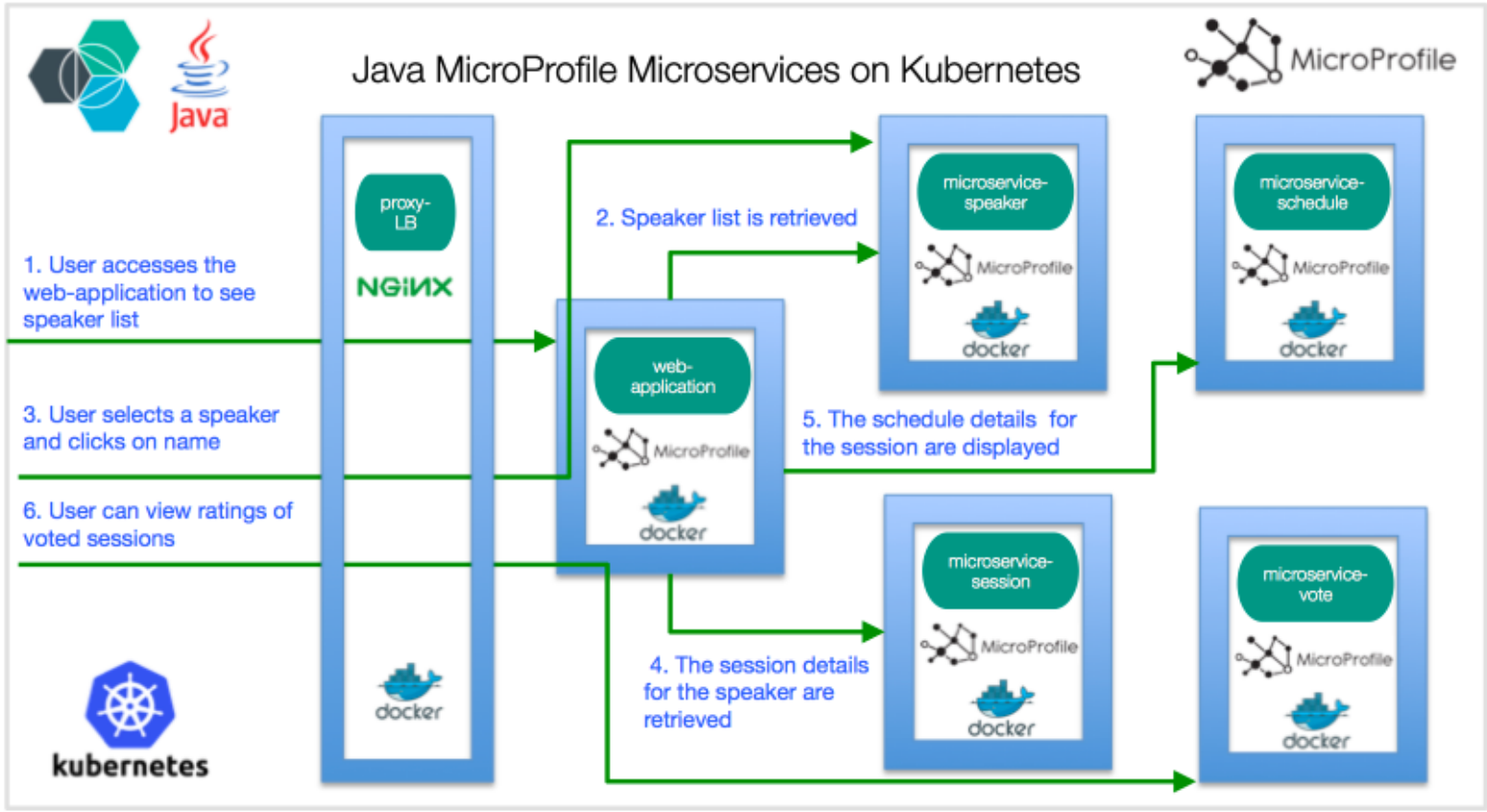
Conclusion

Java Based Code Example – aka pattern

Java based Microservices application using MicroProfile (baseline for Java Microservices architecture) and Microservices Builder on Kubernetes

Developer Works
Code: https://developer.ibm.com/code/journey/deploy-microprofile-java-microservices-on-kubernetes

Github: https://github.com/IBM/java-microprofile-on-kubernetes



Conclusion

- Why containers?
- Docker containers
- Container orchestration

- The following webinar will go in depth with the following subject:
  - **Istio 101 - 7/10**

