ACCELERATE DEVOPS USING OPENSHIFT PAAS

Serge Pagop, Middleware Specialist
spagop@redhat.com

23.06.2015

THE WORLD WE LIVE IN TODAY

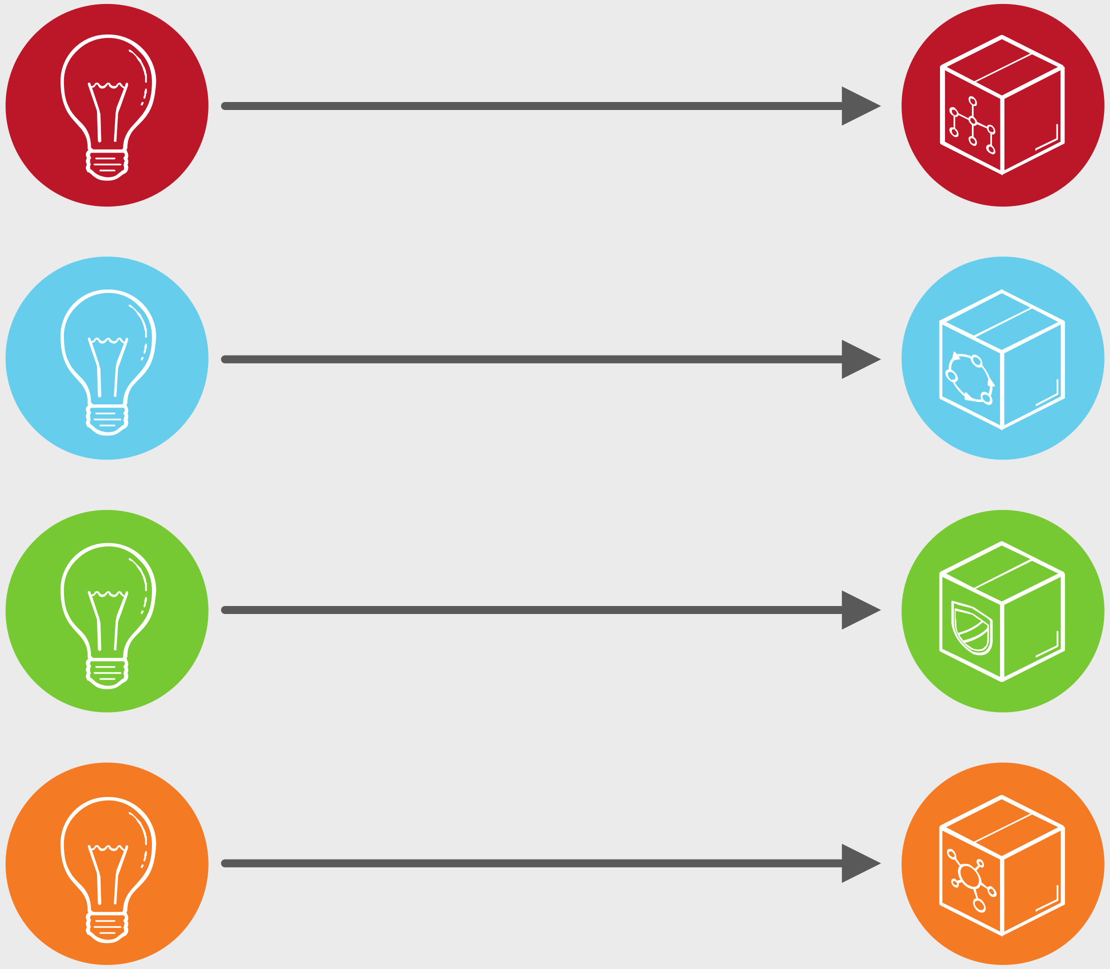
Customers and consumers

- Ubiquitous access to data and services
- Impatient, want everything NOW
- Increased QoS expectations

Businesses

- New opportunities and markets
- Threat of being disrupted, intense competition
- Small time frames to get products and services out

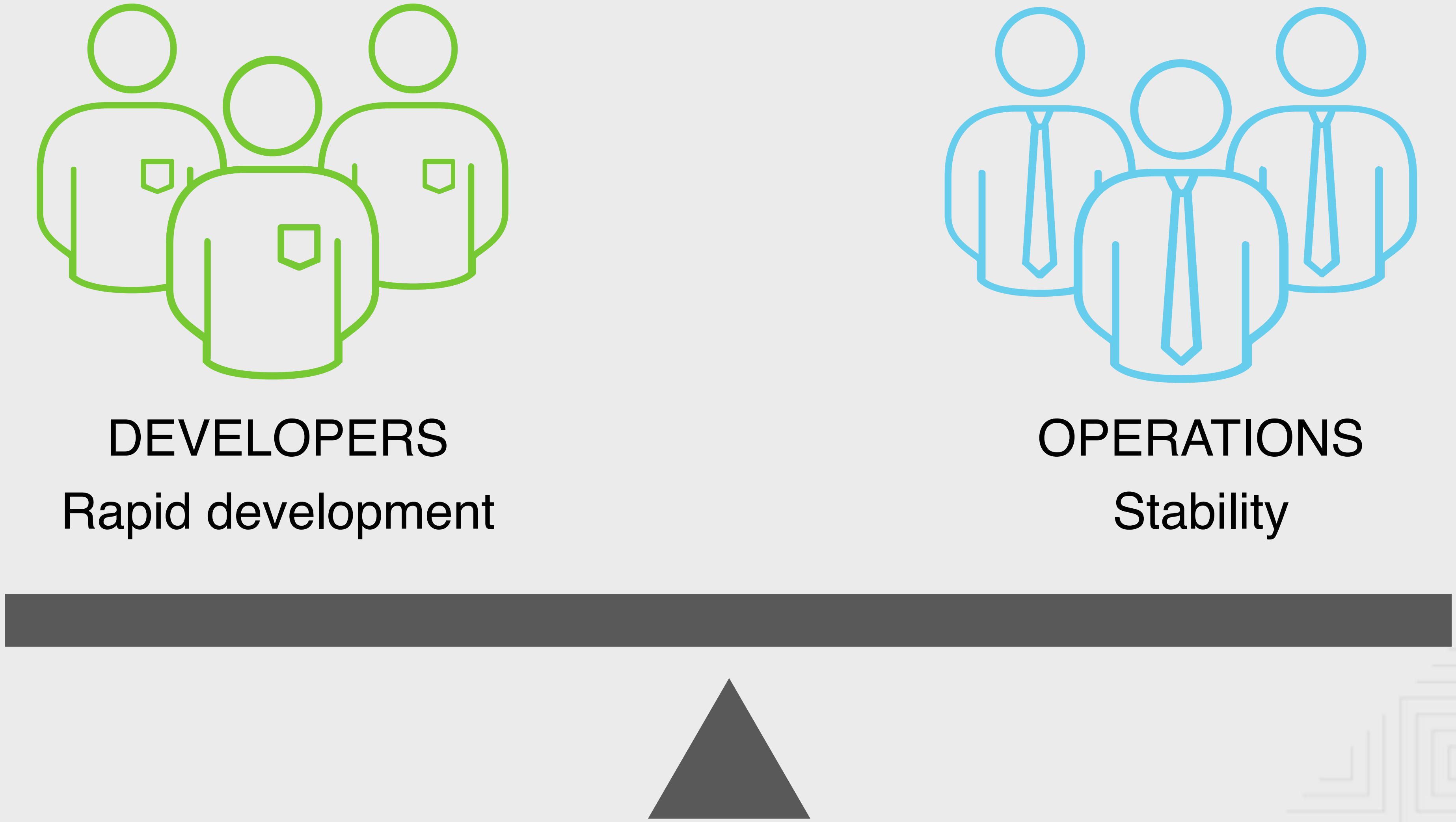
THE WORLD WE LIVE IN TODAY



Increased quality

Rapid delivery of product features and service

Doing more with less



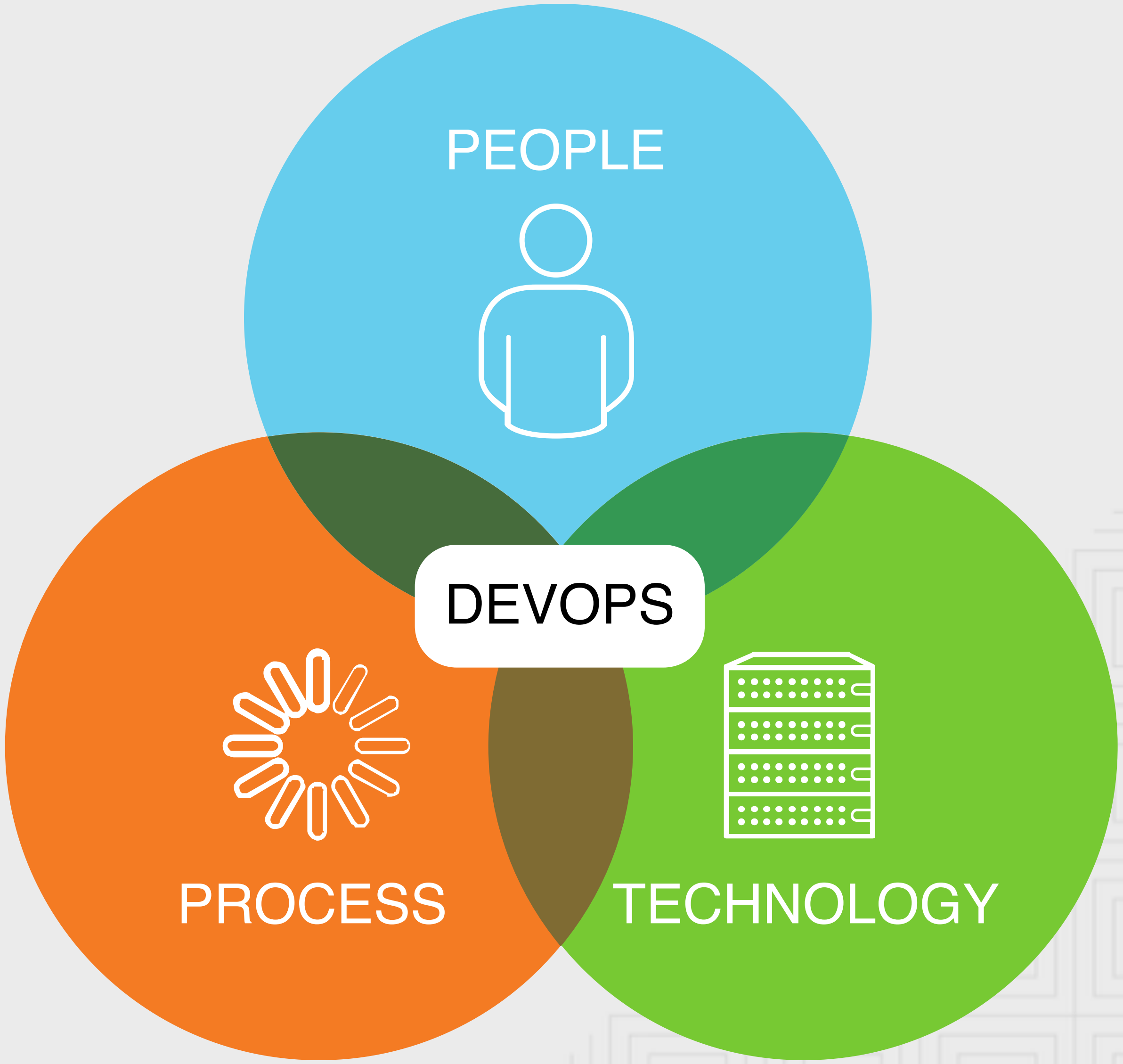
DEVELOPERS
Rapid development

OPERATIONS
Stability

BOTH TEAMS ARE THERE TO ENABLE THE BUSINESS

WHAT IS DEVOPS?

A methodology to deliver software more efficiently by emphasizing collaboration, communication, and integration across different teams (Dev, QA, Ops) in an IT organization.



TRADITIONAL SOFTWARE DELIVERY ENVIRONMENT

TYPICAL ASSUMPTIONS AND EXPECTATIONS

Ops teams are not required in application design discussions.

Software should never break.

Production environments are provisioned/ through a mostly manual process.

