



ShowIT
31/01/2018

Openshift for DevOps

Jiří Kolář
Solution Architect CZ/SK/CEE
jkolar@redhat.com

When all buzzwords come to a party ...



Devops!

Containers!

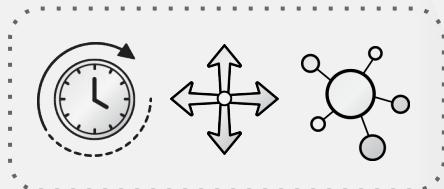
Microservices!

Modern App Dev?

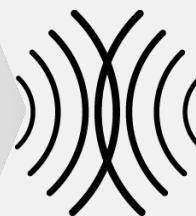


DevOps

PROBLEM:

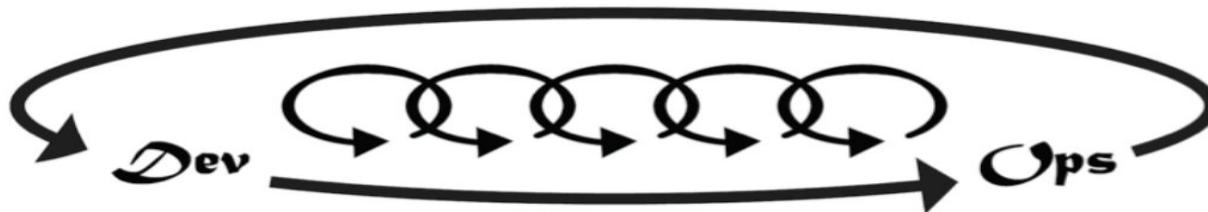


DEVELOPERS



I.T. OPERATIONS

SOLUTION:



Key concepts:

- Small changes -> Less Risk
- Delivery pipeline = **Automation!**
- Culture change: Acceptance of failure
- Team takes **ownership and responsibility!**

Containers

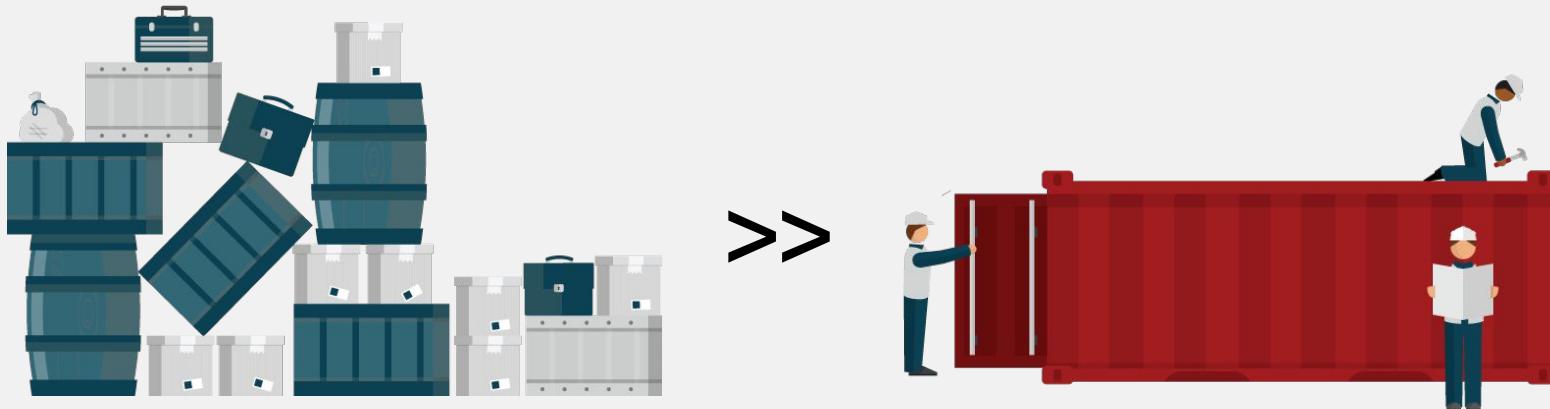
PROBLEM:



>>

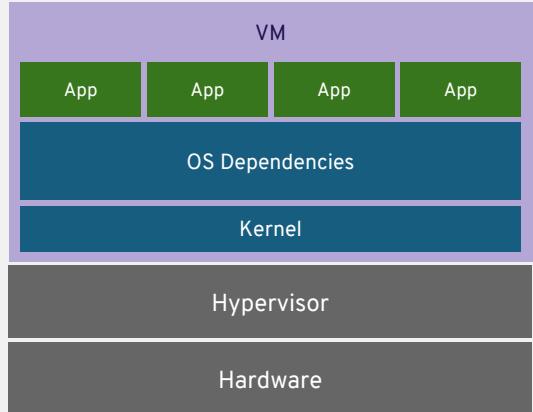
??

SOLUTION:

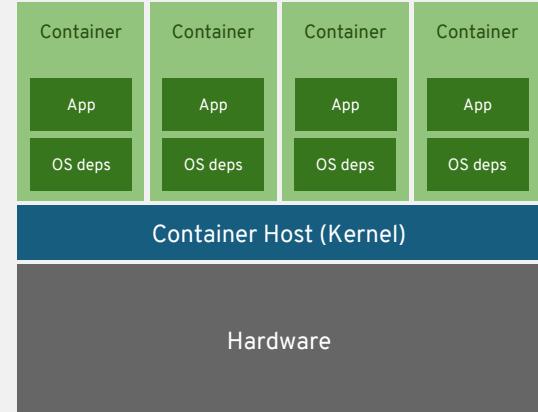


Containers!

VIRTUAL MACHINES



CONTAINERS



virtual machines are isolated
apps are not

containers are isolated
so are the apps



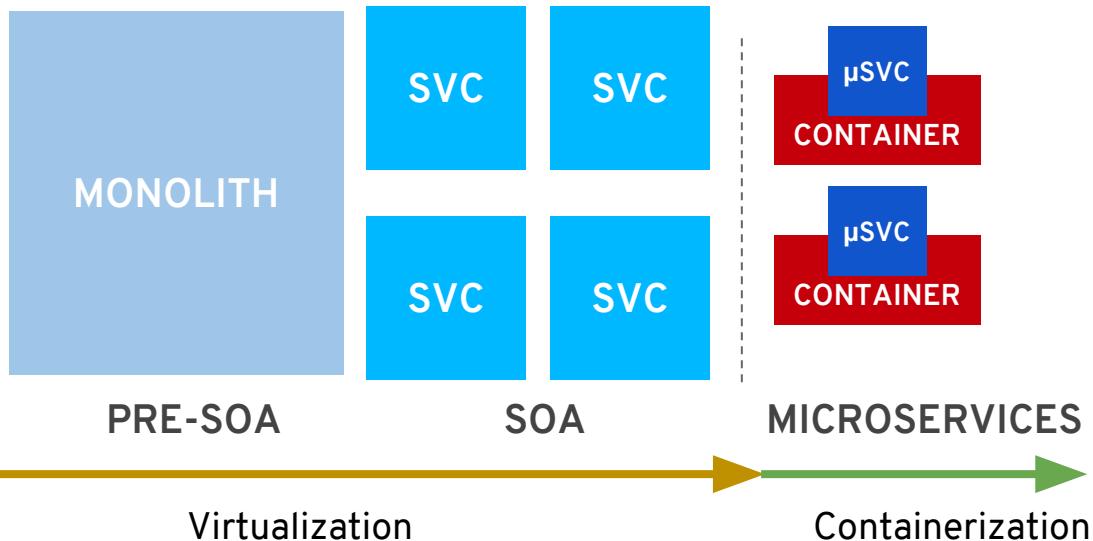
>>



Microservices

ARCHITECT FOR SMALLER APPLICATION COMPONENTS

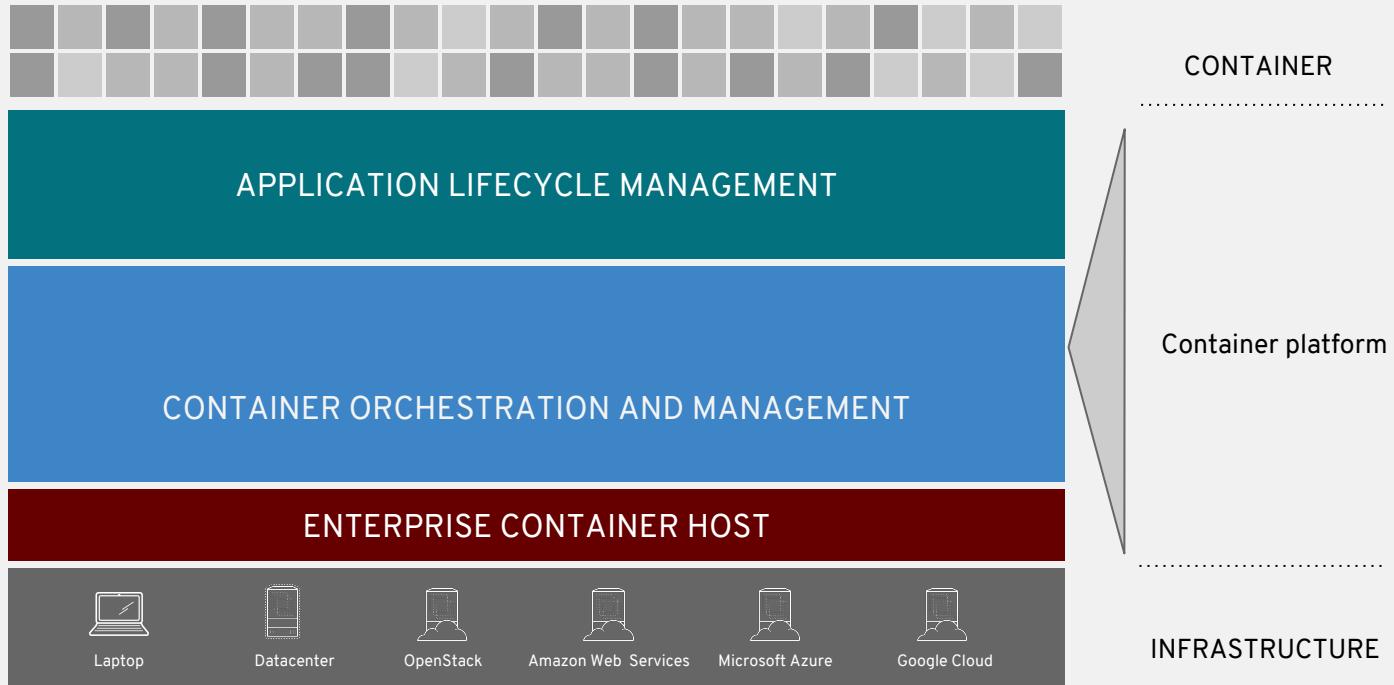
([microservices primer](#))



