



The Containers and Cloud-Native Roadshow Developer Track





Software driven innovation explosion

The IDC predicts that from 2018 to 2023,
500 million new logical apps will be created, equal to
the number built over the past 40 years.

Customers and businesses expect...

ON-DEMAND
SERVICE

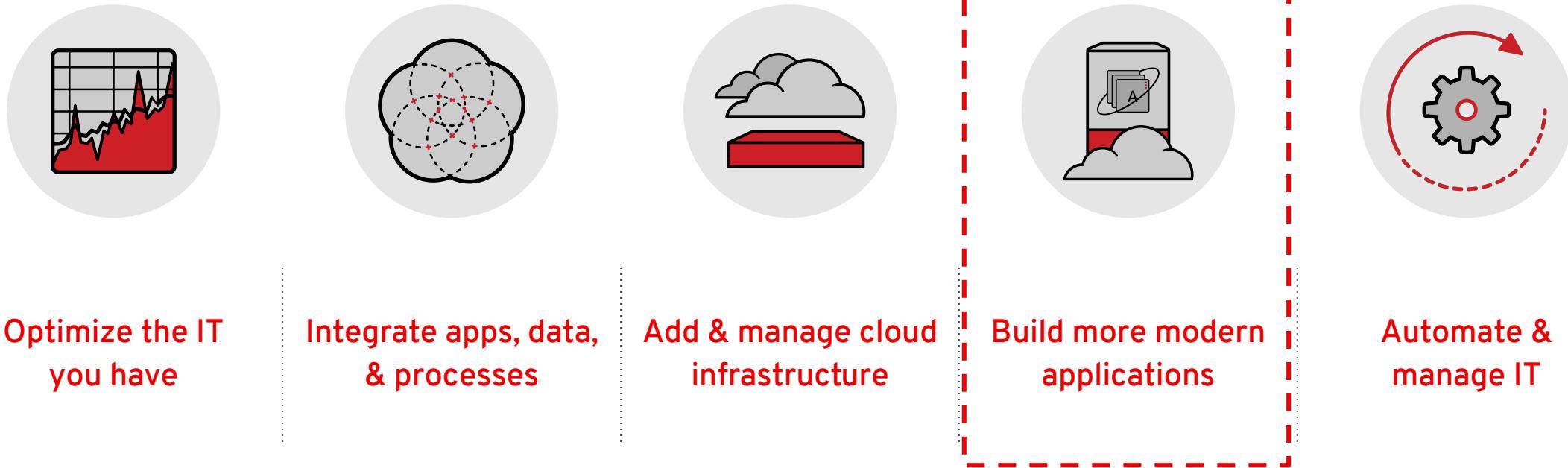
DELIGHTFUL
INTERACTIONS

ACCESS FROM
ANYWHERE

PERSONALIZED
EXPERIENCE

Creating value depends on your ability
to develop and deliver
high quality applications faster.

How do you drive innovation to meet these expectations
while keeping the lights on?



Leveraging the cloud becomes a key strategy for success

WHY CLOUD-NATIVE APPS?



**FASTER
SERVICE DELIVERY**



**INCREASE
SERVICE QUALITY**



**REDUCE
RISK OF DELIVERY**

How does one build apps for the cloud?



Write once, run anywhere?



THE PATH TO CLOUD-NATIVE APPS

A DIGITAL DARWINISM

RE-ORG TO
DEVOPS

SELF-SERVICE
ON-DEMAND
INFRA

AUTOMATION

CONTINUOUS
DELIVERY

ADVANCED
DEPLOYMENT
TECHNIQUES

MICROSERVICES
.....
FAST
MONOLITH

RED HAT CLOUD-NATIVE DEV PLATFORM

Our vision is to simplify the creation of cloud-native services and serverless functions with a rich set of components and tools to match the **workloads** of modern cloud native apps.

Automate Kubernetes application operations with DevOps in mind

Cloud-native middleware applications services and service mesh

Tools and standard processes to increase developer productivity on Kubernetes



Red Hat Developer Tools

Tools and standard processes to increase developer productivity on Kubernetes

OPENSHIFT DEVELOPER SERVICES

IDE

Eclipse Che, VSCode, IntelliJ and Eclipse IDE provide web-based and desktop IDEs integrated with OpenShift and Kubernetes

DEV CONSOLE

Developer and application-centric web console for interacting with OpenShift

DEV CLI

Developer CLI to provide an application-centric way of interacting with OpenShift

BUILDS

Automated and secure image builds from application source code, binary or Dockerfile

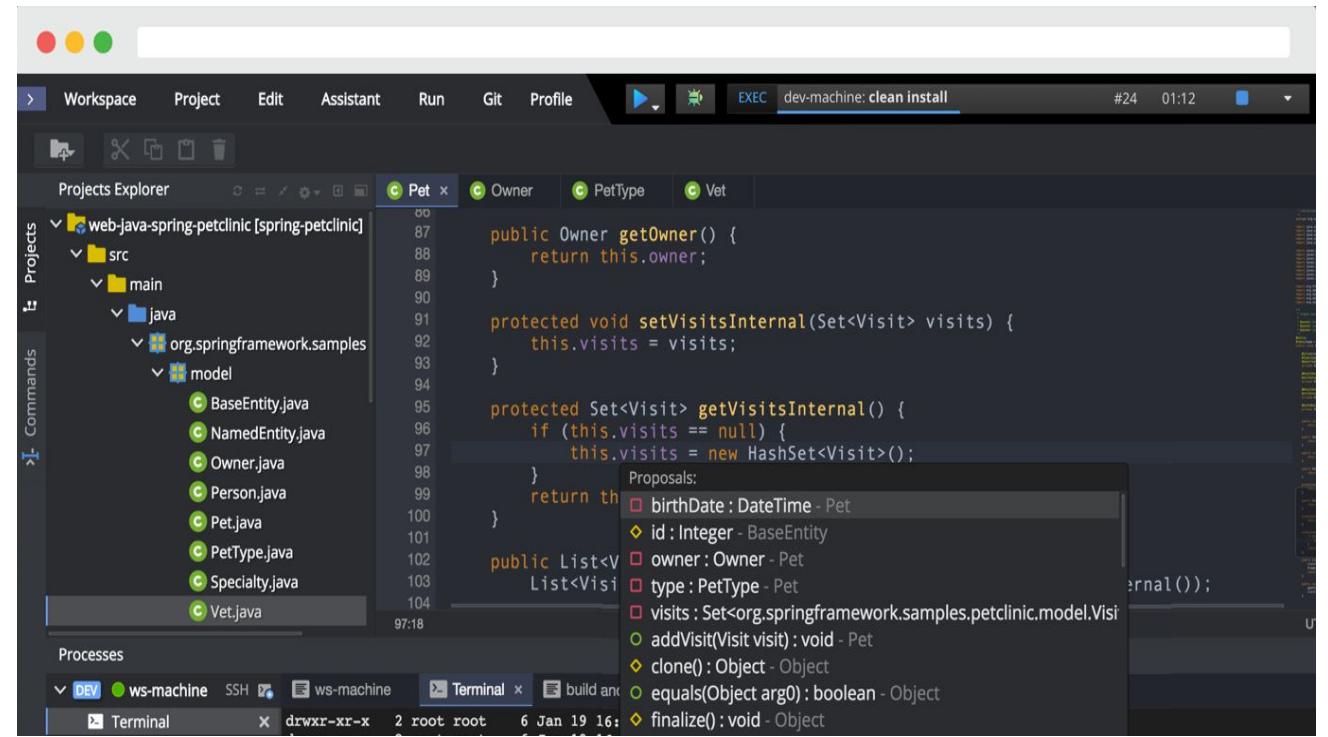
CI/CD

Continuous delivery pipelines optimized for Kubernetes (via the Tekton project)

Enabling developer productivity to build and deploy apps on Kubernetes

Red Hat CodeReady Workspaces

- Browser-based web IDE and dev environment in Kube pods
- Red Hat supported Eclipse Che
- Bundled with OCP/OSD SKU
- Available on OCP and OSD
- Enabled via an operator
- Stacks based on Red Hat Linux and Middleware
- Replaces VDI



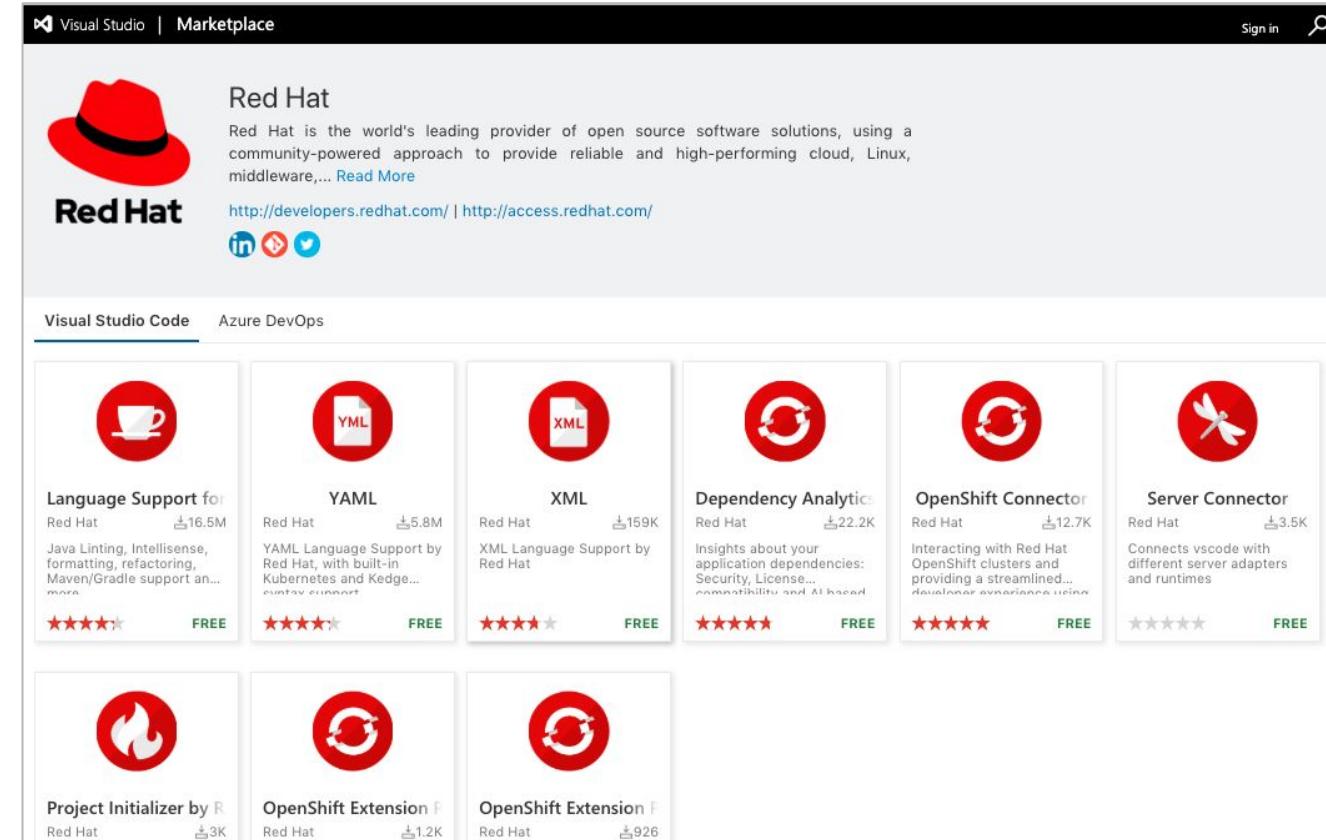
<https://www.youtube.com/watch?v=VwKEVeDy9TA>

Red Hat Plugins for Microsoft VS Code

Red Hat plugins for VSCode add IDE superpowers for Java, Kubernetes YAML and XML.