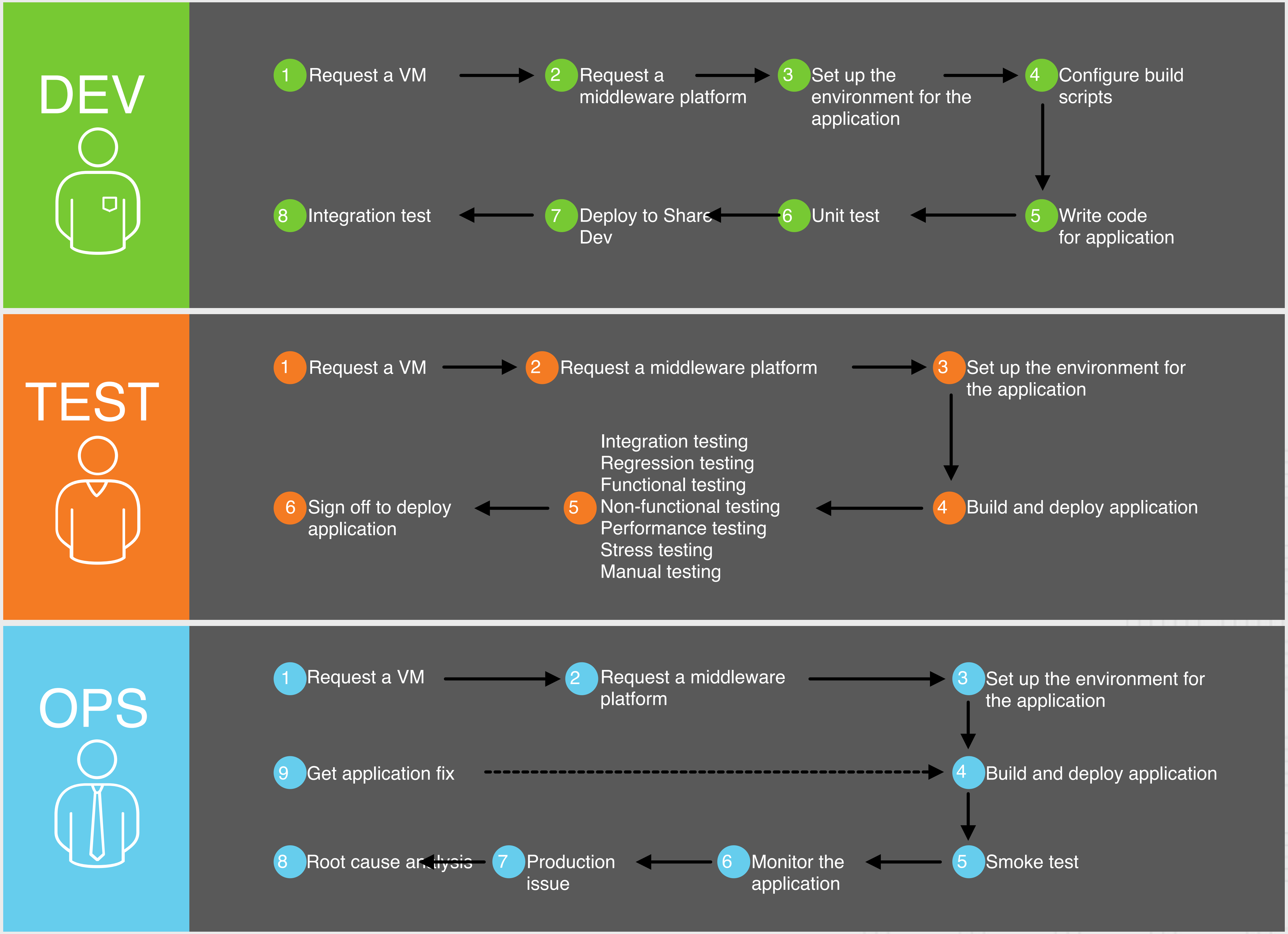
Developers should not have any access to the production environment.

You have to give a lot of lead time for getting an application environment.

An application is deployed to production after all development is complete.

Deployments are a headache—software is deployed using a mostly manual process.

We cannot keep deploying code to production on a regular basis.



DEV

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Configure build scripts

5 Write code for application

6 Unit test

7 Deploy to Shared Dev

8 Integration test

TEST

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Build and deploy application

5 Integration testing
Regression testing
Functional testing
Non-functional testing
Performance testing
Stress testing
Manual testing

6 Sign off to deploy application

OPS

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Build and deploy application

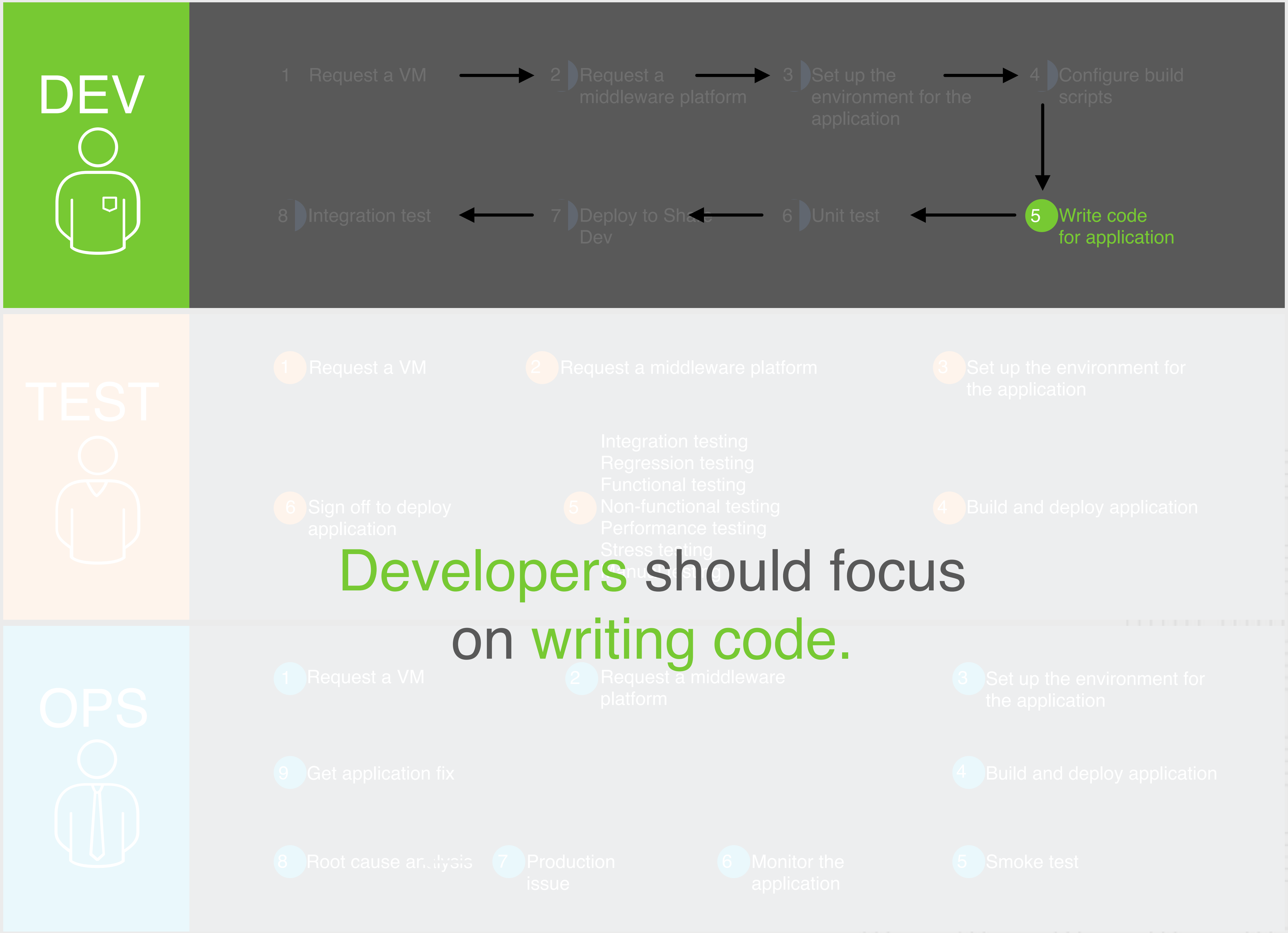
5 Smoke test

6 Monitor the application

7 Production issue

8 Root cause analysis

9 Get application fix



DEV

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Configure build scripts

5 Write code for application

6 Unit test

7 Deploy to Share Dev

8 Integration test

TEST

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Build and deploy application

5 Integration testing
Regression testing
Functional testing
Non-functional testing
Performance testing
Stress testing
Manual testing

6 Sign off to deploy application

OPS

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Build and deploy application

5 Smoke test

6 Monitor the application

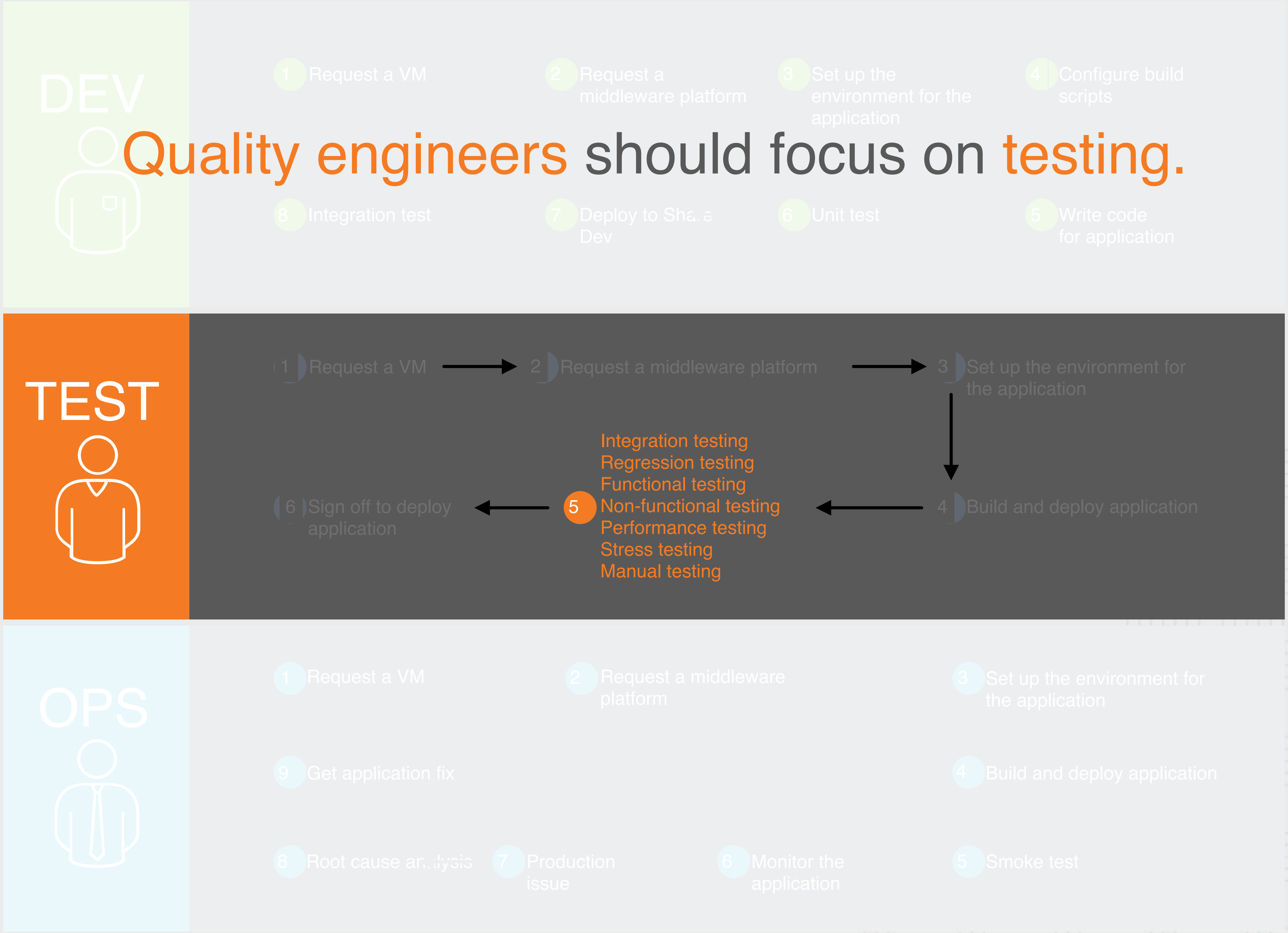
7 Production issue

8 Root cause analysis

9 Get application fix

Developers should focus on writing code.

Quality engineers should focus on testing.