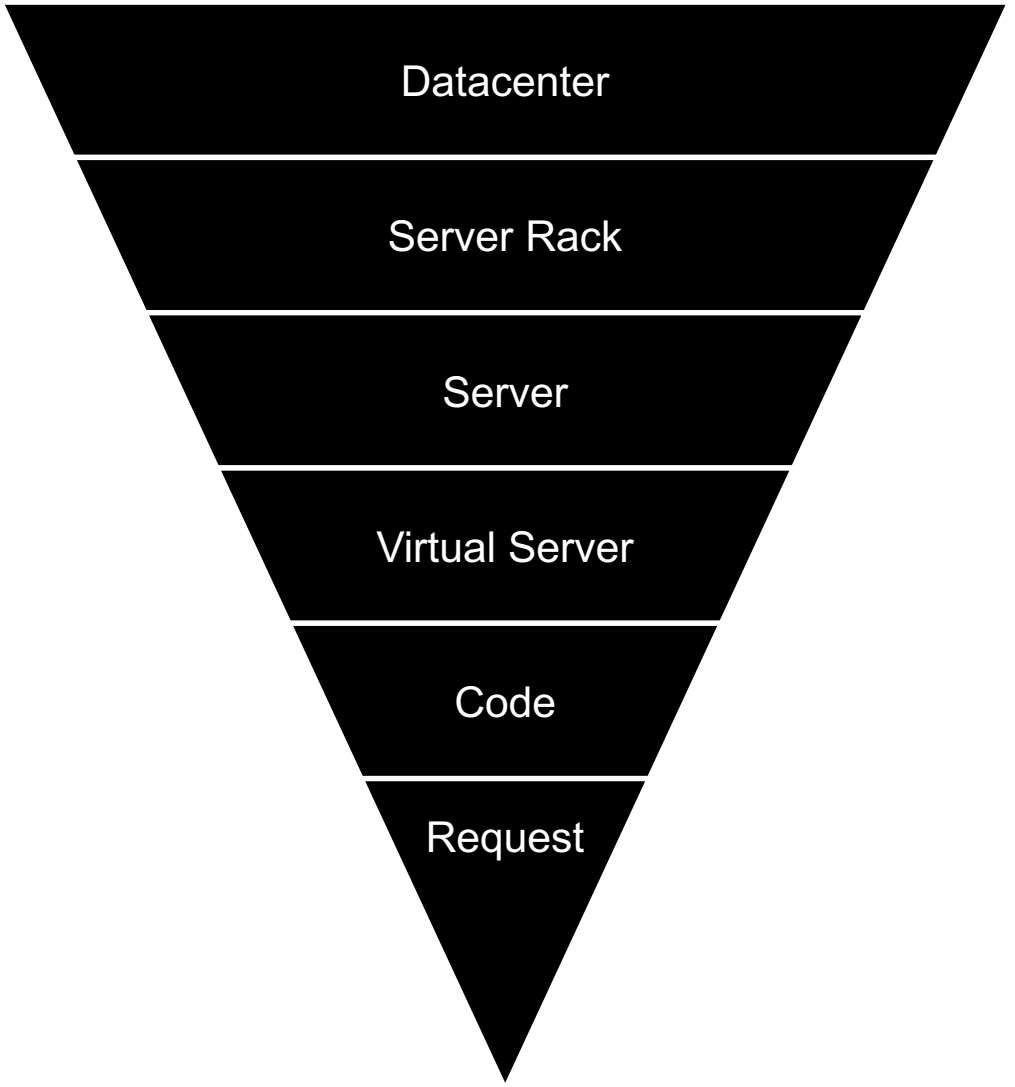
Resources

- Docker tutorial - https://docs.docker.com/get-started/
- Kubernetes tutorial - https://kubernetes.io/docs/tutorials/kubernetes-basics/
- The Evolution of Linux Containers and Their Future
  - https://dzone.com/articles/evolution-of-linux-containers-future
- Introduction to container orchestration
  - https://www.exoscale.ch/syslog/2016/07/26/container-orch/
- TNS Research: The Present State of Container Orchestration
  - https://thenewstack.io/tns-research-present-state-container-orchestration/
- Large-scale cluster management at Google with Borg
  - https://research.google.com/pubs/pub43438.html
- Call for code: http://developer.ibm.com/callforcode
- The other SF City events: https://developer.ibm.com/code/community/cities/san-francisco/

http://ibm.biz/kubernetes070318

Many possibilities for failures – join our Istio webinar

- Failures WILL happen
- Their root causes can be complex and affect different number of users
- Reduce the likelihood of correlated failures
- Recover quickly
- Proactive detection of problems before they happen



Thank you

twitter.com/blumareks

github.com/blumareks

developer.ibm.com/code