The **OpenShift** plugin allows developers to quickly connect and deploy to OpenShift instances locally or remotely.

Dependency Analytics adds license and CVE package alerts.



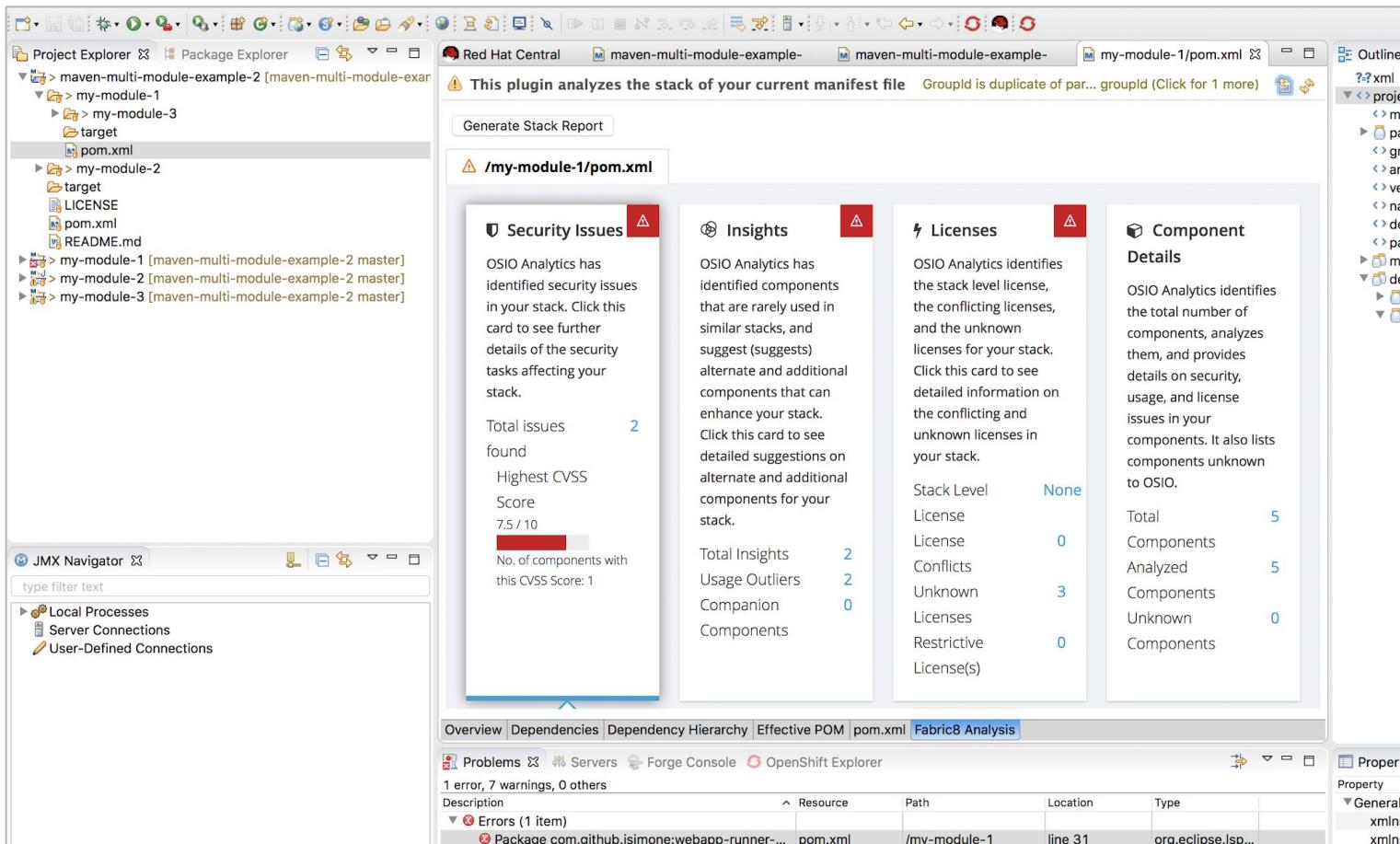
Use It To: Get the most out of Red Hat's products in the VS Code IDE.



Source Code Dependency Analytics

The dependency analytics service provides security and license warnings for any dependency in a project. This helps developers to fix problems earlier in the cycle.

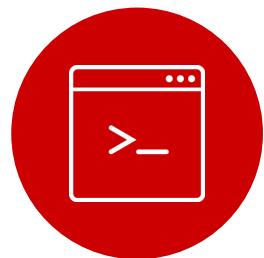
- Find CVEs in any package
- Discover license mismatches
- Supported for Java and Node



Use It To: Help developers find critical issues before they hit production.



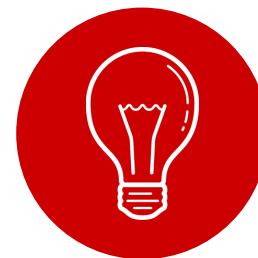
As a Red Hat Customer You Can Benefit from the Red Hat Developer Program and Tools Every Day.



LEVERAGE INTUITIVE
DEVELOPER TOOLS FOR
YOUR KEY PLATFORMS



RELY ON RED HAT SUPPORT
FROM DEVELOPMENT
THROUGH PRODUCTION

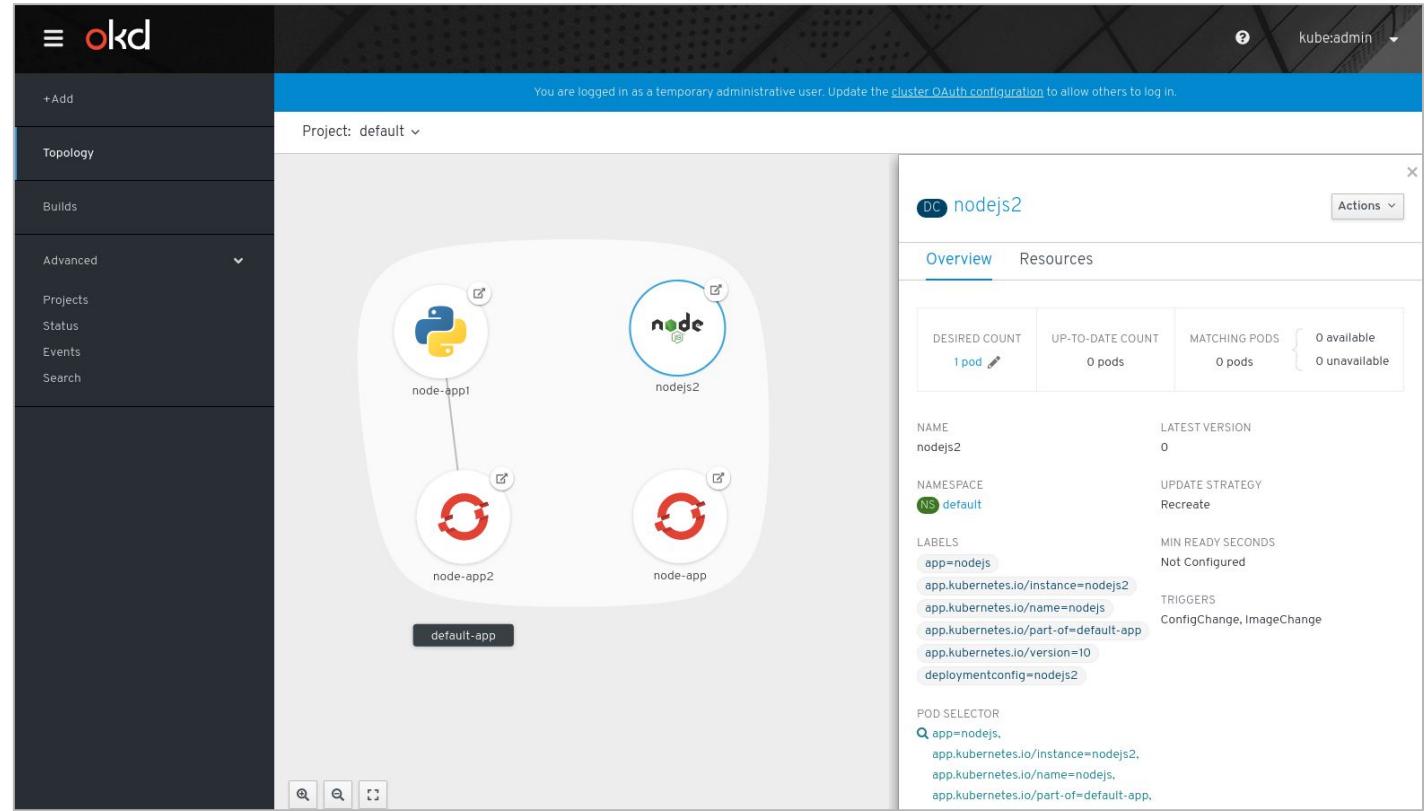


LEARN FROM RED HAT'S
EXPERIENCE TO INFORM
YOUR OWN DECISIONS

OpenShift Developer Console

Provide developers with an application-centric UI that enables them to quickly import code, create containers, link services and build their projects.

Will leverage OpenShift Pipelines for the CI/CD, and use Istio and Kiali project to provide a graphical view of container interactions for an application.



Use It To: Share an application- and code-centric UI with your development teams.



BUILD SOFTWARE THE RED HAT WAY IN OPEN INNOVATION LABS



EXPERIMENT
Rapidly build prototypes,
do DevOps, and be agile.