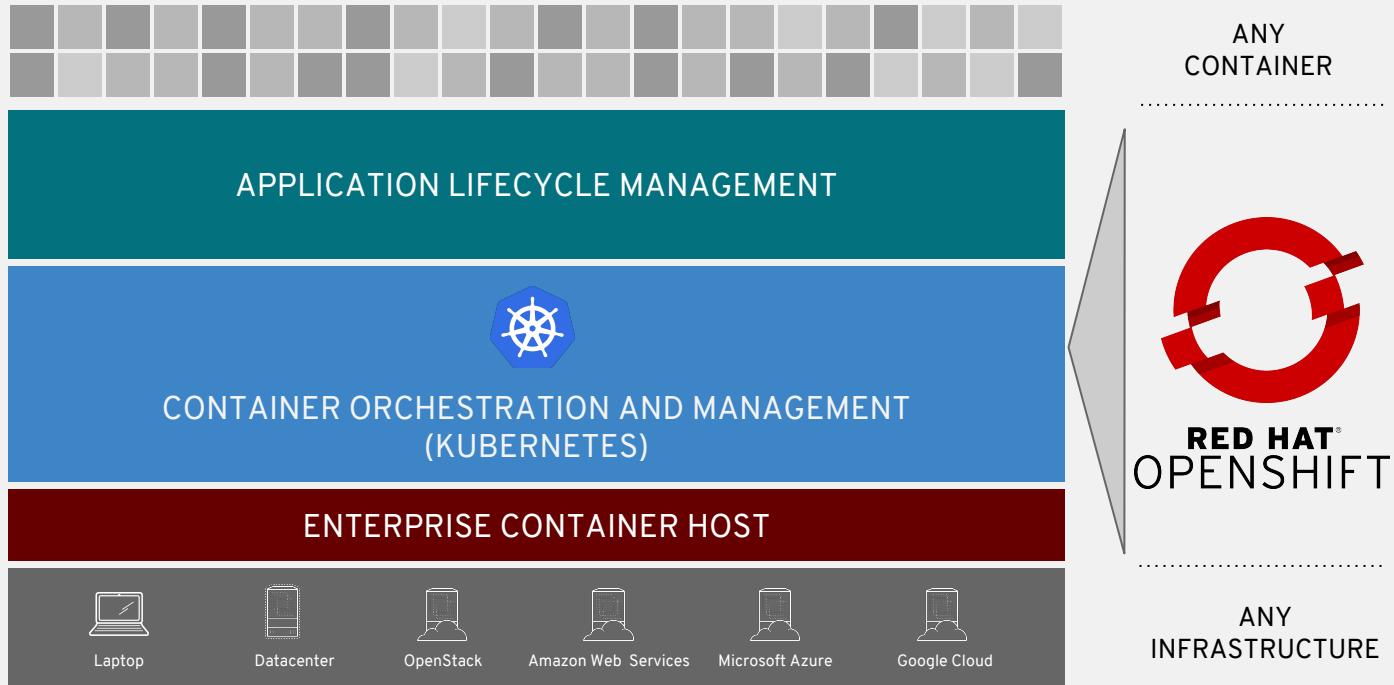
There is good and bad

- Good
 - Agile, DevOps
 - Polyglot
 - New Architectures
- Bad
 - Complexity
 - Dependencies
 - Consistency
 - Dealing with data

CONTAINER PLATFORM



OPENShift CONTAINER PLATFORM





CONTAINERS or PODS ?

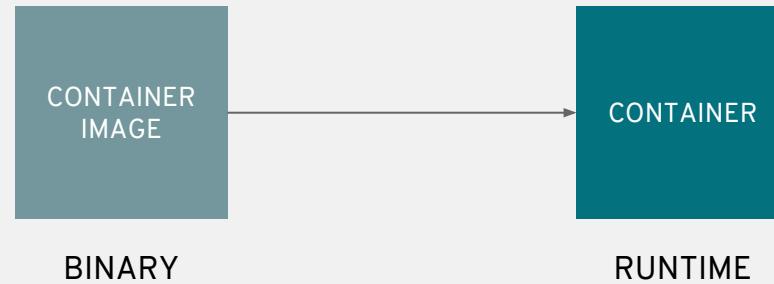


A container is the smallest compute unit

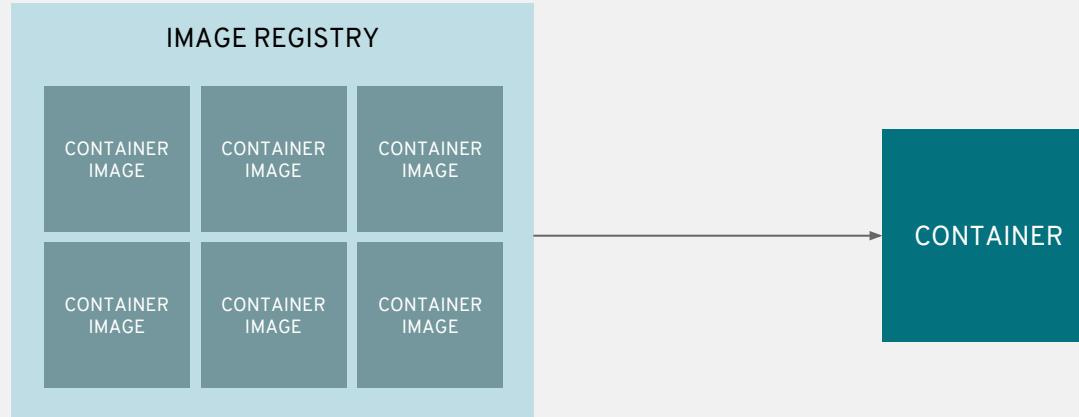




containers are created from container images



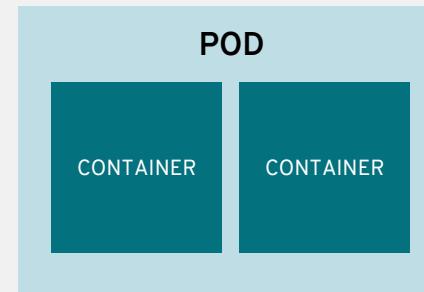
container images are stored in an image registry