DEV

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Configure build scripts

5 Write code for application

6 Unit test

7 Deploy to Shared Dev

8 Integration test

TEST

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Build and deploy application

5 Integration testing
Regression testing
Functional testing
Non-functional testing
Performance testing
Stress testing
Manual testing

6 Sign off to deploy application

OPS

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Build and deploy application

5 Smoke test

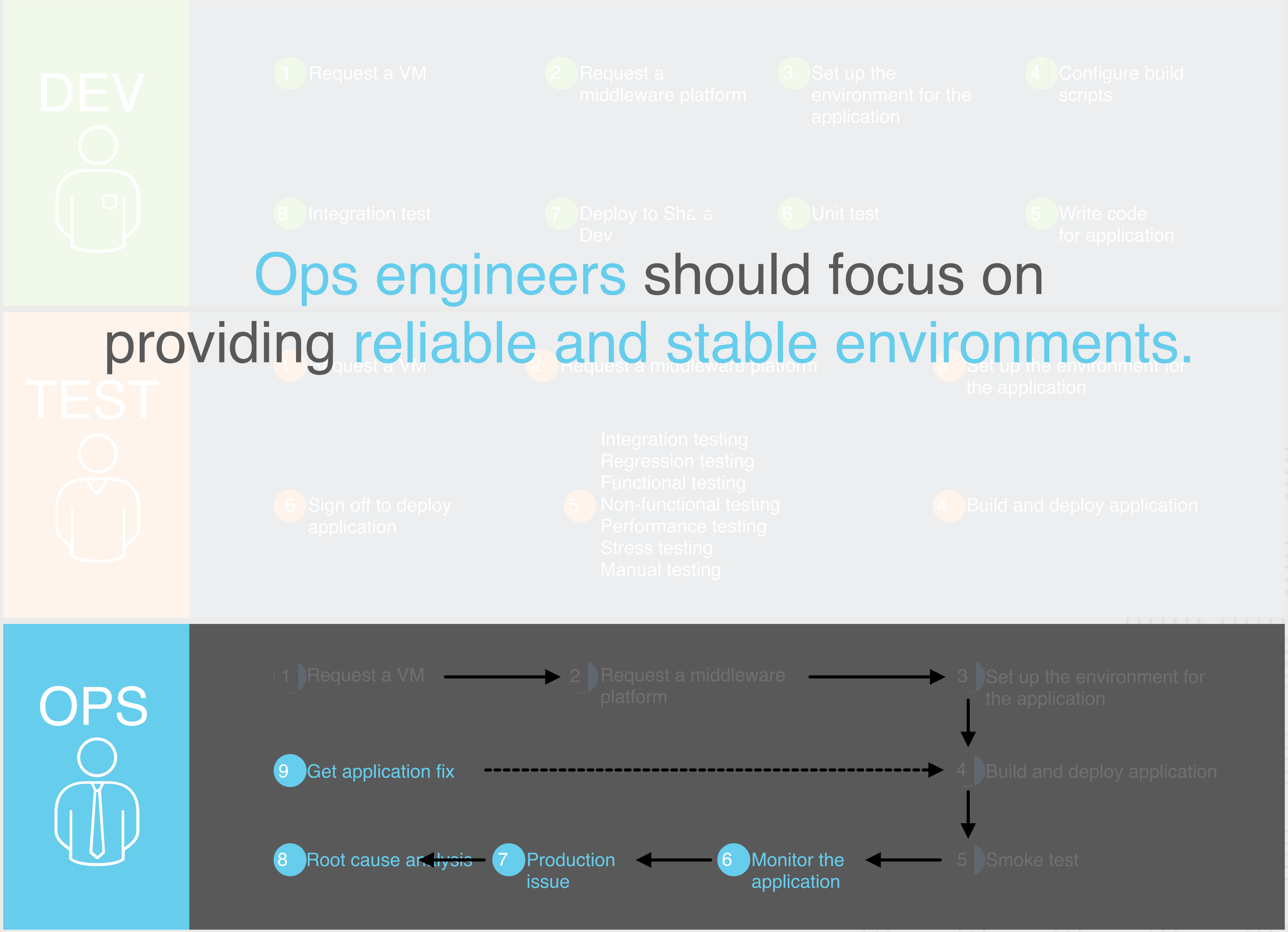
6 Monitor the application

7 Production issue

8 Root cause analysis

9 Get application fix

Ops engineers should focus on providing reliable and stable environments.



DEV

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Configure build scripts

5 Write code for application

6 Unit test

7 Deploy to Share Dev

8 Integration test

TEST

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Build and deploy application

5 Integration testing
Regression testing
Functional testing
Non-functional testing
Performance testing
Stress testing
Manual testing

6 Sign off to deploy application

OPS

1 Request a VM

2 Request a middleware platform

3 Set up the environment for the application

4 Build and deploy application

5 Smoke test

6 Monitor the application

7 Production issue

8 Root cause analysis

9 Get application fix

REALIZING EFFICIENCIES

STANDARDIZATION

AUTOMATION

CONTINUOUS IMPROVEMENT

STANDARDIZATION

STANDARDIZATION

STANDARDIZE TECHNOLOGY

- Operating systems (with patch levels)
- Application servers
- Java/JDK/JRE
- Common libraries
- Build and packaging technologies

STANDARDIZE PROCESSES

- SDLC
- Release management
- Monitoring
- Escalation management

AUTOMATION

redhat.

THREE LEVELS OF AUTOMATION

APPLICATION LIFE CYCLE AUTOMATION
Application

MIDDLEWARE PLATFORM AUTOMATION
Web/app servers | Libraries

INFRASTRUCTURE AUTOMATION
Virtualization | OS | Bare metal

THREE LEVELS OF AUTOMATION

APPLICATION LIFE CYCLE AUTOMATION

Application life cycle

- Software features, enhancements, versions
- Release management version control, build, release management, IDE, continuous
- Integration frameworks, common frames of references for monitoring, configuration management

Typical use cases

- Continuous integration
- Continuous delivery
- Automated testing

MIDDLEWARE PLATFORM AUTOMATION

Web/app servers I Libraries

INFRASTRUCTURE AUTOMATION

Virtualization I OS I Bare metal

THREE LEVELS OF AUTOMATION

APPLICATION LIFE CYCLE AUTOMATION

Application

MIDDLEWARE PLATFORM AUTOMATION

Provisioning middleware platforms

- Load balancers
- Application servers
- Java/JDK environments
- Stand-alone frameworks

Typically provided by PaaS capabilities such as OpenShift

Typical use cases

- Developers, testers, and ops teams requesting middleware platforms
- Auto-scaling
- Compute governance policies and automatic set up and tear down of resources
- Resource optimization
- Standard operating environment

INFRASTRUCTURE AUTOMATION

Virtualization I OS I Bare metal

THREE LEVELS OF AUTOMATION

APPLICATION LIFE CYCLE AUTOMATION

Application

MIDDLEWARE PLATFORM AUTOMATION

Web/app servers I Libraries

INFRASTRUCTURE AUTOMATION

Provisioning resources operating system and down

- Operating systems
- Network
- Disk and storage
- CPU, RAM, and compute

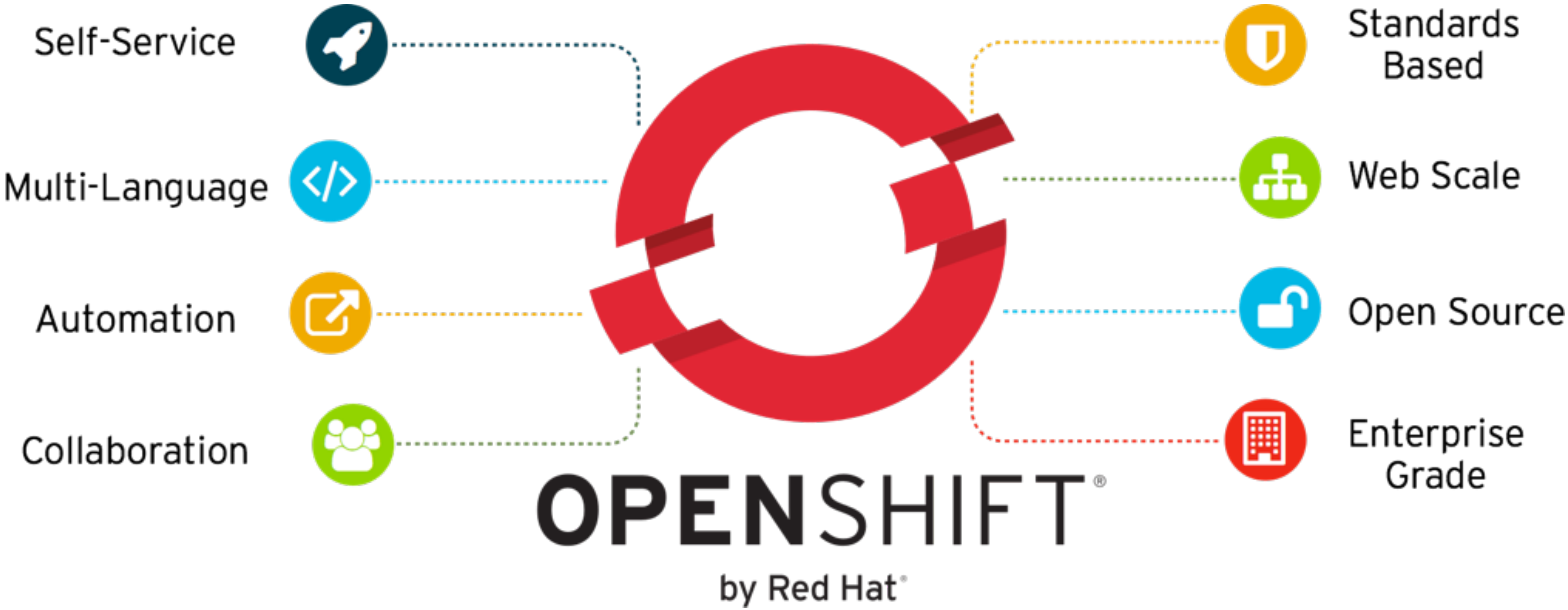
Typically provided by IaaS capabilities such as OpenStack

Virtualization – Limitations

Typical use cases

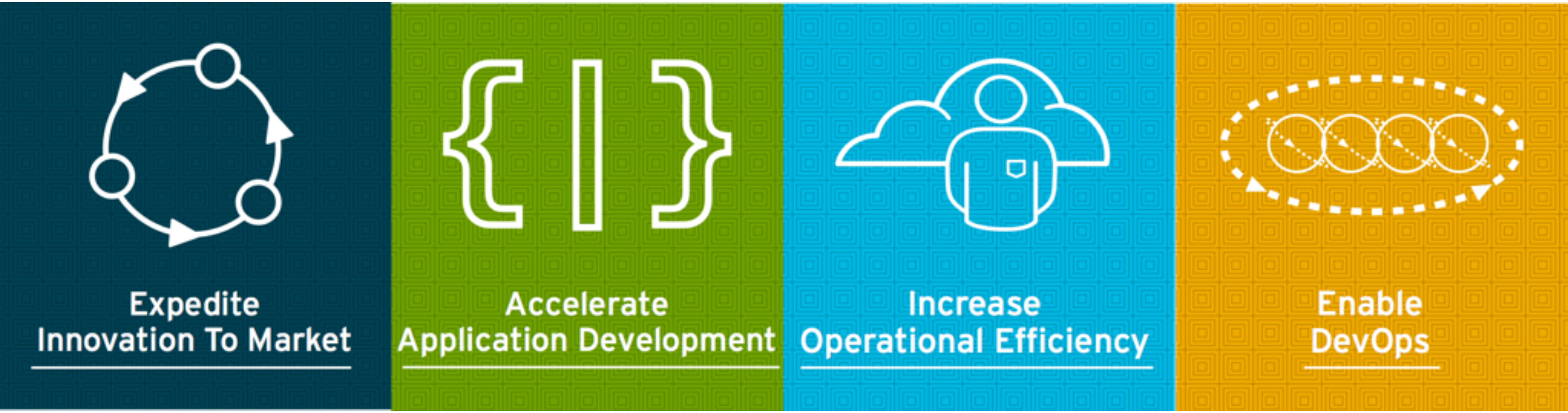
- Developers, testers, and ops teams requesting VMs
- Allocating compute power to your applications during peak load times
- Dynamically adding storage based on consumption
- Compute governance policies and automatic set up and tear down of resources
- Utility-based consumption models, pay what you use
- Does not include application platforms (only VM and down)
- Standard operating environment

