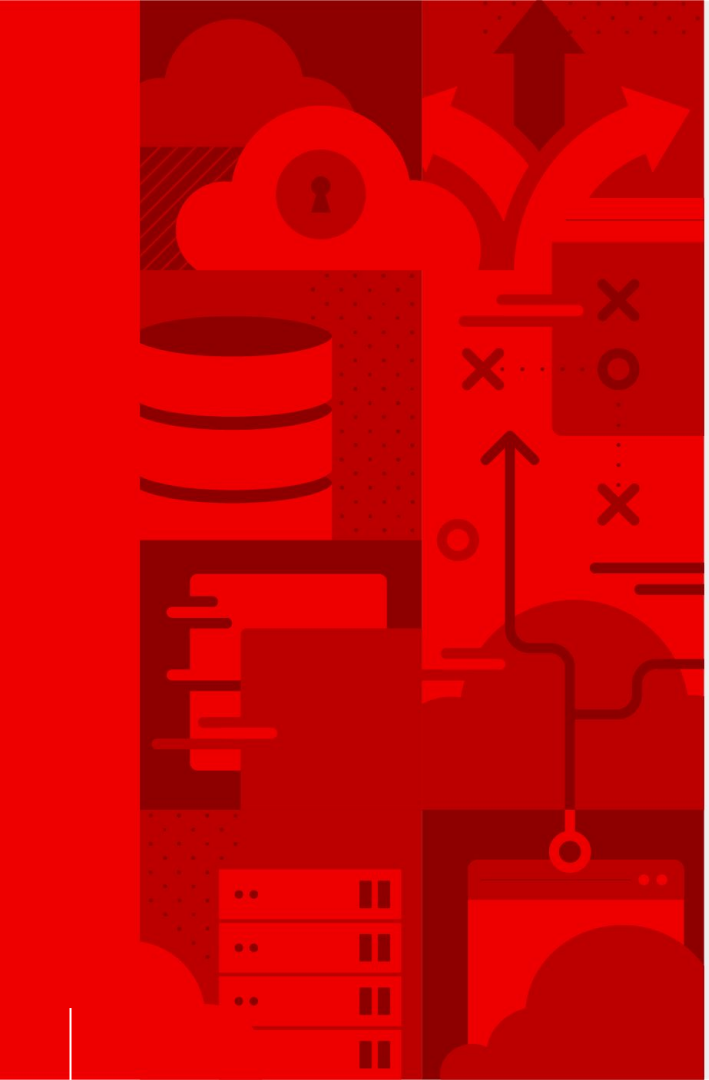


OpenShift Developer

Architecture Workshop

Introduction to OpenShift for Developers

Usecase description & Quarkus demo-app

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat

Self introduction

Name: Wanja Pernath

Email: wpernath@redhat.com

Base: Germany (very close to the Alps)

Role: EMEA Technical Partner Development Manager - OpenShift and MW

Experience: Years of Consulting, Training, PreSales at Red Hat and before

Agenda

Agenda

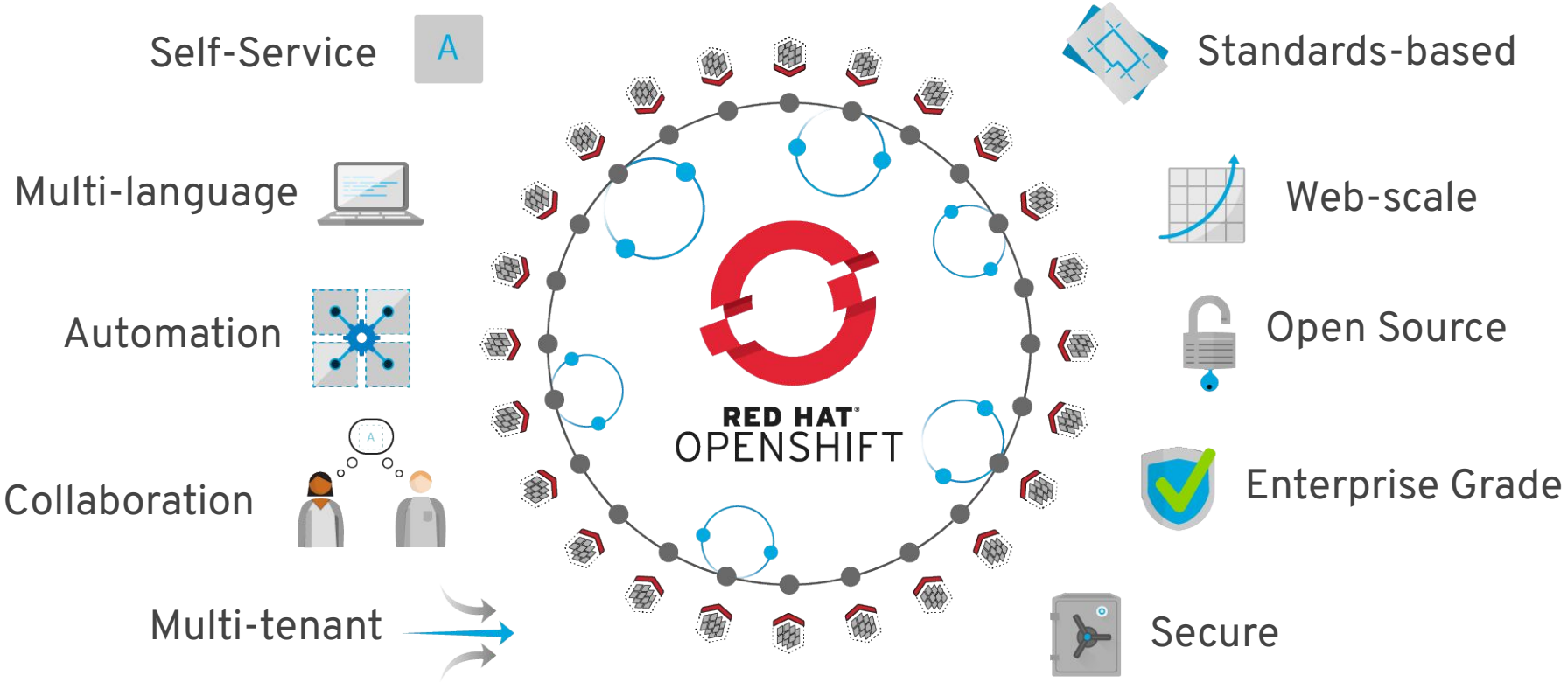
- What is this all about
- The OpenShift Ecosystem for Developers
  - Basics
  - Advanced topics
- How to demo OpenShift for Developers
- Quarkus Business Value
- Quarkus Technical Value

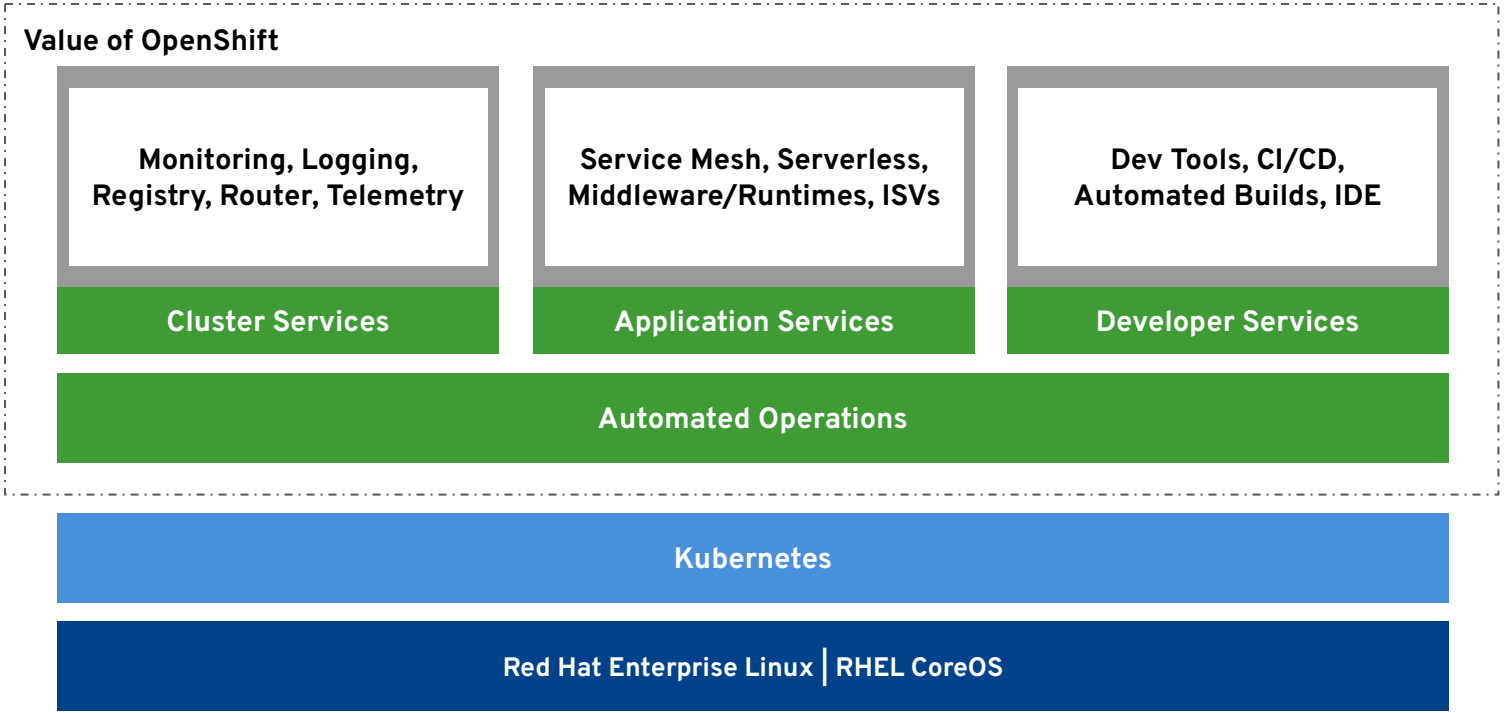
What is this all about?

Learning Goals

- Getting an overview of the whole OpenShift ecosystem for developers
- Learn about the benefits of OpenShift prior plain Kubernetes
- Learn how to effectively and efficiently demo OpenShift for developers
- Learn to use OpenShift and its ecosystem
- Learn to start coding with OpenShift in your preferred language
- Learn to do proper release management with OpenShift; understand the basics and how to include it into your release management process but also learn how to benefit from OpenShift Pipelines and when to use what
- Understanding and using Operators for releases
- Understand and make use of advanced OpenShift features like Istio and Serverless

OpenShift Developers Basics

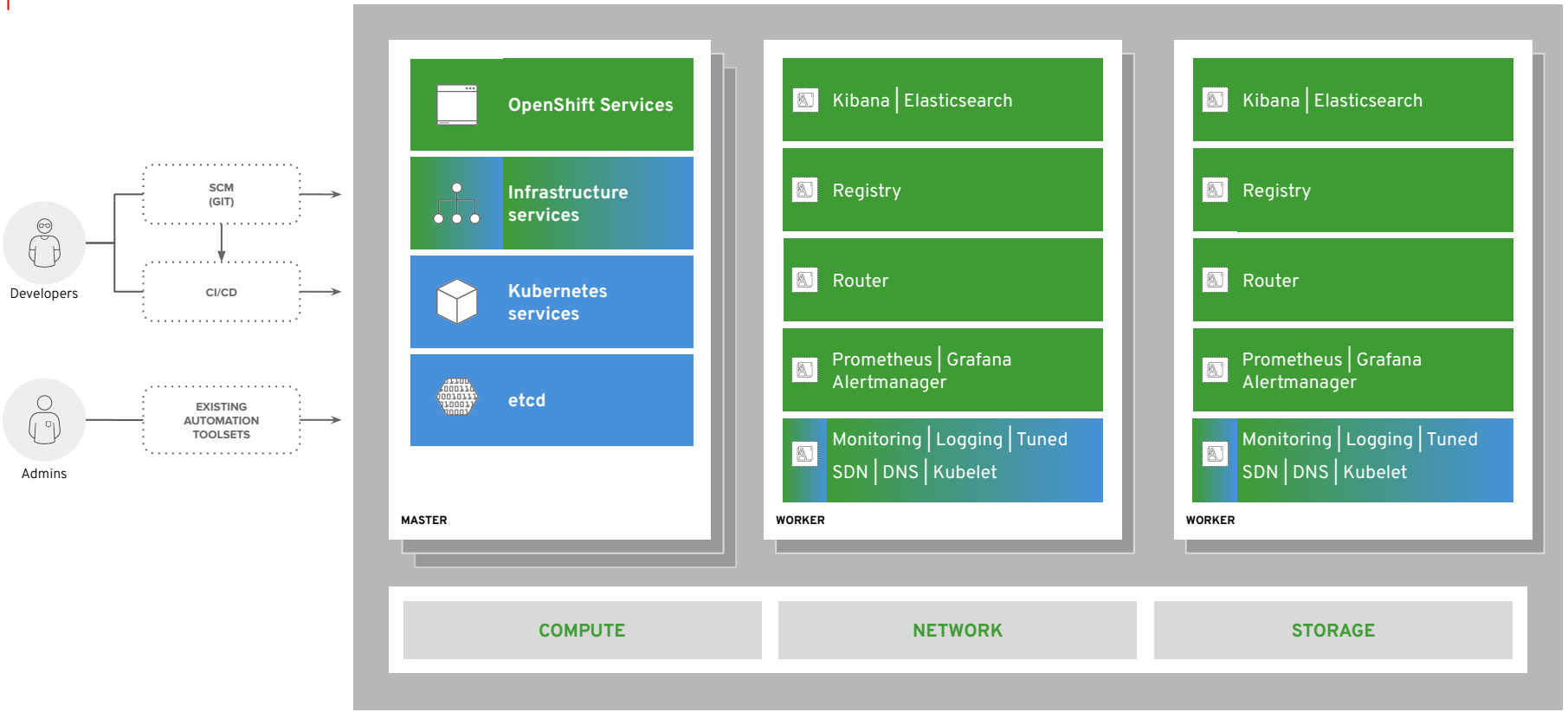

Self-Service
Multi-language
Automation
Collaboration
Multi-tenant
RED HAT
OPENSHIFT
Standards-based
Web-scale
Open Source
Enterprise Grade
Secure