containers are wrapped in pods which are units of deployment and management



IP: 10.1.0.11



IP: 10.1.0.55

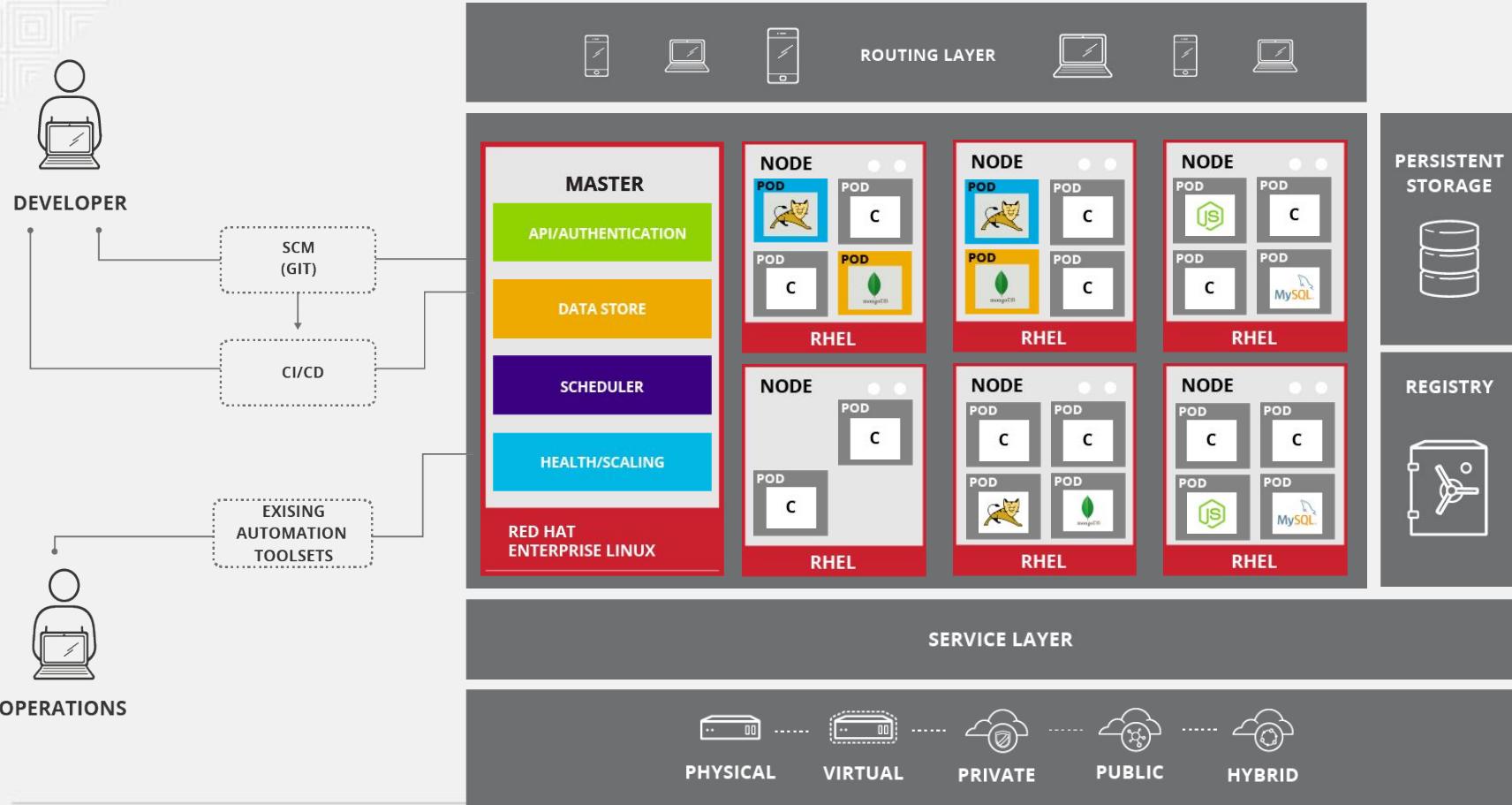
A photograph of a large stack of shipping containers in a port terminal. The containers are stacked high, filling the frame. The perspective is from a low angle looking up at the top of the stack. The sky is clear and blue.

OpenShift

How it works



Openshift architecture



Source 2 Image Walk Through

Code



git



DEV

Build

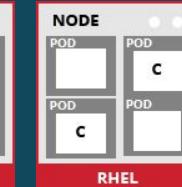
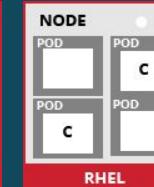
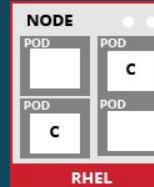


Container
Image



Registry

Deploy

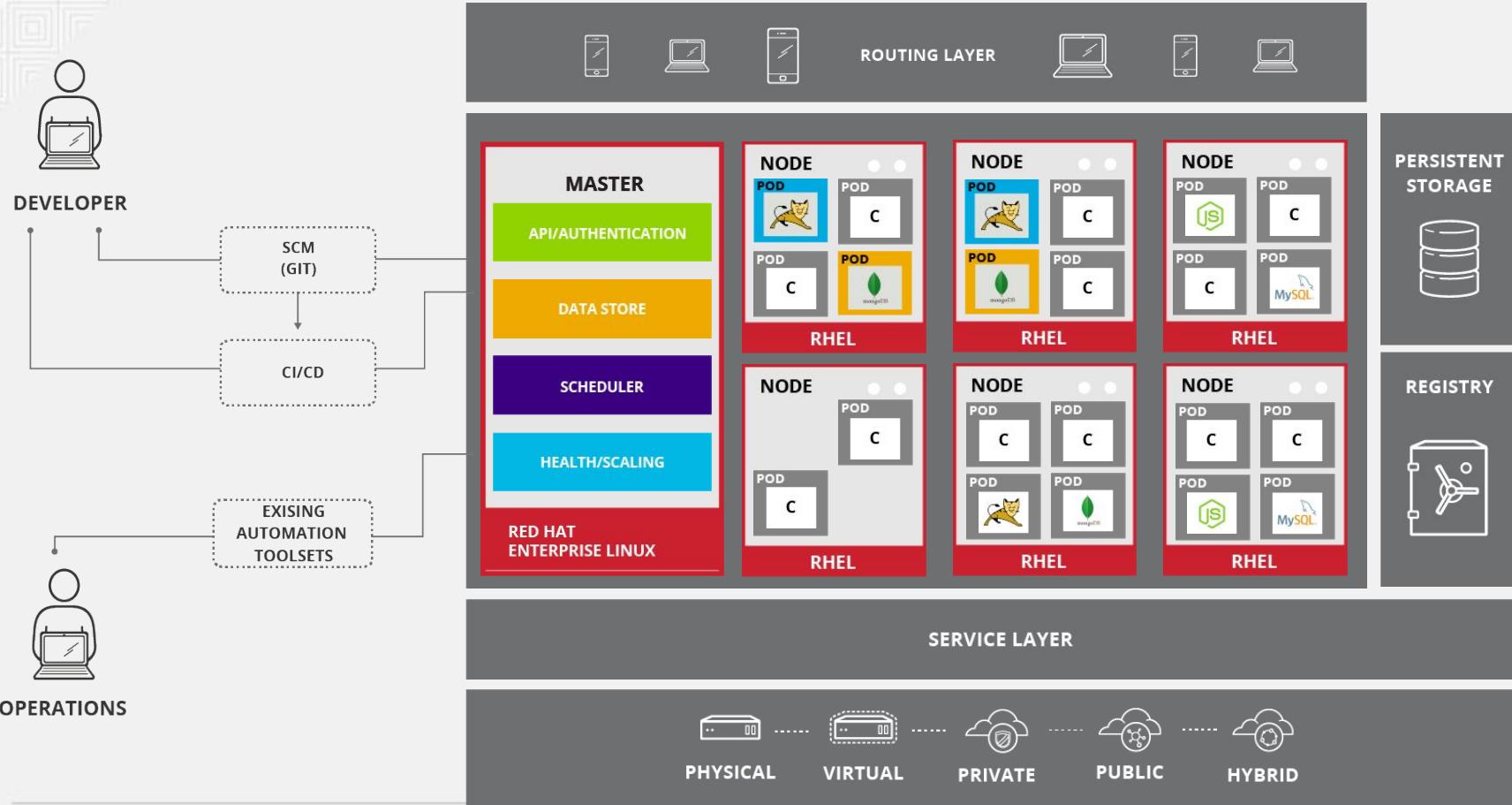


OPS

TRUE POLYGLOT PLATFORM

LANGUAGES	Java	NodeJS	Python	PHP	Perl	Ruby	.NET Core	Third-party Language Runtimes	
DATABASES	MySQL	PostgreSQL	MongoDB	Redis	<p>...and virtually any docker image out there!</p>				CrunchyData
WEB SERVERS	Apache HTTP Server	nginx	Varnish	Phusion Passenger					GitLab
MIDDLEWARE	Spring Boot	Wildfly Swarm	Vert.x	JBoss Web Server	JBoss EAP	JBoss A-MQ	JBoss Fuse	Iron.io	Couchbase
	3SCALE API mgmt	JBoss BRMS	JBoss BPMS	JBoss Data Virt	JBoss Data Grid	RH Mobile	RH SSO	Sonatype	EnterpriseDB
								Third-party Middleware	NuoDB
								Third-party Middleware	Fujitsu
									and many more

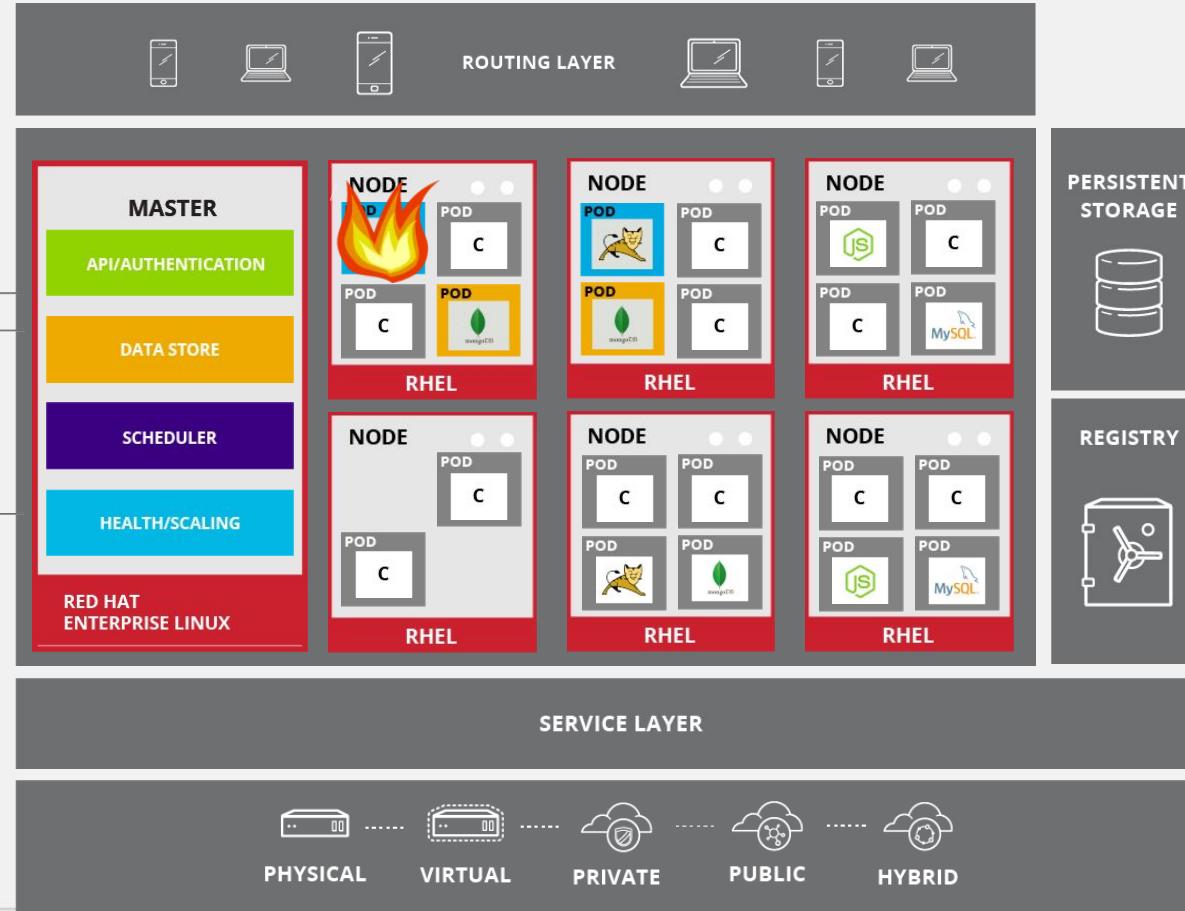
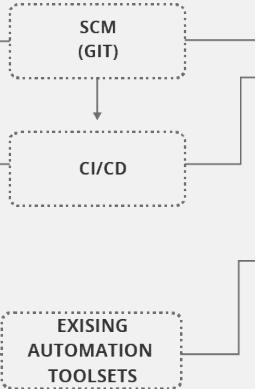
Openshift architecture



