



What the Containers?

Arun Gupta, @arungupta

Docker Captain

Java Champion

JavaOne Rock Star (4 years)

NetBeans Dream Team

Silicon Valley JUG Leader

Author

Runner

Lifelong learner











Malcom McLean

Father of Containerization

Intermodalism

*“pertaining to or suitable for transportation involving **more than one form of carrier**”*



Advantages of Containerization

- Standard transport product
- Flexibility of usage
- Economies of scale
- Speed
- Security
- Lower labor costs
- Efficient stacking
- Boom in international trade

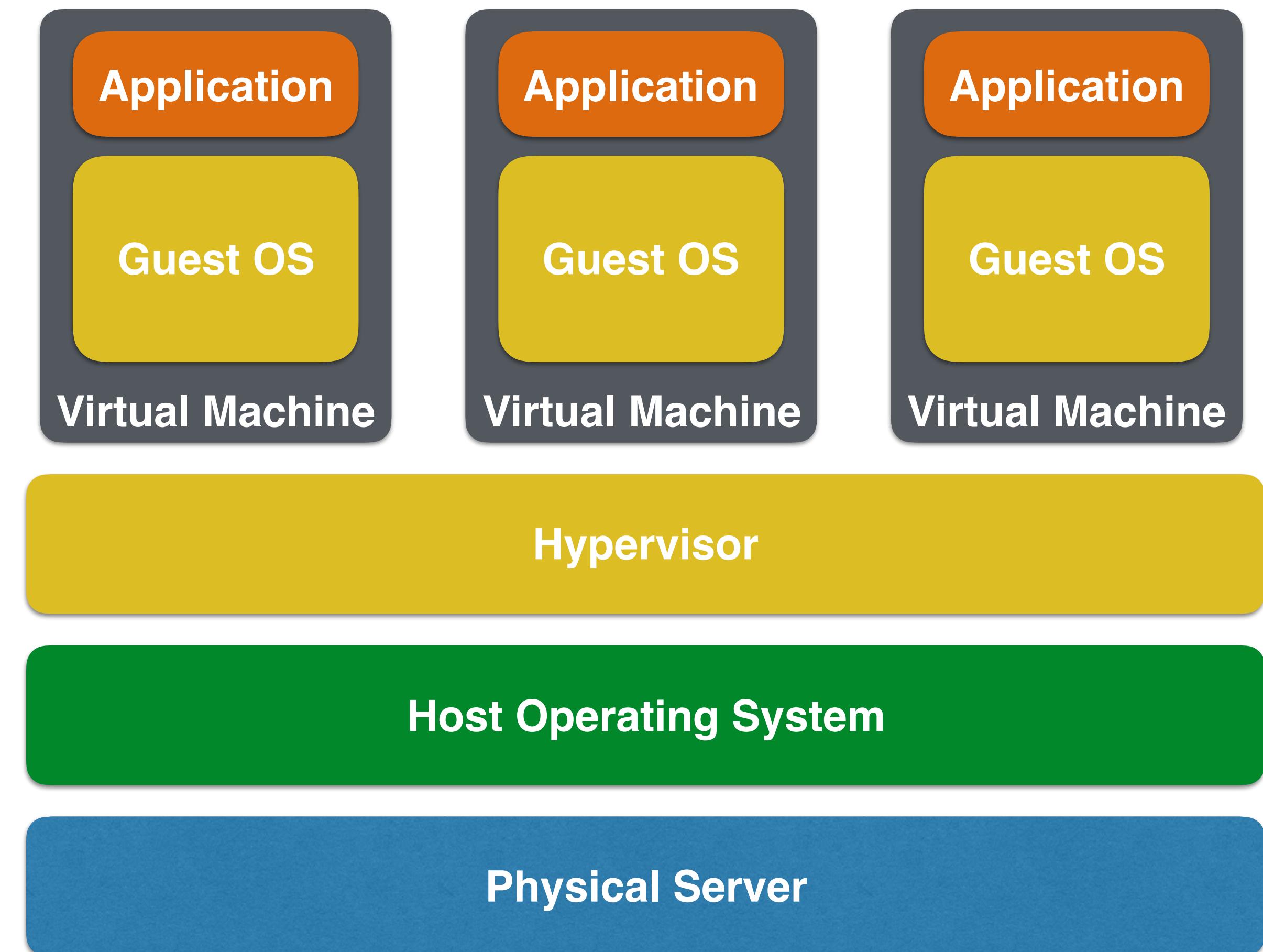
Bare Metal

Application

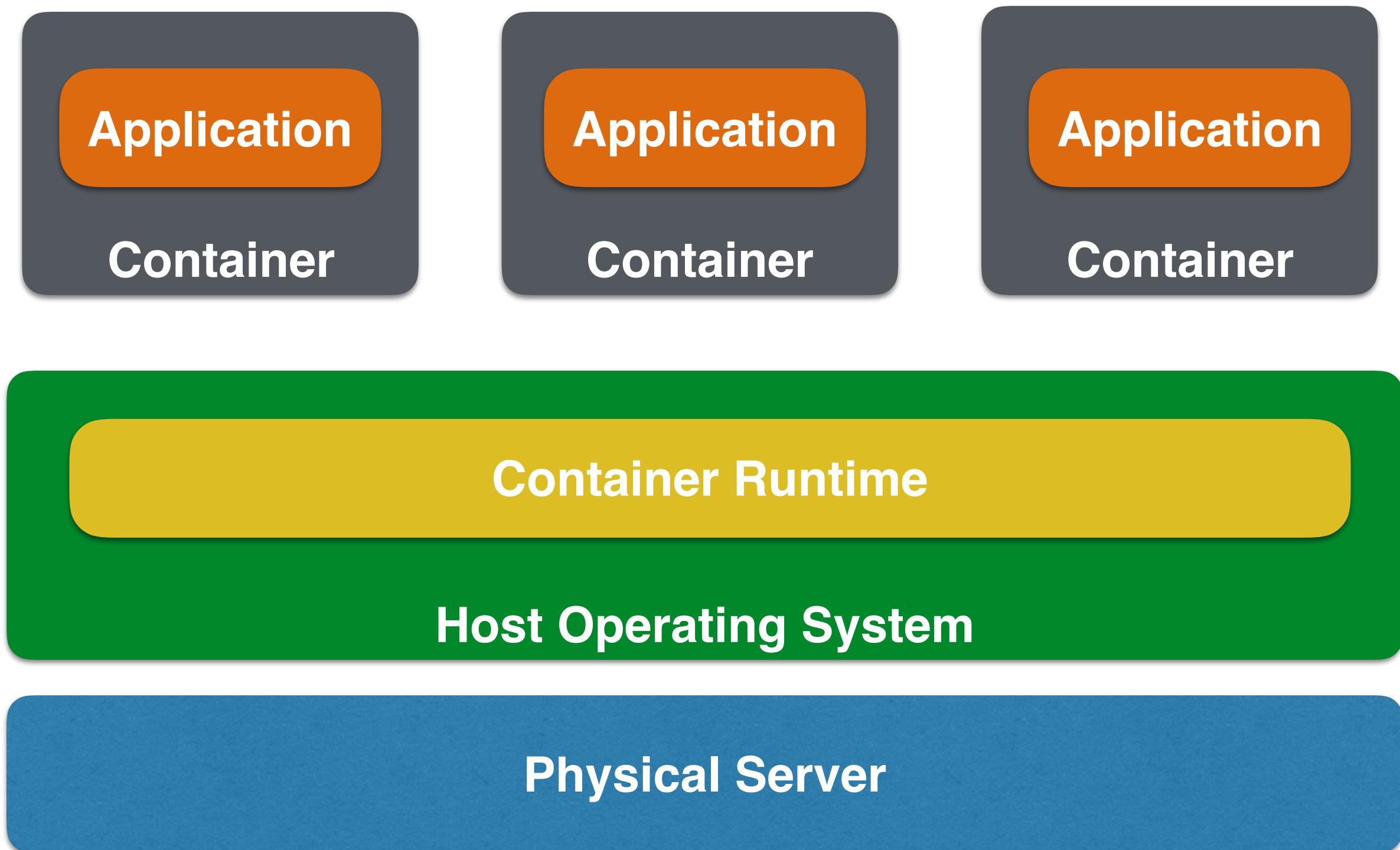
Operating System

Physical Server

Virtual Machines



Containers



Advantages of Containerization

- Standard transport product
- Flexibility of usage
- Economies of scale
- Speed
- Security
- Lower labor costs
- Efficient stacking
- Boom in international trade
- Standard packaging
- Application agnostic
- High density
- Faster deployment
- Security sandbox
- Easy portability
- Orchestration frameworks
- Tipping point