Best IT Ops Experience

CaaS ⟷ PaaS ⟷ FaaS

Best Developer Experience


SCM (GIT)
CI/CD
Developers
EXISTING AUTOMATION TOOLSETS
Admins
OpenShift Services
Infrastructure services
Kubernetes services
etcd
MASTER
Kibana | Elasticsearch
Registry
Router
Prometheus | Grafana Alertmanager
Monitoring | Logging | Tuned
SDN | DNS | Kubelet
WORKER
Kibana | Elasticsearch
Registry
Router
Prometheus | Grafana Alertmanager
Monitoring | Logging | Tuned
SDN | DNS | Kubelet
WORKER
COMPUTE
NETWORK
STORAGE

Red Hat

CNCF Cloud Native Landscape
2019-07-12T03:51:02Z 024e93b

Overwhelmed? Please see the CNCF Trail Map. That and the interactive landscape are at l.cncf.io

Greyed logos are not open source

App Definition and Development

Database

Streaming & Messaging

Application Definition & Image Build

Continuous Integration & Delivery

Orchestration & Management

Scheduling & Orchestration

Coordination & Service Discovery

Remote Procedure Call

Service Proxy

API Gateway

Service Mesh

Runtime

Cloud-Native Storage

Container Runtime

Cloud-Native Network

Provisioning

Automation & Configuration

Container Registry

Security & Compliance

Key Management

Cloud

Public

Special

Kubernetes Certified Service Provider

Platform

Certified Kubernetes - Distribution

Certified Kubernetes - Hosted

Certified Kubernetes - Installer

PaaS/Container Service

Kubernetes Training Partner

Observability and Analysis

Monitoring

Logging

Tracing

Chaos Engineering

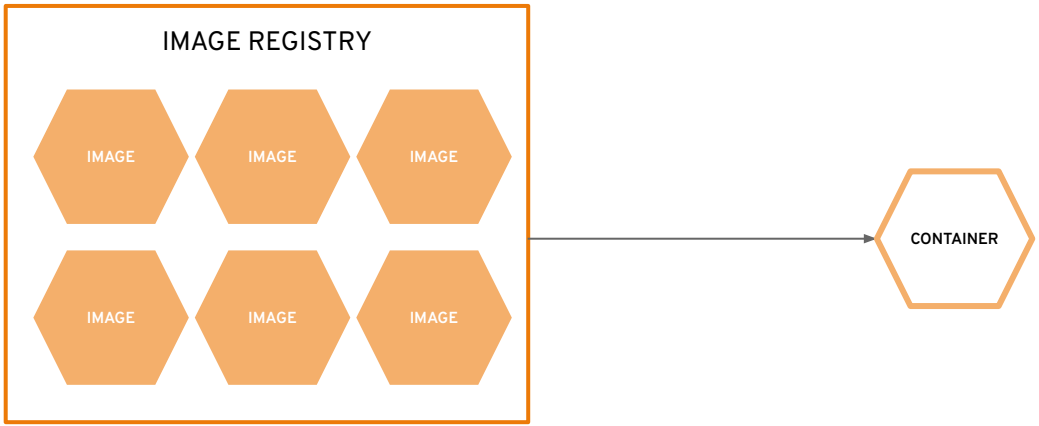
Serverless

Core Concepts

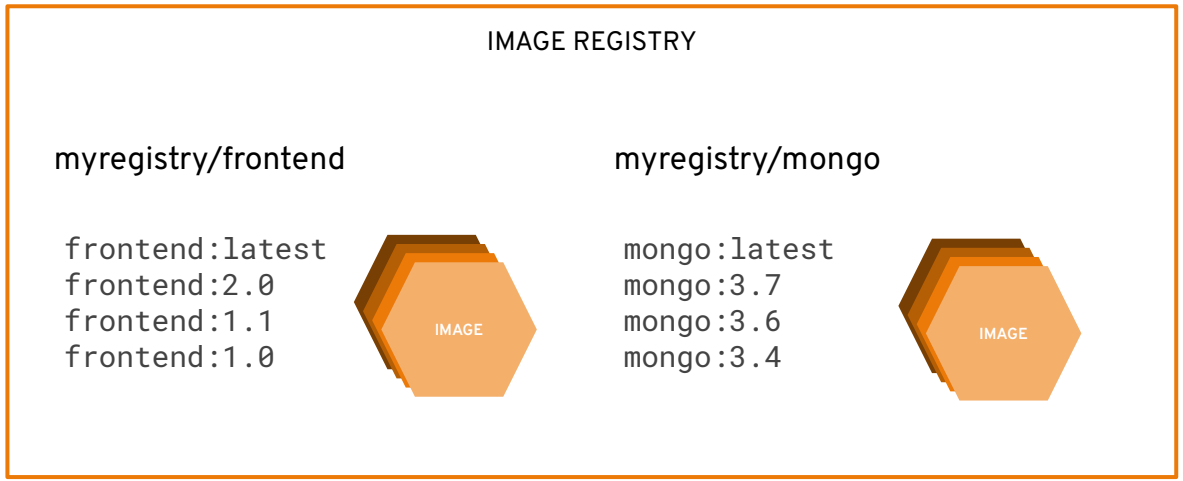
a container is the smallest compute unit

containers are created from container images

container images are stored in an image registry

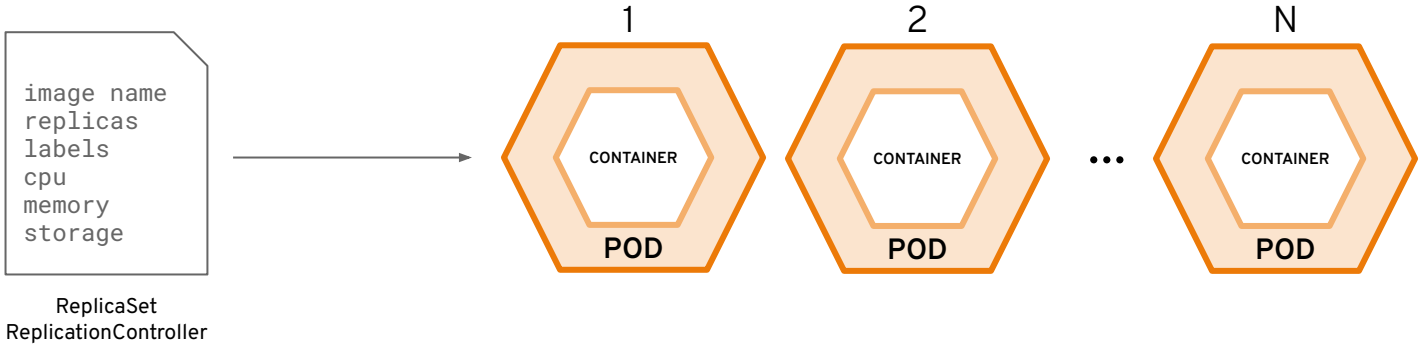


an image repository contains all versions of an image in the image registry

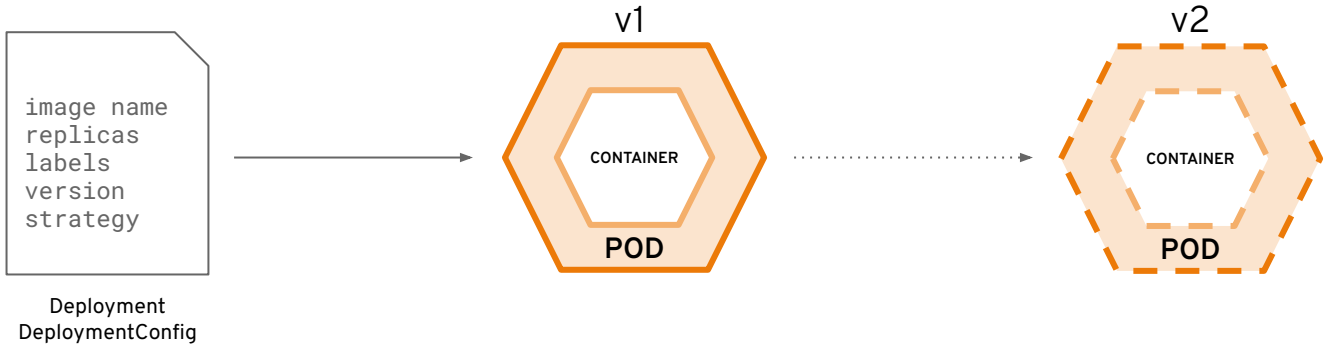


containers are wrapped in pods which are units of deployment and management

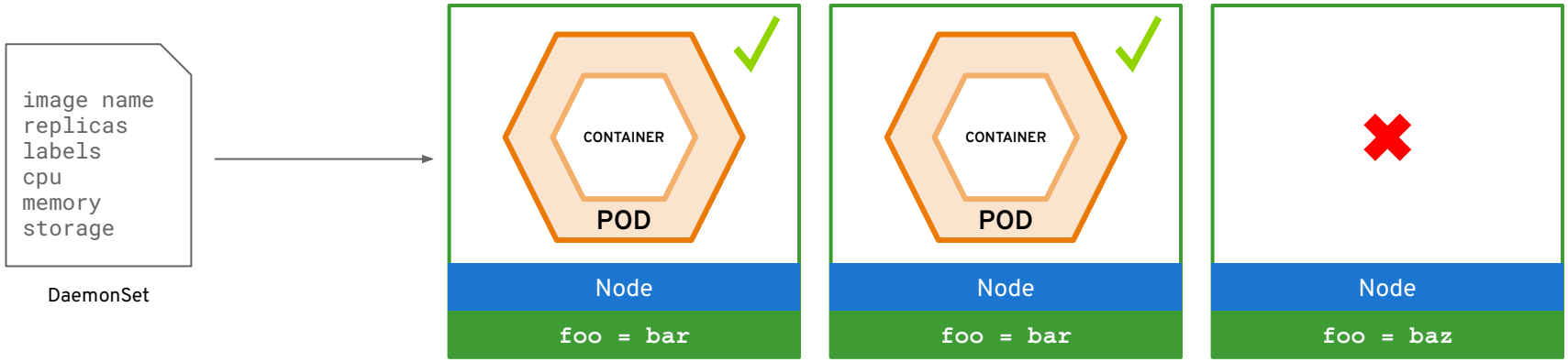
`ReplicationControllers` & `ReplicaSets` ensure a specified number of pods are running at any given time



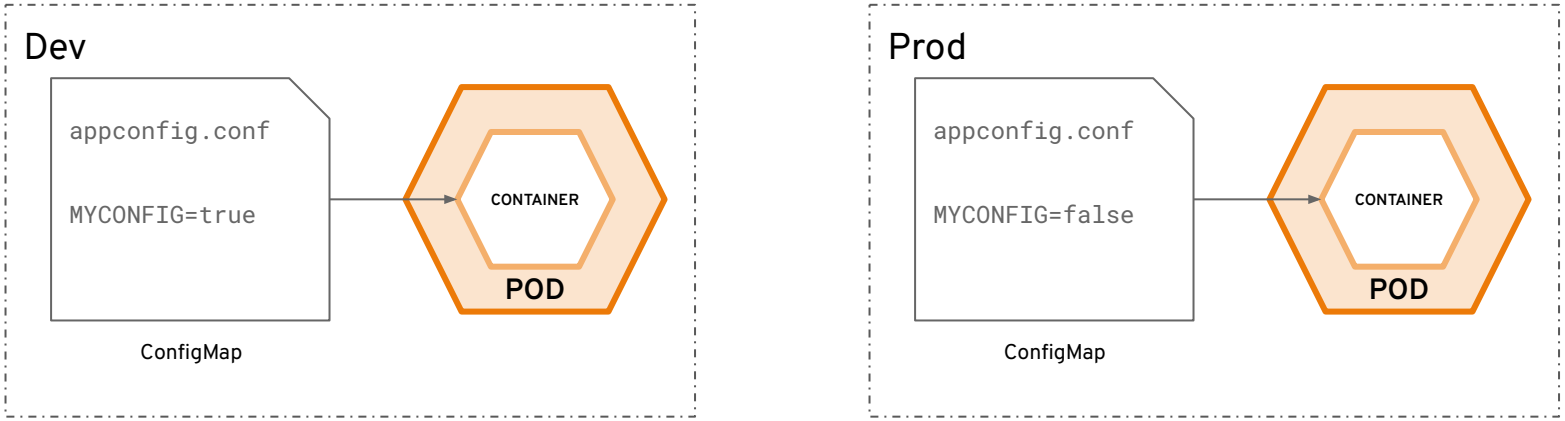
`Deployments` and `DeploymentConfigurations` define how to roll out new versions of Pods