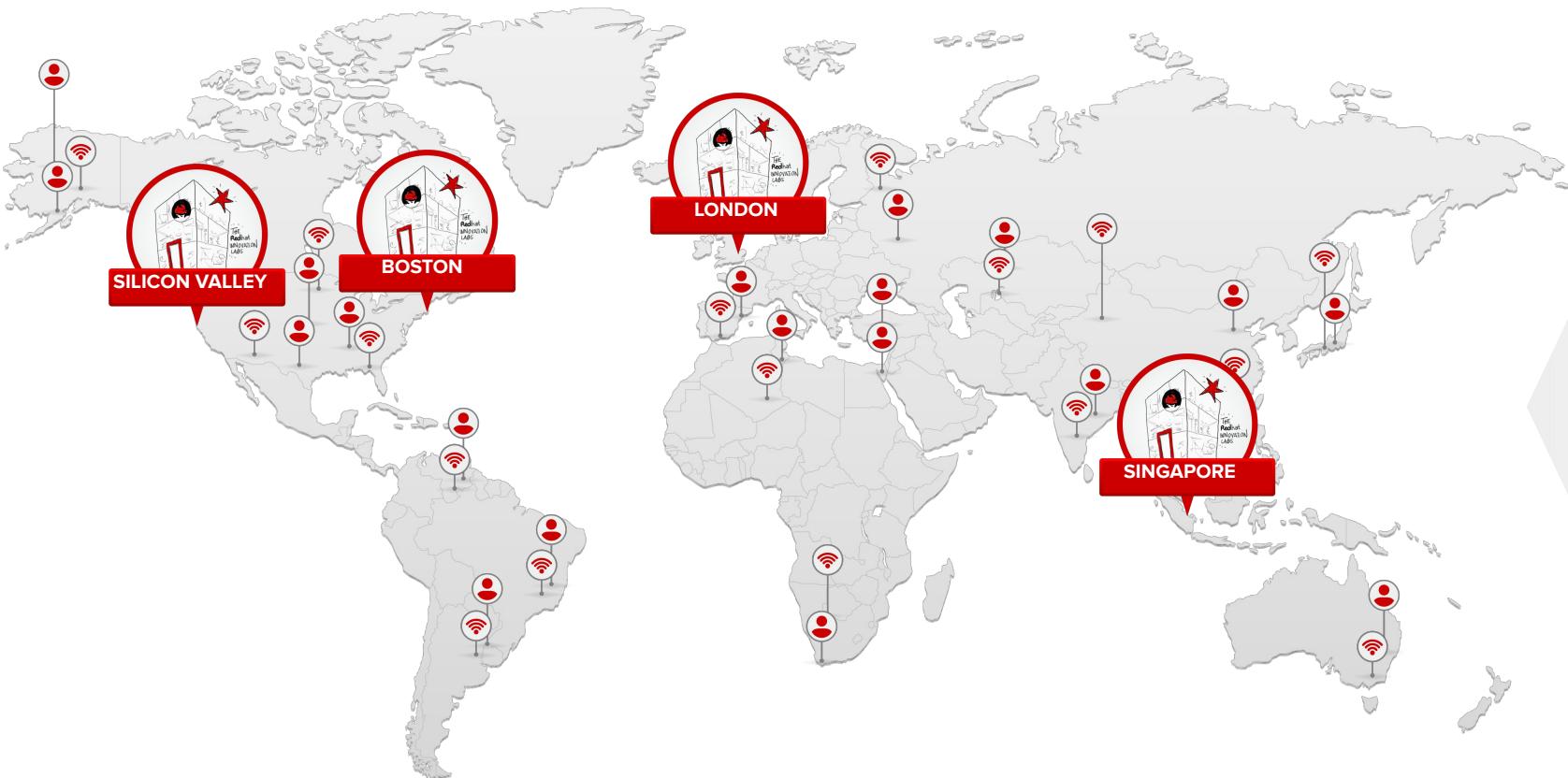


IMMERSE YOUR TEAM
Work side-by-side with experts
in a residency-style engagement.



CATALYZE INNOVATION
Bring modern application
development back to your
team.

DRIVE A CULTURE OF INNOVATION THROUGH A SPACE THAT FOSTERS COLLABORATION

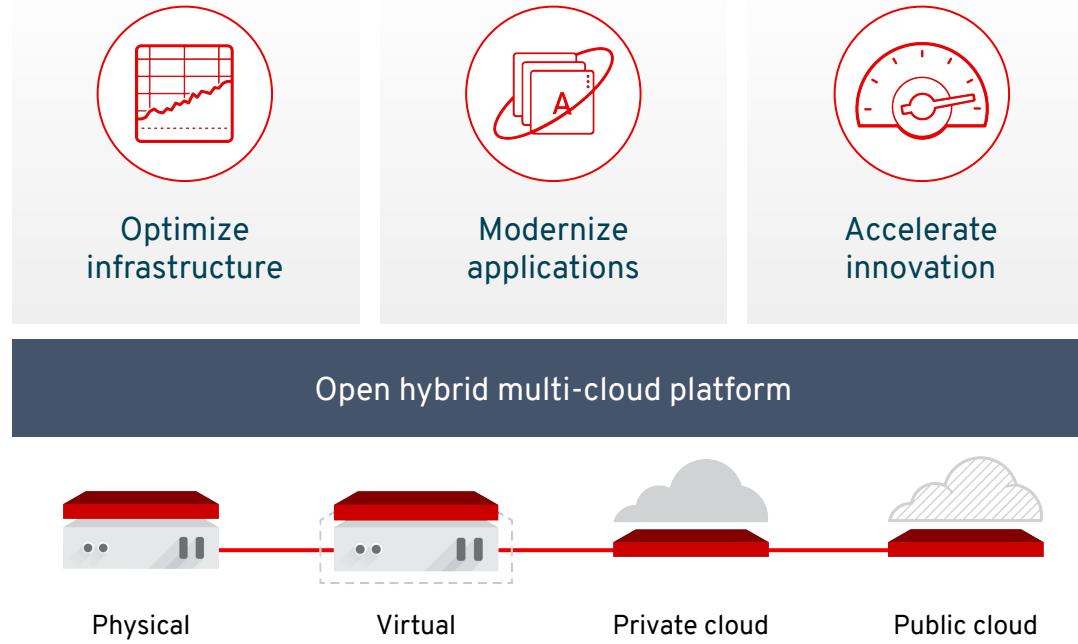


INNOVATE ANYWHERE

- Purpose-driven
- Collaborate and make
- Network and share
- Flex and adapt
- Rejuvenate and connect

Red Hat modernization and migration solutions

Innovation labs | Analytics service | Migration tooling | Open practice library



Establish platforms for the future

Accelerate adoption of next generation technologies

Reduce cost of existing applications and infrastructure

Transform development and operations practices and culture



DEVELOPER TRACK MODULES

1 OPTIMIZING EXISTING APPLICATIONS

Migrate an existing monolithic Java application from a legacy platform to Red Hat.

Modernize by incrementally refactoring to microservices architecture and modern Java platform

2 DEBUGGING, MONITORING AND CONTINUOUS DELIVERY

Debug, instrument and monitor a modern microservice application.

Deploy continuously using Pipelines

3 CONTROL CLOUD NATIVE APPS WITH SERVICE MESH

Gain a deep understanding of app behavior through service mesh instrumentation and visualization

4 ADVANCED CLOUD NATIVE WITH EVENT-DRIVEN SERVERLESS

Dynamically respond to events and scale applications using powerful Kubernetes constructs

OpenShift Concepts

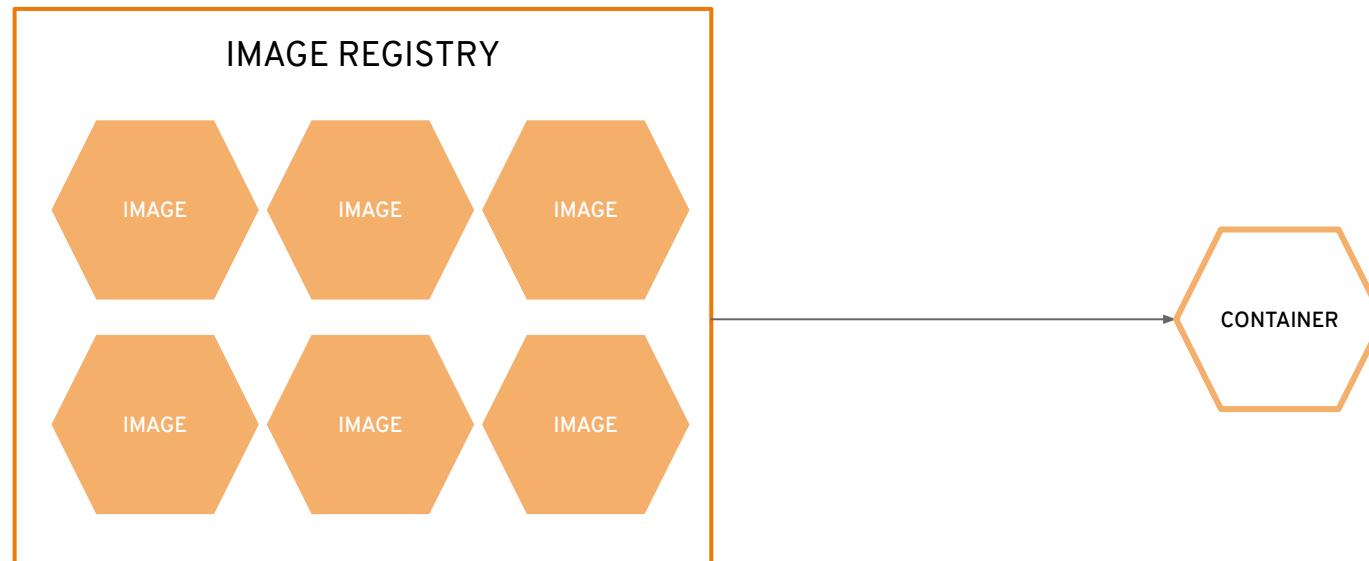
a container is the smallest compute unit



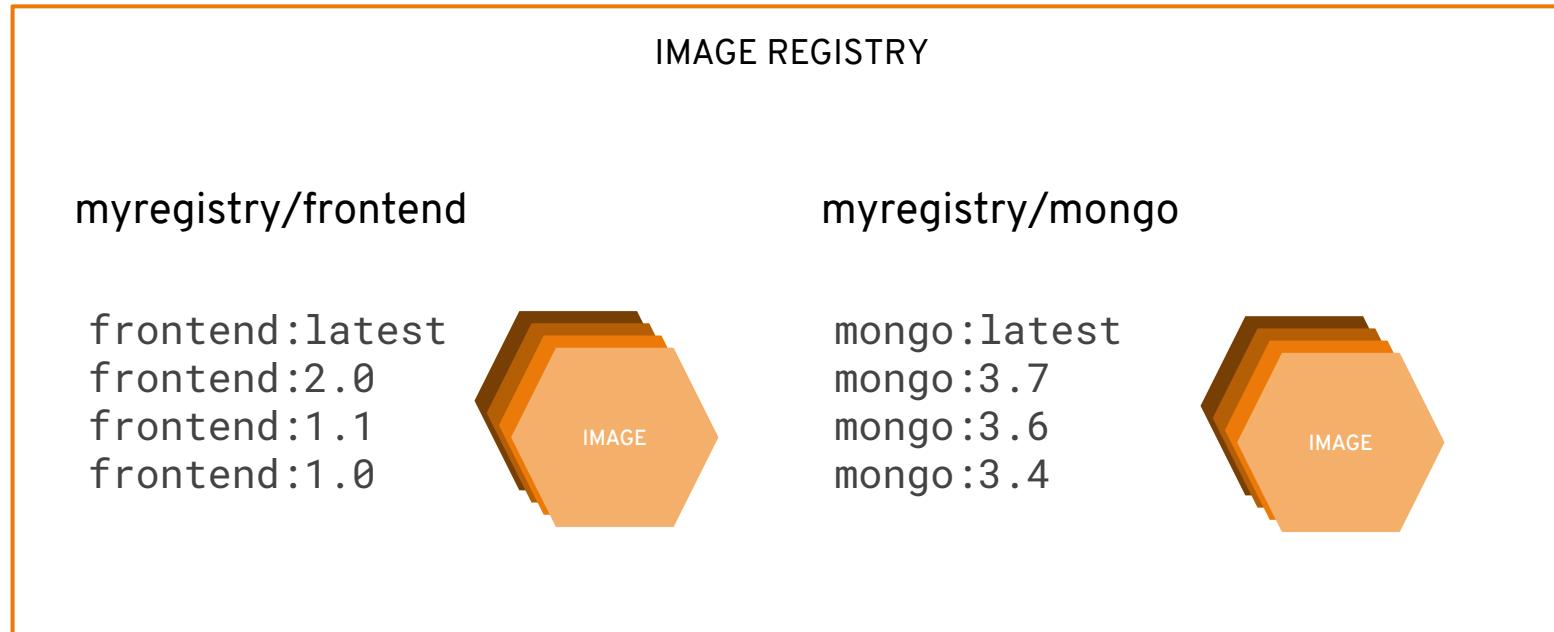
containers are created from container images