NATIONAL BESTSELLER

The TIPPING POINT

*How Little Things Can
Make a Big Difference*

MALCOLM
GLADWELL



- Cloud-native
- Microservices
- DevOps
- Docker

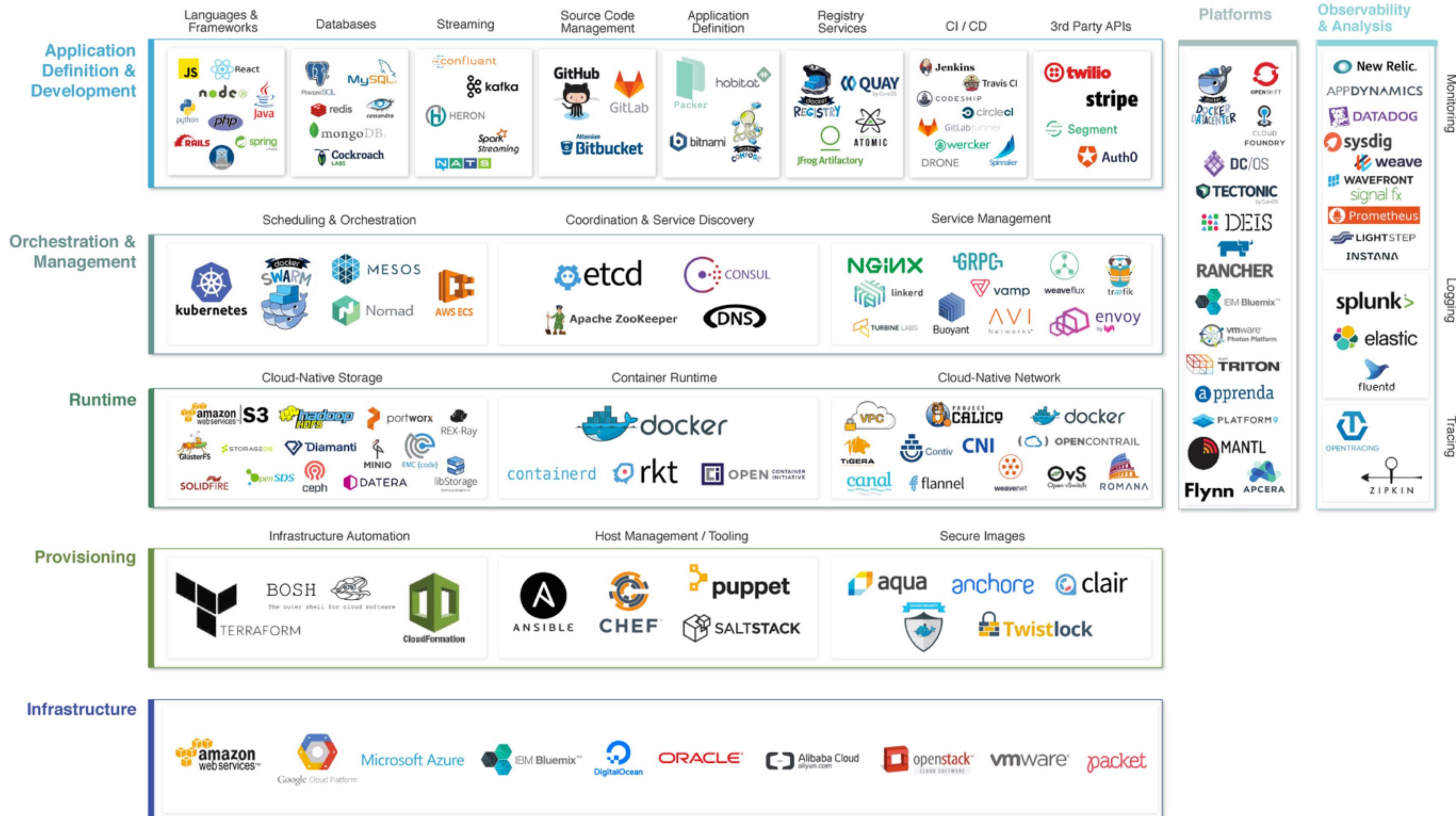


Create and drive adoption of computing paradigm optimized for distributed systems capable of scaling to tens of thousands of self-healing multi-tenant nodes



- Uses an open source software stack to
 - deploy applications as **microservices**
 - package each part into its own **container**
 - dynamically **orchestrate containers** for optimum resource utilization

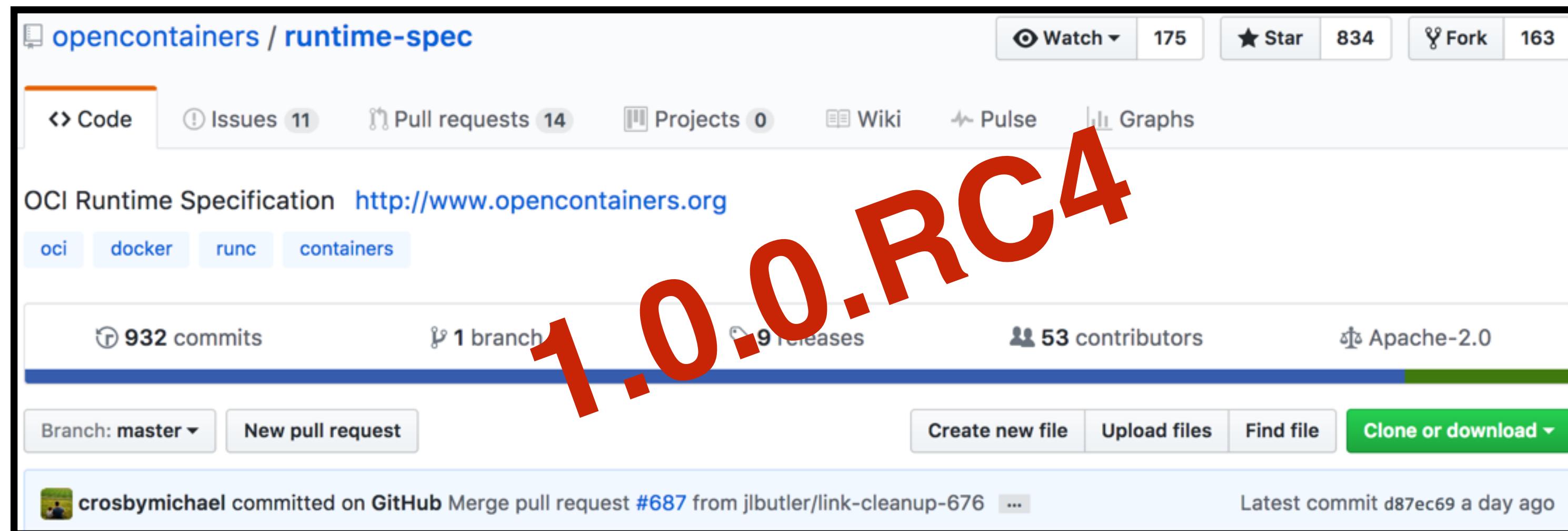
Cloud Native Landscape v0.9.3





promote a set of common, minimal,
open standards and specifications
around container technology