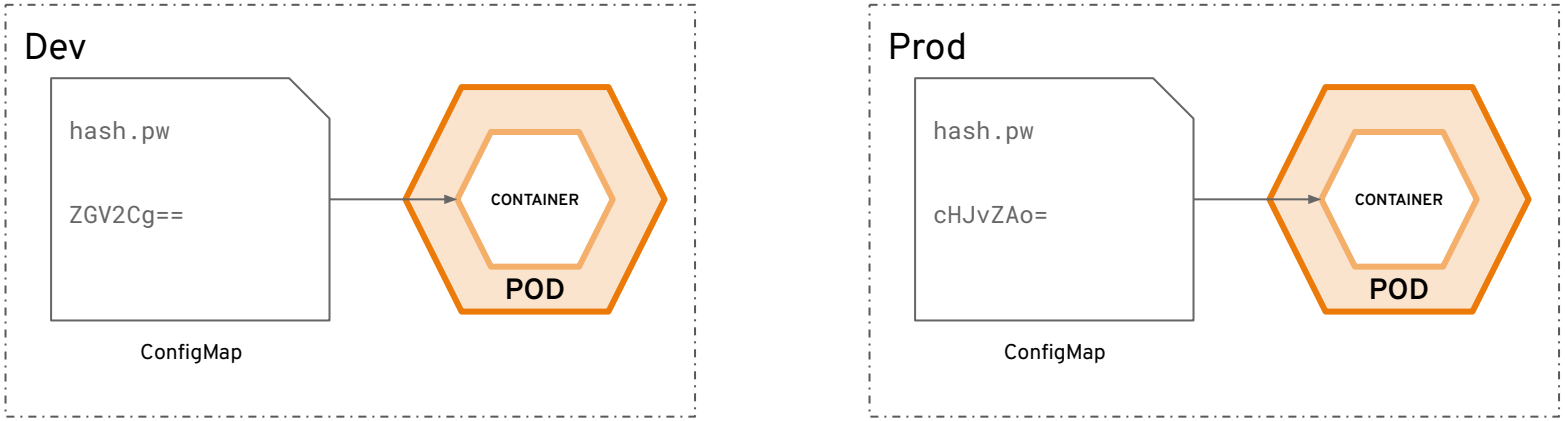
a daemonset ensures that all (or some) nodes run a copy of a pod



`configmaps` allow you to decouple configuration artifacts from image content

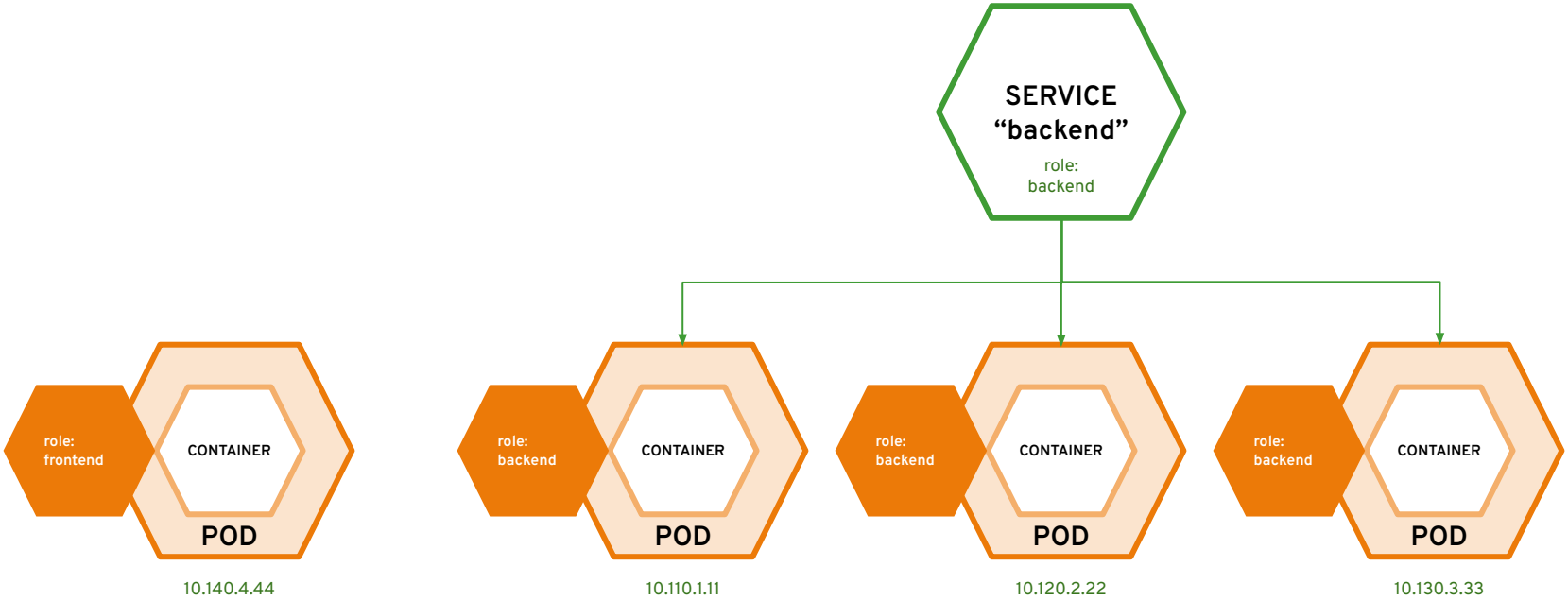


`secrets` provide a mechanism to hold sensitive information such as passwords

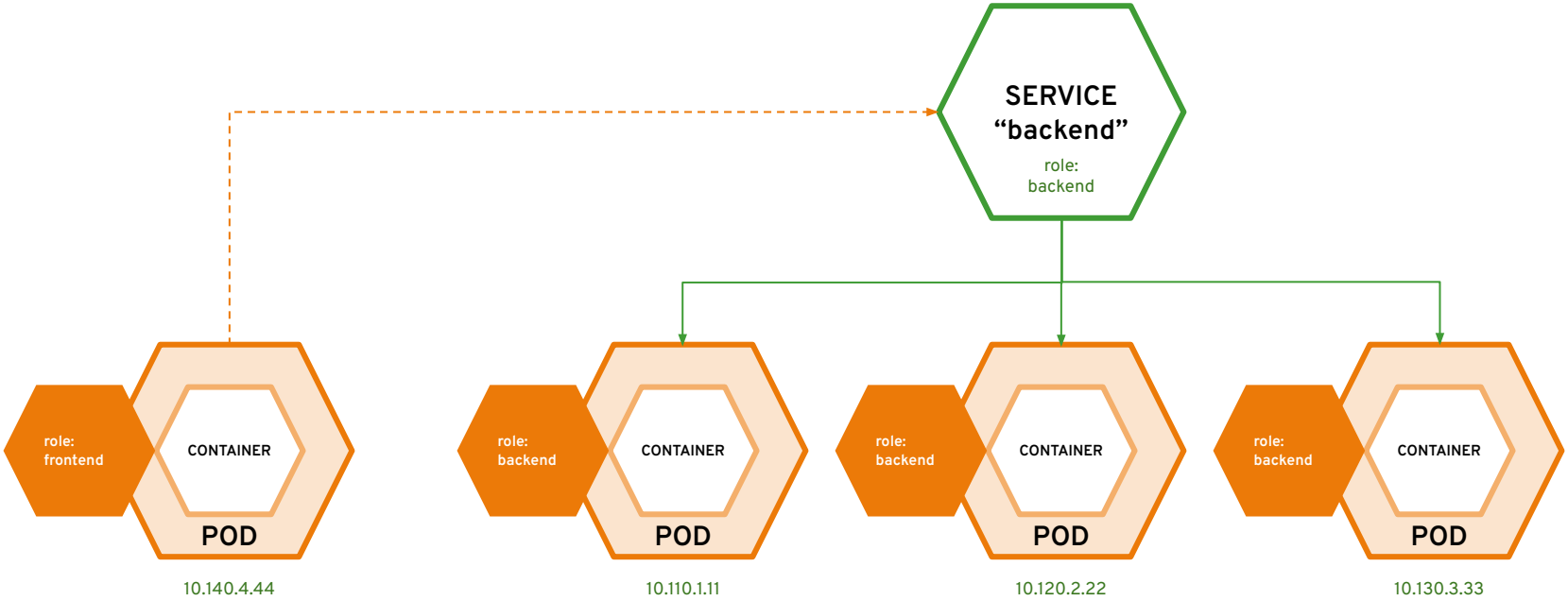


The etcd datastore can be encrypted for additional security
https://docs.openshift.com/container-platform/4.6/security/encrypting-etcd.html

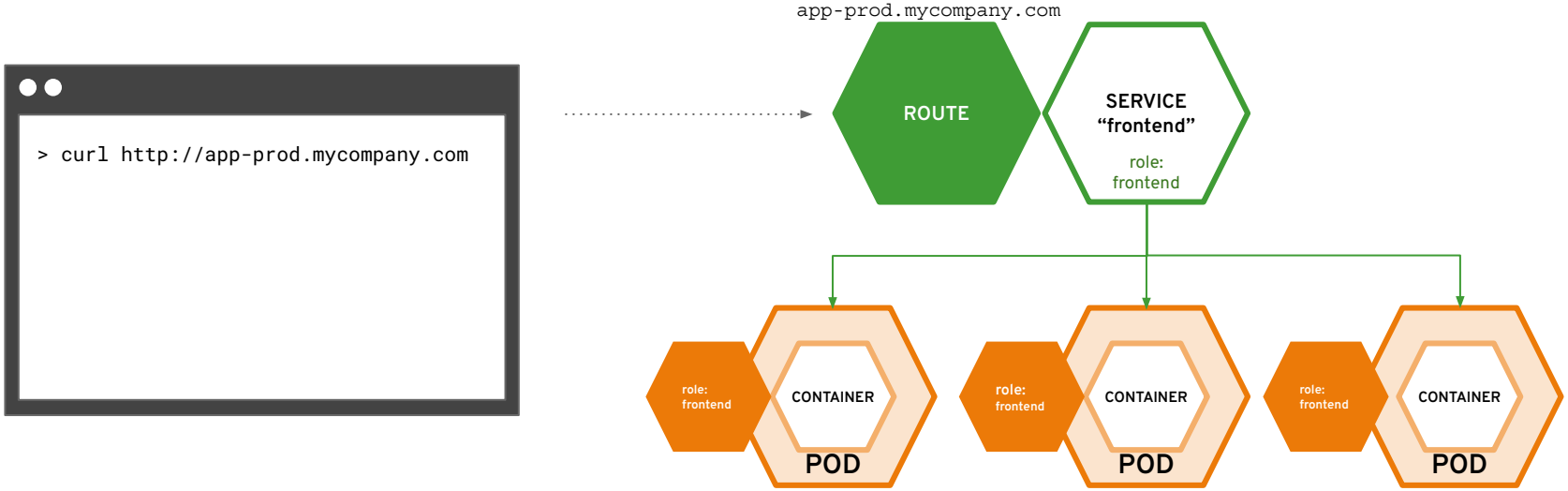
services provide internal load-balancing and service discovery across pods



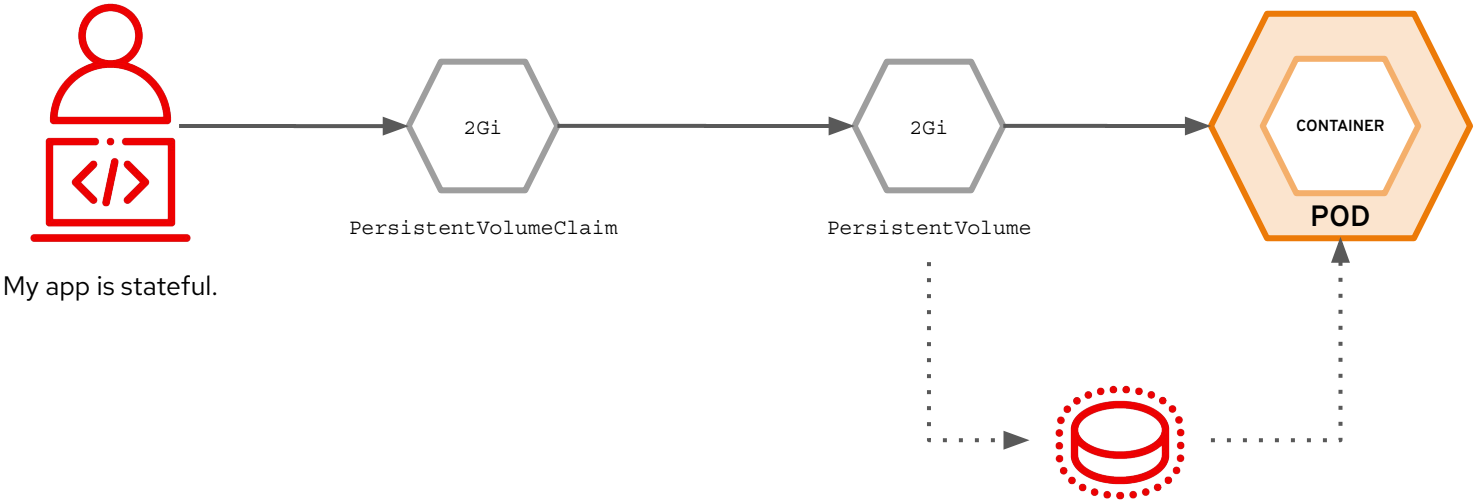
apps can talk to each other via services