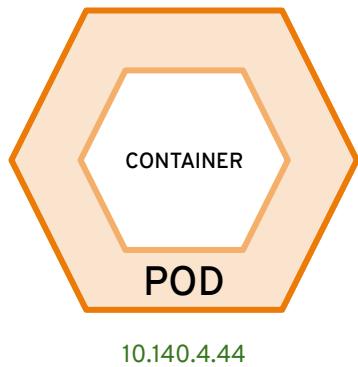
container images are stored in an image registry



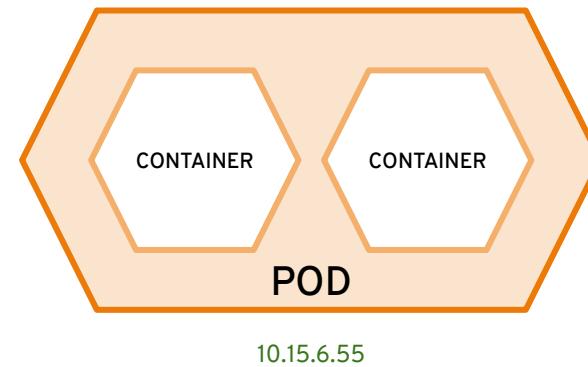
an image repository contains all versions of an image in the image registry



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

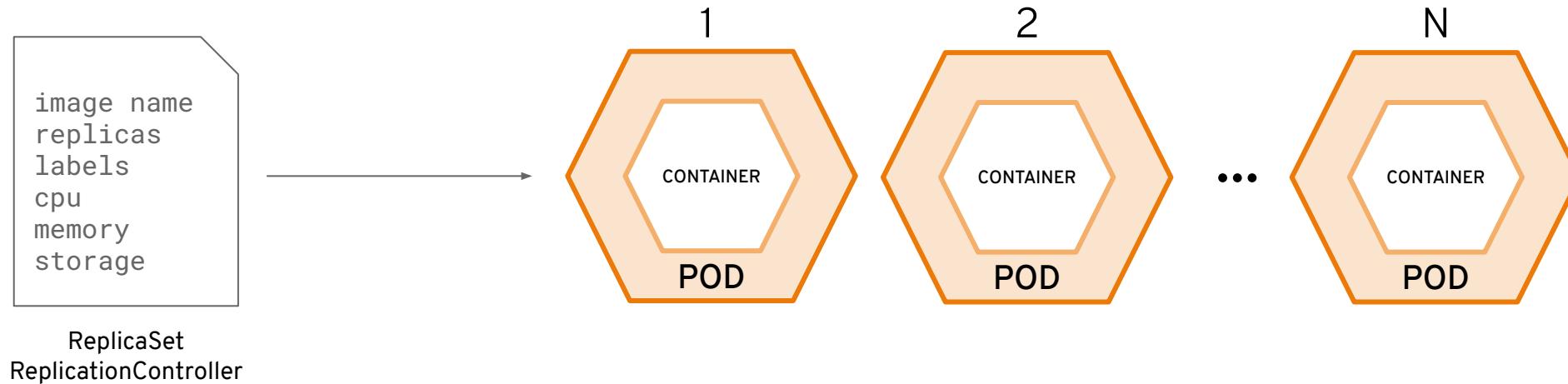


10.140.4.44

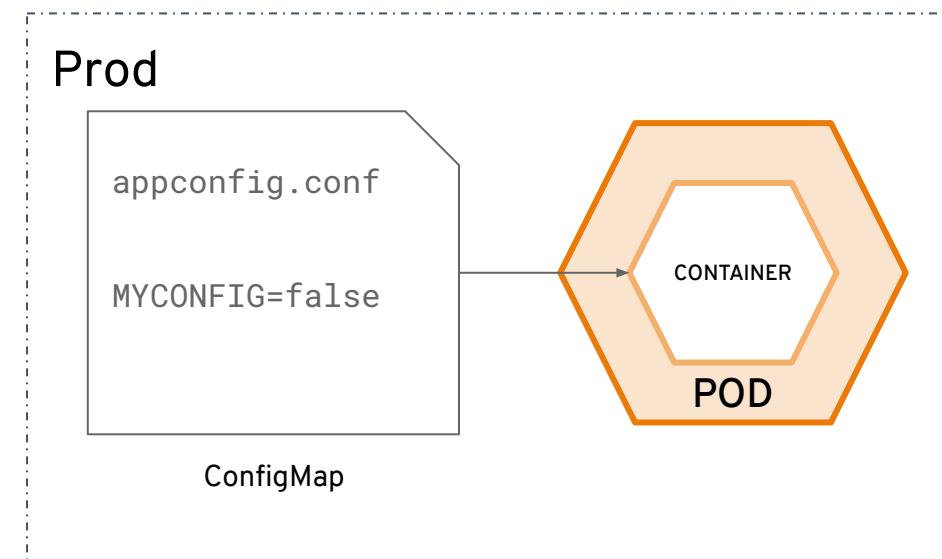
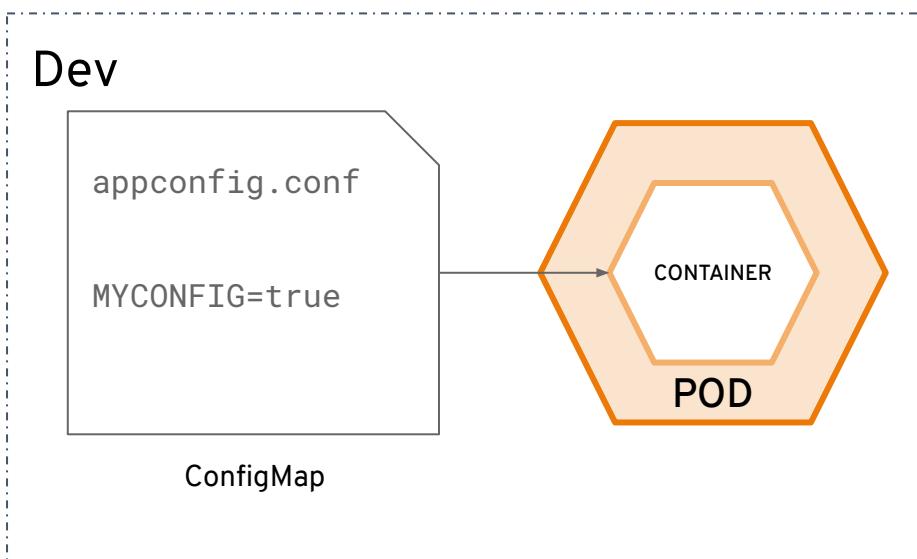


10.15.6.55

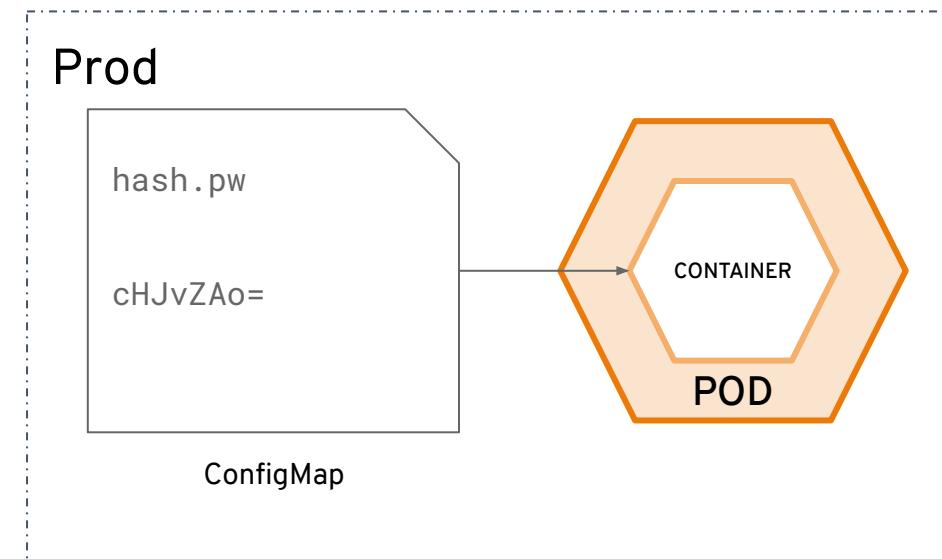
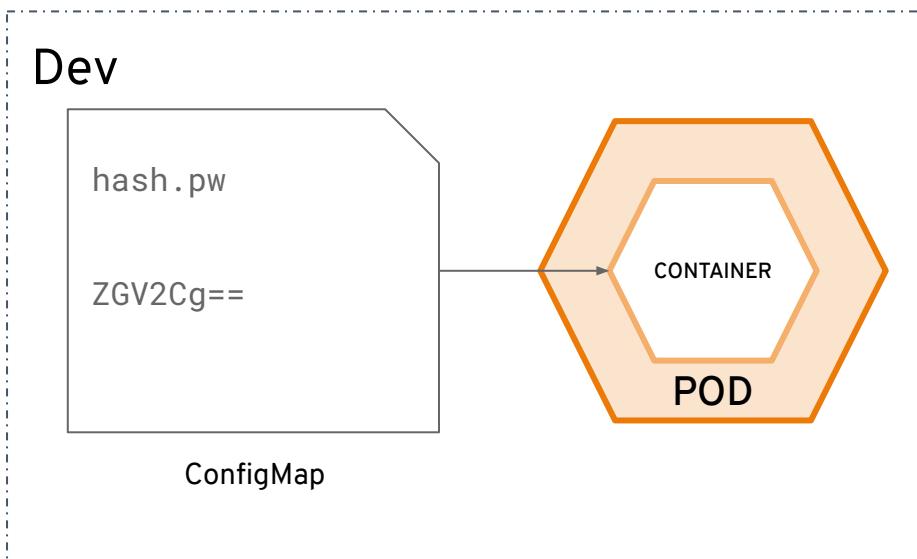
ReplicationControllers & ReplicaSets ensure a specified number of pods are running at any given time



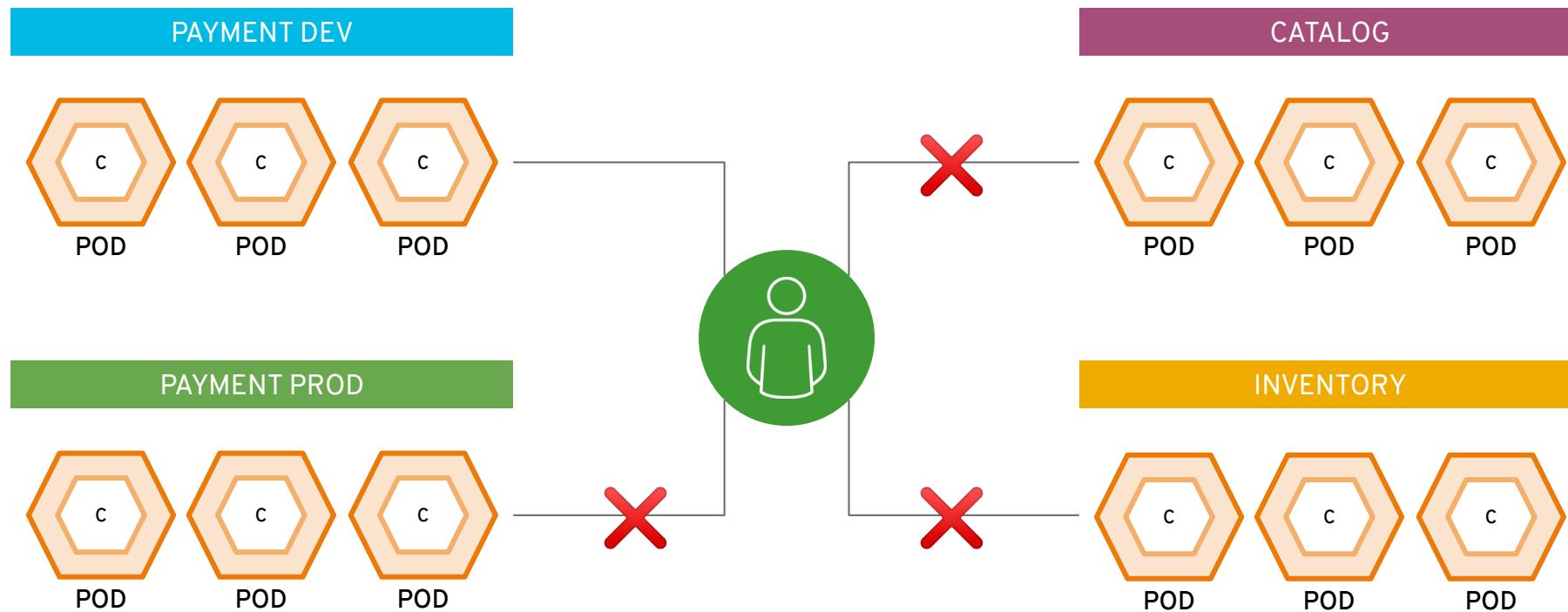
configmaps allow you to decouple configuration artifacts from image content