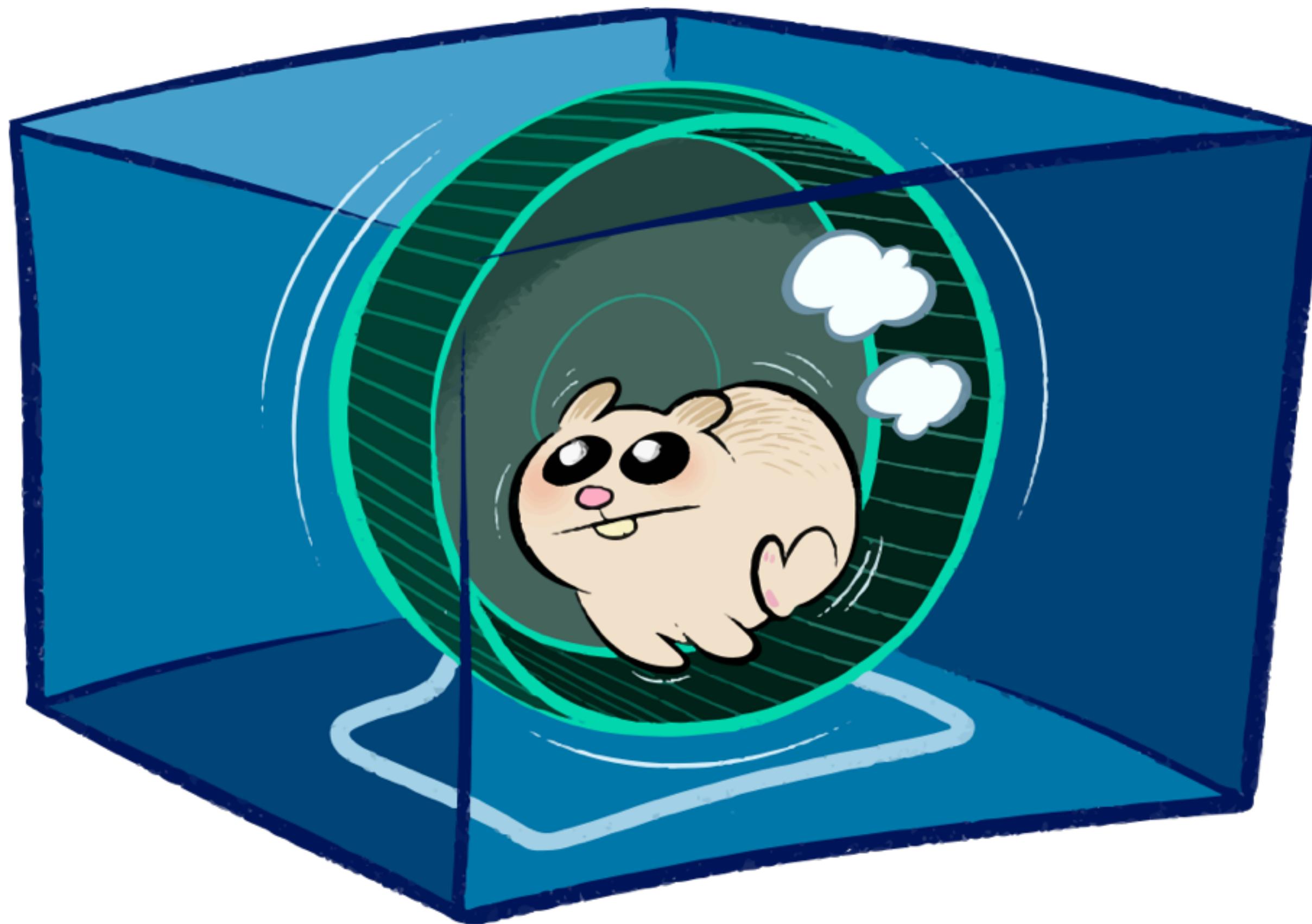
OCI Projects



OCI Image Spec

- Image format
 - Docker - `docker save` and `load`
- Registry
 - rkt - supports pull only
 - AWS ECR - push and pull OCI images
- Tools to generate OCI image
 - acbuild: github.com/containers/build
 - umoci: github.com/openSUSE/umoci

OCI Runtimes



runC

- Reference Implementation of the OCI runtime specification
 - Donated by Docker
 - Based on original *libcontainer* project
 - Interfaces with *cgroups* and *namespaces*
 - Manage the lifecycle of container
 - Lockstep with OCI specifications major version



 [opencontainers / runc](#)

[Code](#) [Issues 68](#) [Pull requests 39](#) [Projects](#) [Wiki](#) [Pulse](#) [Graphs](#)

[Watch ▾ 320](#) [Star 3,574](#) [Fork 557](#)

RC2

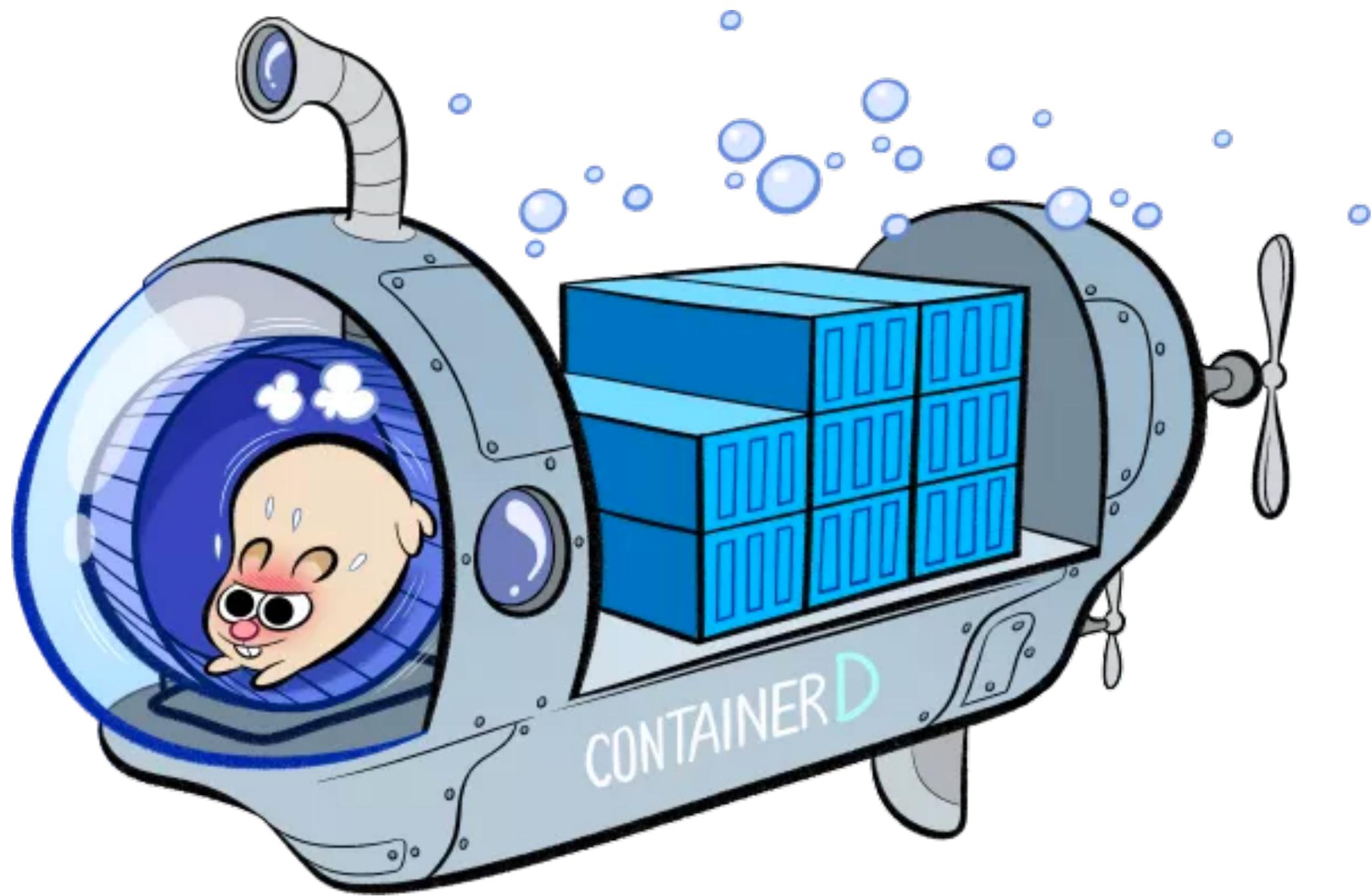
CLI tool for spawning and running containers according to the OCI specification <https://www.opencontainers.org/>

[containers](#) [docker](#) [oci](#)

©2016 Couchbase Inc.

containerd

- Daemon that uses runC to manage containers
- Exposes its functionality over gRPC
- Docker Engine uses API to run containers
 - Adds volumes, networks, images etc
 - Exposes a full-blown REST API