`routes` make services accessible to clients outside the environment via real-world urls

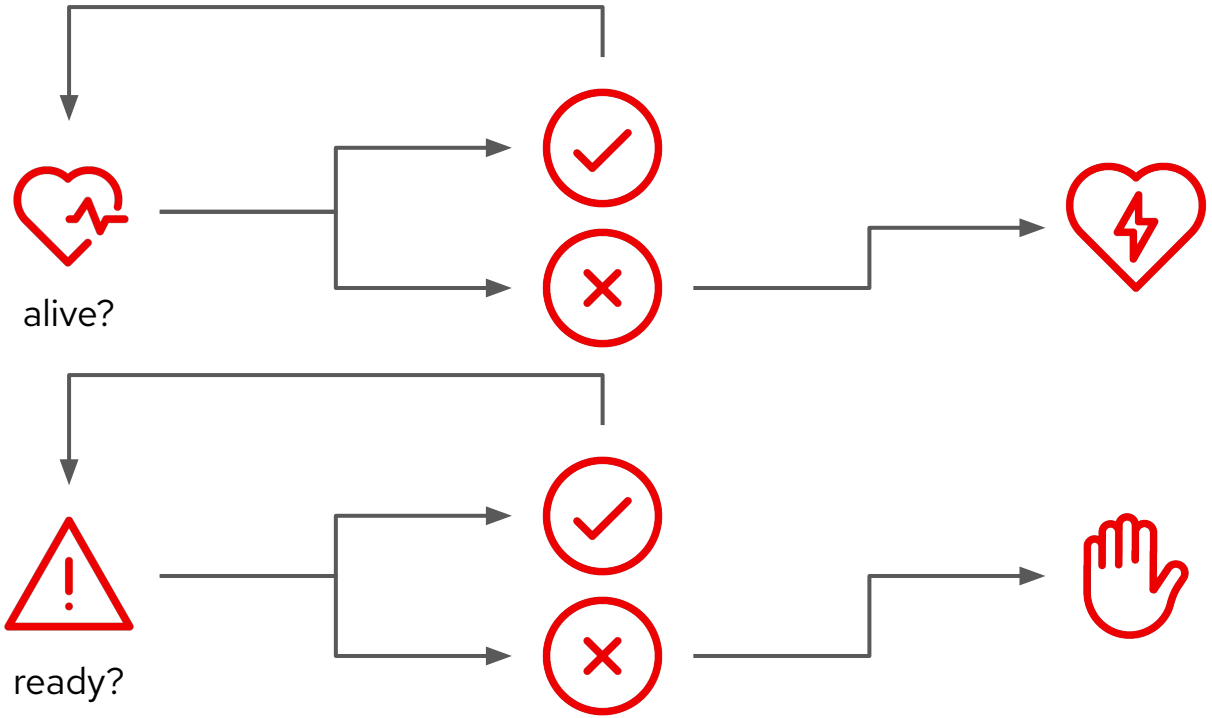


Persistent Volume and Claims

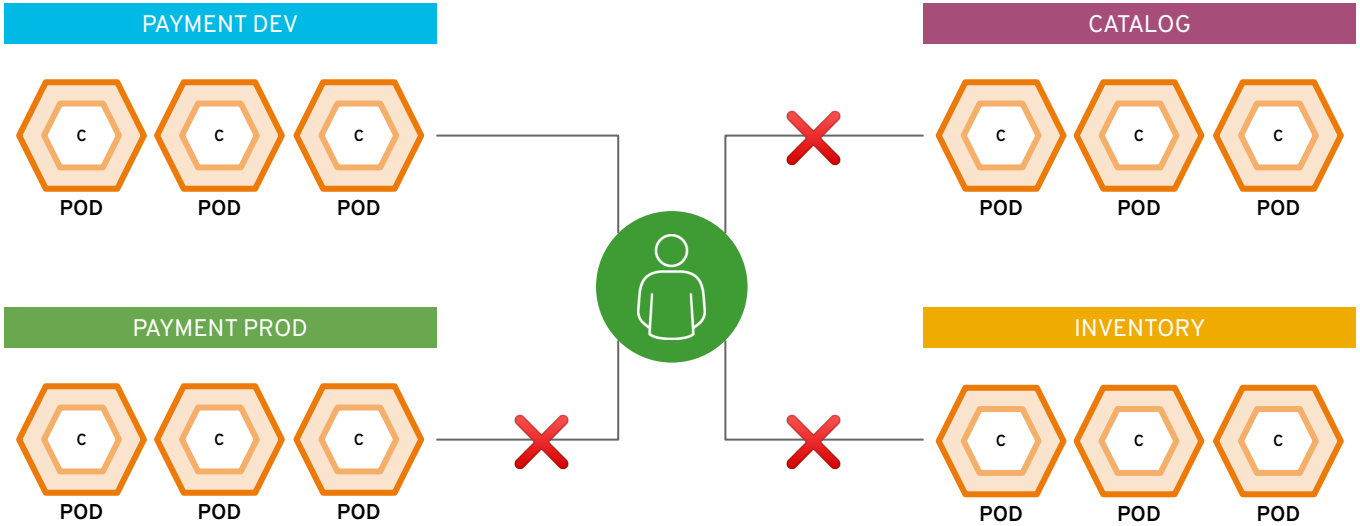


My app is stateful.

Liveness and Readiness



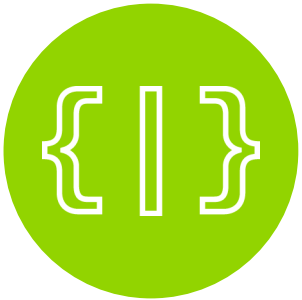
projects isolate apps across environments, teams, groups and departments