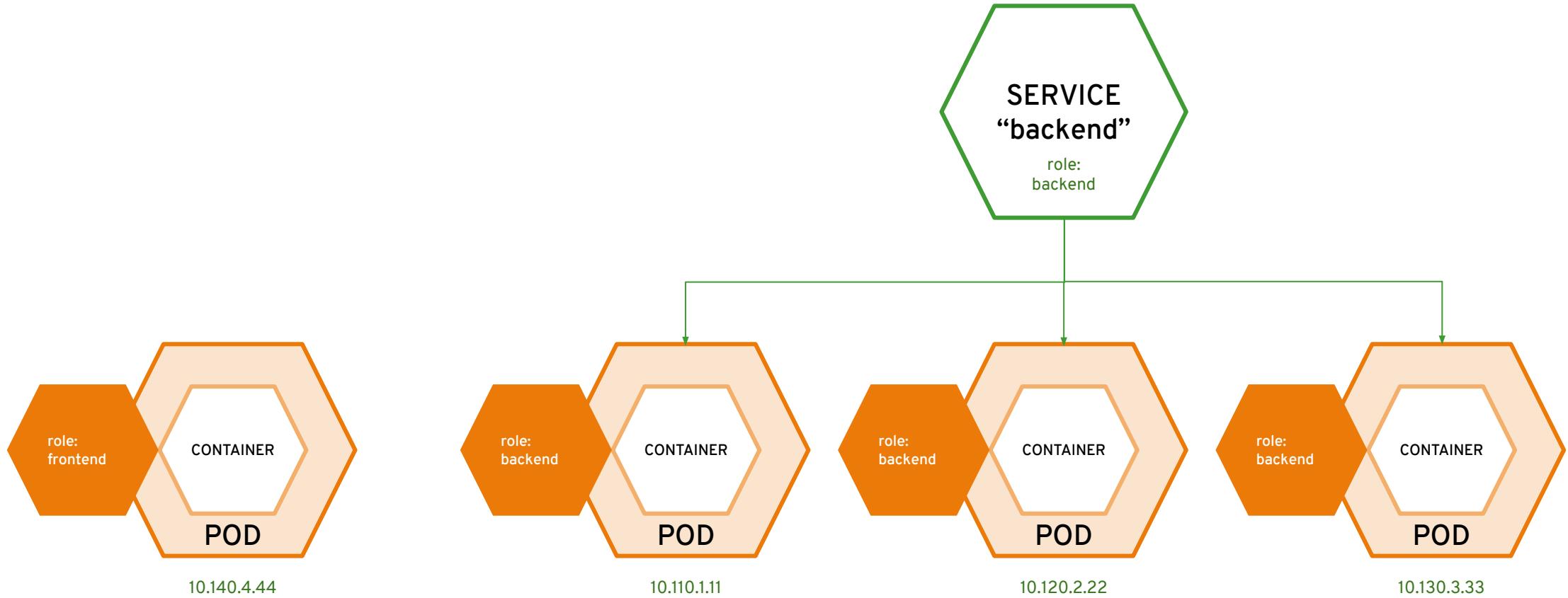
secrets provide a mechanism to hold sensitive information such as passwords



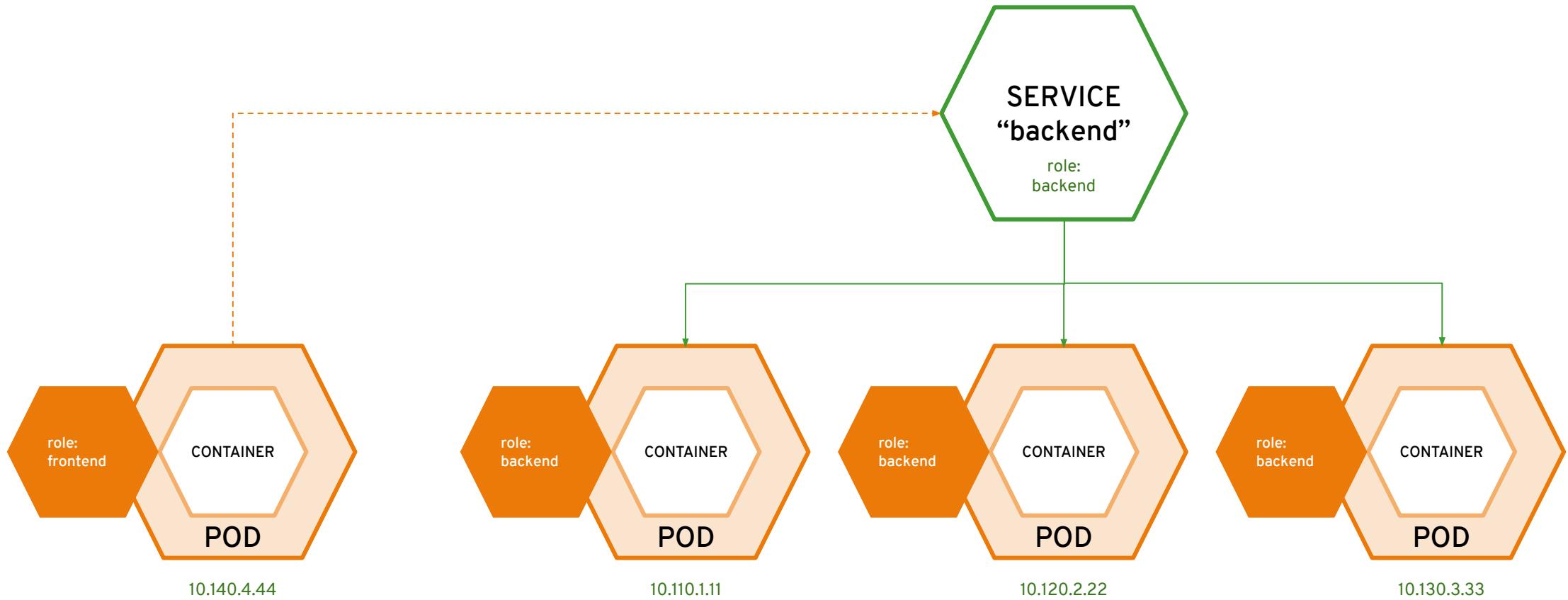
projects isolate apps across environments,
teams, groups and departments



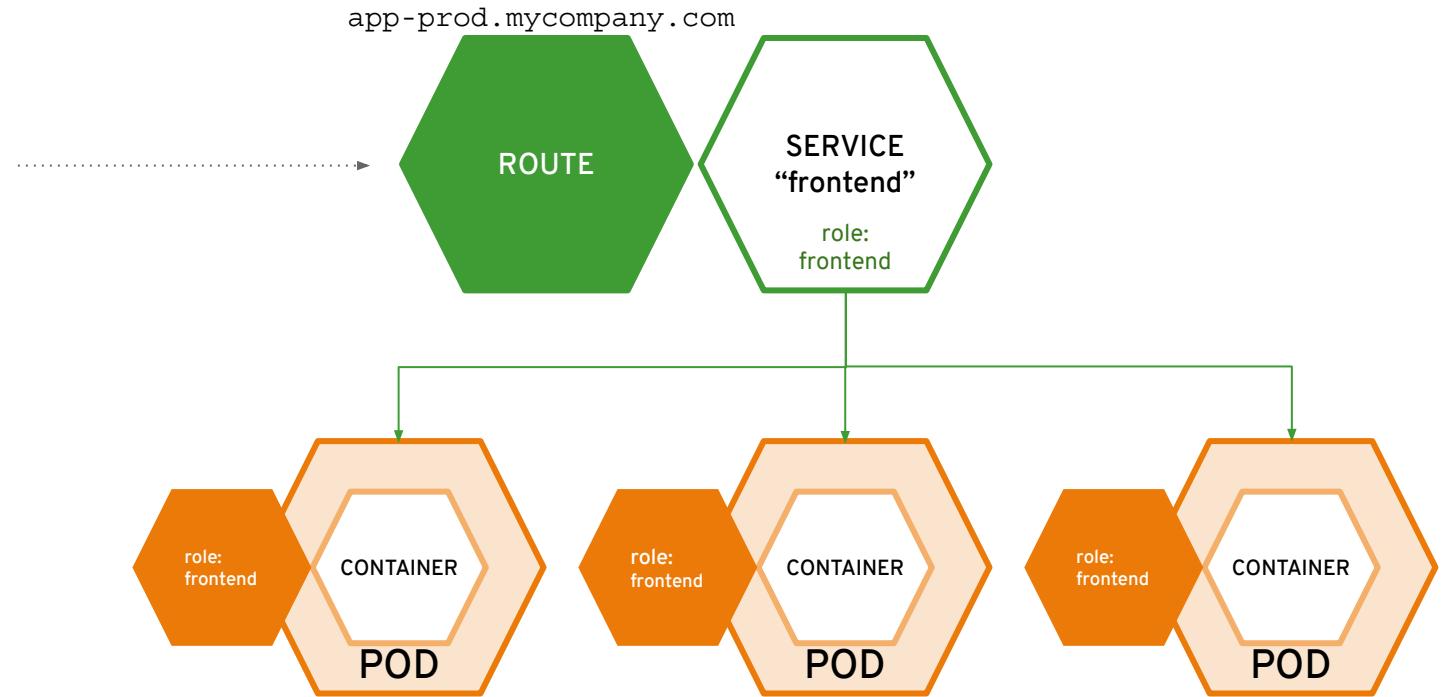
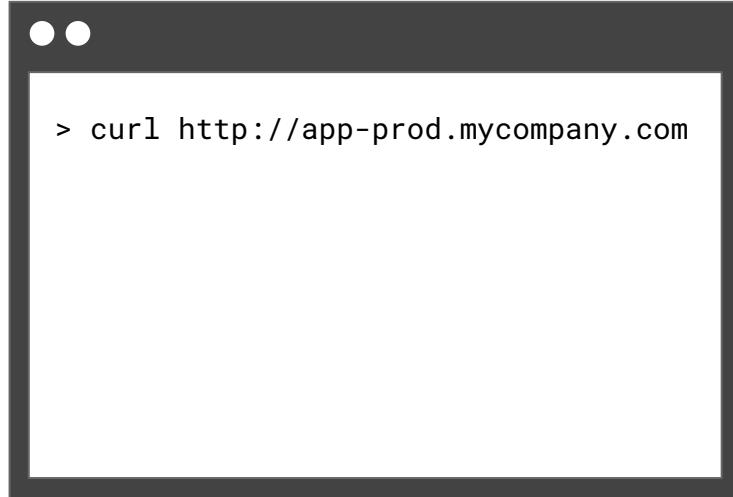
services provide internal load-balancing and service discovery across pods