What if ..



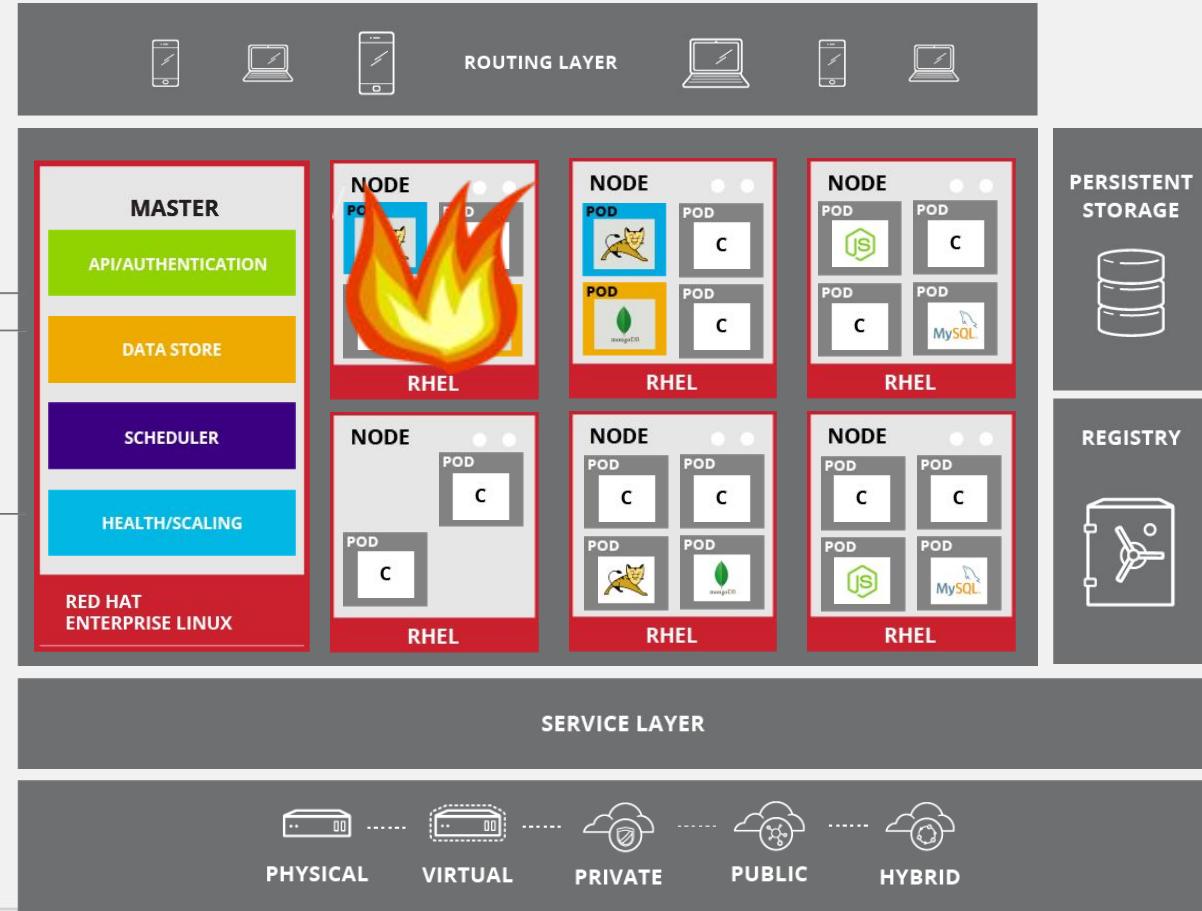
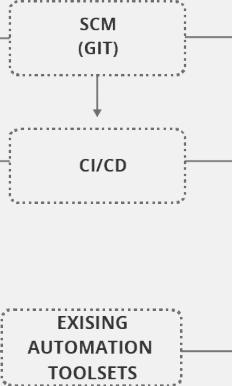
DEVELOPER



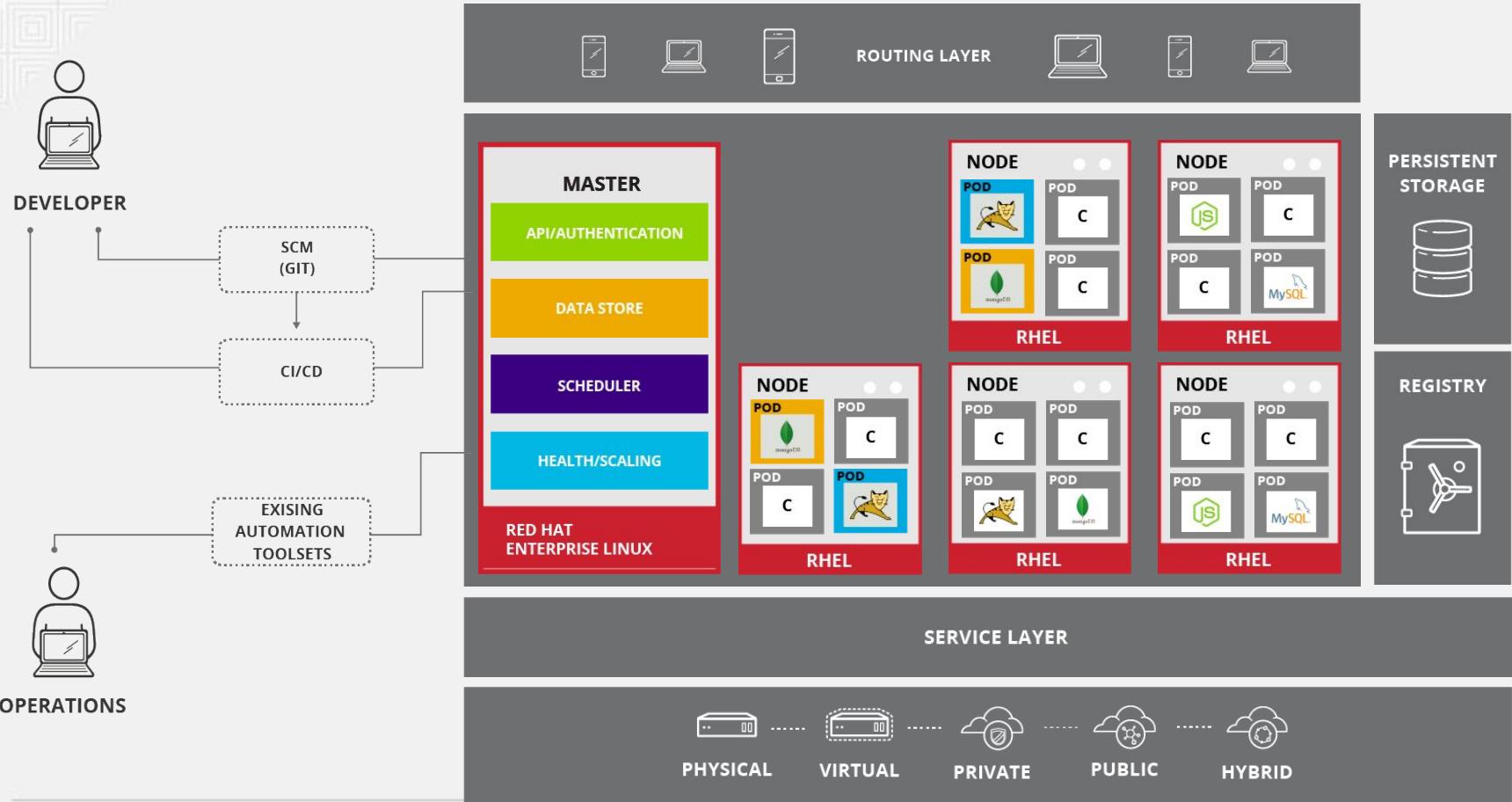
What if ...