Red Hat

BUILD & DEPLOY

BUILD AND DEPLOY CONTAINER IMAGES

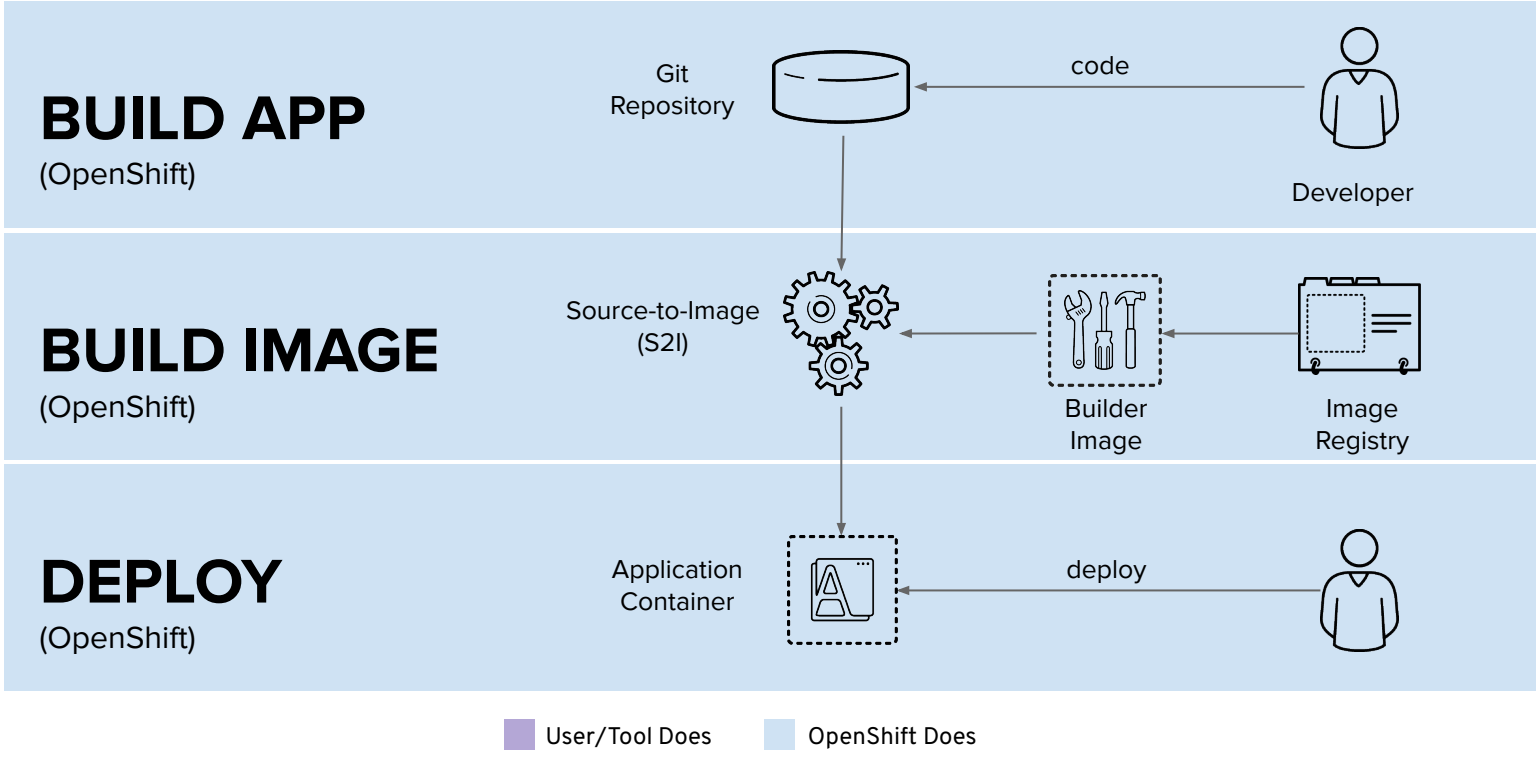


DEPLOY YOUR SOURCE CODE

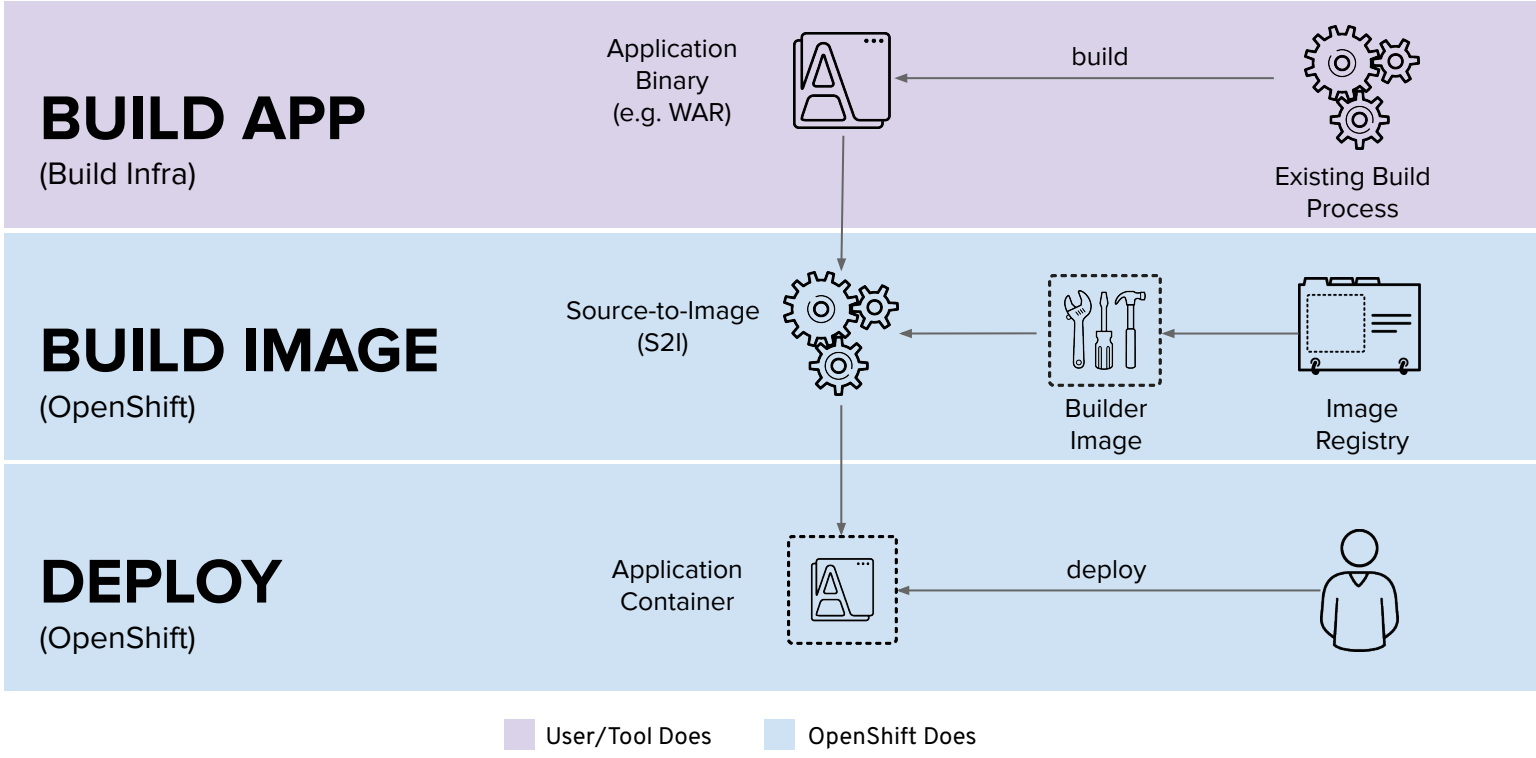
DEPLOY YOUR APP BINARY

DEPLOY YOUR CONTAINER IMAGE

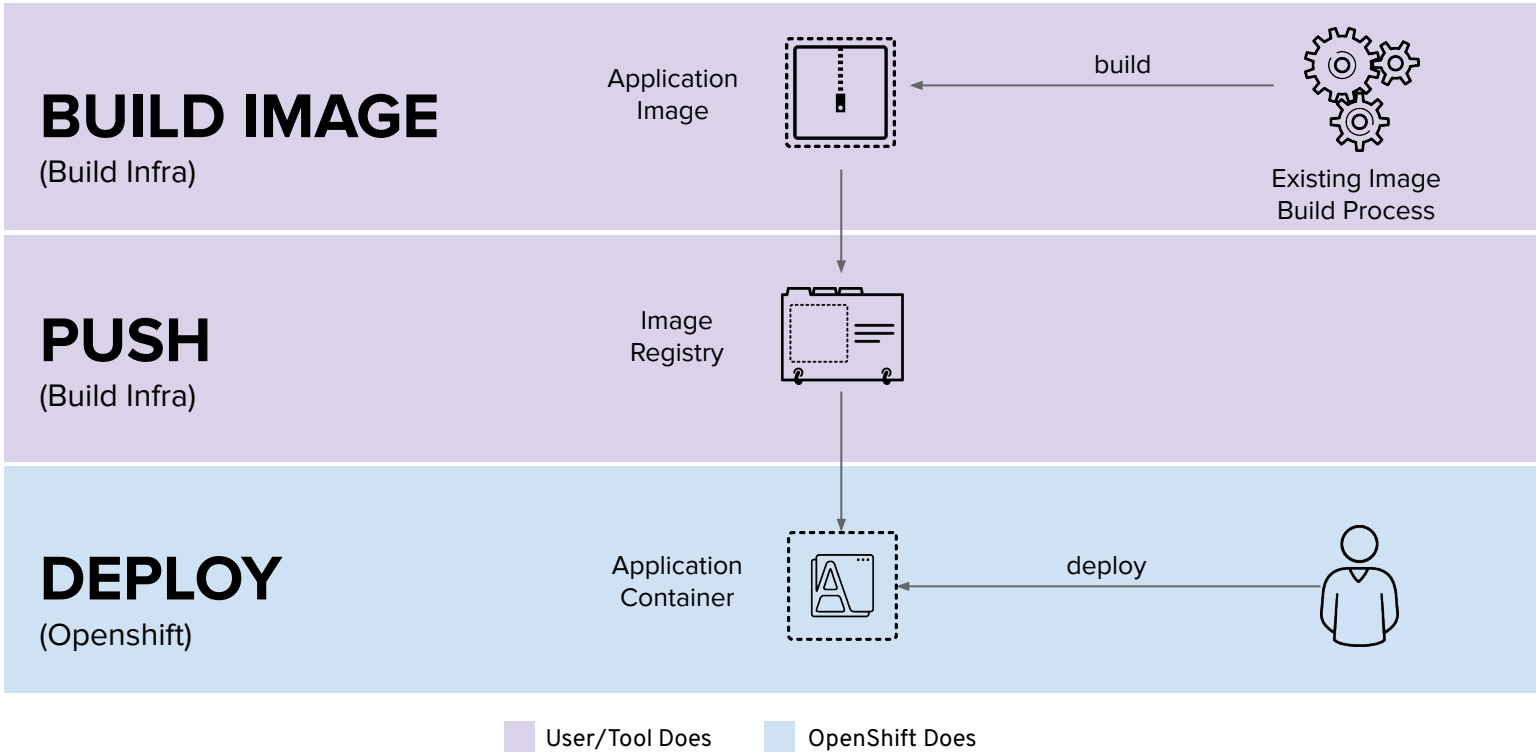
DEPLOY SOURCE CODE WITH SOURCE-TO-IMAGE (S2I)



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

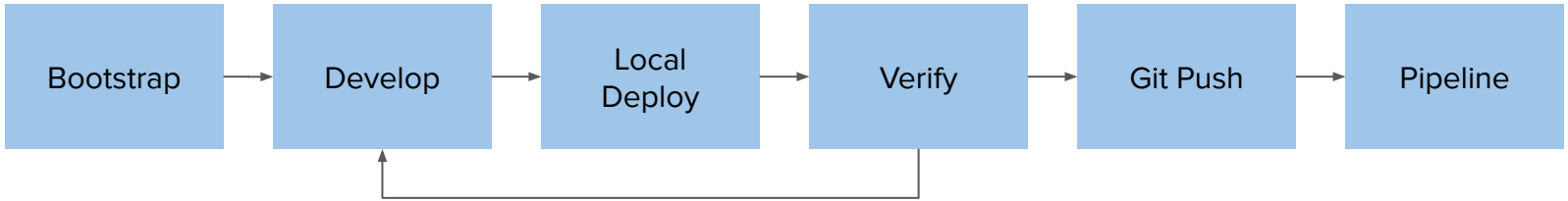


DEPLOY DOCKER IMAGE



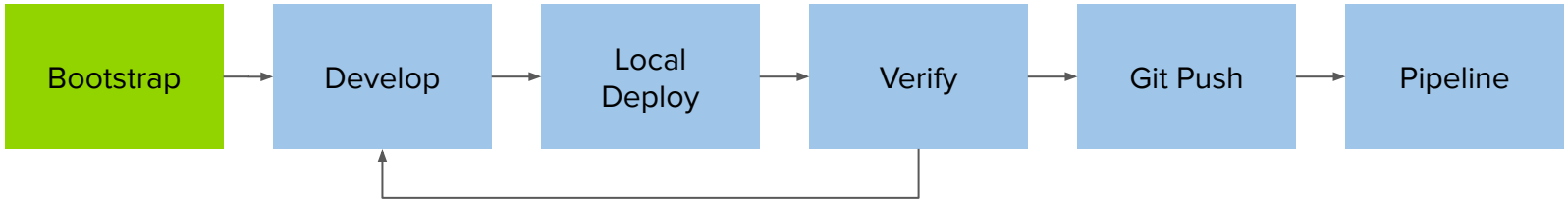
DEVELOPER WORKFLOW

LOCAL DEVELOPMENT WORKFLOW



RED HAT®
OPENSHIFT
Container Platform

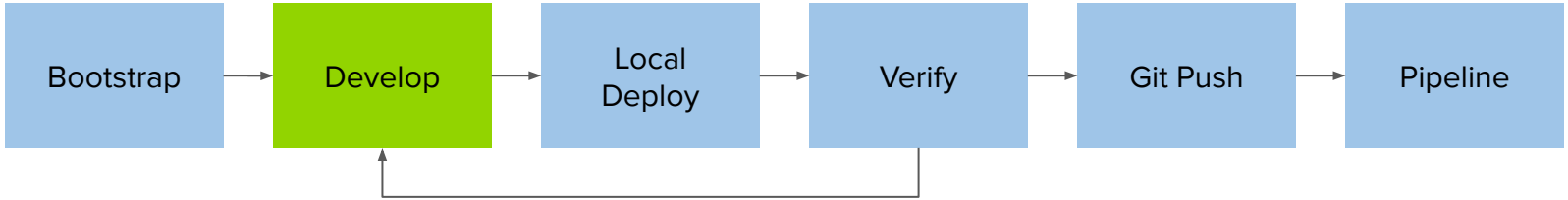
LOCAL DEVELOPMENT WORKFLOW



BOOTSTRAP

- Pick your programming language and application runtime of choice
- Create the project skeleton from scratch or use a generator such as
  - Maven archetypes
  - Quickstarts and Templates
  - OpenShift Generator
  - Spring Initializr

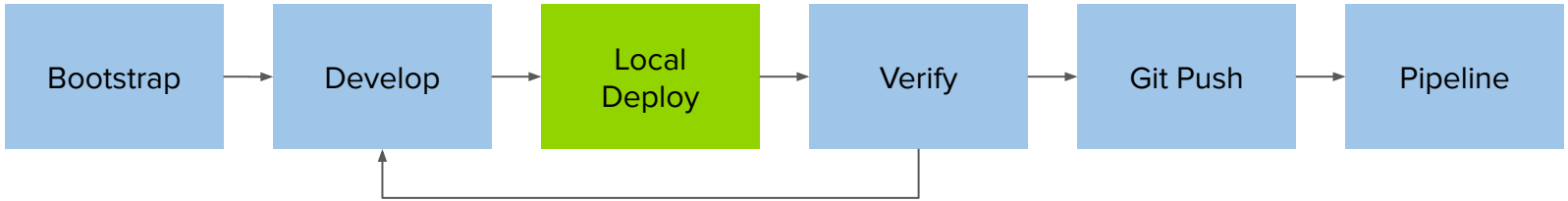
LOCAL DEVELOPMENT WORKFLOW



DEVELOP

- Pick your framework of choice such as Java EE, Spring, Ruby on Rails, Django, Express, ...
- Develop your application code using your editor or IDE of choice
- Build and test your application code locally using your build tools
- Create or generate OpenShift templates or Kubernetes objects

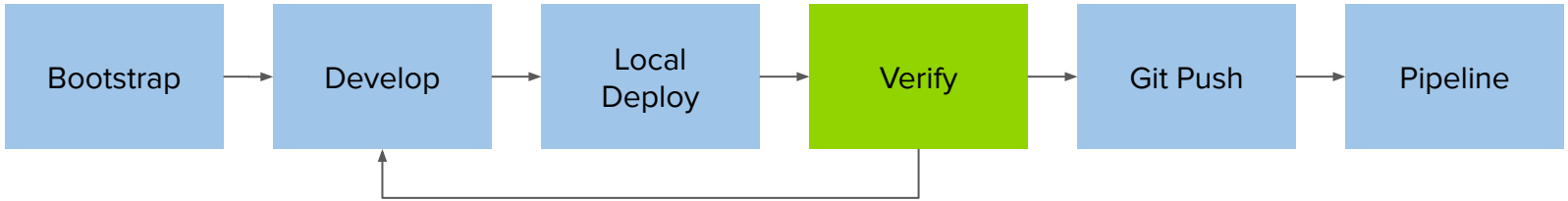
LOCAL DEVELOPMENT WORKFLOW



LOCAL DEPLOY

- Deploy your code on a local OpenShift cluster
  - Red Hat Container Development Kit (CDK), minishift and oc cluster
- Red Hat CDK provides a standard RHEL-based development environment
- Use binary deploy, maven or CLI rsync to push code or app binary directly into containers

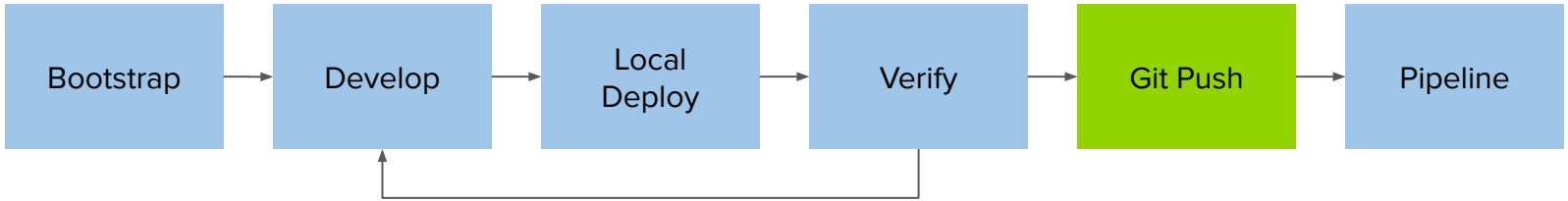
LOCAL DEVELOPMENT WORKFLOW



VERIFY

- Verify your code is working as expected
- Run any type of tests that are required with or without other components (database, etc)
- Based on the test results, change code, deploy, verify and repeat

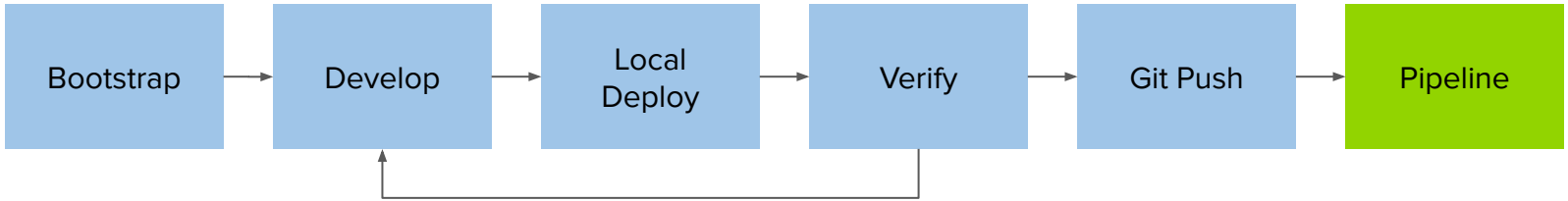
LOCAL DEVELOPMENT WORKFLOW



GIT PUSH

- Push the code and configuration to the Git repository
- If using Fork & Pull Request workflow, create a Pull Request
- If using code review workflow, participate in code review discussions

LOCAL DEVELOPMENT WORKFLOW



PIPELINE

- Pushing code to the Git repository triggers one or multiple deployment pipelines
- Design your pipelines based on your development workflow e.g. test the pull request
- Failure in the pipeline? Go back to the code and start again

OpenShift Developer Ecosystem

OpenShift Container Platform

Increase developer productivity on Kubernetes with developer-focused tools and services

Developer Console
Application-centric dashboard for onboarding applications and visibility into application components.