apps can talk to each other via services



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



OpenShift Features

Enabling Agile Development

Trusted enterprise Kubernetes

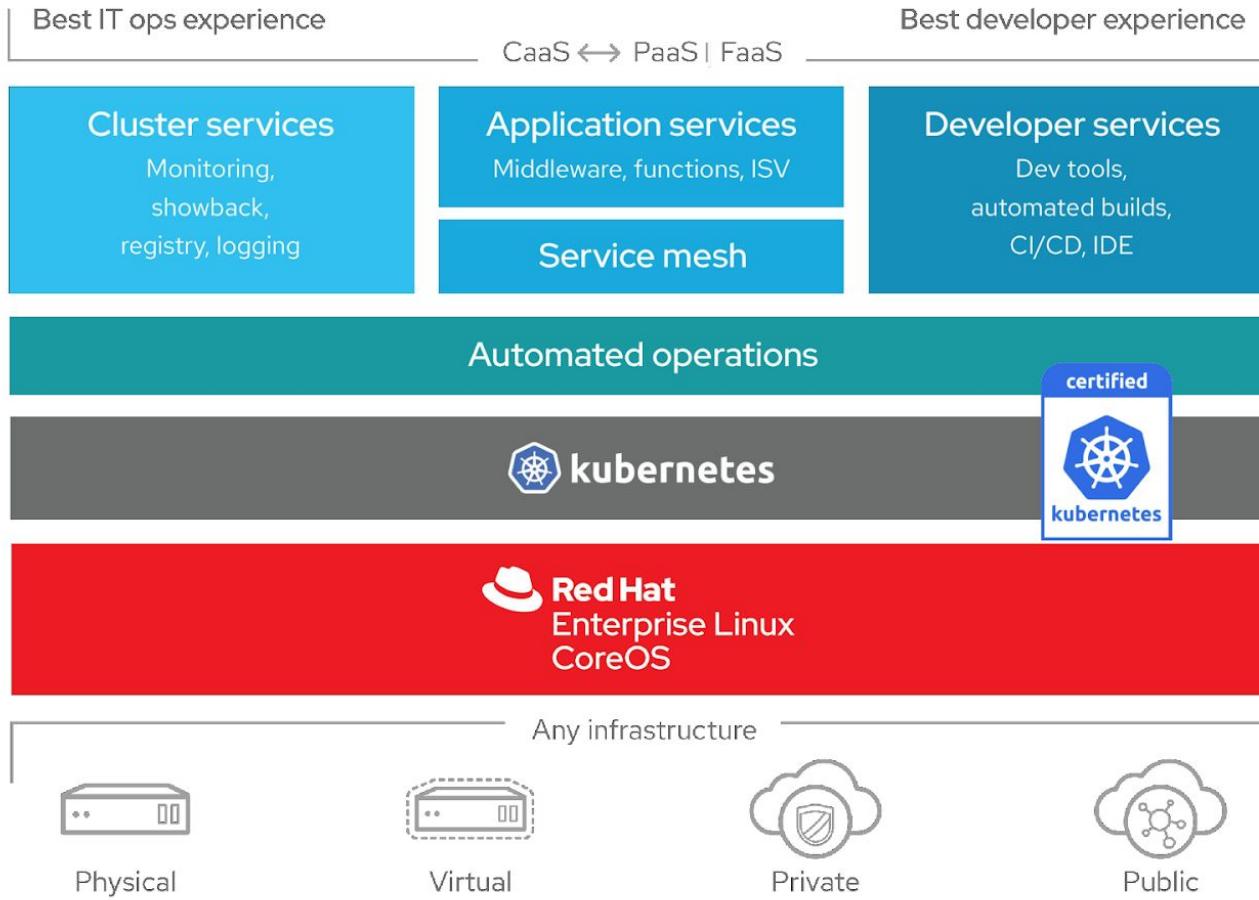


Trusted host, content,
platform

Full-stack automated
installation

Seamless updates

OpenShift 4 - A smarter Kubernetes platform



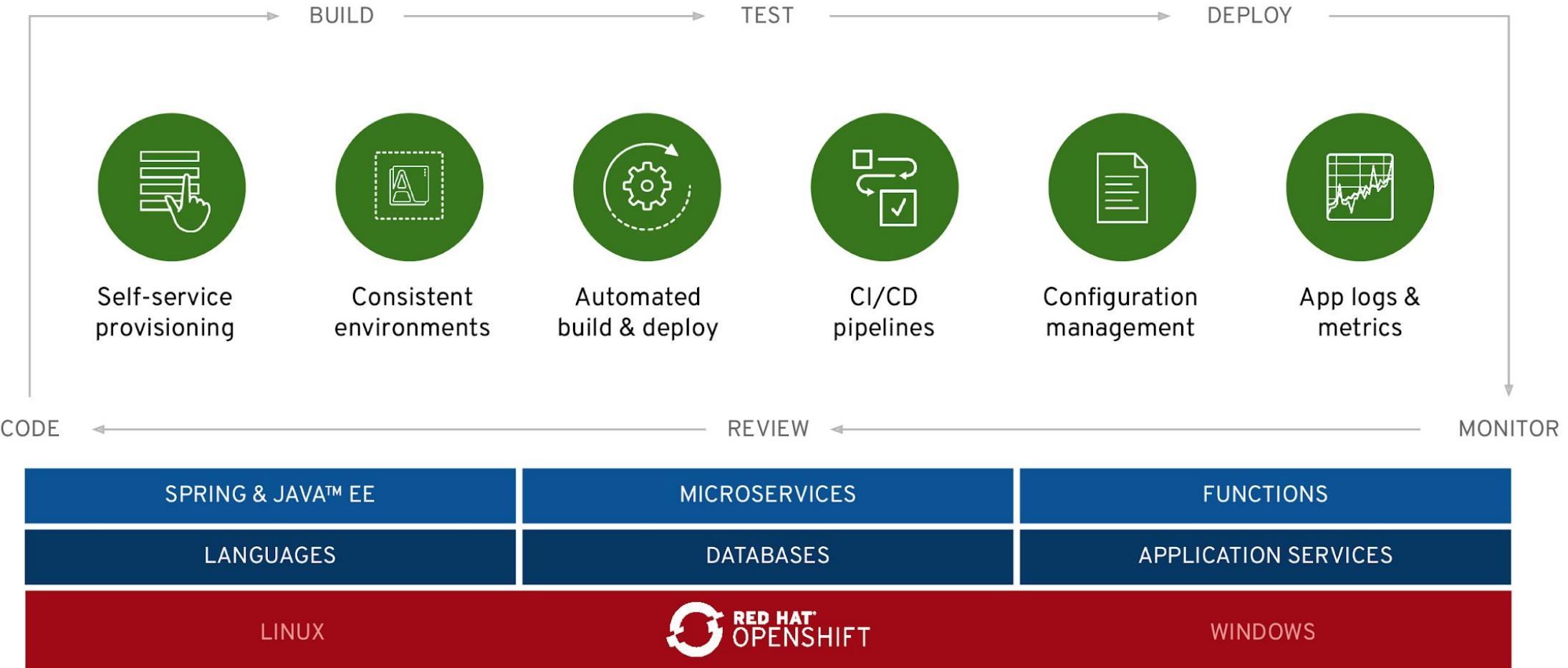
Automated, full-stack installation from the container host to application services

Seamless Kubernetes deployment to any cloud or on-premises environment

Autoscaling of cloud resources

One-click updates for platform, services, and applications

OpenShift enables developer productivity



Building next-gen applications

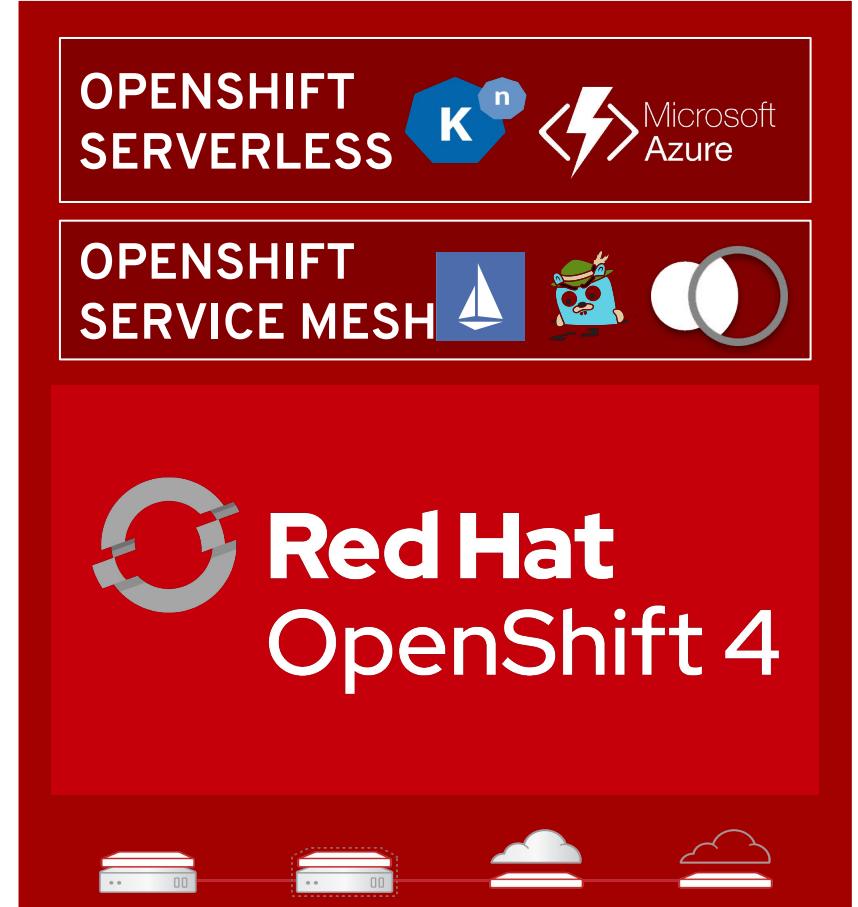
OpenShift Service Mesh

- Integrated Service Mesh for enhanced security and network segmentation of microservices applications. Combines Istio, Kiali (UI), and Jaeger (Tracing) projects.

August 25th!

OpenShift Serverless

- Integrated serverless, enabling scale-to-zero FaaS services and event sources - built on the Knative framework.
- Support for Azure Functions
- Integrated with Camel-k for rich set of initial event sources: HTTP, Kafka, AMQP



Why customers choose Red Hat OpenShift

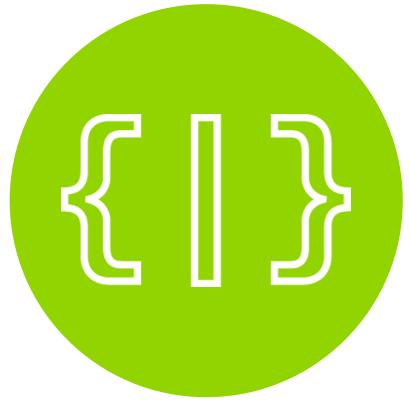
The infographic is organized into three main sections, each enclosed in a dotted box:

- Trusted enterprise Kubernetes**: Features a diagram of a server rack with a network of nodes connected to it.
- Cloud-like experience everywhere**: Features the Red Hat OpenShift logo (a red circle with a white gear) above two clouds, one containing a network icon and the other a shield icon.
- Empowering developers to innovate**: Features logos for Spring, Node.js, Docker, and Kafka, along with a smartphone displaying various app icons.