OpenShift is PaaS



HOW OPENSHIFT ACCELERATES DEVOPS

Value of OpenShift



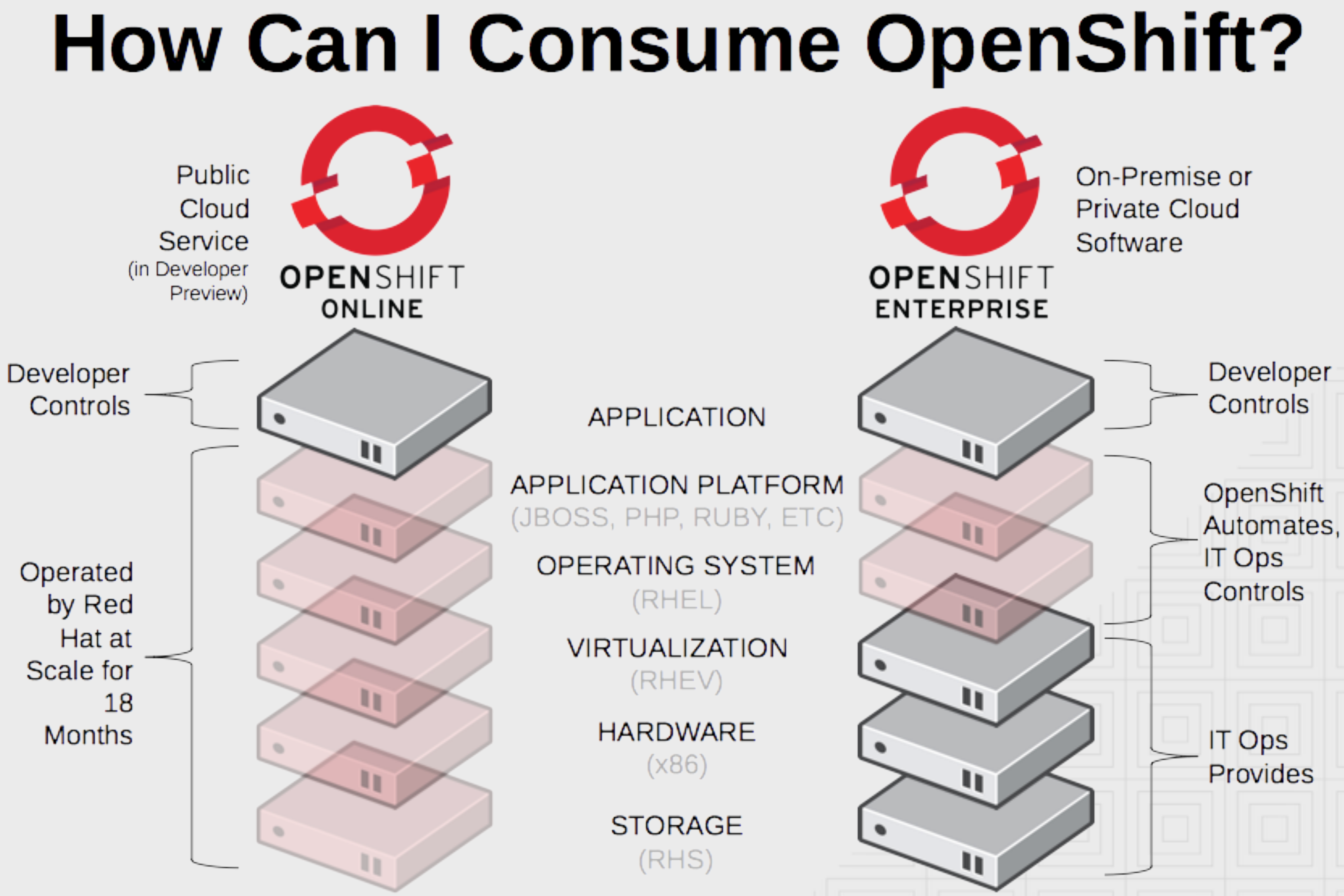
Expedite Innovation To Market

Accelerate Application Development

Increase Operational Efficiency

Enable DevOps

HOW OPENSHIFT ACCELERATES DEVOPS



How Can I Consume OpenShift?

Public Cloud Service (in Developer Preview)

OPENSHIFT ONLINE

OPENSHIFT ENTERPRISE

On-Premise or Private Cloud Software

Developer Controls

Operated by Red Hat at Scale for 18 Months

APPLICATION

APPLICATION PLATFORM (JBOSS, PHP, RUBY, ETC)

OPERATING SYSTEM (RHEL)

VIRTUALIZATION (RHEV)

HARDWARE (x86)

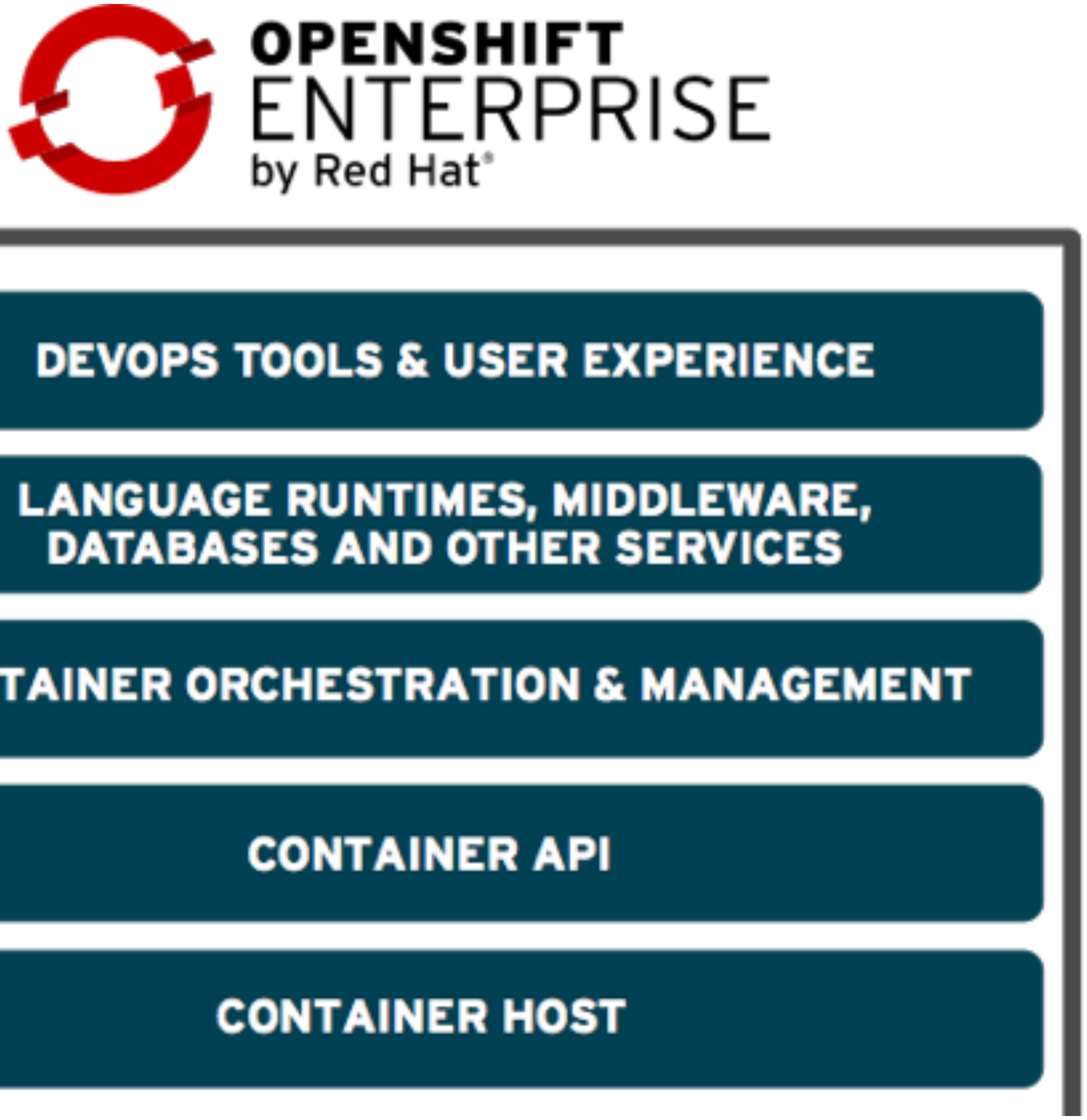
STORAGE (RHS)

Developer Controls

OpenShift Automates, IT Ops Controls

IT Ops Provides

OpenShift 3 Architecture


OPENSHIFT ENTERPRISE by Red Hat
DEVOPS TOOLS & USER EXPERIENCE
LANGUAGE RUNTIMES, MIDDLEWARE, DATABASES AND OTHER SERVICES
CONTAINER ORCHESTRATION & MANAGEMENT
CONTAINER API
CONTAINER HOST