DEVELOPER



Self-healing AKA. Contant HA



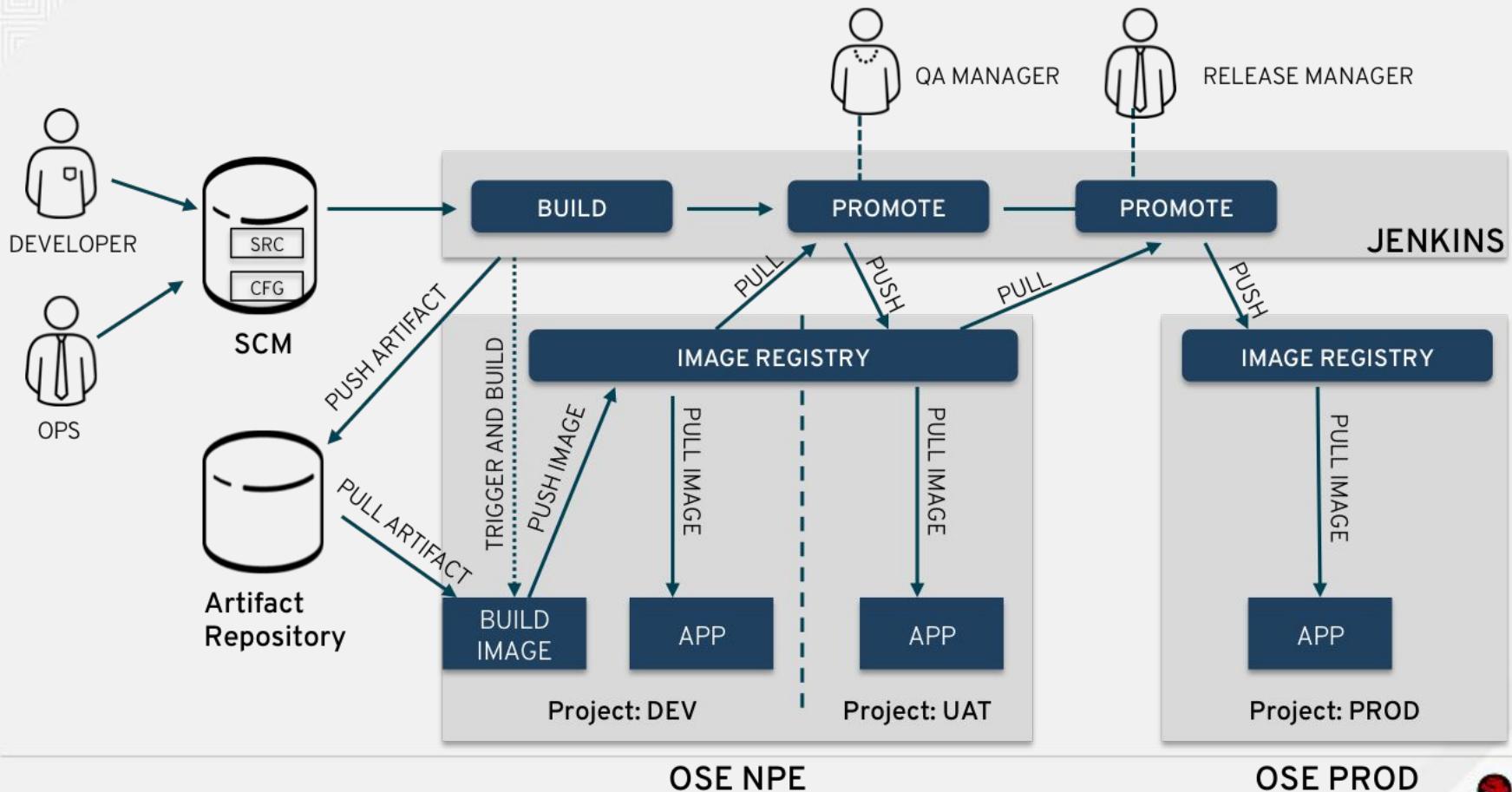
A photograph of a large stack of shipping containers in a port terminal. The containers are stacked high, filling the frame. They are primarily white and yellow, with some green ones interspersed. The perspective is from a low angle, looking up at the towering stacks.

OpenShift

Full CI/CD platform



OpenShift CI/CD flow



OPENShift PIPELINES

- OpenShift Pipelines allow defining a CI/CD workflow via a Jenkins pipeline which can be started, monitored, and managed similar to other builds
- Dynamic provisioning of Jenkins slaves
- Auto-provisioning of Jenkins server
- OpenShift Pipeline strategies
 - Embedded Jenkinsfile
 - Jenkinsfile from a Git repository

```
apiVersion: v1
kind: BuildConfig
metadata:
  name: app-pipeline
spec:
  strategy:
    type: JenkinsPipeline
    jenkinsPipelineStrategy:
      jenkinsfile: |->
        node('maven') { <----->
          stage('build app') {
            git url: 'https://git/app.git'
            sh "mvn package"
          }
          stage('build image') {
            sh "oc start-build app --from-file=target/app.jar"
          }
          stage('deploy') {
            openshiftDeploy deploymentConfig: 'app'
          }
        }
```

Provision a Jenkins slave for running Maven

OpenShift Pipelines in Web Console

app-pipeline created 32 minutes ago

[Start Build](#) [Actions](#)

[Summary](#) Configuration

✓ Latest build #11 complete. [View Log](#)
started 16 minutes ago

A bar chart comparing the duration of recent builds. The y-axis represents Duration in seconds, ranging from 20s to 2m 20s. The x-axis lists build numbers #2 through #11. Builds #2 and #3 are red, indicating failure, while builds #8, #9, #10, and #11 are blue, indicating completion. The average duration is 1m 55s.

Build Number	Status	Duration
#2	Failed	2m 20s
#3	Failed	2m 0s
#8	Complete	1m 40s
#9	Complete	1m 20s
#10	Complete	1m 0s
#11	Complete	40s

— Average: 1m 55s

The timeline view shows the stages of two completed builds: Build #11 and Build #10. Each build consists of three stages: build app, build image, and deploy. The stages are represented by green horizontal bars with checkmarks at the start. The times for each stage are listed below the bars: 25s for build app, 16s for build image, and 45s for deploy for Build #11; and 26s for build app, 16s for build image, and 47s for deploy for Build #10.

Build	Stage	Time
Build #11	build app	25s
	build image	16s
	deploy	45s
Build #10	build app	26s
	build image	16s
	deploy	47s



OpenShift

What is inside?



Trusted Container OS



Enterprise Container Host

Container Runtime & Packaging
(Docker)

Atomic Host

Red Hat Enterprise Linux

Trusted by Fortune Global
500 companies



Enterprise Kubernetes



Container Orchestration & Cluster Management
(kubernetes)

Networking Storage Registry Logs & Metrics Security

Infrastructure Automation & Mg



Enterprise Container Host

Container Runtime & Packaging
(Docker)

Atomic Host

Red Hat Enterprise Linux



kubernetes
Cloudforms
Red Hat Storage



Enterprise Container Platform



Self-Service

Service Catalog
(Language Runtimes, Middleware, Databases)

Build Automation Deployment Automation

OpenShift Application Lifecycle Management
(CI/CD)



Container Orchestration & Cluster Management
(kubernetes)

Networking Storage Registry Logs & Metrics Security

Infrastructure Automation & Cockpit



Enterprise Container Host

Container Runtime & Packaging
(Docker)

Atomic Host Red Hat Enterprise Linux

**Source-2-Image
Application Pipelines
Dev Tools**

Traditional, Stateful, and Microservices-based Apps

Business Automation

Integration

Data & Storage

Web & Mobile

Container

Container

Container

Container



Self-Service

Service Catalog
(Language Runtimes, Middleware, Databases)

Build Automation Deployment Automation

OpenShift Application Lifecycle Management
(CI/CD)



Container Orchestration & Cluster Management
(kubernetes)

Networking Storage Registry Logs & Metrics Security

Infrastructure Automation & Cockpit



Enterprise Container Host

Container Runtime & Packaging
(Docker)