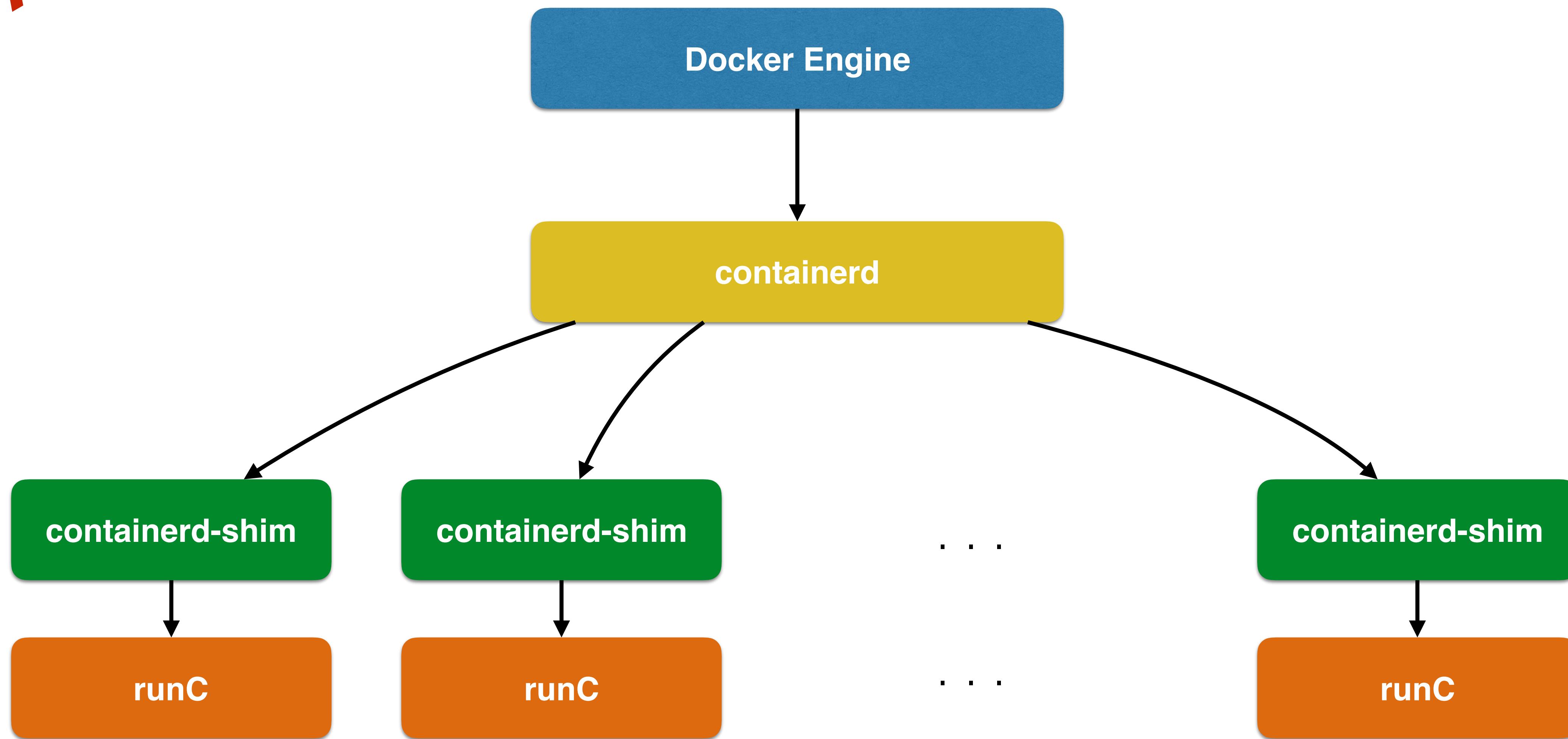
[docker / containerd](#)