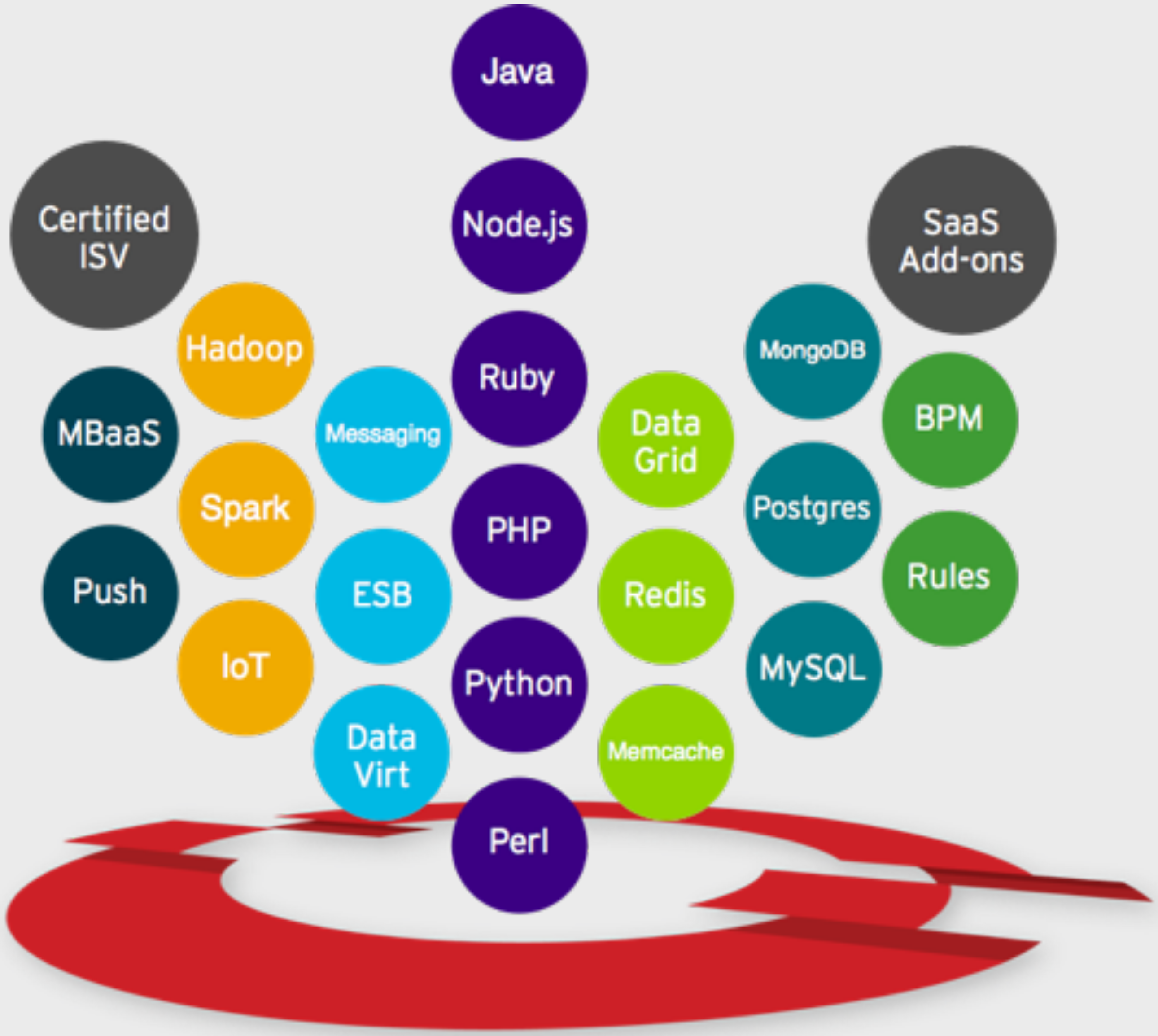
OPENSHIFT by Red Hat

RED HAT ENTERPRISE LINUX 7

docker

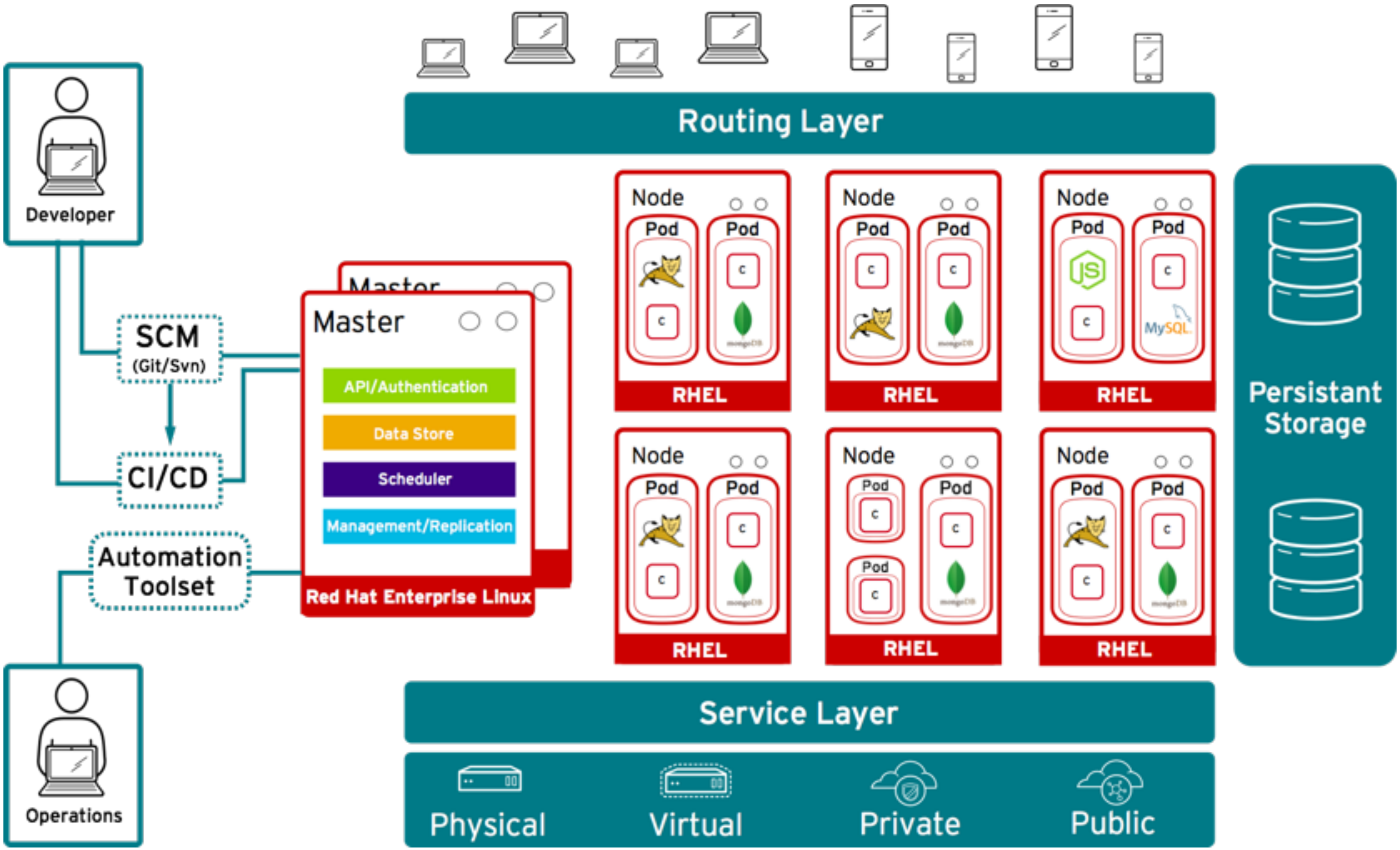
kubernetes

OPENSHIFT APPLICATION SERVICES



- From Red Hat
- From ISV Partners
- From the Community

HOW OPENSHIFT ACCELERATES DEVOPS

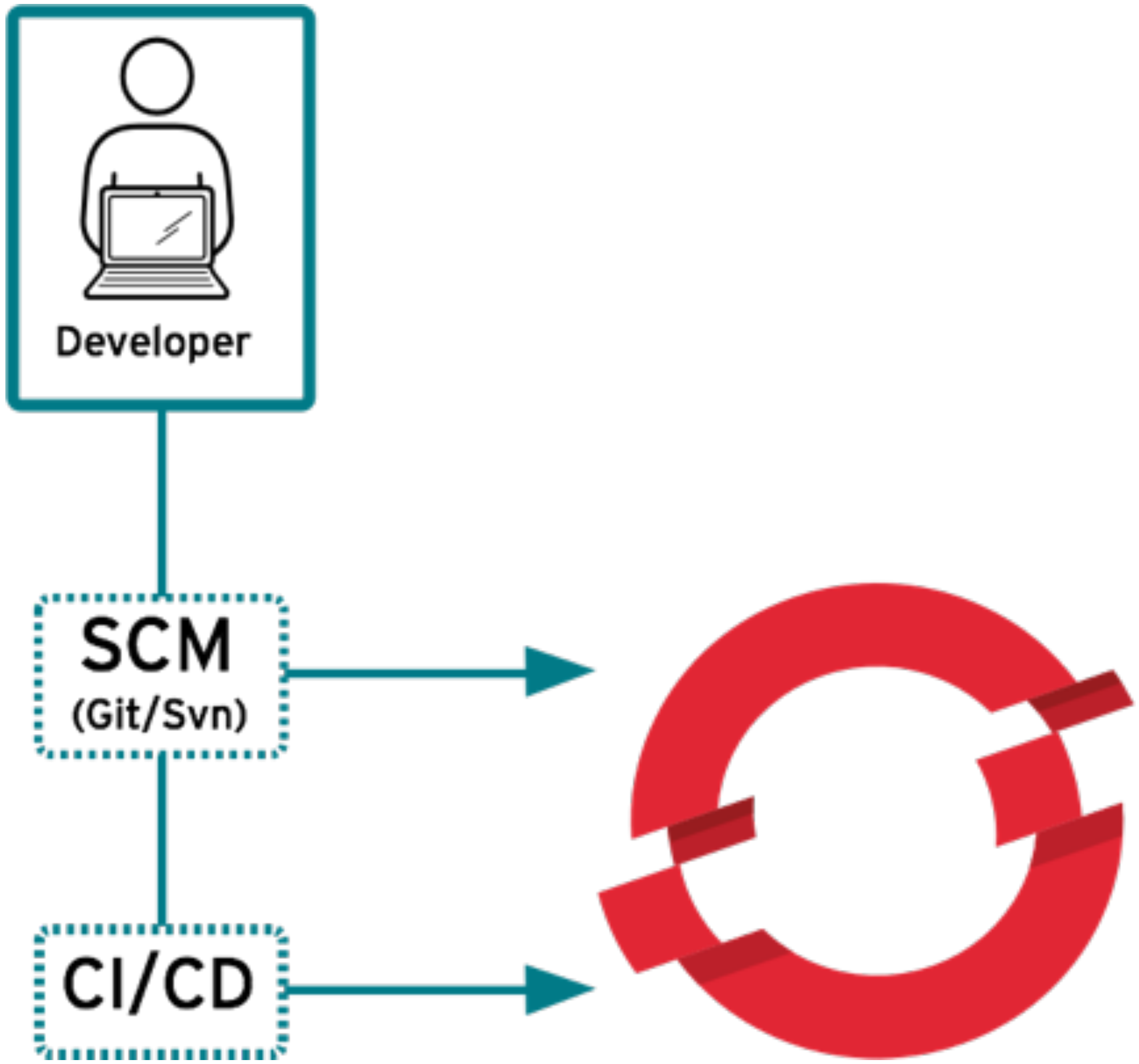


OPENSHIFT by Red Hat

HOW OPENSHIFT ACCELERATES DEVOPS

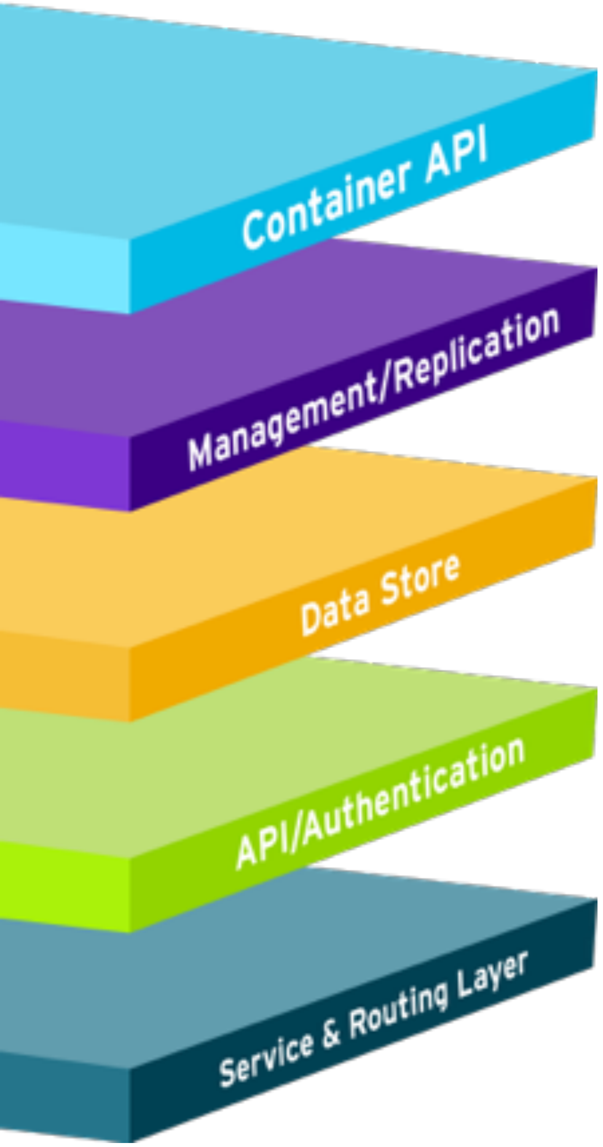
Benefits for Developers

- Access a broad selection of application components
- Deploy application environments on-demand
- Leverage your choice of interface & integrate with existing tools
- Automate application deployments, builds and source-to-image
- Enable collaboration across users, teams & projects



HOW OPENSHIFT ACCELERATES DEVOPS

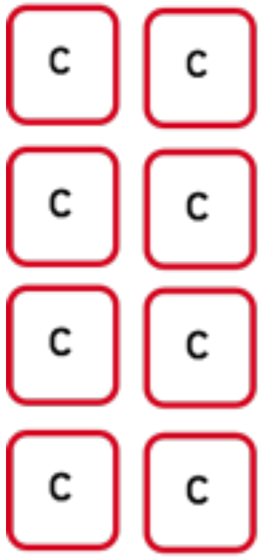
Benefits for IT Operations



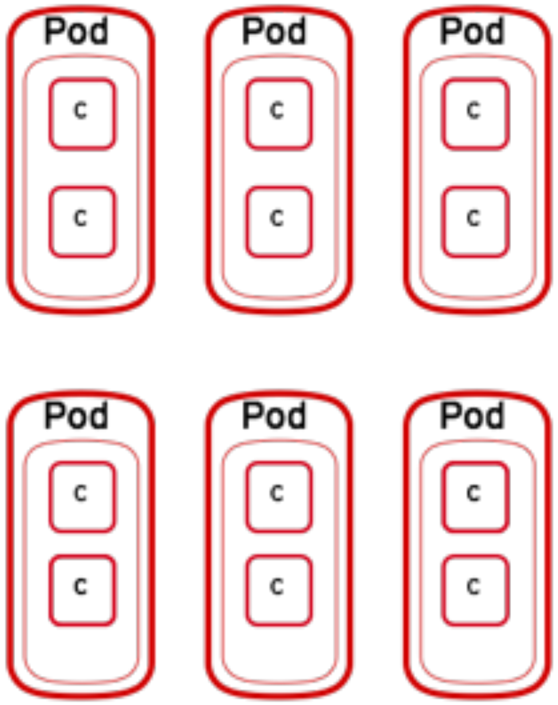
- Deploy a secure, enterprise-grade container-based application platform
- Enable application developers while improving operational efficiency & infrastructure utilization
- Utilize advanced scheduling and automated placement with regions and zones for HA
- Leverage powerful declarative management for application services
- Manage user & team access and integrate with enterprise authentication systems