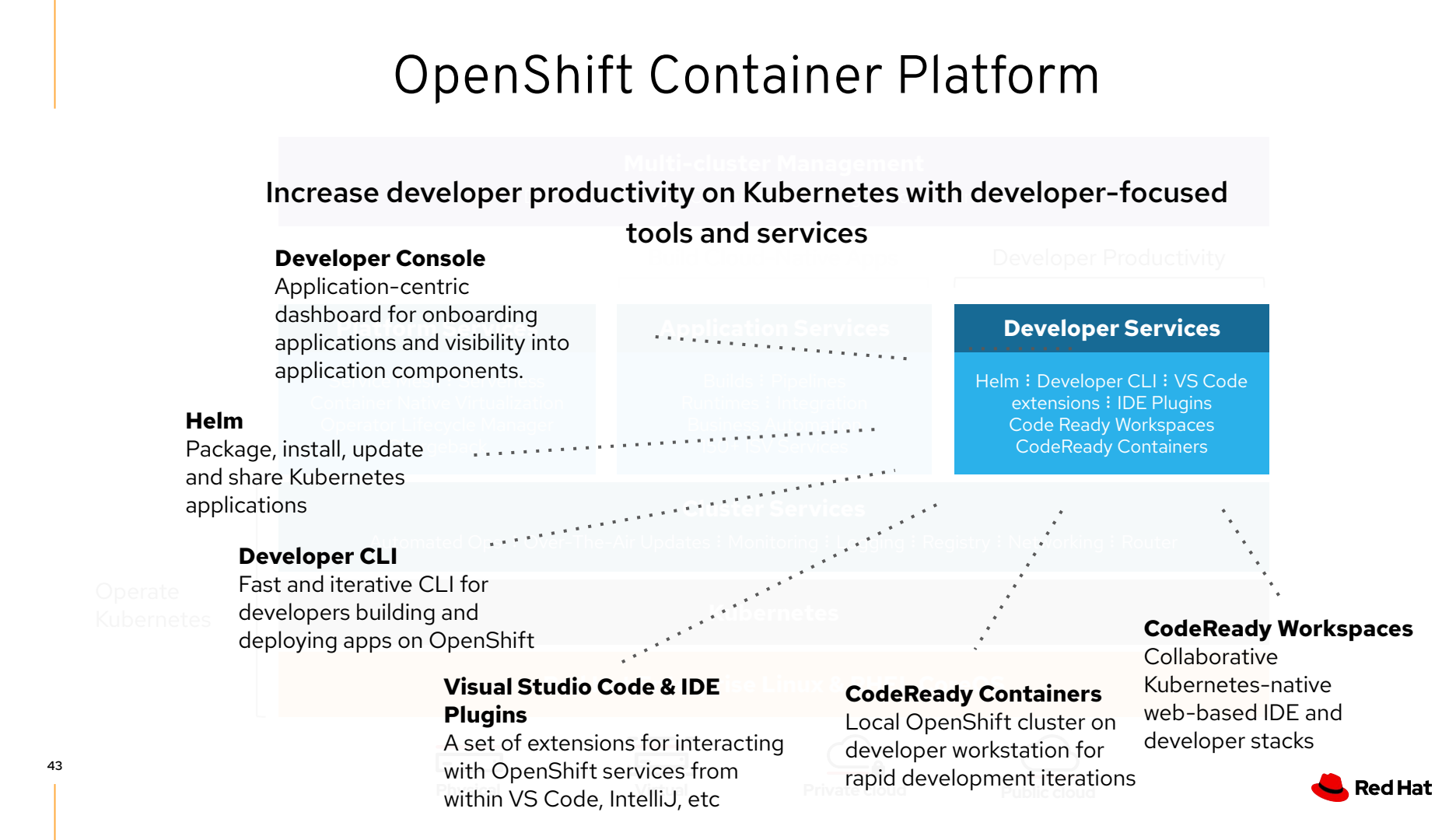
Helm
Package, install, update and share Kubernetes applications

Developer CLI
Fast and iterative CLI for developers building and deploying apps on OpenShift

Visual Studio Code & IDE Plugins
A set of extensions for interacting with OpenShift services from within VS Code, IntelliJ, etc

CodeReady Containers
Local OpenShift cluster on developer workstation for rapid development iterations

CodeReady Workspaces
Collaborative Kubernetes-native web-based IDE and developer stacks

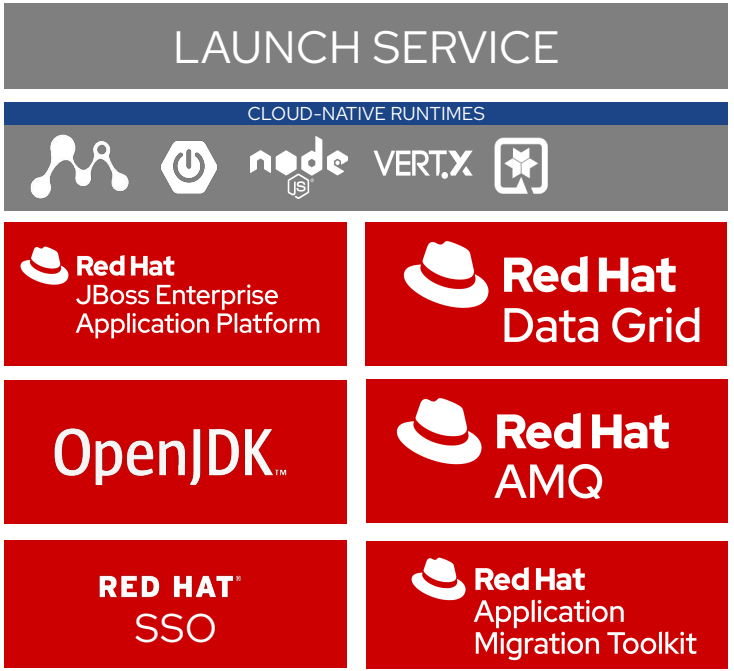


Red Hat

Red Hat Runtimes

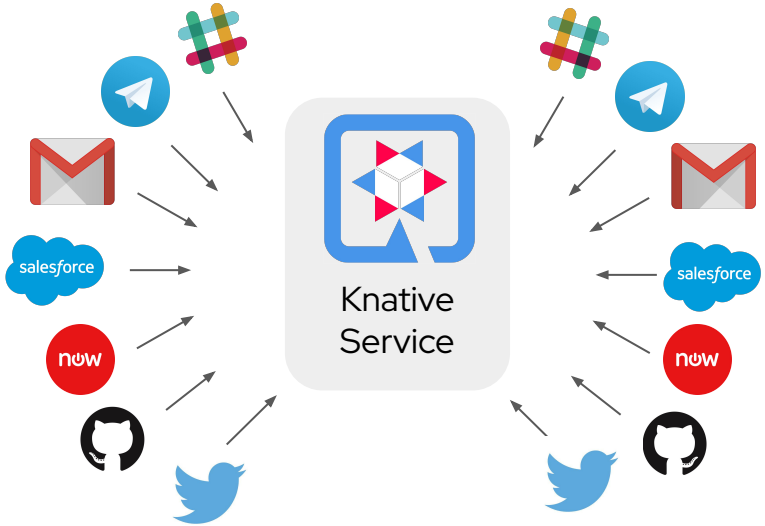
- **Quarkus GA** - Supersonic, Subatomic Java. Native support coming in Q3 via Mandrel.
- **Red Hat SSO** - Support for OpenShift secret Vaults, WebAuthn protocol.
- **Data Grid 8.1** - Cross-site cluster support
- **Z-Series support for Runtimes** - Use existing Z & Cloud Pak investment with Runtimes & OpenShift.
- **JBOSS EAP** expansion pack for MicroProfile
- **Spring Boot 2.2** - New AMQ Starters, GA of Reactive support and Kubernetes Java annotations.

[What's New in Red Hat Runtimes [Q2]]



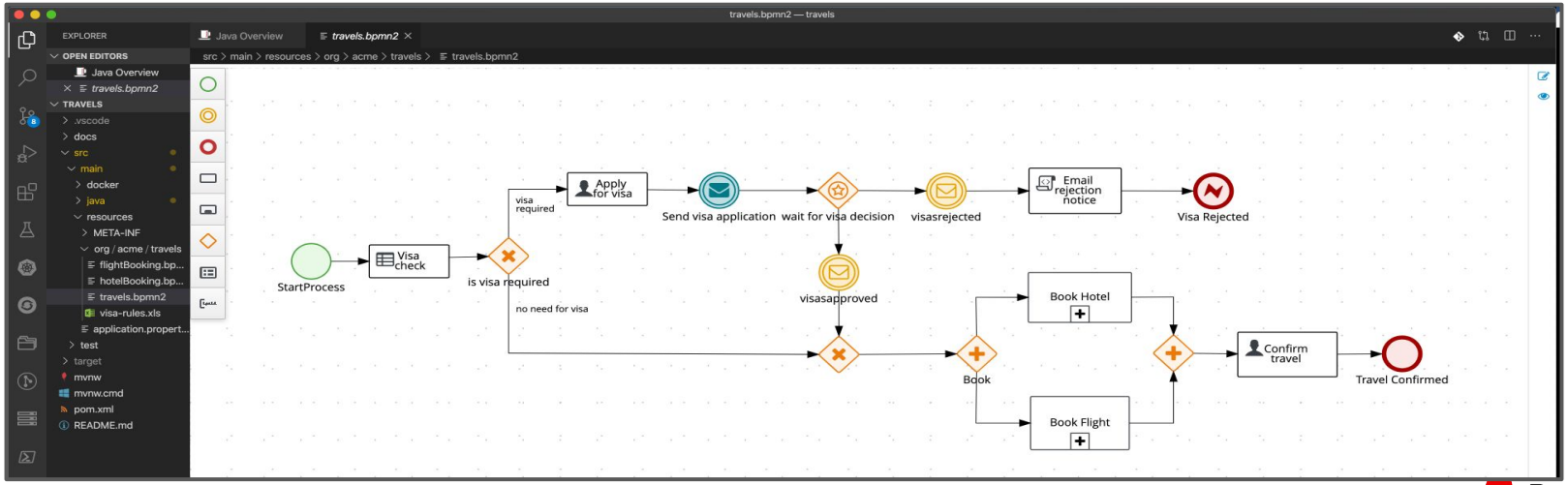
Red Hat Integration

- **Service Registry GA** - schema and API registry to support Kafka and API-based workloads on OpenShift.
- **Mirror Maker 2.0** - major evolution in data replication for Kafka on OpenShift now at full support in Strimzi
- **Camel K for Serverless (TP)** - leverage the huge Camel connector catalog to drive events into your OpenShift Serverless applications based on Camel K and Knative Eventing.



Red Hat Process Automation

- **Kogito -** moves to developer preview for next-gen business automation based on Quarkus
- **Trusty AI -** first MVP of Trusty AI in the next Kogito developer preview release. With runtime dashboards and metrics.
- **Red Hat Business Automation Bundle -** released in the Visual Studio Code Marketplace as Developer Preview



Product Manager: Karena Angell (on behalf of the Red Hat Application Services team)

Red Hat

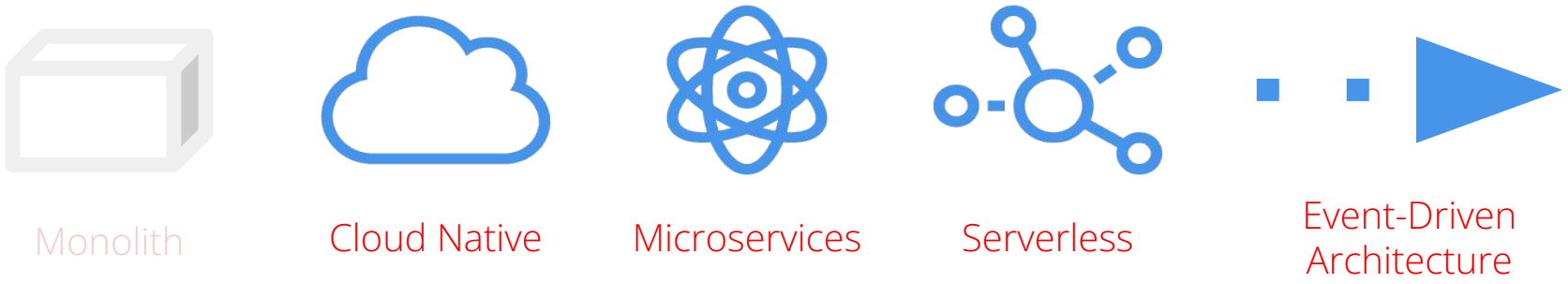
Quarkus Business Value

QUARKUS

Supersonic. Subatomic. Java.