Code Issues 36 Pull requests 11 Projects 0 Wiki Pulse Graphs

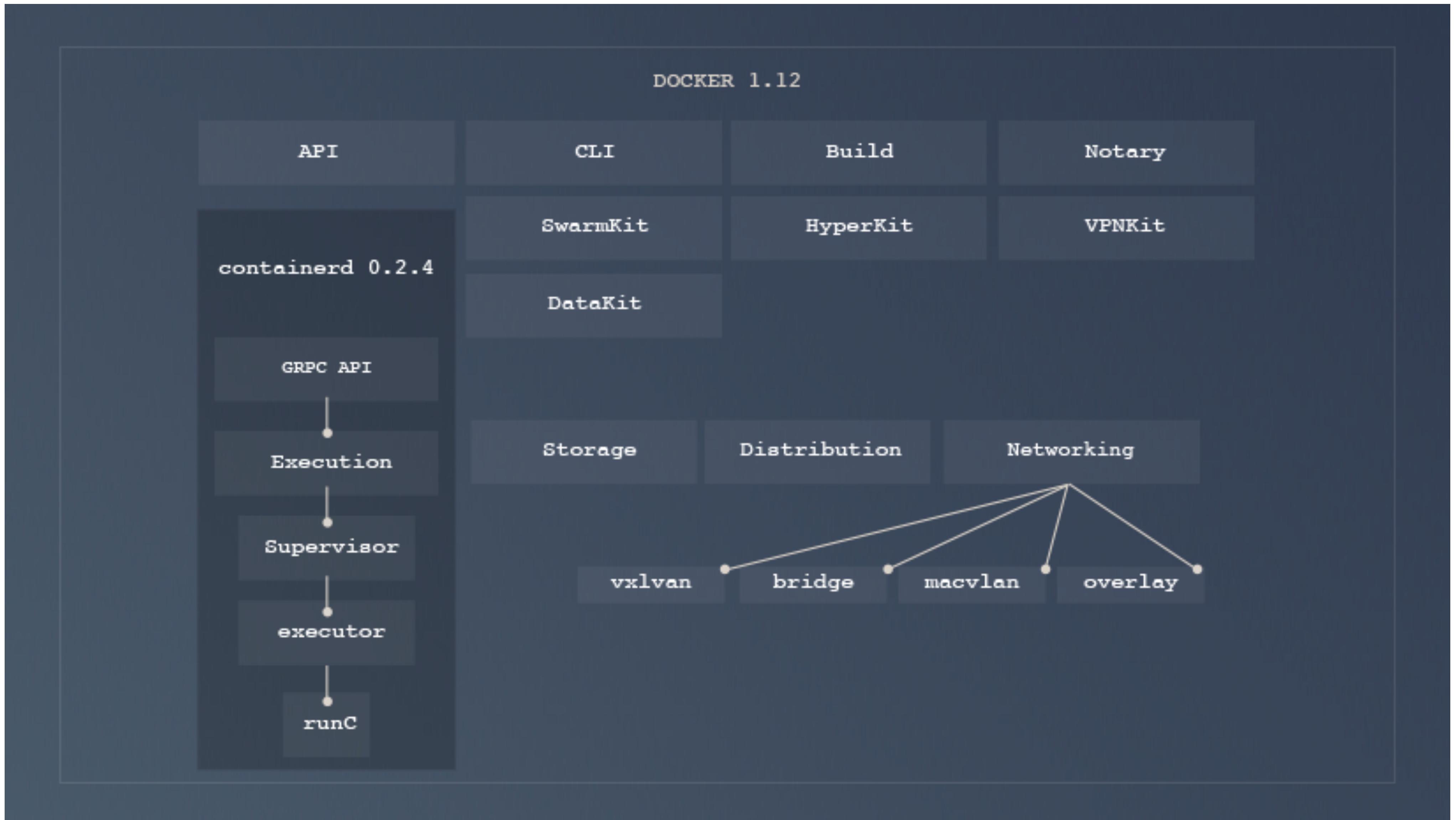
Watch 141 Star 1,238 Fork 216

0.2.5

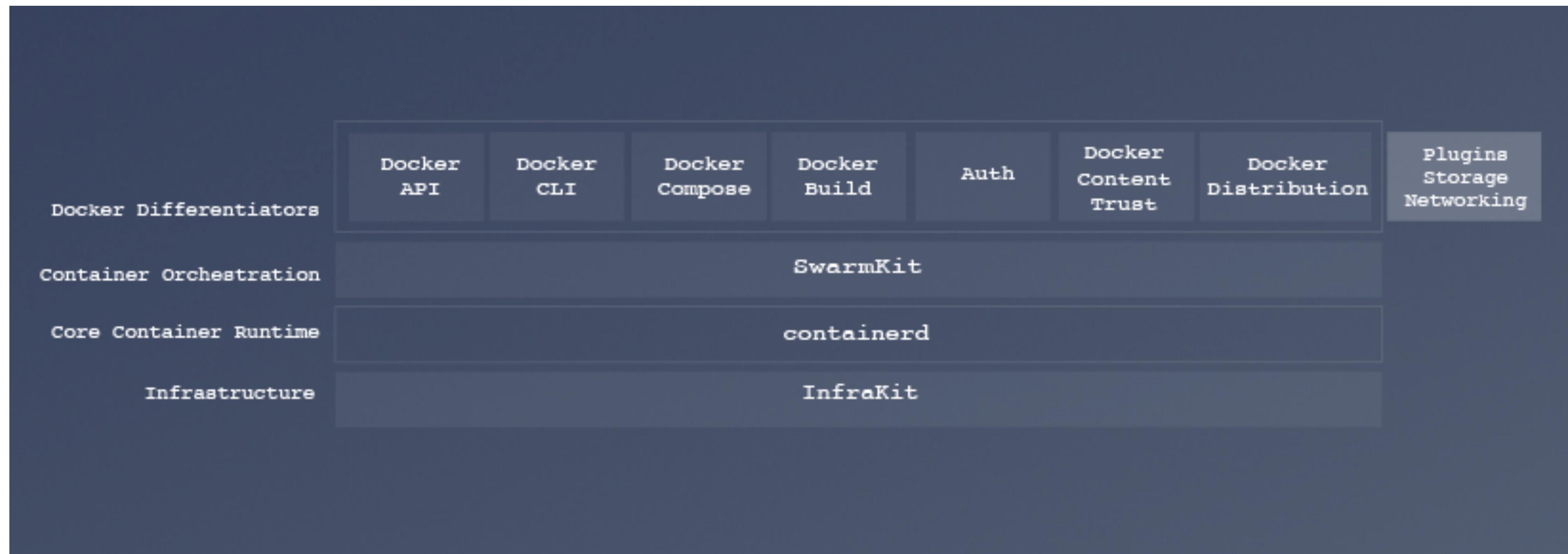
TODAY



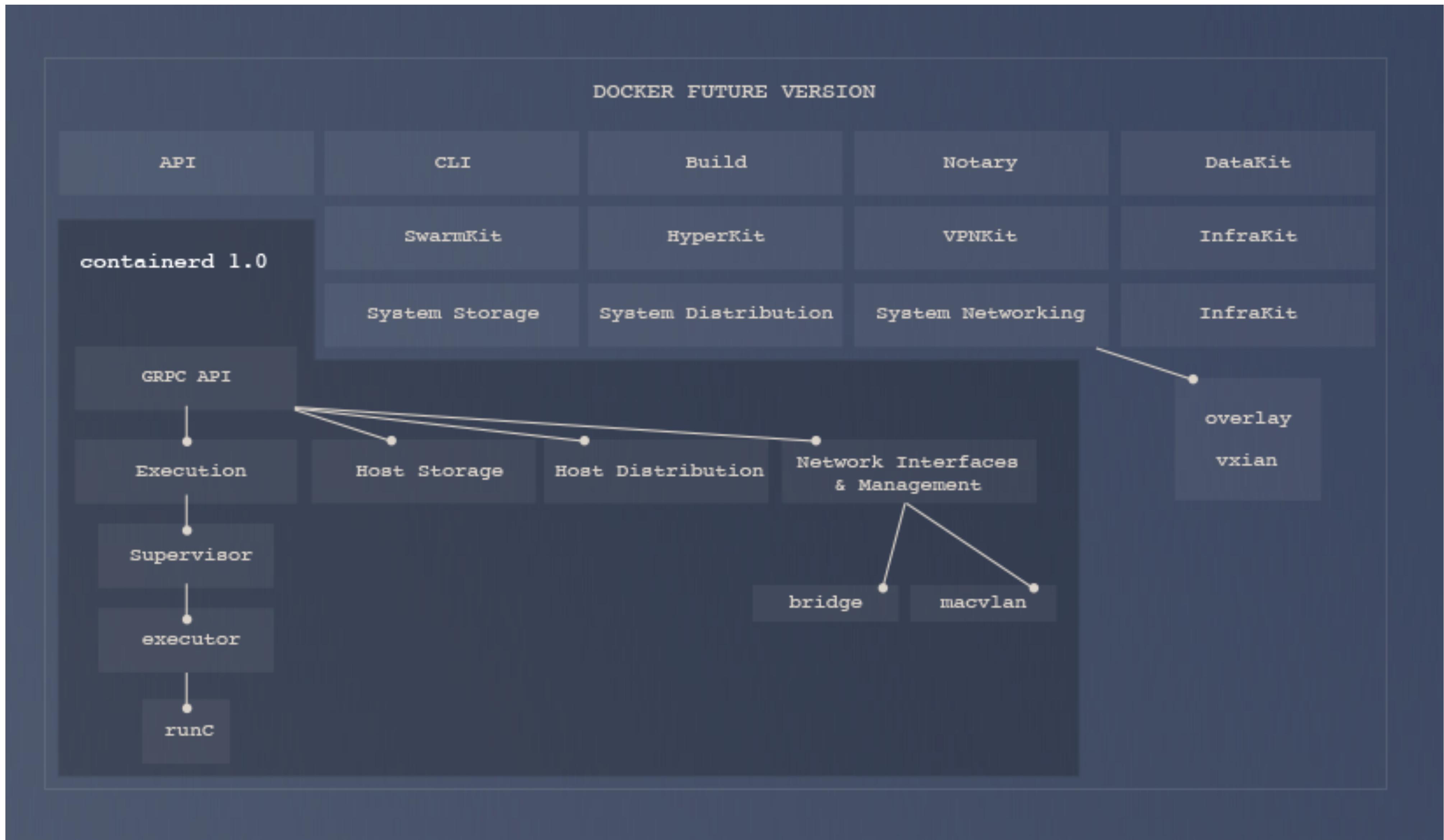
TODAY



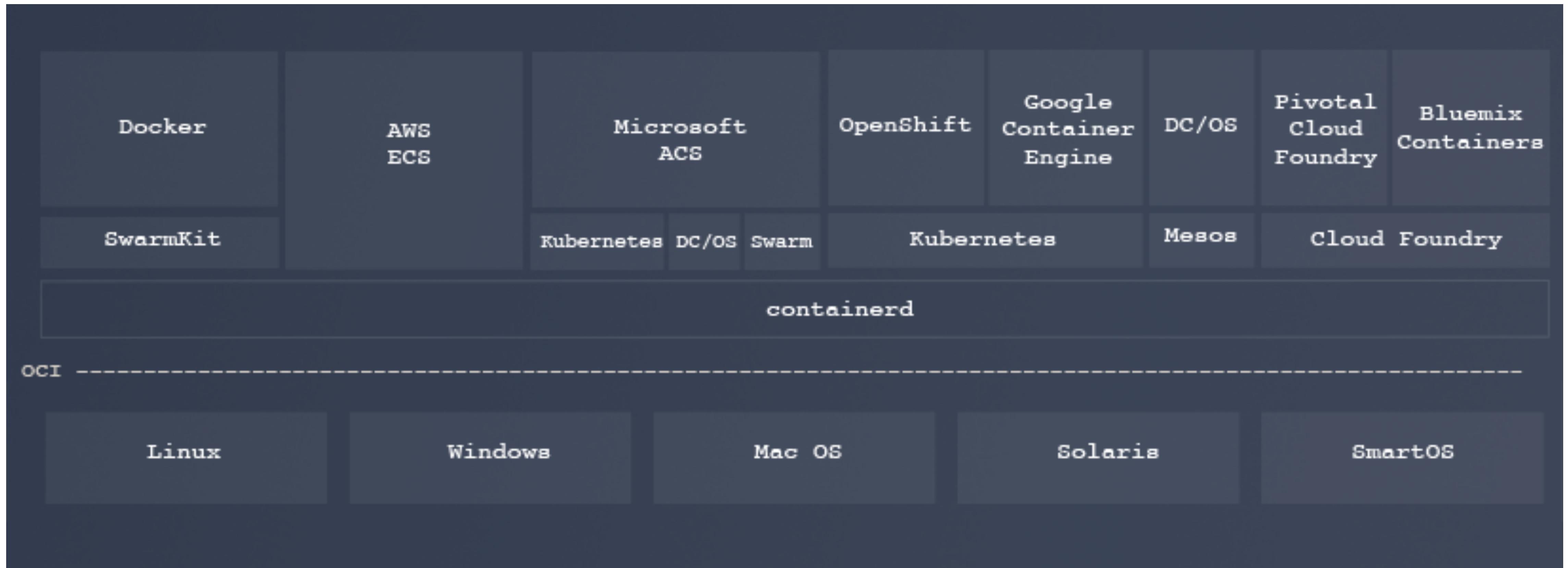
Containererd and Docker



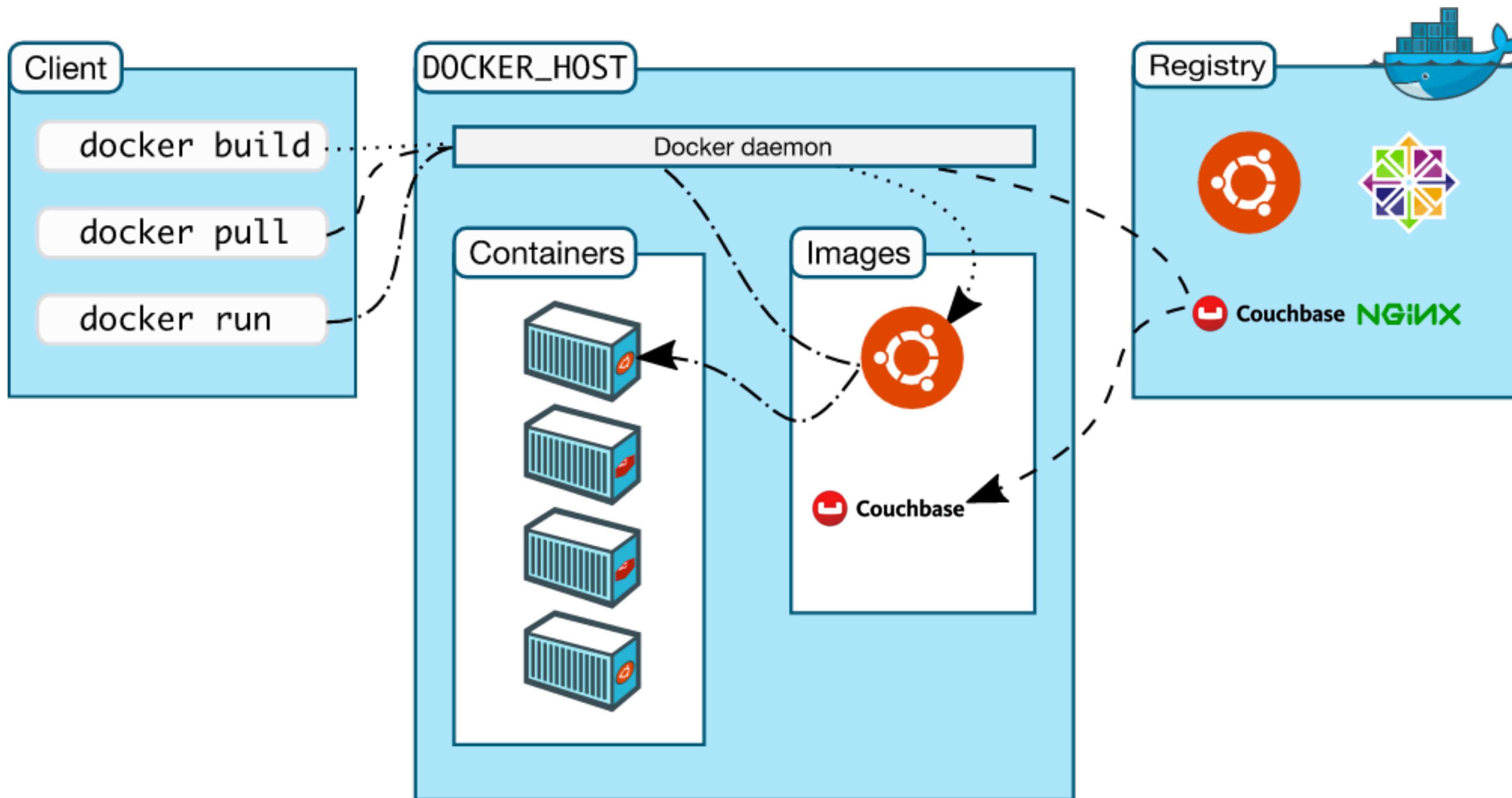
FUTURE

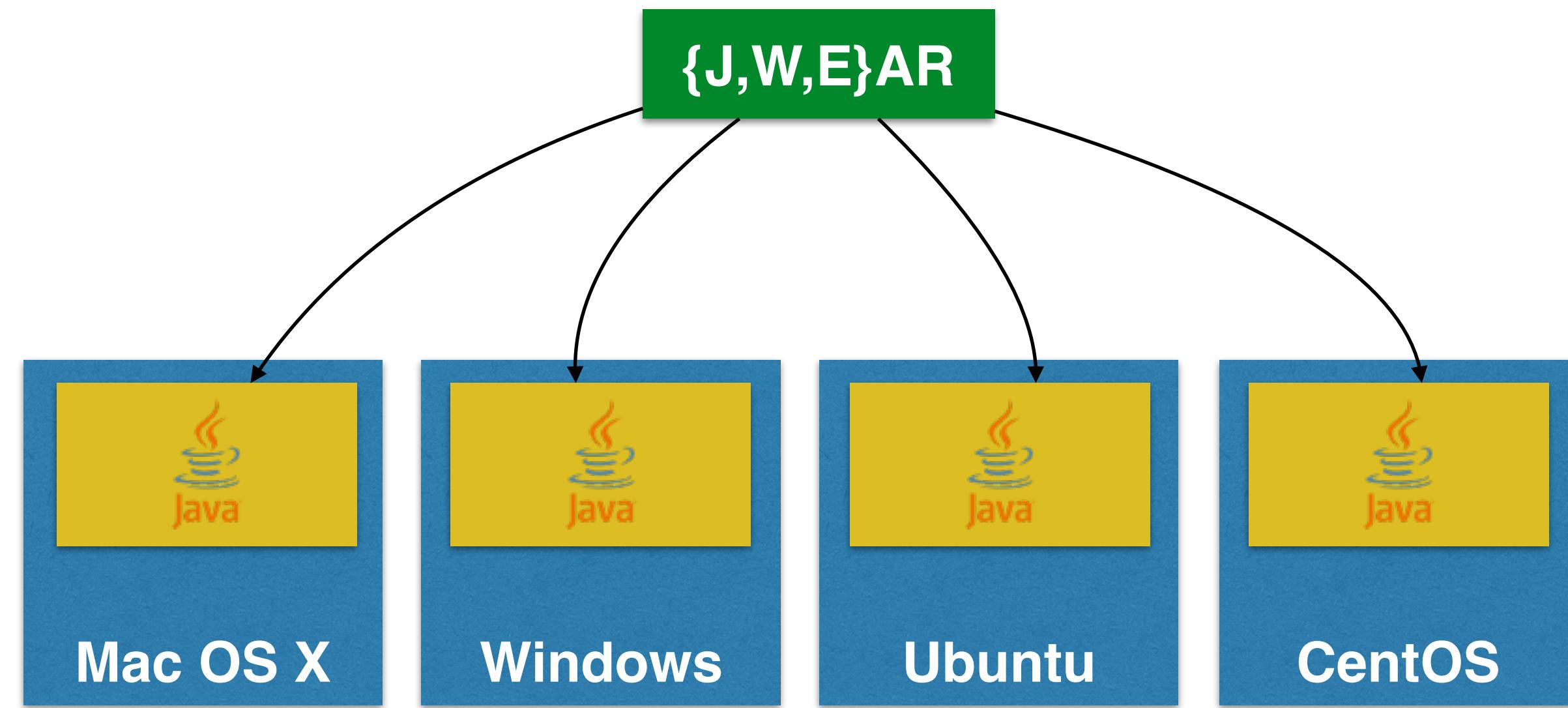


FUTURE Containerd and Orchestration Frameworks

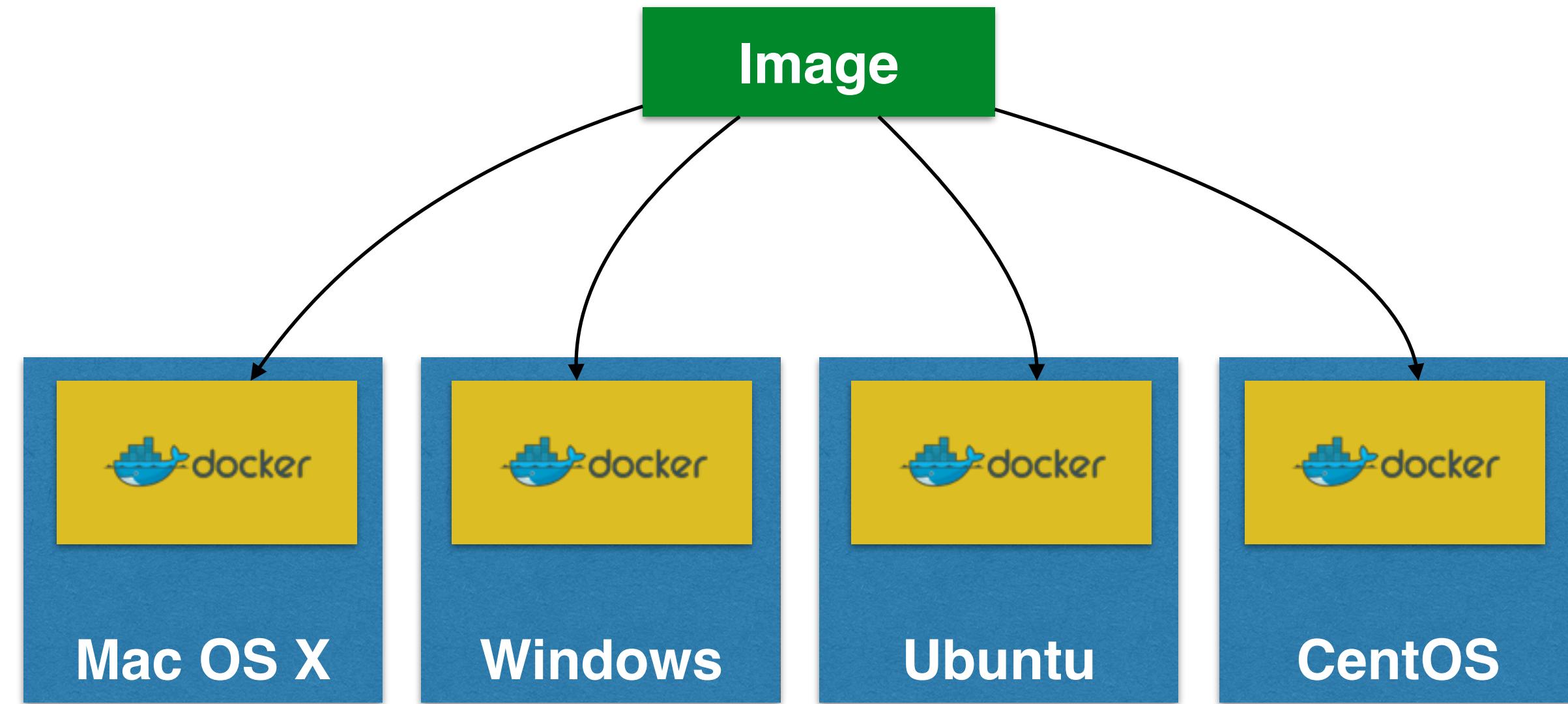


Container Workflow





WORA = Write Once Run Anywhere

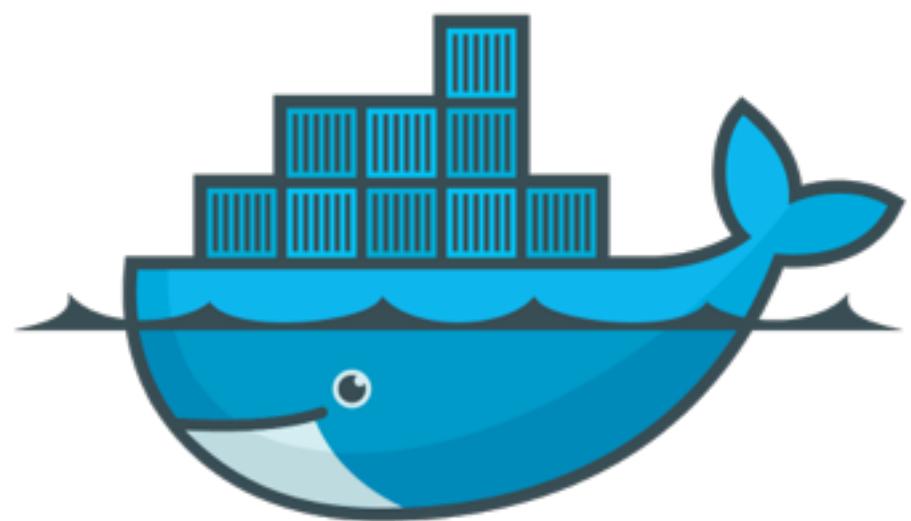


PODA = Package Once Deploy Anywhere

Orchestration Frameworks

- Local development
- Core concepts
- Declarative state
- Schedule containers