HOW OPENSHIFT ACCELERATES DEVOPS

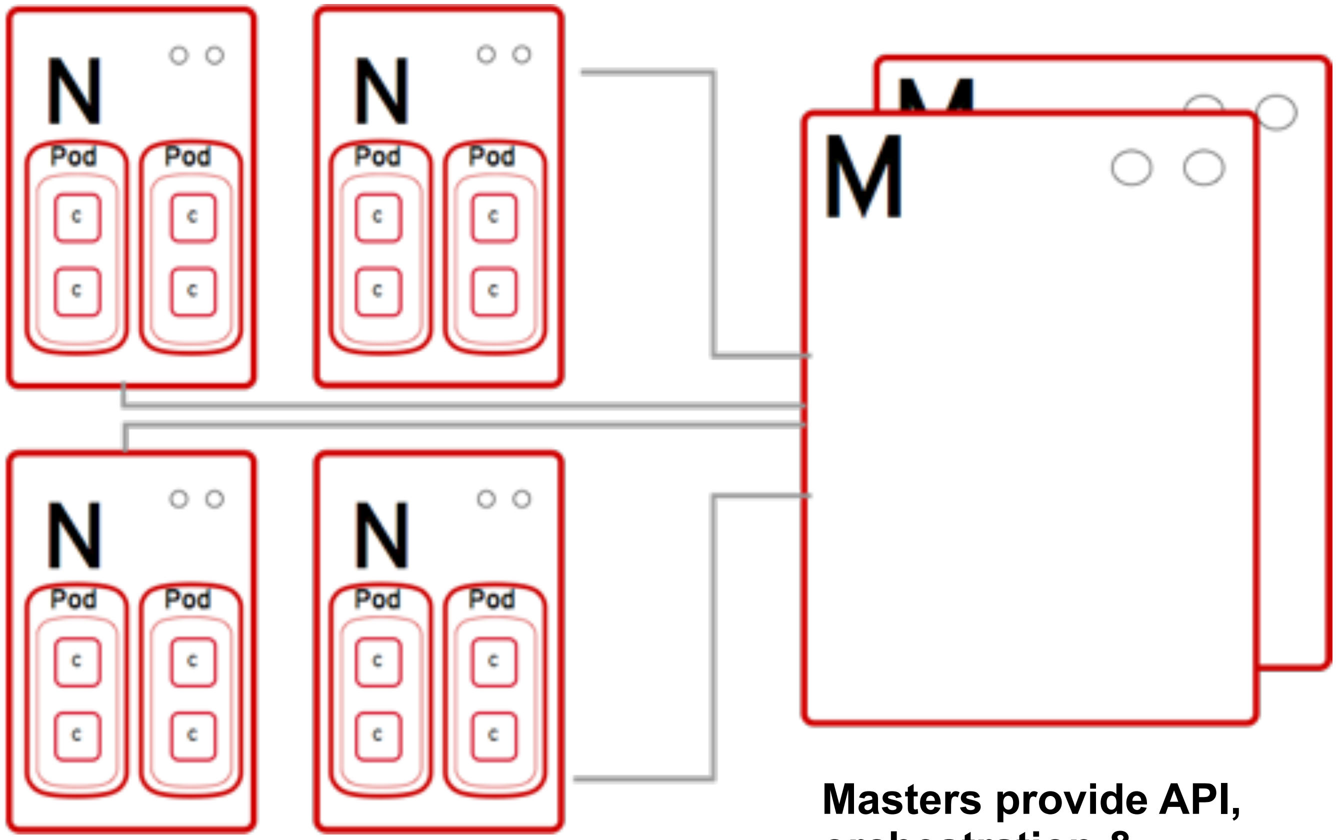
How It Works - OpenShift Concepts



Containers run lang/framework middleware, database & other runtimes.



Pods run one or more containers as a single unit. Each pod has an IP and mapped storage volumes.

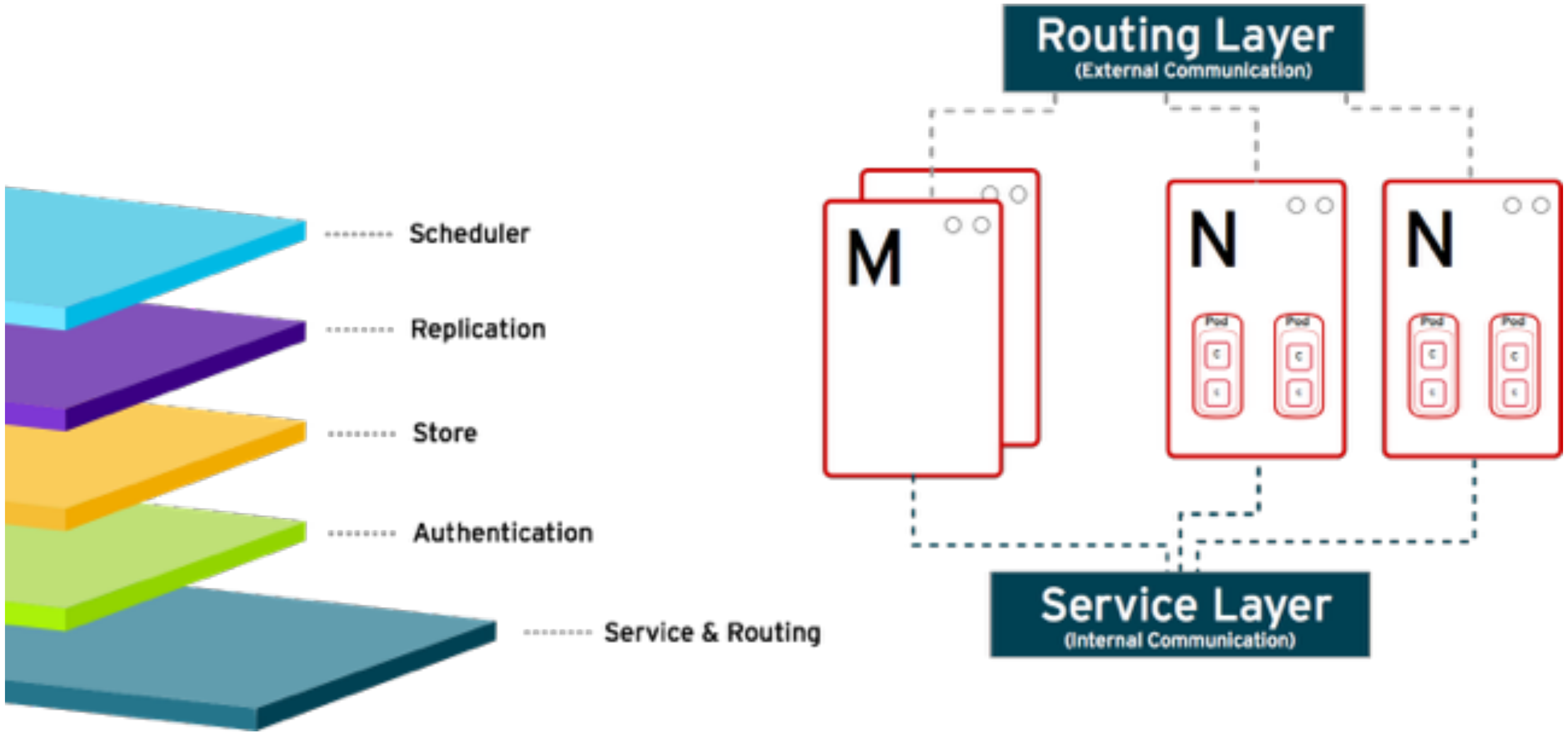


Nodes are Linux container hosts that run Pods assigned by the Master.

Masters provide API, orchestration & scheduling, maintain state and manage Pods & Services.

HOW OPENSHIFT ACCELERATES DEVOPS

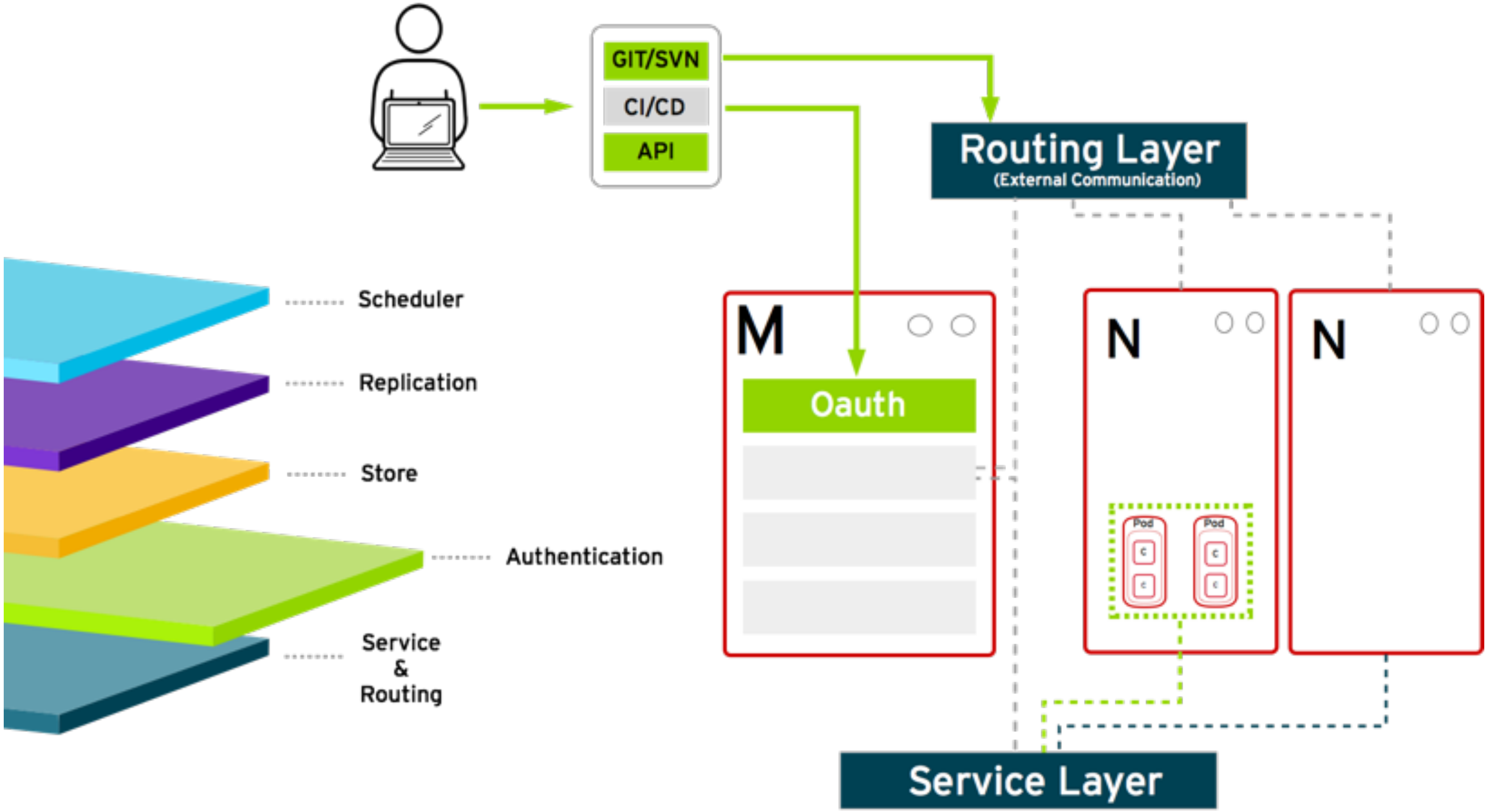
OpenShift - How It Works



- Masters orchestrate containers
- Services, routes and pods all orchestrated and scheduled by the masters