Red Hat

Quarkus - Kubernetes Native Java



Red Hat

WHAT IS QUARKUS?

QUARK: elementary particle / **US**: hardest thing in computer science

Quarkus Business Value

“Supersonic, Subatomic Java”

Cost Savings

Cloud efficiency (low memory, fast startup, high density), serverless deployments

Faster Time to Market

Developer productivity, extensions ecosystem, low learning curve, keep competitive edge

Reliability

Trusted technology, active community, trusted sponsor, proven Java libraries and standards

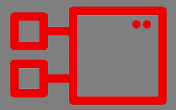
Red Hat

Why Quarkus?

"Quarkus is a great fit if you are looking for..."

IMPROVED APP PERFORMANCE & COST

- ❏ Reduce memory consumption
- ❏ Increase startup speeds
- ❏ Reduce application size
- ❏ Looking for Spring or Java alternatives



DIGITAL TRANSFORMATION

- ❏ Modernize or optimize existing applications
- ❏ Move to a cloud-native or microservices architecture
- ❏ Need lightweight connectivity between services
- ❏ Next generation business automation

SERVERLESS

- ❏ Have seldom used but critical services
- ❏ Interested in serverless or FaaS (function as a service)

Quarkus Technical Value

"Historical" Enterprise Java Stack

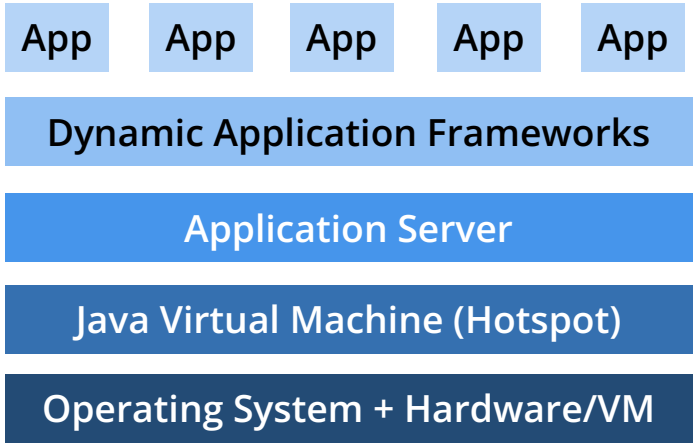
Architecture: Monoliths

Deployment: multi-app, appserver

App Lifecycle: Months

Memory: 1GB+ RAM

Startup Time: 10s of sec



"Modern" Enterprise Java Stack

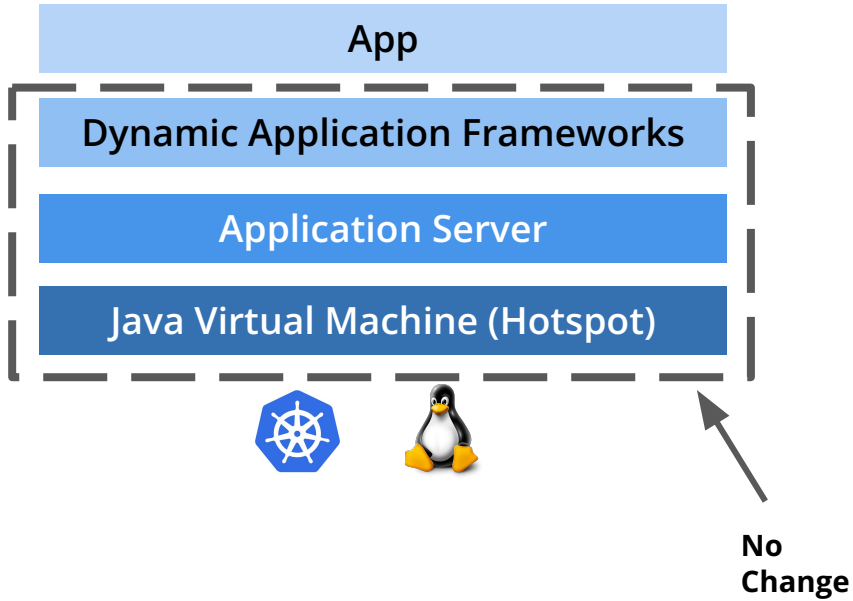
Architecture: Microservices

Deployment: Single App

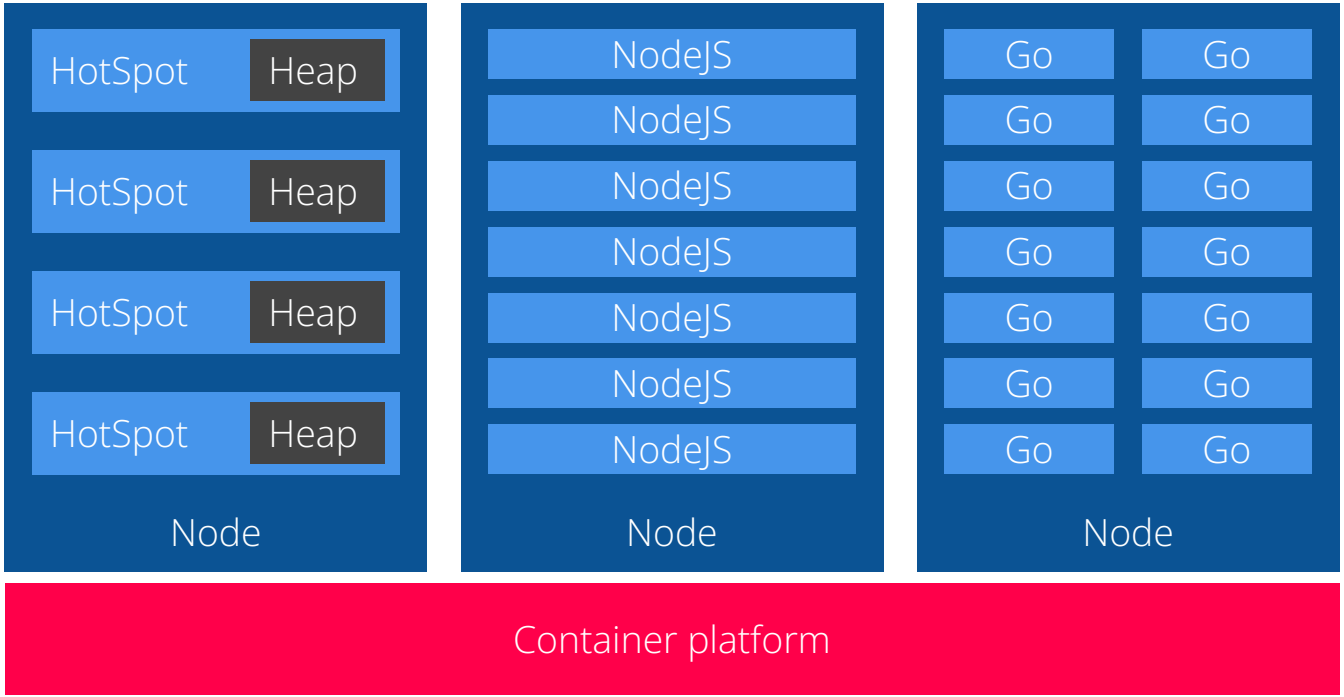
App Lifecycle: Days

Memory: 100MBs+ RAM

Startup Time: Seconds



Hidden Truth About Java + Containers



Red Hat

THERE IS A NEED FOR A NEW JAVA STACK FOR CLOUD-NATIVE AND SERVERLESS

Experts from cloud-native Java OS projects

Eclipse Vert.x | Hibernate | RESTEasy | Eclipse MicroProfile | WildFly | Undertow | OpenJDK

QUARKUS

Differentiators

Container First

Tailors your app for HotSpot & GraalVM

Fast boot time and low RSS memory

Serverless fit

Developer Joy

Live coding

Unified configuration

Unifies Imperative & Reactive

Combines blocking and non-blocking

Built-in event bus

Best of Breed Libraries & Standards

90+ extensions

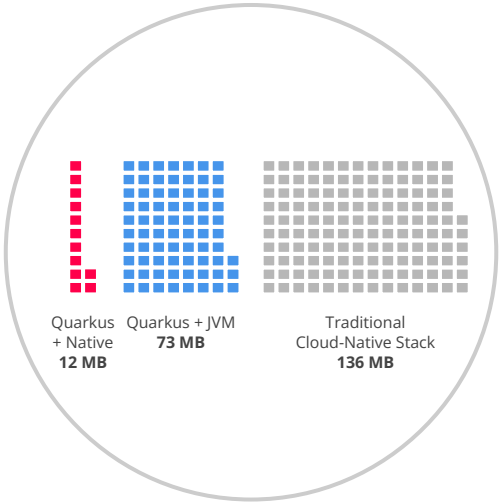
“Powered by Quarkus” applications

Red Hat

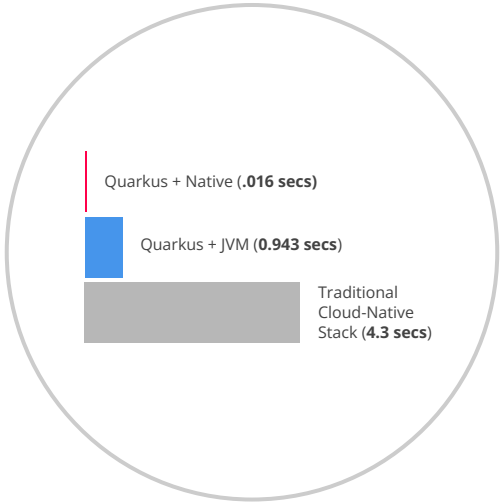
Benefit No. 1: Container First

*"We went from **1-min** startup times to **400 milliseconds**"*

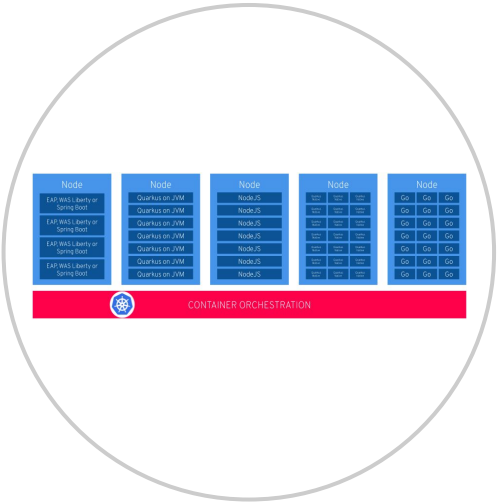
Reduced Memory Footprint



Fast Startup Time



Smaller Disk Footprint



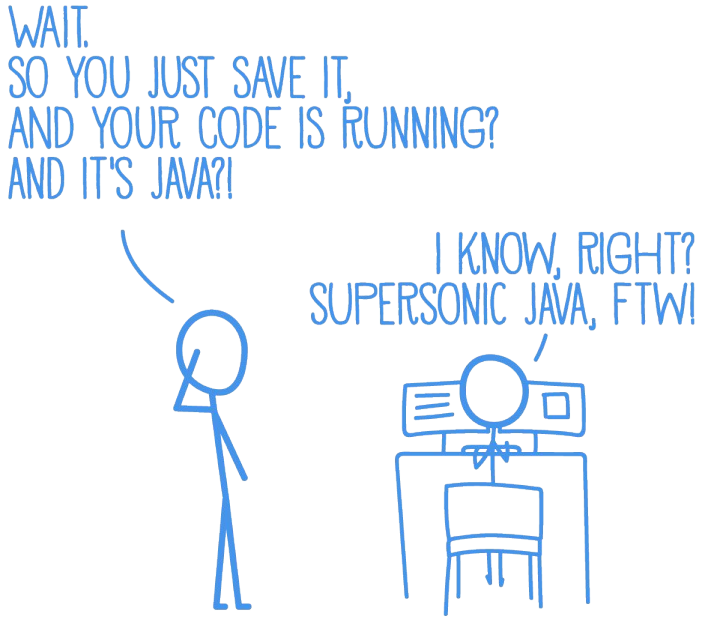
Red Hat

Benefit No. 2: Developer Joy

*"Our developers used to wait **2 to 3 mins** to see their changes. **Live coding** does away with this."*

A cohesive platform for optimized developer joy:

- Based on standards and more
- Unified configuration
- Live coding
- Streamlined code for the 80% common usages, flexible for the 20%
- No hassle native executable generation



Red Hat

Benefit No. 3: Unifies Imperative and Reactive

```
@Inject
SayService say;

@GET
@Produces(MediaType.TEXT_PLAIN)
public String hello() {
      return say.hello();
}
```

```
@Inject @Stream("kafka")
Publisher<String> reactiveSay;

@GET
@Produces(MediaType.SERVER_SENT_EVENTS)
public Publisher<String> stream() {
      return reactiveSay;
}
```

- Combine both Reactive and imperative development in the same application
- Inject the EventBus or the Vertx context
- Use the technology that fits your use-case
- Key for reactive systems based on event driven apps

Benefit No. 4: Best of Breed Frameworks & Standards

"When you adopt Quarkus, you will be productive from day one since you don't need to learn new technologies."