- Service discovery
- Load balancing
- System monitoring
- Persistent Volumes
- Multi-host networking
- Multiple master
- Rolling Update
- Rules and constraints
- Cloud/commercial support

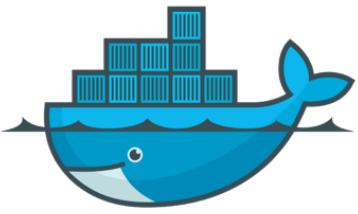
Orchestration Frameworks



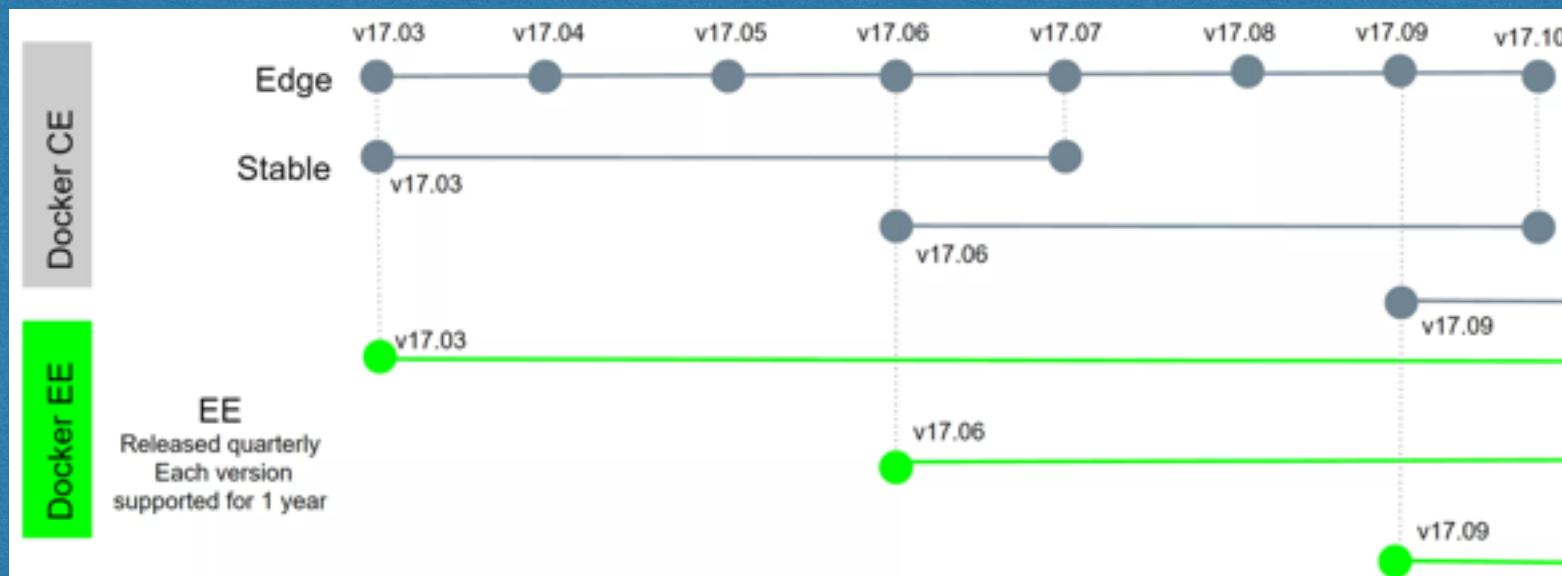
docker



Local Development



- Docker Community Edition
 - Docker for Mac/Windows/Linux
 - Monthly edge and quarterly stable
 - Native desktop or cloud provider experience



- Single-node cluster
 - Minikube
- Multi-node cluster
 - kops (AWS)
 - kube-aws (CoreOS + AWS)
 - kube-up (deprecated)
 - GCE, Azure, Tectonic, ...

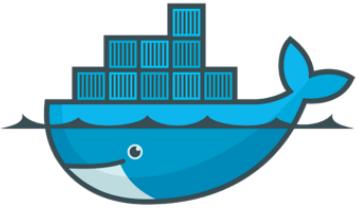


- Vagrant image
- Cloud
 - AWS CloudFormation templates