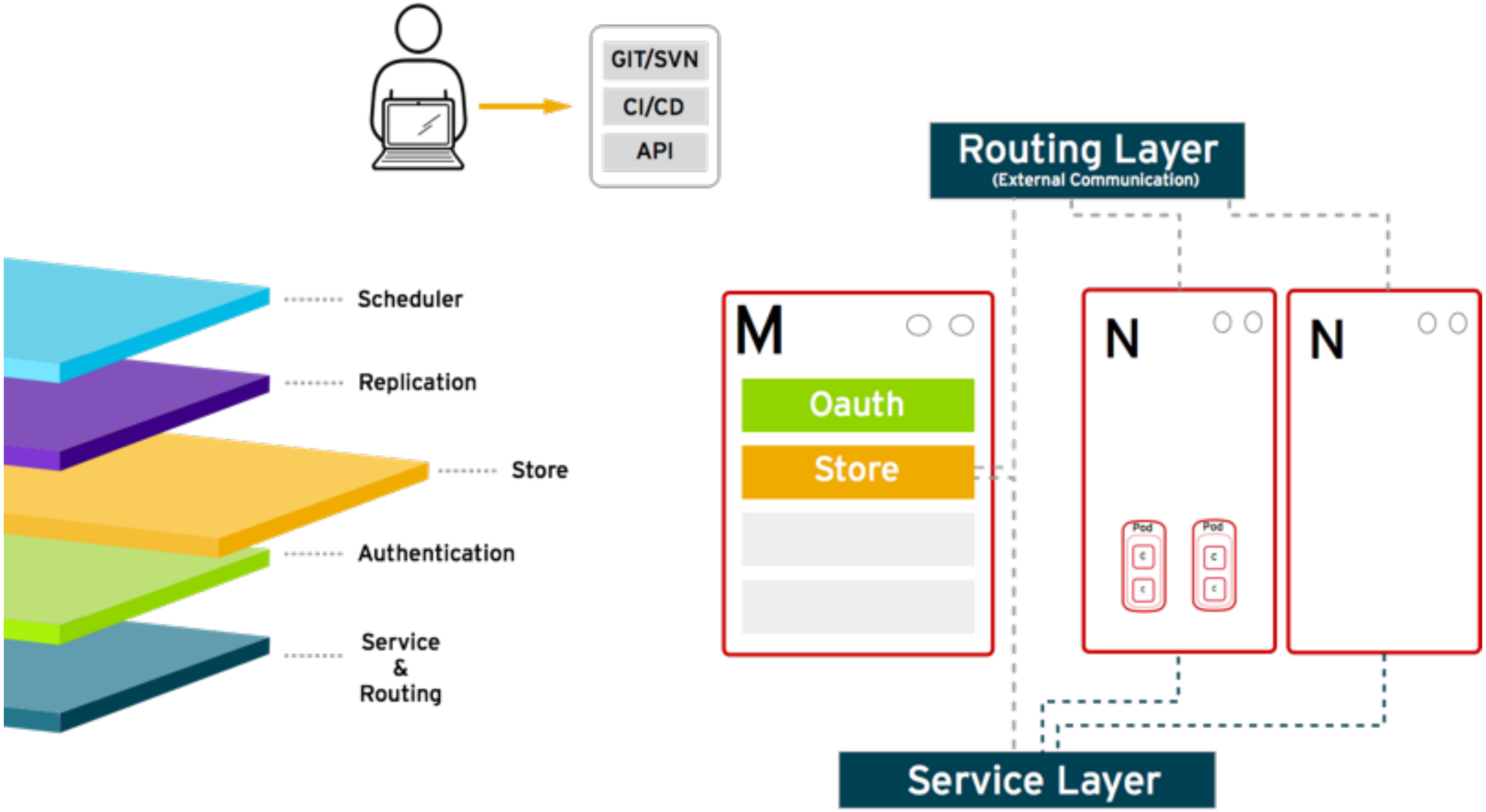
HOW OPENSHIFT ACCELERATES DEVOPS

OpenShift - How It Works



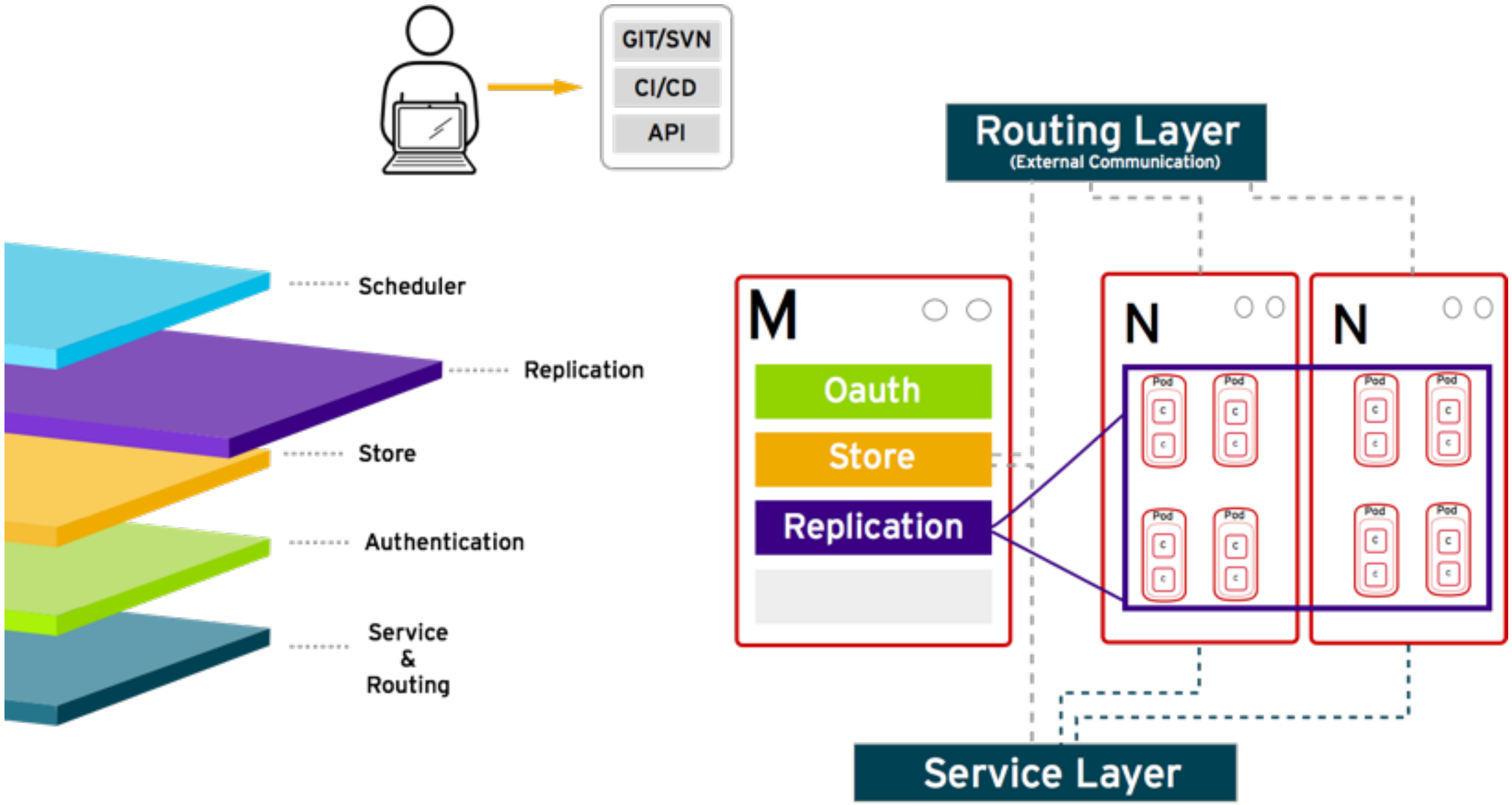
HOW OPENSHIFT ACCELERATES DEVOPS

OpenShift - How It Works



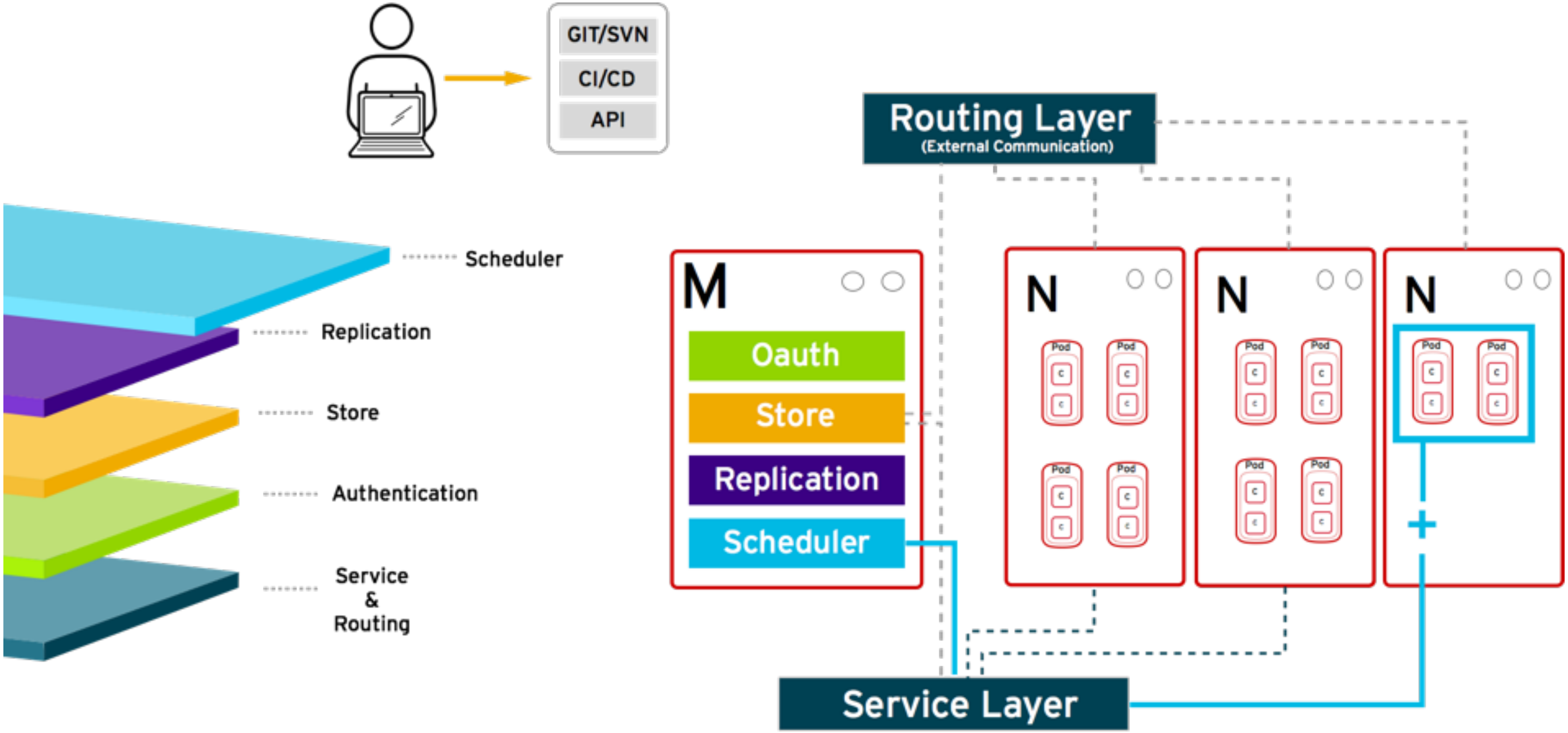
HOW OPENSHIFT ACCELERATES DEVOPS

OpenShift - How It Works



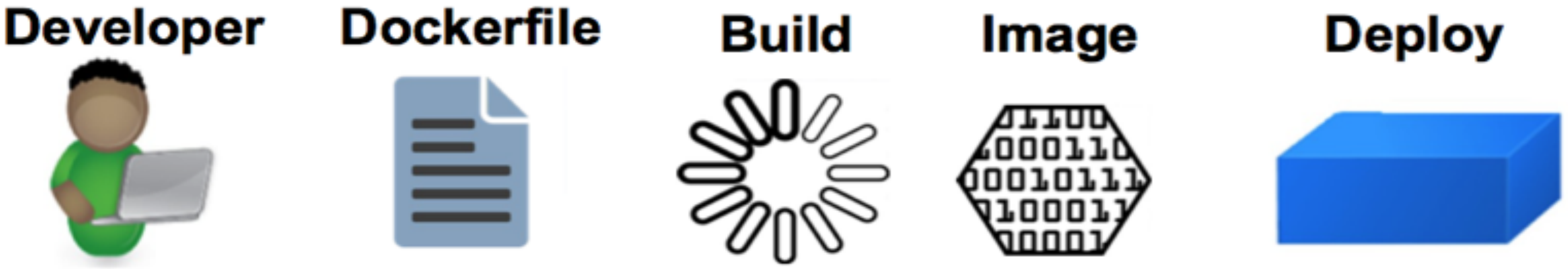
HOW OPENSHIFT ACCELERATES DEVOPS

OpenShift - How It Works



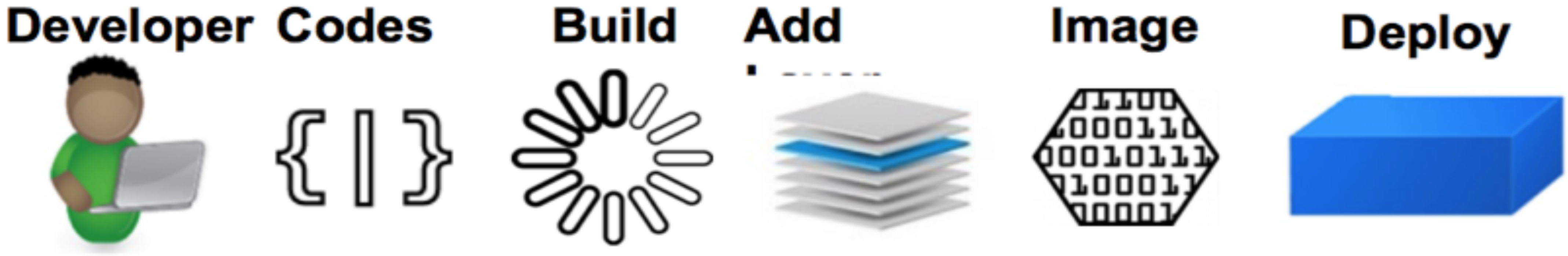
Source to image

Integrated Docker Builds



Developer Dockerfile Build Image Deploy

Source to Image Builds



Developer Codes Build Add Image Deploy

HOW OPENSHIFT ACCELERATES DEVOPS

JBoss Middleware Services on OpenShift

Application Container Services

- JBoss Enterprise Application Platform
- JBoss Web Server / Tomcat
- JBoss Developer Studio

Integration Services

- Fuse
- A-MQ
- Data Virtualization

Business Process Services

- Business Process Management *
- Business Rules Management System *

Mobile Services

- Red Hat Mobile / FeedHenry *

* = Coming Soon

SO, HOW DO WE DO ALL THAT?