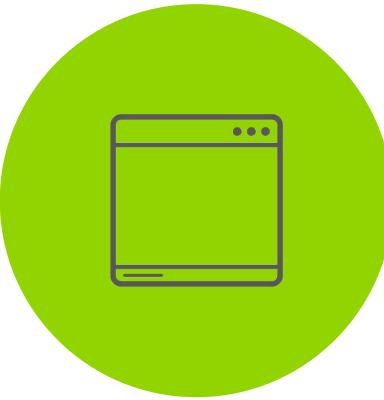
Below these sections is a horizontal row of open source technology logos:

- Shipyard (blue hexagon)
- Sailboat (blue sailboat)
- Flame (red flame)
- Lightning bolt (red lightning bolt)
- Ceph (red circle with a white ring)
- CoreOS (blue circle with a white triangle)
- etcd (blue hexagon with a gear icon)
- Snowflake (blue hexagon with a snowflake icon)

The bottom section is a dark teal bar with the text "Open source innovation".



**DEPLOY YOUR
SOURCE CODE**

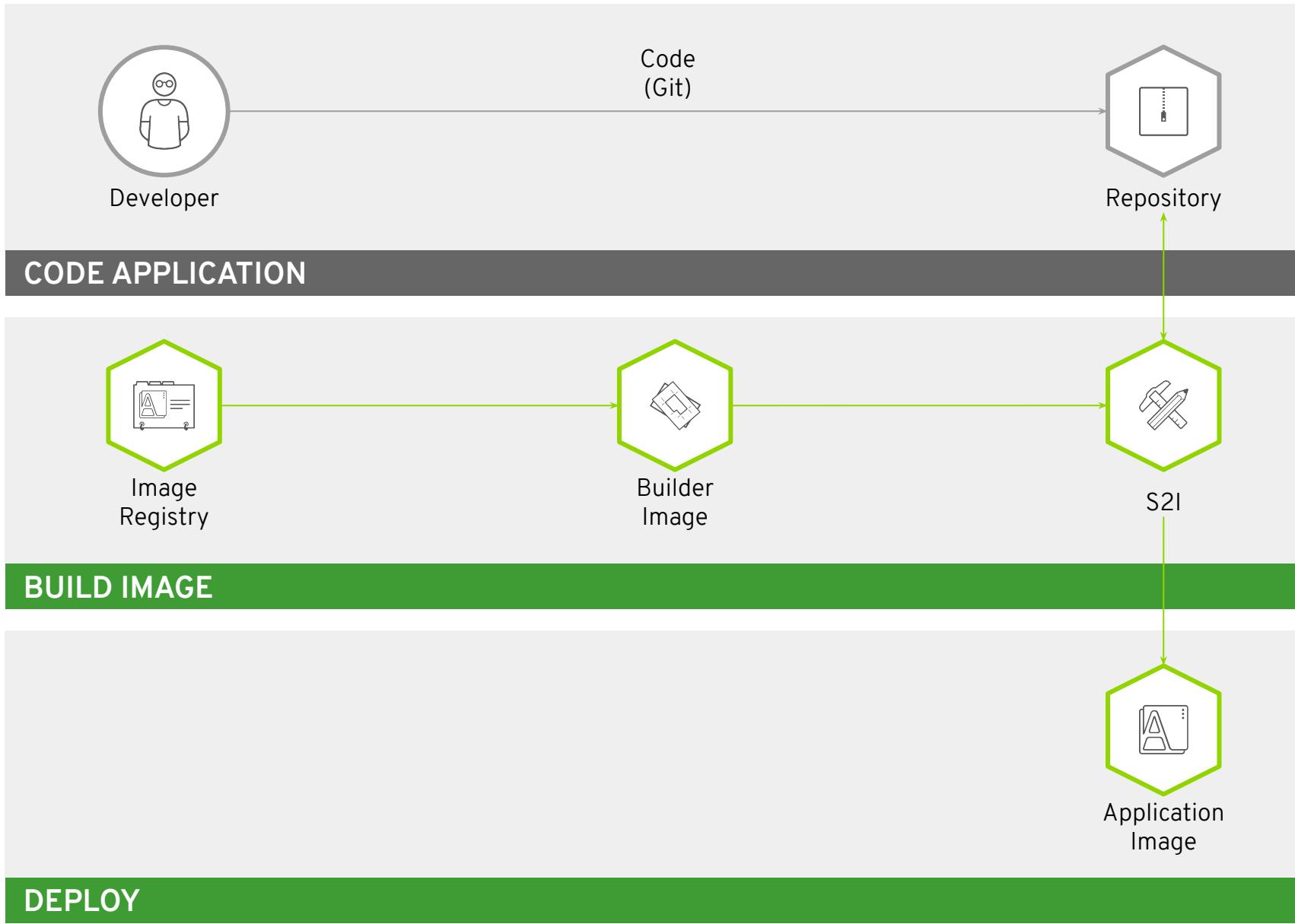


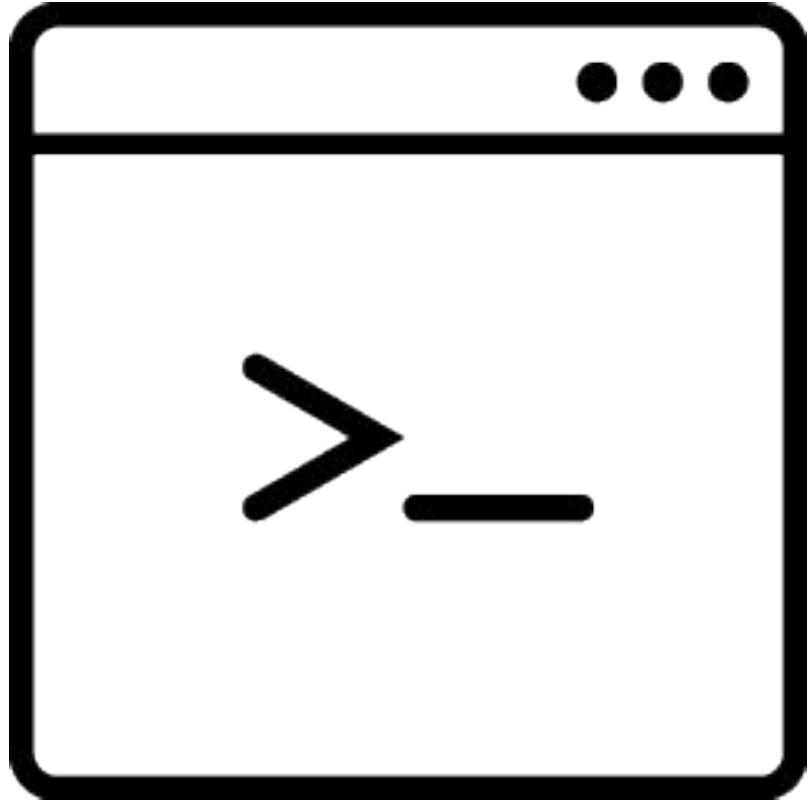
**DEPLOY YOUR
APP BINARY**



**DEPLOY YOUR
CONTAINER IMAGE**

BUILD AND DEPLOY | Source-to-Image (S2I) for building and deploying from code





**OPENSHIFT UI BRINGS
KUBERNETES POWER
TO YOUR FINGERTIPS**



**APPLICATION METRICS
AND LOGS BRING CLARITY
ABOUT APP HEALTH**

Module 2: Debugging, Monitoring and Continuous Delivery

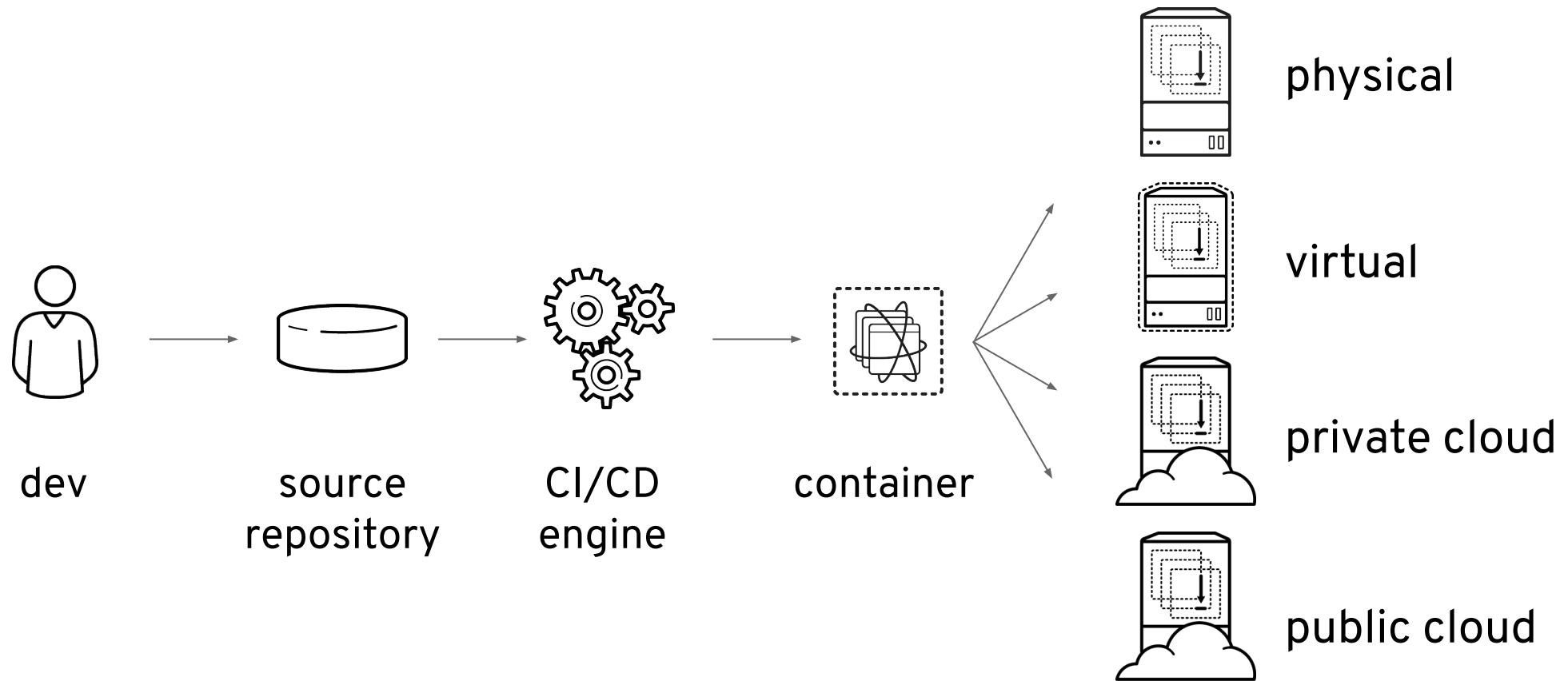
- 1. Optimizing Existing Applications
- 2. Debugging, Monitoring, and Continuous Delivery
- 3. Control Cloud Native Apps with Service Mesh
- 4. Advanced Cloud Native with Event-Driven Serverless

THE PATH TO CLOUD-NATIVE APPS

A DIGITAL DARWINISM



DEPLOYMENT PIPELINES



OPENSHIFT PIPELINES

Jenkins is still the most used CI/CD platform in enterprises and can be used from inside OpenShift.