Atomic Host

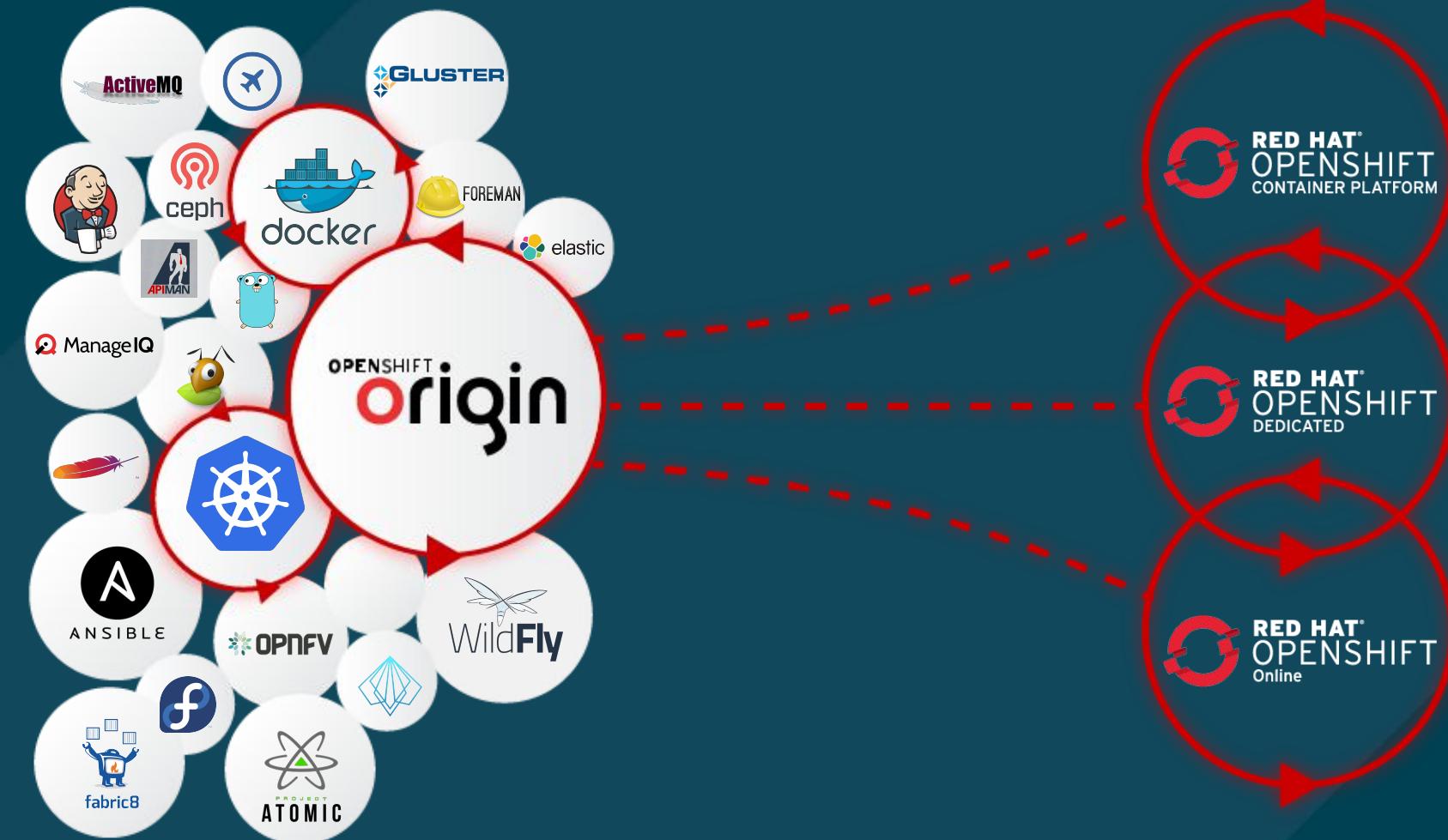
Red Hat Enterprise Linux

JBOSS EAP
JBOSS DATA GRID
JBOSS DATA VIRTUALIZATION
JBOSS AM-Q
JBOSS BRMS
JBOSS BPM
JBOSS FUSE
RED HAT MOBILE
3 Scale



redhat.[®]

Community Powered Innovation





Red Hat Training & Certification



Classroom



Virtual
training



Online
training



On-site
training

DO080 Deploying Containerized Applications Technical Overview (FREE!)

DO092 Developing Cloud-Native Applications with Microservices Architectures (FREE!)

DO180 Introduction to Containers, Kubernetes, and Red Hat OpenShift

DO280 Red Hat OpenShift Administration I

DO285 Containers, Kubernetes, and Red Hat OpenShift Administration I

DO380 Red Hat OpenShift Administration II: High Availability

DO290 Developing and Deploying Applications on OpenShift

Knowledge is the power. Training is the key!

Self-Service



Multi-language



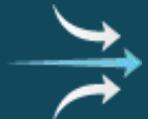
Automation



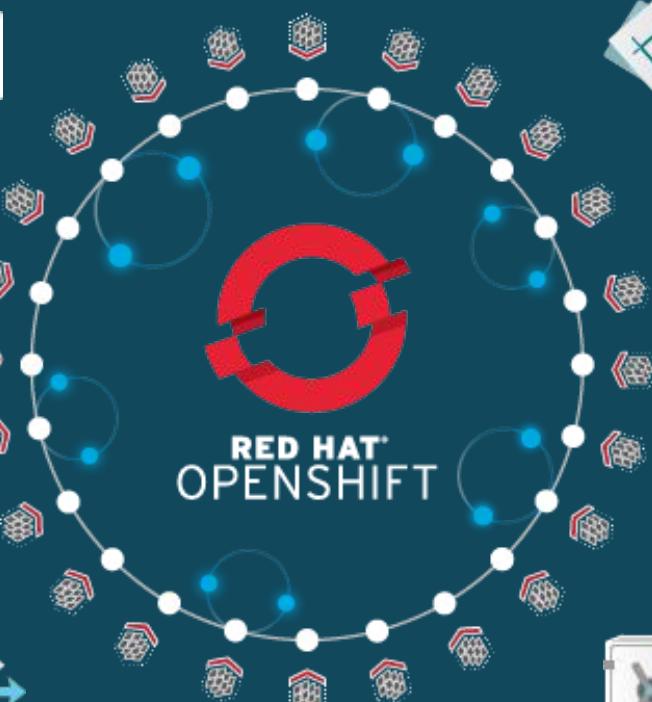
Collaboration



Seamless



RED HAT[®]
OPENSHIFT



Standards-based



Web-scale



Open Source



Enterprise Grade

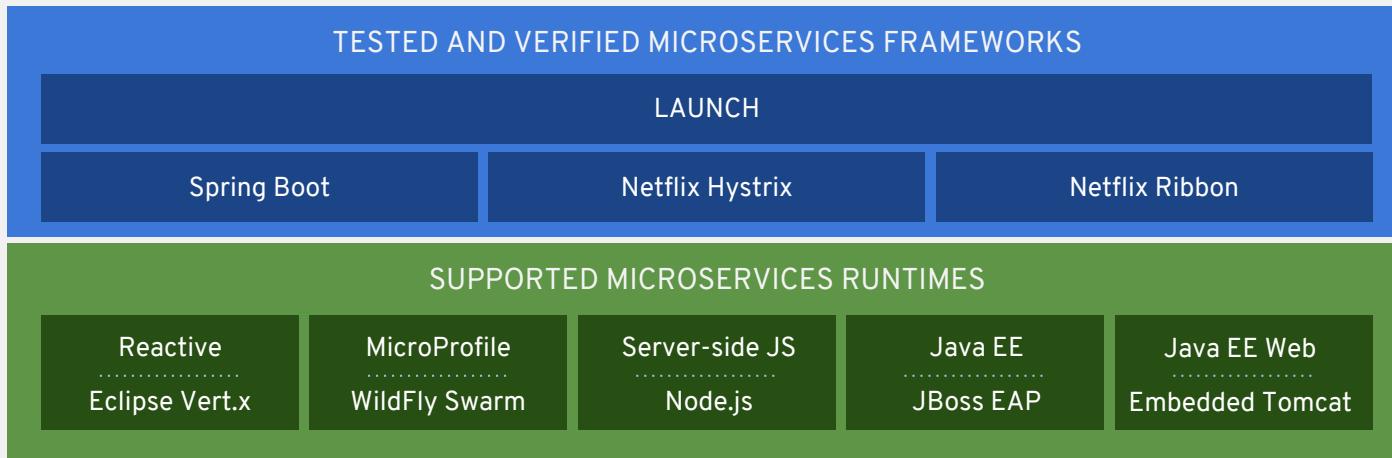


Secure





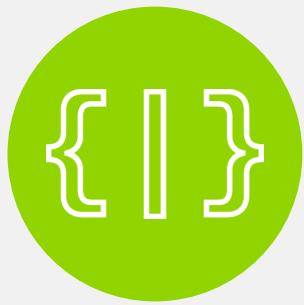
redhat.[®]



Modern, Cloud-Native Application Runtimes and
an Opinionated Developer Experience

DEPLOYMENT OPTIONS?