HOW OPENSHIFT ACCELERATES DEVOPS

Automation is a cornerstone of DevOps practices.

APPLICATION LIFE CYCLE AUTOMATION

Application

DevOps platform automation leads to efficient, repeatable DevOps application life cycle management. OpenShift is an accelerator to application DevOps.

MIDDLEWARE PLATFORM AUTOMATION

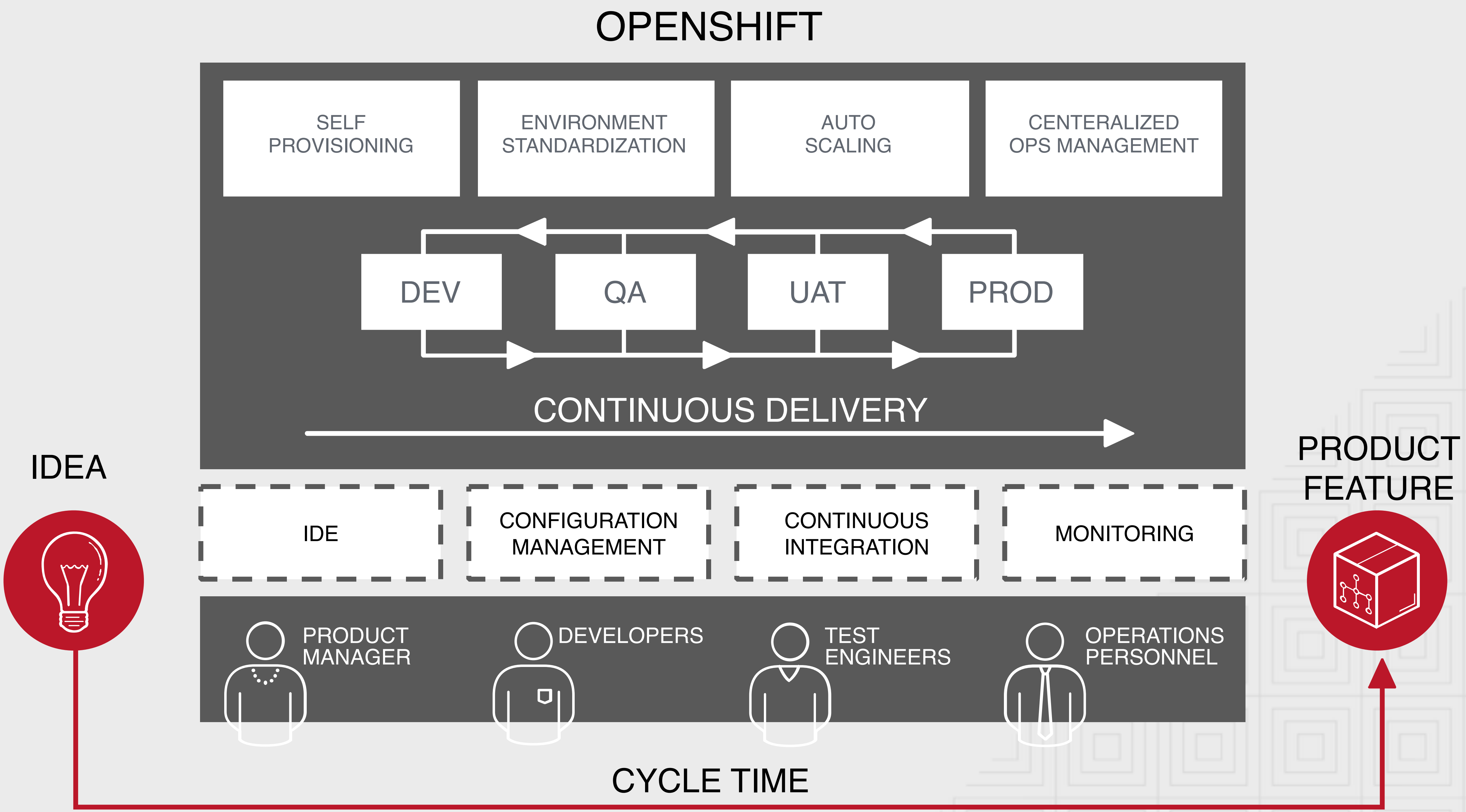
Web/app servers I Libraries

OpenShift is advanced platform automation. Does not replace DevOps collaboration, but provides a framework for it, so you don't have to roll your own.

INFRASTRUCTURE AUTOMATION

Virtualization I OS I Bare metal

CONTINUOUS DELIVERY THROUGH OPENSHIFT



OPENSHIFT

SELF PROVISIONING

ENVIRONMENT STANDARDIZATION

AUTO SCALING

CENTERALIZED OPS MANAGEMENT

DEV

QA

UAT

PROD

CONTINUOUS DELIVERY

IDEA

IDE

CONFIGURATION MANAGEMENT

CONTINUOUS INTEGRATION

MONITORING

PRODUCT FEATURE

PRODUCT MANAGER

DEVELOPERS

TEST ENGINEERS

OPERATIONS PERSONNEL

CYCLE TIME

HOW OPENSHIFT ACCELERATES DEVOPS APPLICATION LIFE CYCLE MANAGEMENT

Solves platform automation…

- Standardized operating environments
- Environment configuration as code
- Self-provisioning

…so the DevOps focus can be on application delivery

Continuous integration/delivery

- Integration with major DevOps tools
- Just-in-time delivery (and teardown) of single-purpose platforms for resource efficiency

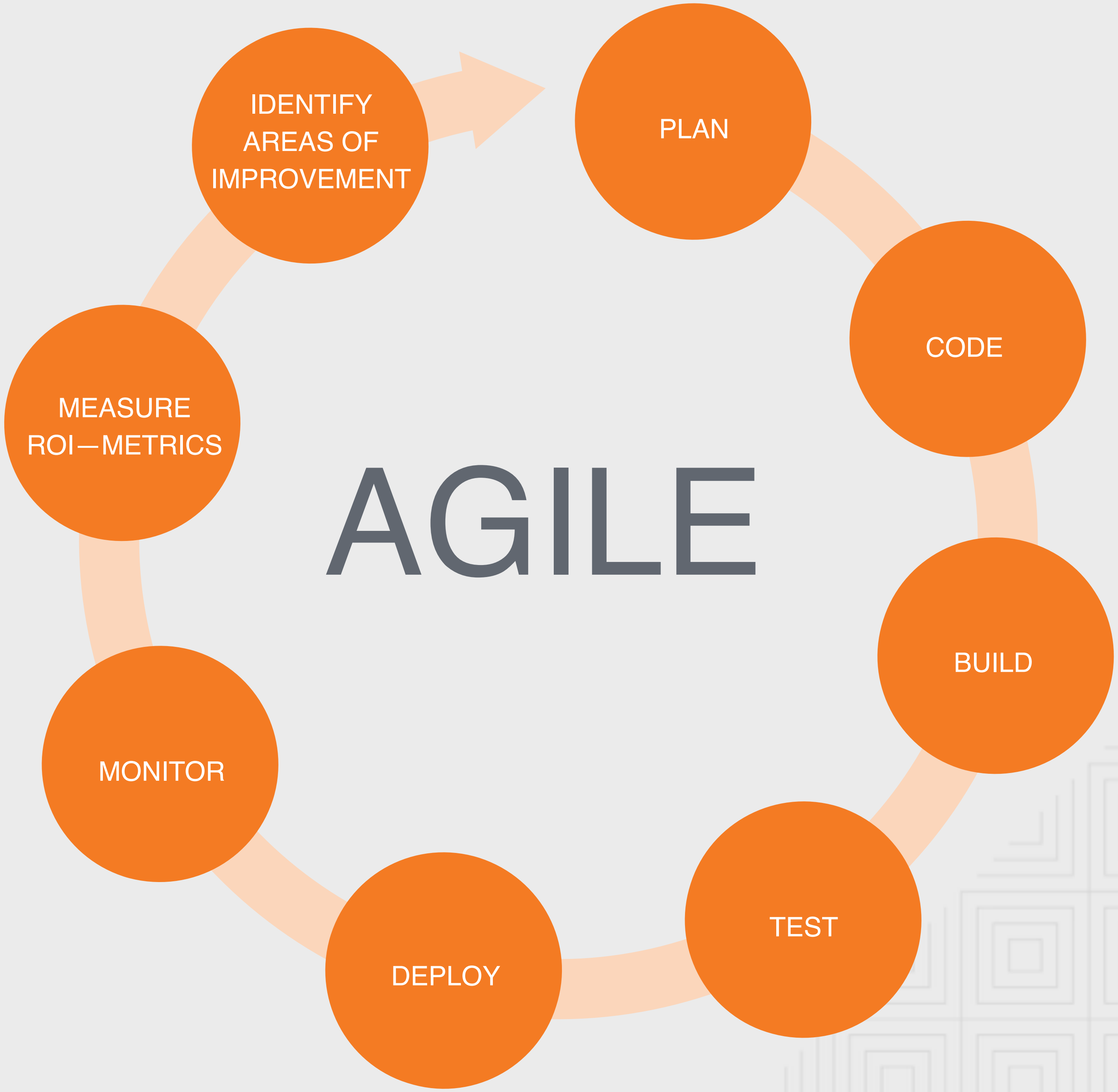
Ready-to-go framework for application configuration as code

- Action hooks
- Cartridges
- Environment variables

Auto-scaling

CONTINUOUS IMPROVEMENT

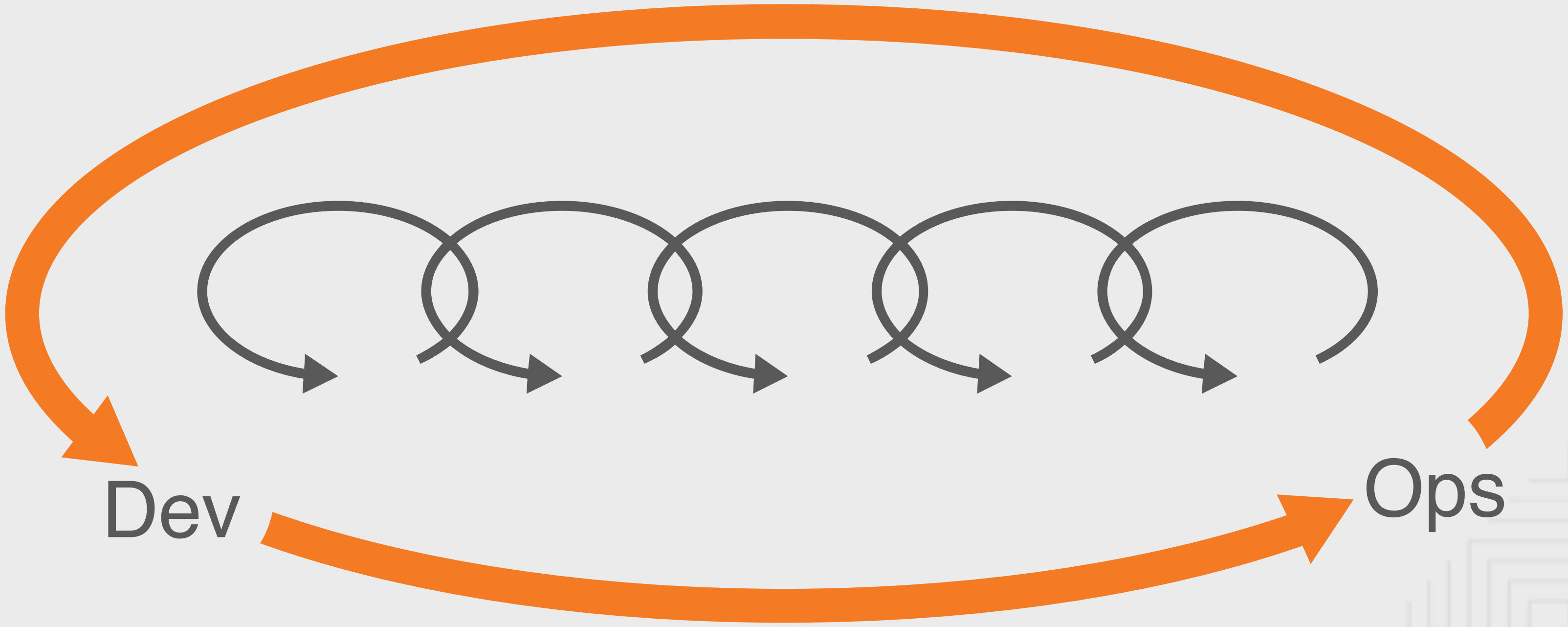
CONTINUOUS IMPROVEMENT



CONTINUOUS IMPROVEMENT

THE THIRD WAY:

Culture of continual experimentation and learning



The Three Ways: The Principles Underpinning DevOps by Gene Kim

QUESTIONS?