Eclipse Vert.x

Hibernate

RESTEasy

Apache Camel

Eclipse MicroProfile

Netty

Kubernetes

OpenShift

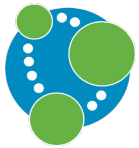
Jaeger

Prometheus

Apache Kafka

Infinispan

Flyway



Neo4j

MongoDB

MQTT

KeyCloak

Apache Tika

Red Hat

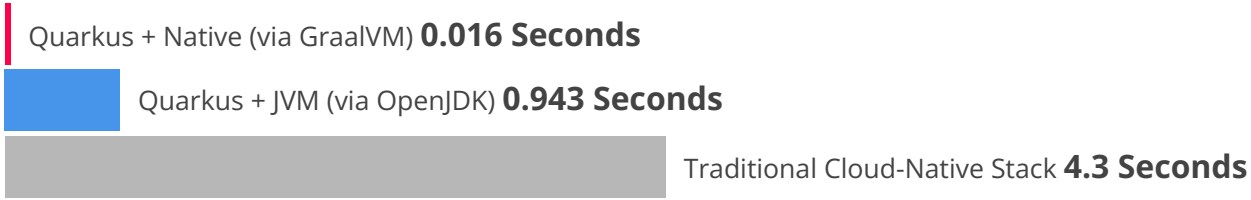
Supersonic, Subatomic

Fast.
Blazing fast to start.
Millisecond fast!

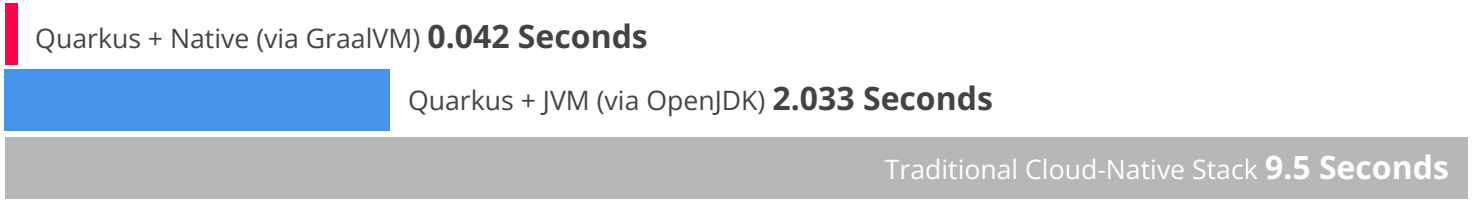
Red Hat

Supersonic, Subatomic Java

REST



REST + CRUD

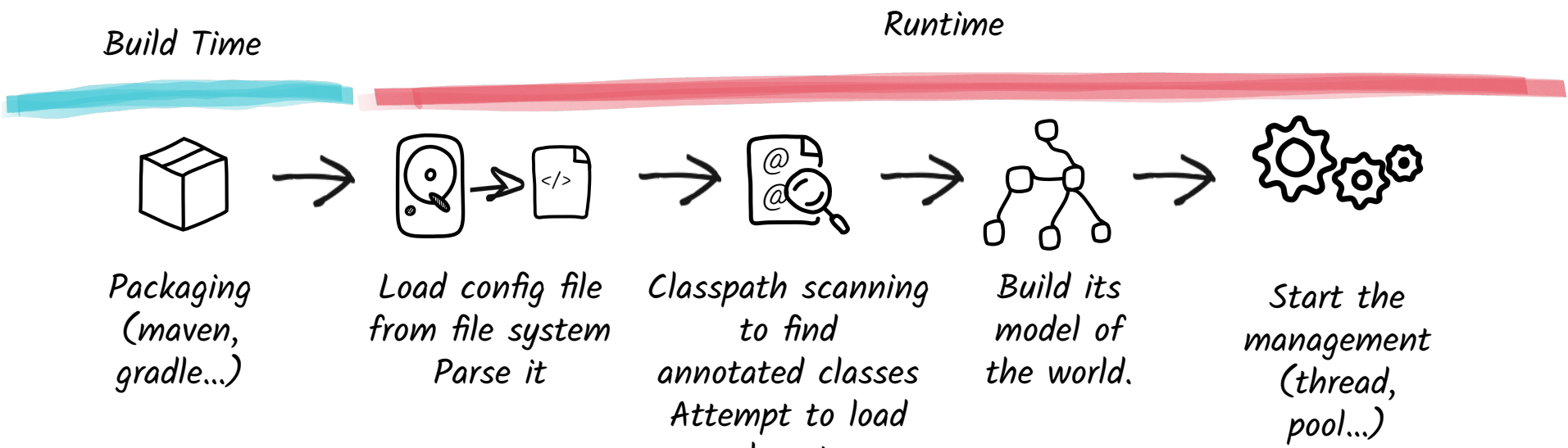


Time to first response

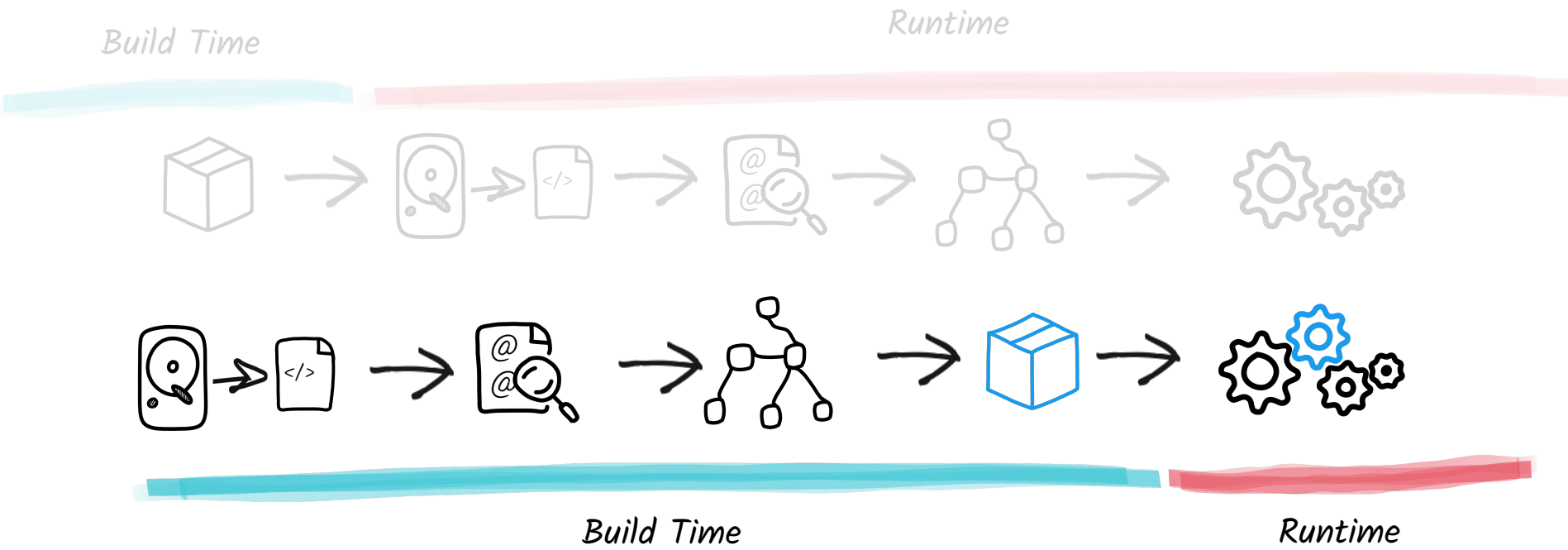
Red Hat

HOW DOES QUARKUS WORK?

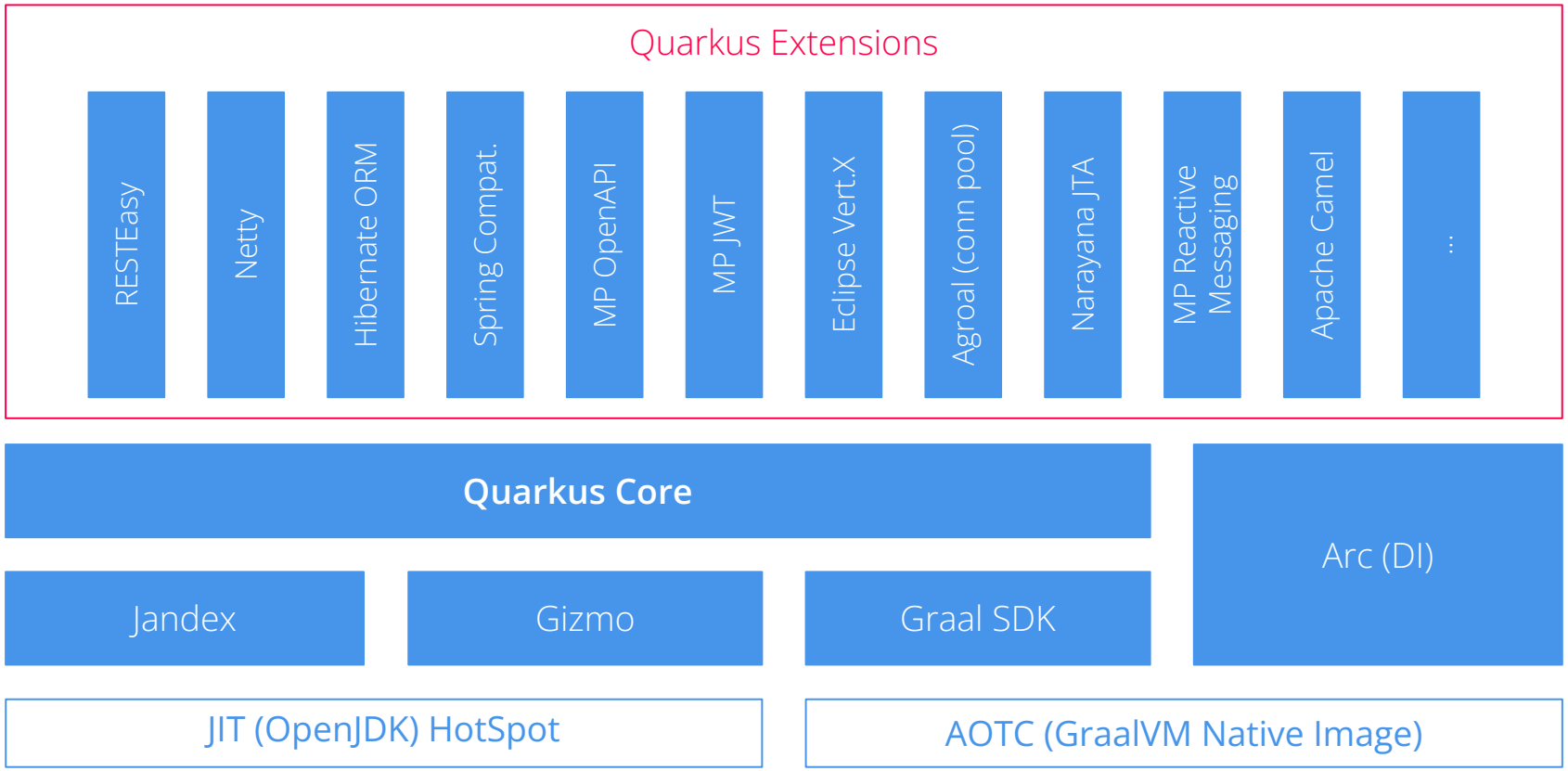
How Does a Framework Start?



The Quarkus Way



An ahead-of-time, build-time, runtime


Quarkus Extensions
RESTEasy
Netty
Hibernate ORM
Spring Compat.
MP OpenAPI
MP JWT
Eclipse Vert.X
Agroal (conn pool)
Narayana JTA
MP Reactive Messaging
Apache Camel
...
Quarkus Core
Arc (DI)
Jandex
Gizmo
Graal SDK
JIT (OpenJDK) HotSpot
AOTC (GraalVM Native Image)

Red Hat

The Right VM For the Right Deployment

JIT (OpenJDK HotSpot)

High memory density requirements
High request/s/MB
Fast startup time

Best raw performance (CPU)
Best garbage collectors
Higher heap size usage

Known monitoring tools
Compile Once, Run anywhere
Libraries that only works in standard JDK

AOT (GraalVM native image)*

Highest memory density requirements
Highest request/s/MB
    for low heap size usages
Faster startup time
    10s of ms for Serverless

*Currently in Tech Preview

Red Hat

Quarkus Tools - Build


```
mvn io.quarkus:quarkus-maven-plugin:1.3.2.Final-redhat-00001:create \
    -DprojectGroupId=org.acme \
    -DprojectArtifactId=getting-started \
    -DplatformGroupId=com.redhat.quarkus \
    -DplatformVersion=1.3.2.Final-redhat-00001 \
    -DclassName="org.acme.quickstart.GreetingResource" \
    -Dpath="/hello"
cd getting-started
```

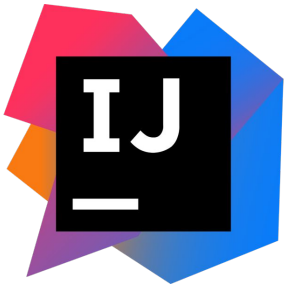
*community supported

Red Hat

Quarkus Tools - IDE

VSCode

Eclipse



IntelliJ

che.openshift.io

DEMO

THANK YOU