BUILD AND DEPLOY CONTAINER IMAGES



DEPLOY YOUR
SOURCE CODE

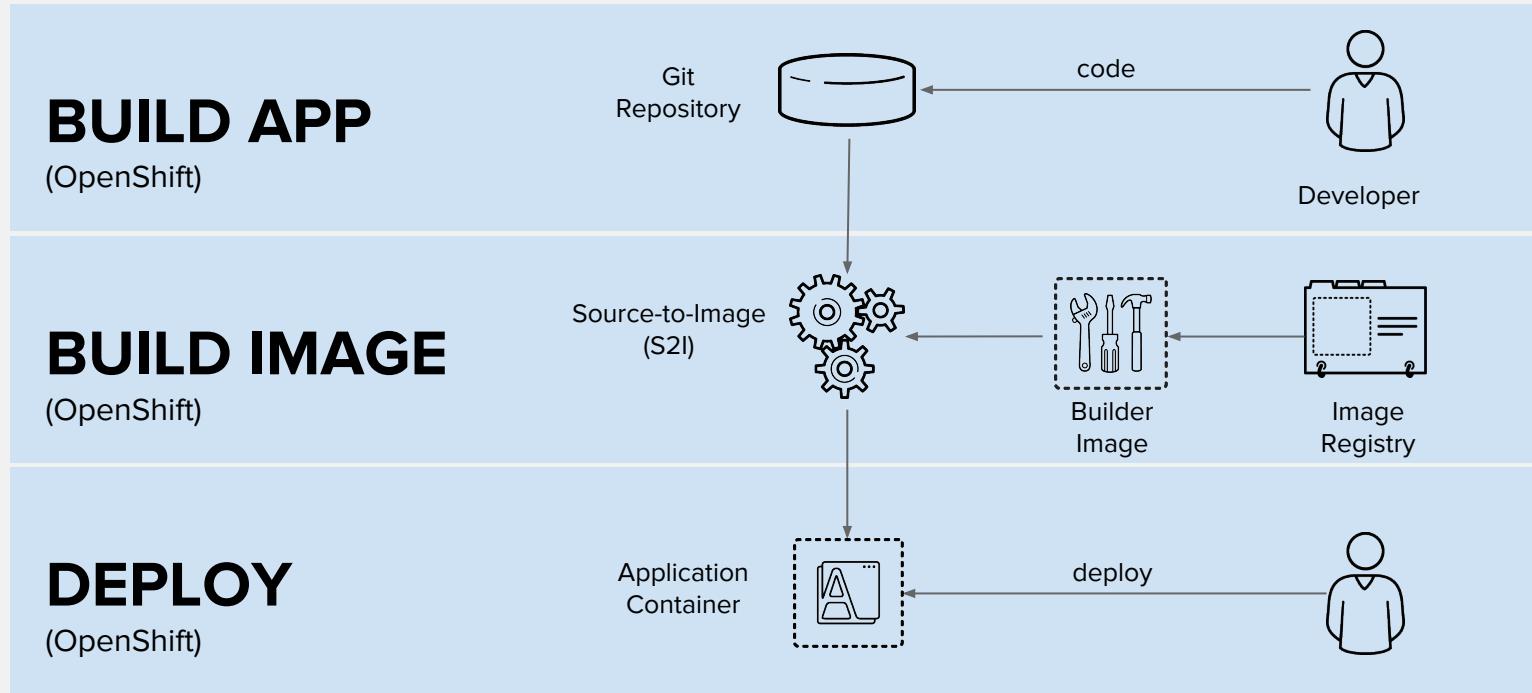


DEPLOY YOUR
APP BINARY

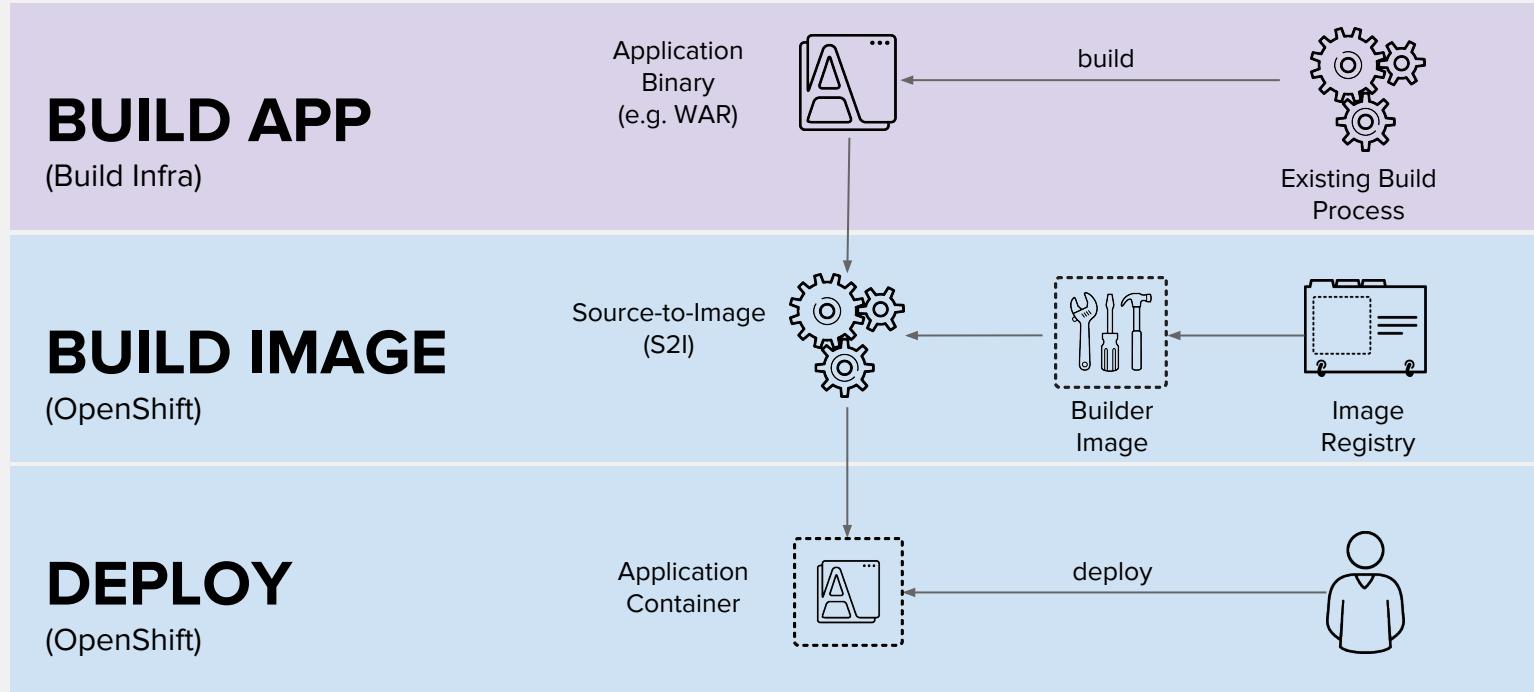


DEPLOY YOUR
CONTAINER IMAGE

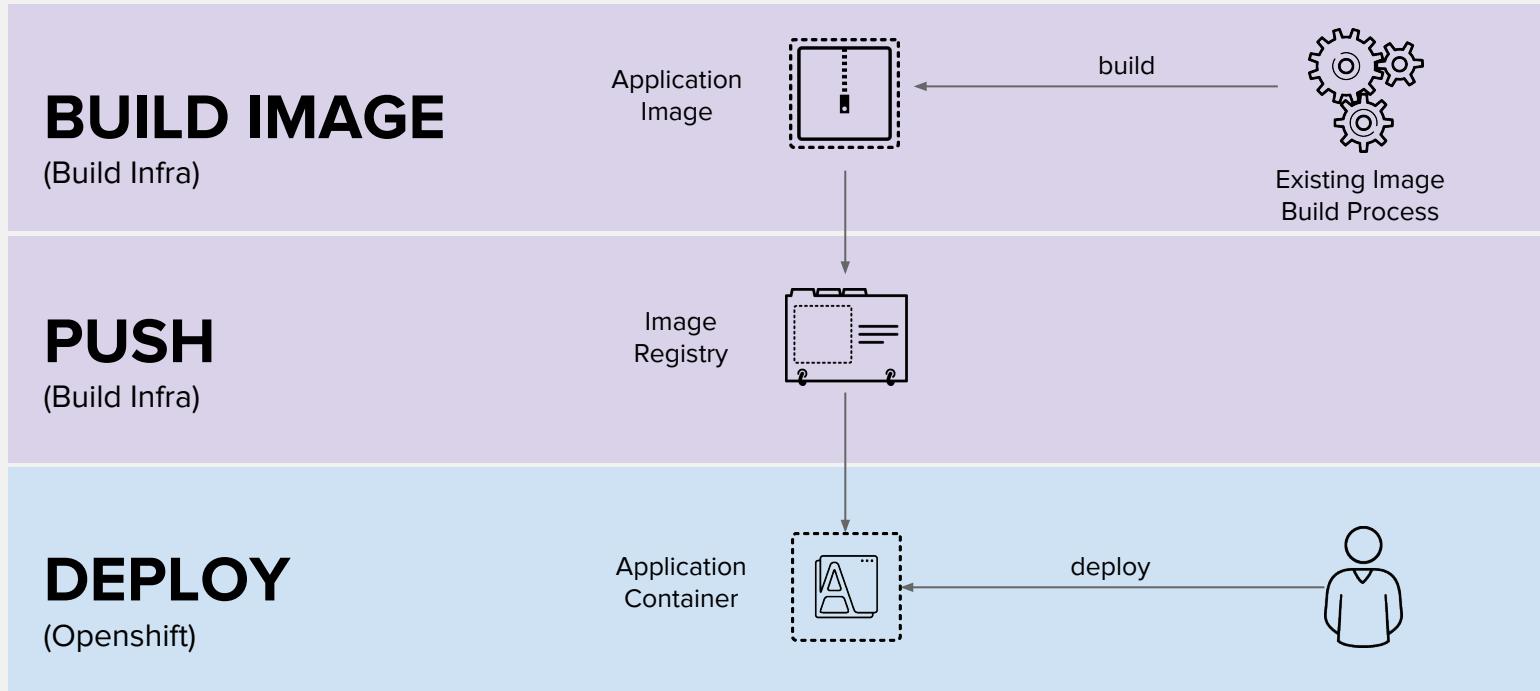
DEPLOY SOURCE CODE WITH SOURCE-TO-IMAGE (S2I)



DEPLOY APP BINARY WITH SOURCE-TO-IMAGE (S2I)