 - Azure Resource Manager templates
 - Digital Ocean using Terraform
 - GCE (using scripts)



- Amazon Web Services

Core Concepts



- Ops
 - Managers
 - Workers
- Developer
 - Replicated and Global Service
 - Tasks
 - Scaling
 - Run-once
 - Compose



- Ops
 - Master
 - Worker
- Developer
 - Pods (Horizontal Pod Autoscaling)
 - Services
 - Deployment
 - Replica Set
 - Daemon Set
 - Job



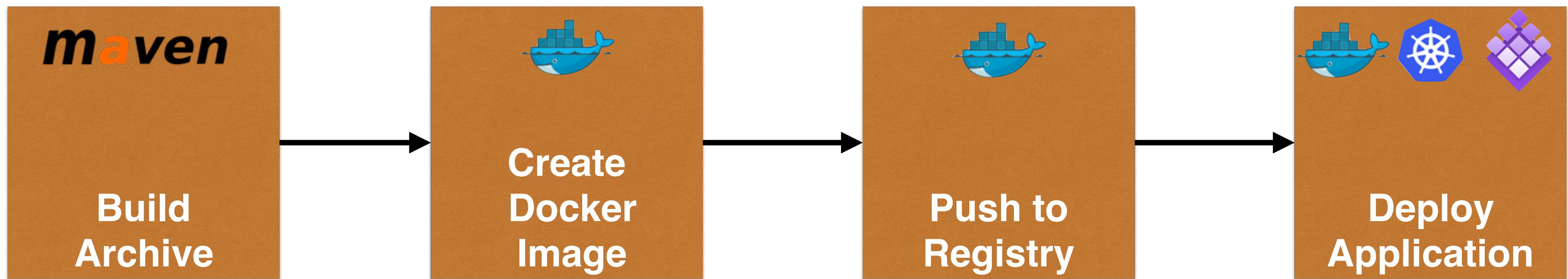
- Ops
 - Master & Slaves (Mesos, public/private)
 - Frameworks (Marathon)
- Developer
 - Application
 - Task
 - Pod (multiple tasks that share)
 - Job



- Ops
 - Container Instance
 - Cluster
- Developer
 - Service
 - AutoScaling - integrated with CloudWatch
 - Task