An intuitive pipeline visualization makes it simple for users to see how builds are progressing.

The full Jenkins UI is also available.

The screenshot shows the OpenShift Pipeline interface. On the left, a sidebar menu includes categories like Workloads, Networking, Storage, Builds (selected), Build Configs, Image Streams, Monitoring, Compute, and Administration. The main area displays a Jenkins pipeline named "tasks-pipeline-1" under the namespace "cicd-smx". The pipeline consists of six stages: Build App, Test, Code Analysis, Archive App, Build Image, and Deploy DEV. Each stage has a green checkmark icon and a timestamp indicating its completion time. A "Build 1" section shows a "Promote to ST..." button with a yellow warning icon and an "Input Required" message. Below the pipeline diagram, detailed information is provided: NAME: tasks-pipeline-1, STATUS: Running, and TYPE: JenkinsPipeline.

NAME	STATUS
tasks-pipeline-1	Running

NAMESPACE	TYPE
NS cicd-smx	JenkinsPipeline

OPENShift PIPELINES

- CI/CD workflow via Jenkins
- Pipelines are started, monitored, and managed similar to other builds
- Auto-provisioning of Jenkins server
- On-demand Jenkins slaves
- Embedded Jenkinsfile or in Git repo

```
pipeline {
    agent {
        label 'maven'
    }
    stages {
        stage('build app') {
            steps {
                git url:
'https://git/app.git'
                sh "mvn package"
            }
        }
        stage('build image') {
            steps {
                script {
                    openshift.withCluster() {
                        openshift.startBuild("...")
                    }
                }
            }
        }
    }
}
```

OPENSHIFT PIPELINES CI/CD PLATFORM

Next-gen Kubernetes CI/CD pipeline that works for containers (including serverless).

Based on the **Tekton** project (which was spun out of the Knative Pipelines project) started by Google, Red Hat and others.

The screenshot shows the Red Hat OpenShift Pipelines interface. On the left, there's a sidebar with 'XYZ Name' and tabs for 'Builds' (selected) and 'Pipelines'. The main area shows a table of pipelines:

NAME	STATUS	STARTED	DURATION	TRIGGER
aa-build-3	Running	10 mins ago	2 min 04 sec	Commit #123456ABC

Below the table, the pipeline 'aa-build-3' is shown as a sequence of steps:

```
graph LR; A((input info)) --> B[Steps - (3/3)  
build-name (30s)]; B --> C[Steps - (3/8)  
Test-st... (6s)]; C --> D[Steps - (2/5)  
Code a... (13s)]; D --> E[Steps - (1/3)  
Security... (20s)]; E --> F[Steps - (0/2)  
Image b... (0s)]; F --> G[Steps - (0/7)  
DeployTo... (0s)];
```

A detailed log window at the bottom displays the execution of the pipeline steps:

```
Downloading six-1.11.0-py2.py3-none-any.whl  
Building wheels for collected packages: tornado, configparser  
Running setup.py bdist_wheel for tornado: started  
Running setup.py bdist_wheel for tornado: finished with status 'done'  
Stored in directory: /root/.cache/pip/wheels/0c/21/02/8cd6a381450df92b449ea7c57be653dd7aa80ba42c716212c  
Running setup.py bdist_wheel for configparser: started  
Running setup.py bdist_wheel for configparser: finished with status 'done'  
Stored in directory: /root/.cache/pip/wheels/1c/bd/b4/277af3f6c40645661b4cd1c21df26aca0f2e1e9714a1d4cda8  
Successfully built tornado configparser  
Installing collected packages: six, singledispatch, certifi, backports-abc, tornado, enum34, configparser, mccabe, pyflakes, pycodestyle, flake8  
Found existing installation: six 1.8.0  
Uninstalling six-1.8.0:  
Successfully uninstalled six-1.8.0  
Successfully installed backports-abc-0.5 certifi-2017.11.5 configparser-3.5.0 enum34-1.1.6 flake8-3.5.0 mccabe-0.6.1 pycodestyle-2.3.1 pyflakes-1.6.0  
singledispatch-3.4.0.3 six-1.11.0 tornado-4.5.3  
$ python -c 'print("Hello, world")'  
Hello, world  
Job succeeded
```

OpenShift Application Monitoring



Metrics collection and storage via Prometheus, an open-source monitoring system time series database.

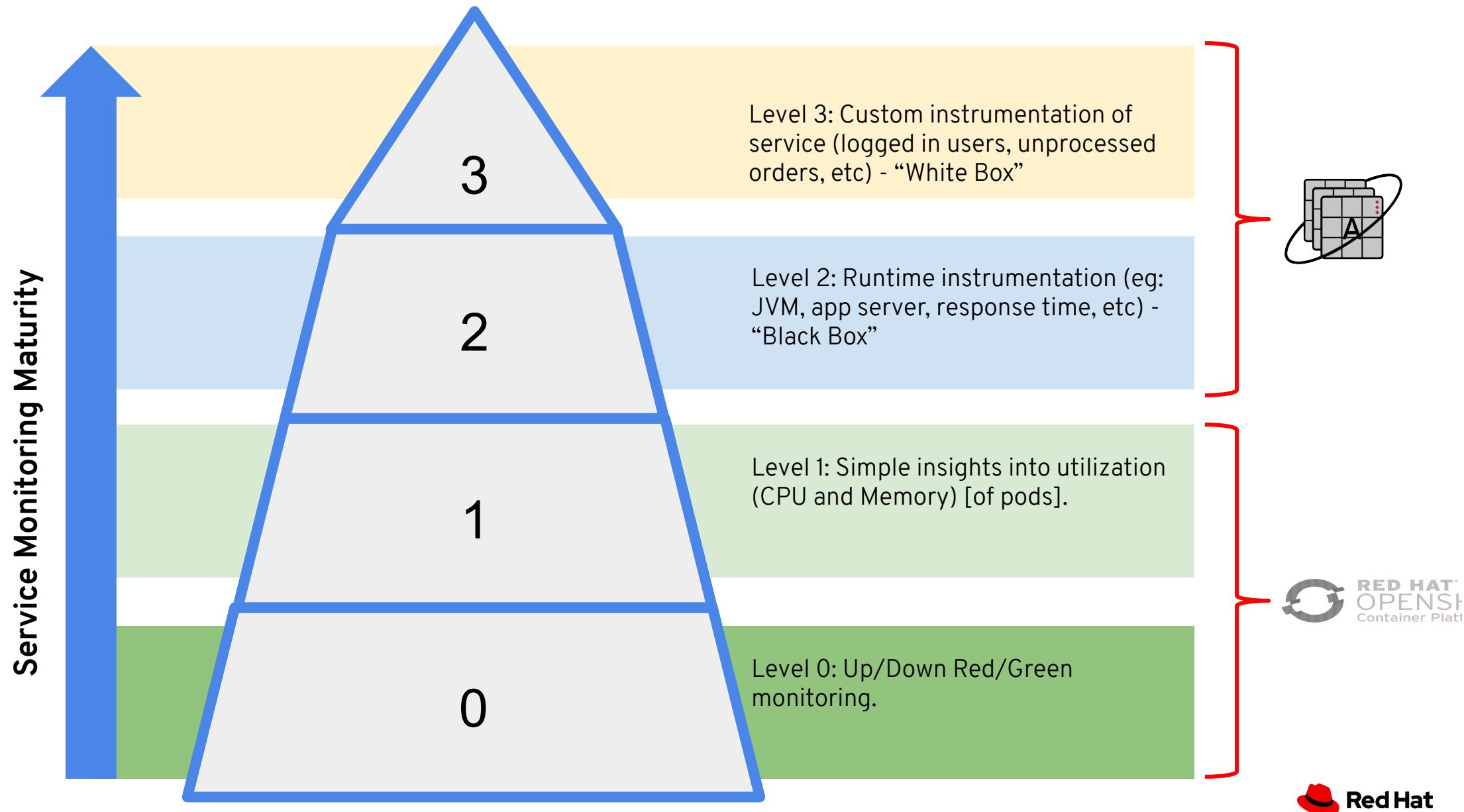


Alerting/notification via Prometheus' Alertmanager, an open-source tool that handles alerts sent by Prometheus.

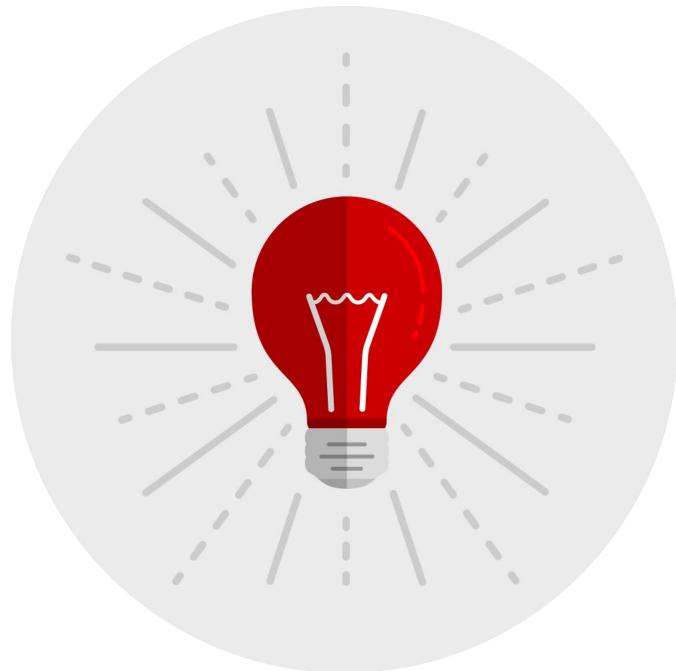


Metrics visualization via Grafana, the leading metrics visualization technology.

Service Monitoring Maturity Model



Middleware OOTB Instrumentation



Prometheus Endpoint

Runtimes (Thorntail, Node.js, vert.x, Quarkus, Spring Boot),
Decision & Process Servers.
Coming in EAP 7.3

Health Endpoint

EAP, Runtimes (Thorntail, Node.js, vert.x, Spring Boot,
Quarkus), Decision & Process Servers.

Distributed Tracing Client

EAP, Runtimes (Thorntail, Node.js, vert.x, Spring Boot,
Quarkus), Fuse.

JSON logging format

EAP, Spring Boot, Decision & Process Servers

GOAL FOR MODULE

In this module you will learn:

- How to create a CI/CD pipeline using Jenkins to automate the deployment of updates to microservice applications
- How to use CodeReady Workspaces to debug Java apps
- How to create health probes to allow the platform to act on unhealthy apps
- How to use distributed tracing and Jaeger to gain insight into application behavior
- How to instrument and monitor application performance using Prometheus (metrics ingestion and alerting) and Grafana (visualization)

LAB INSTRUCTIONS

- **Everything is done in browser** - no local commands or installs needed on your laptop
- Tested with **Chrome 75.0.3770.142, Firefox 60.8.0esr**. → **Safari 12.x does not work!**
- If things get weird, just reload browser page
- **Turn off VPN** (we use websockets extensively), **pause AdBlock** for the lab domain (there are no ads)
- To recreate the lab locally, visit

github.com/RedHat-Middleware-Workshops/cloud-native-workshop-v2-infra

To get started:

<https://bit.ly/CCNm2-July2020>

Password: r3dh4t1!

If you get stuck, raise a hand!