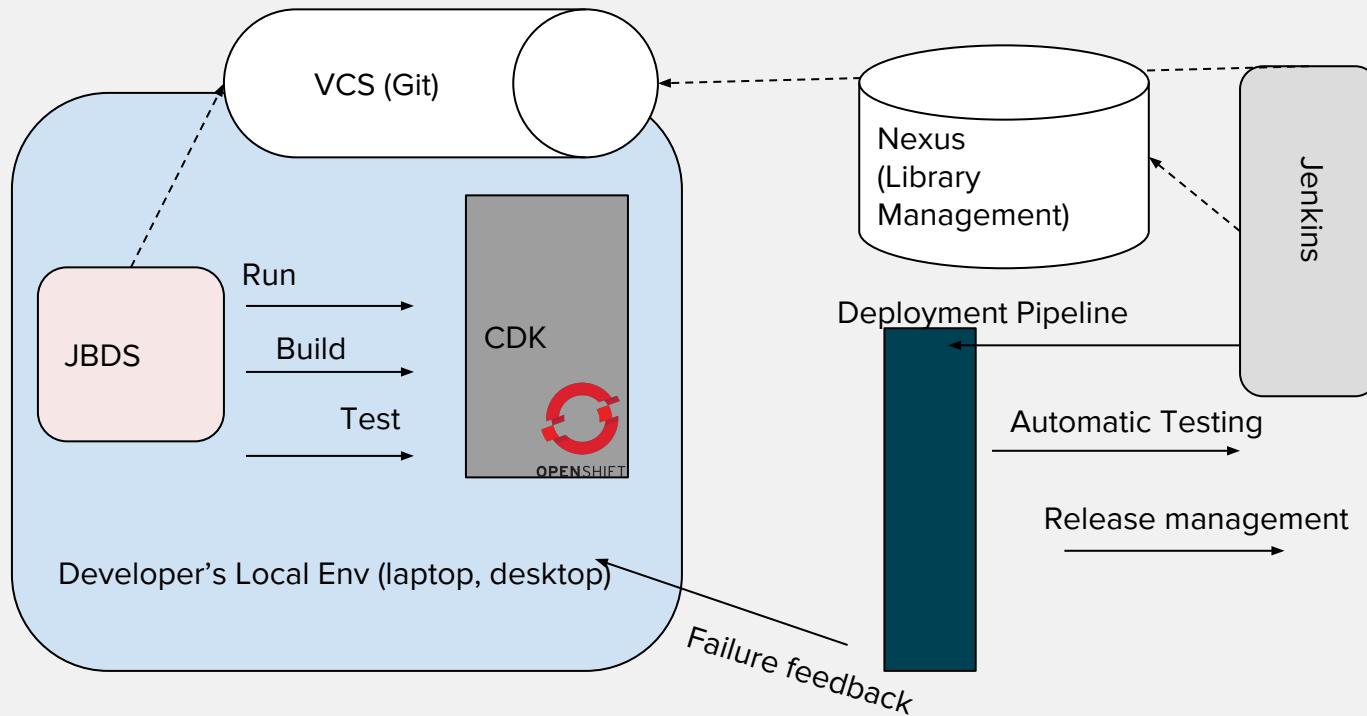


DEPLOY DOCKER IMAGE



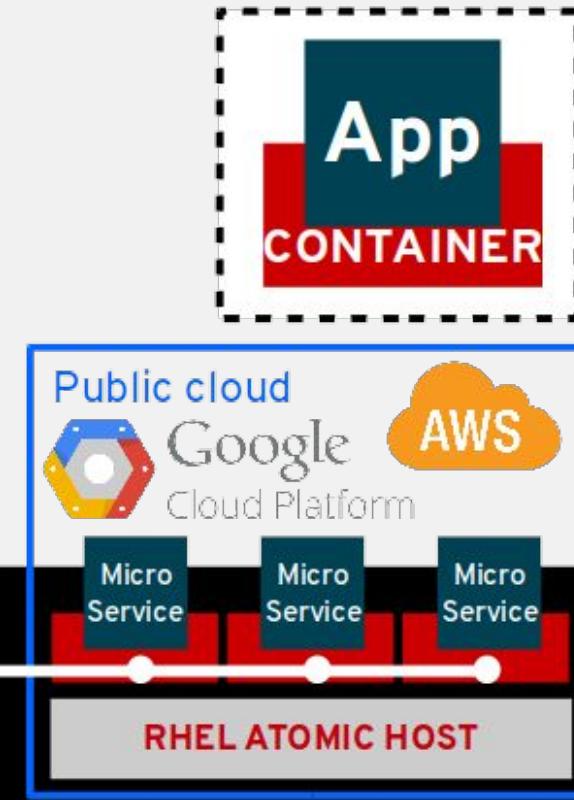
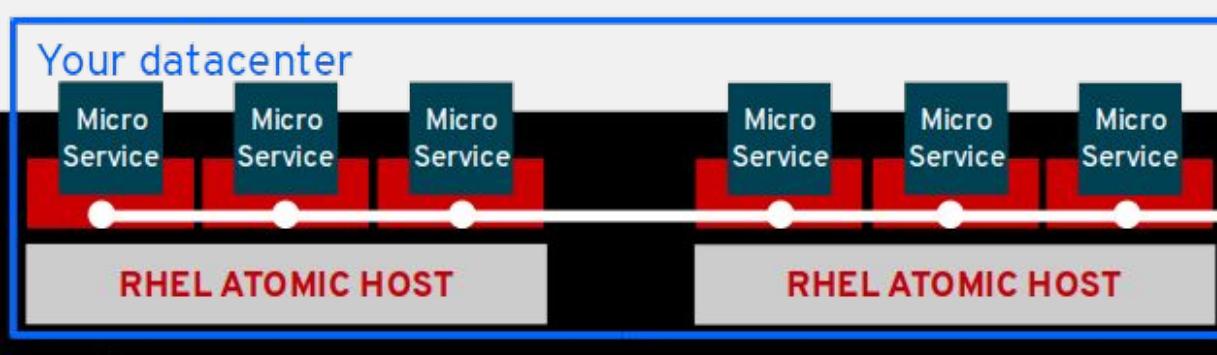
**PIPELINE AFTER EVERY COMMIT??
DEBUGGING??**

LOCAL DEVELOPMENT WITH CONTAINERS



Openshift summary

- Automates whole container lifecycle
- On Premise / In cloud / Both (aka. Hybrid)
- Containers orchestration (Kubernetes)
- Microservices O-o-the-box
- CI/CD automation, Dev Ops
- Scalability & HA O-o-the-box





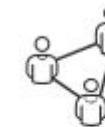
redhat.[®]



Red Hat Training & Certification



Classroom



Virtual
training



Online
training



On-site
training

DO080 Deploying Containerized Applications Technical Overview (FREE!)

DO092 Developing Cloud-Native Applications with Microservices Architectures (FREE!)

DO180 Introduction to Containers, Kubernetes, and Red Hat OpenShift

DO280 Red Hat OpenShift Administration I

DO285 Containers, Kubernetes, and Red Hat OpenShift Administration I

DO380 Red Hat OpenShift Administration II: High Availability

DO290 Developing and Deploying Applications on OpenShift

Knowledge is the power. Training is the key!

Self-Service



Multi-language



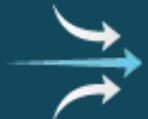
Automation



Collaboration



Seamless



RED HAT[®]
OPENSHIFT



Standards-based



Web-scale



Open Source



Enterprise Grade



Secure





redhat.[®]

The background image shows the interior of a large industrial ship's cargo hold. Numerous shipping containers are stacked in several layers. The containers are primarily white and green. The ship's metal structure, including beams, ladders, and walkways, is visible throughout the frame.

OpenShift

Customer References



Evolving Application Architecture at Volvo

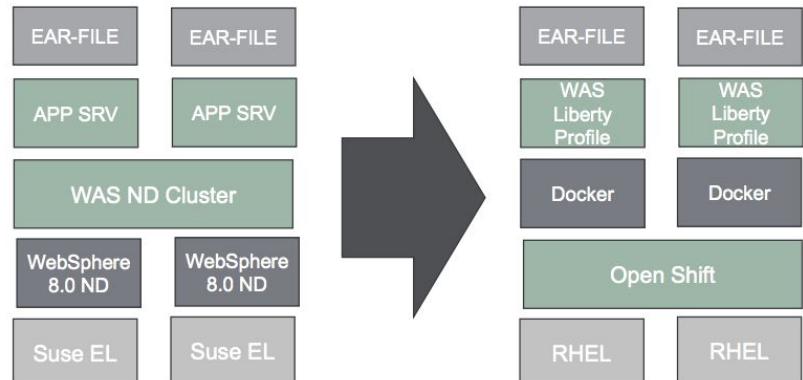
785 apps across 560 app servers

OpenShift provides build, distribution & runtime environment

Platform for DevOps and Microservices

Running OpenShift on Azure,
automatically provisioned with
Ansible

OUR NEW ENVIRONMENT



DEUTSCHE BANK - Technology Transformation



“Delivering Everything as a Service. From 20% adoption to 40% in 1yr; planning to move 85% of all applications to OpenShift platform. We won the hearts and minds of developers.” -- Pat Healy, CTO, Deutsche Bank

- Macro trends are radically reshaping the banking industry
- Need to regain software expertise that was previously outsourced
- OpenShift replaced an internal, homegrown PaaS platform
- Over 300 internal projects moved to OpenShift
- 6x better efficiency of computing resources using containers and OpenShift. Driving overall utilization up via multi-tenancy.
- Leverage OpenShift across multiple public clouds.

Ideas to Production, safely in a day.

View the [Deutsche Bank keynote](#)

BMW GROUP - Evolving the Connected Car



Digital customer experience, connected and automated driving and digitalized business processes lead to a transformation of the BMW Group towards software and services (Tech).



- Global manufacturer of luxury automobiles, motorcycles and engines. “The Ultimate Driving Machine”.
- Evolving in-vehicle communications and telematics for 15yrs.
- OpenShift platform enables BMW ConnectedDrive service.
- Enables Electric Cars, Service Calls, Real-Time Traffic, Driving Assistance, Anonymous Cars.
- Moving from Monolithic applications to Microservices, Containers and Kubernetes



Read the [press release](#) and view Red Hat Summit [presentation](#).



redhat.[®]