Containers and Java Developers

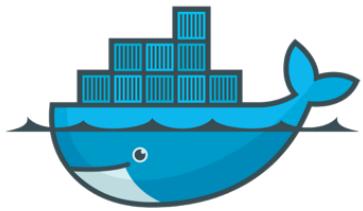


Docker Image with Maven

```
<plugin>
  <groupId>io.fabric8</groupId>
  <artifactId>docker-maven-plugin</artifactId>
  <version>0.19.0</version>
  <configuration>
    <images>
      <image>
        <name>hellojava</name>
        <build>
          <from>openjdk:latest</from>
          <assembly>
            <descriptorRef>artifact</descriptorRef>
            <assembly>
              <cmd>java -jar maven/${project.name}-${project.version}.jar</cmd>
              </build>
              <run>
                <wait>
                  <log>Hello World!</log>
                </wait>
              </run>
            </image>
          </images>
        </configuration>
```

```
<executions>
  <execution>
    <id>docker:build</id>
    <phase>package</phase>
    <goals>
      <goal>build</goal>
    </goals>
  </execution>
  <execution>
    <id>docker:start</id>
    <phase>install</phase>
    <goals>
      <goal>run</goal>
      <goal>logs</goal>
    </goals>
  </execution>
</executions>
</plugin>
```

Application Definition



```
version: "3"
services:
  db:
    image: arungupta/couchbase:travel
    ports:
      - 8091:8091
      - 8092:8092
      - 8093:8093
      - 11210:11210
  web:
    image: arungupta/wildfly-couchbase-javaee:travel
    environment:
      - COUCHBASE_URI=db
  ports:
    - 8080:8080
    - 9990:9990
```

Application Definition



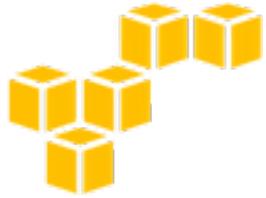
```
apiVersion: v1
kind: Service
metadata:
  name: couchbase-service
spec:
  selector:
    app: couchbase-rc-pod
  ports:
    - name: admin
      port: 8091
    - name: views
      port: 8092
    - name: query
      port: 8093
    - name: memcached
      port: 11210
---
apiVersion: v1
kind: ReplicationController
metadata:
  name: couchbase-rc
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: couchbase-rc-pod
    spec:
      containers:
        - name: couchbase
image: arungupta/wildfly-couchbase-javaee:travel
ports:
  - containerPort: 8091
  - containerPort: 8092
  - containerPort: 8093
  - containerPort: 11210
---
apiVersion: batch/v1
kind: Job
metadata:
  name: web
  labels:
    name: web-pod
spec:
  template:
    metadata:
      name: web-pod
    spec:
      containers:
        - name: web-pod
          image: arungupta/couchbase:travel
          env:
            - name: COUCHBASE_URI
              value: couchbase-service
  restartPolicy: Never
```

Application Definition



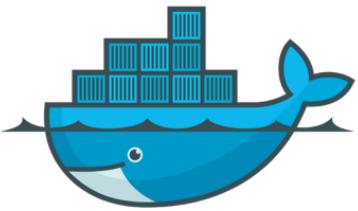
```
{  
  "id":"/webapp",  
  "apps": [  
    {  
      "id":"database",  
      "cpus":4,  
      "mem":4096,  
      "instances":1,  
      "container":{  
        "type": "DOCKER",  
        "docker":{  
          "image": "arungupta/couchbase:travel",  
          "network": "USER"  
        }  
      },  
      "ipAddress":{  
        "networkName": "dcos"  
      }  
    },  
    {  
      "id": "web",  
      "dependencies": [  
        "/webapp/database"  
      ],  
      "cpus":2,  
      "mem":4096,  
      "instances":1,  
      "container":{  
        "type": "DOCKER",  
        "docker":{  
          "image": "arungupta/wildfly-couchbase-  
javaee:travel",  
          "network": "USER",  
          "portMappings": [  
            {  
              "hostPort":0,  
              "containerPort":8080,  
              "protocol": "tcp"  
            }  
          ],  
          "ipAddress":{  
            "networkName": "dcos"  
          },  
          "env":{  
            "COUCHBASE_URI": "database-  
webapp.marathon.containerip.dcos.thisdcos.directory"  
          },  
          "labels":{  
            "HAPROXY_0_VHOST": "DCOS-PublicSlaveLo-  
LZ6PIP10I080-1004309391.us-west-1.elb.amazonaws.com",  
            "HAPROXY_GROUP": "external"  
          }  
        }  
      }  
    }  
  ]  
}
```

Application Definition



```
version: "3"
services:
  db:
    image: arungupta/couchbase:travel
    ports:
      - 8091:8091
      - 8092:8092
      - 8093:8093
      - 11210:11210
  web:
    image: arungupta/couchbase-wildfly-javaee:travel
    environment:
      - COUCHBASE_URI=db
    ports:
      - 8080:8080
      - 9990:9990
```

Production



- Docker Enterprise Edition
 - Certified infrastructure
 - Certified containers from 3rd-party ISVs
 - Certified plugins from networking and storage vendors
 - End-to-end security
- AWS, Azure, GCP
- Linux*
- Windows Server 2016



- Mesosphere Enterprise DC/OS
 - Advanced operational & troubleshooting
 - Multi-tenancy
 - Networking, storage, security
 - DC/OS Universe

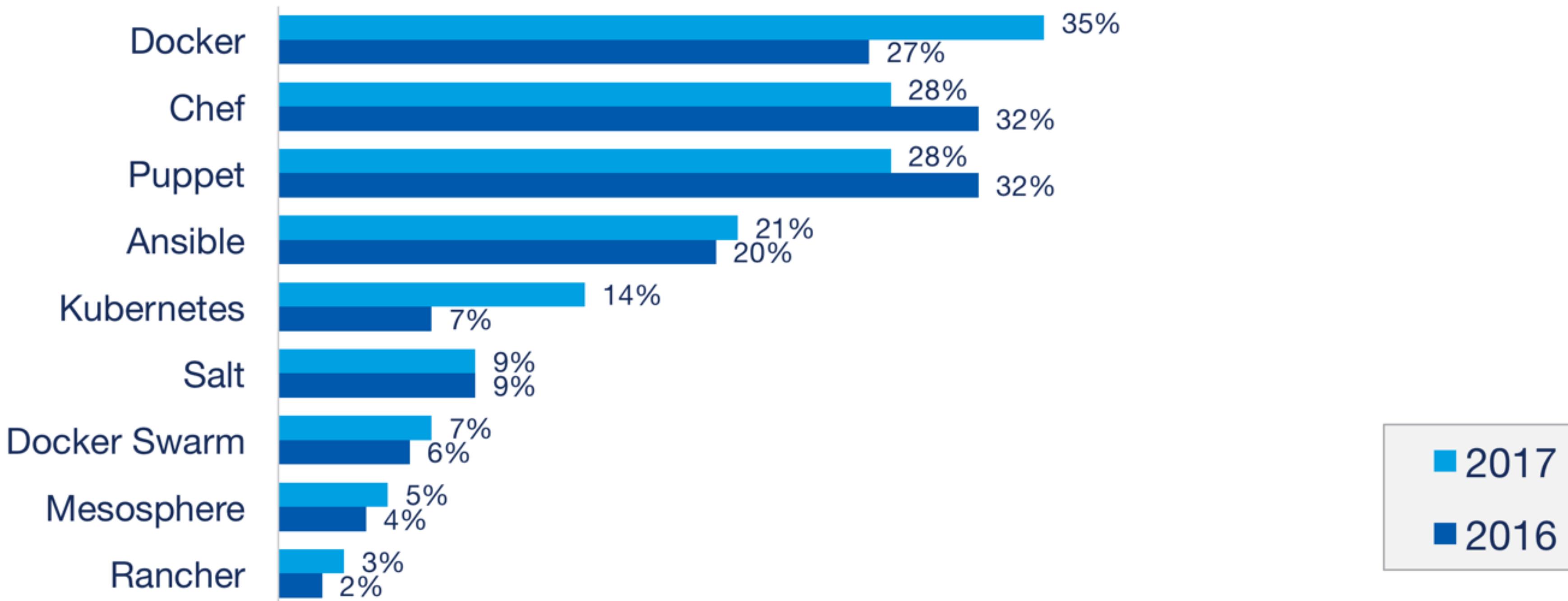


- AWS, Azure, GCP
- Red Hat OpenShift

- Amazon Web Services

DevOps

Respondents Using DevOps Tools



Source: RightScale 2017 State of the Cloud Report



References

- Docker: docs.docker.com
- Kubernetes: kubernetes.io
- DC/OS: dcos.io
- AWS ECS: aws.amazon.com/ecs
- Slides: github.com/arun-gupta/docker-java