ACCELERATING DEVOPS THROUGH OPENSHIFT BY RED HAT

Trevor Quinn
PaaS and DevOps Practice Lead, North America
Red Hat Consulting

redhat.

RED HAT SUMMIT

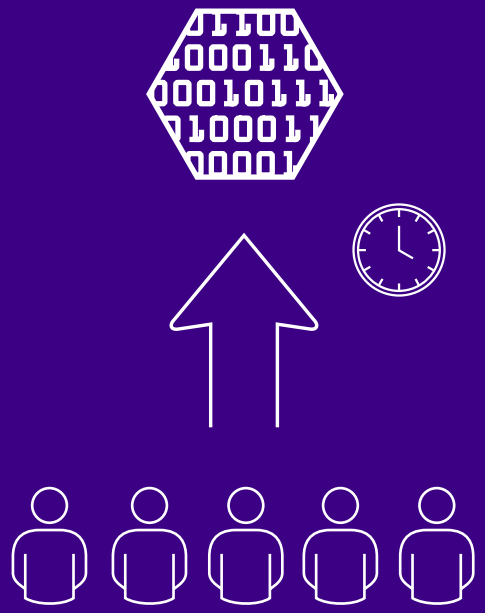
Agenda

- Current IT Landscape
- Traditional IT organization
- State of DevOps
- Cloud automation
- Application lifecycle automation
- Demo

CURRENT IT LANDSCAPE

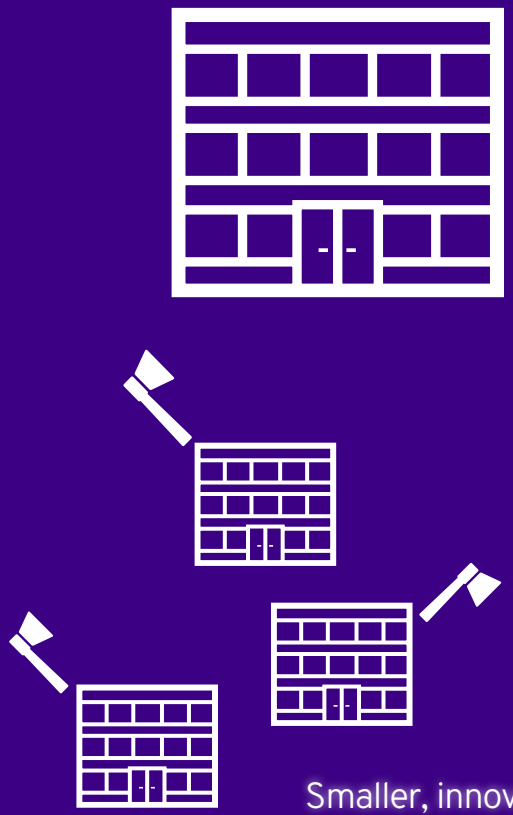
redhat.

Customers

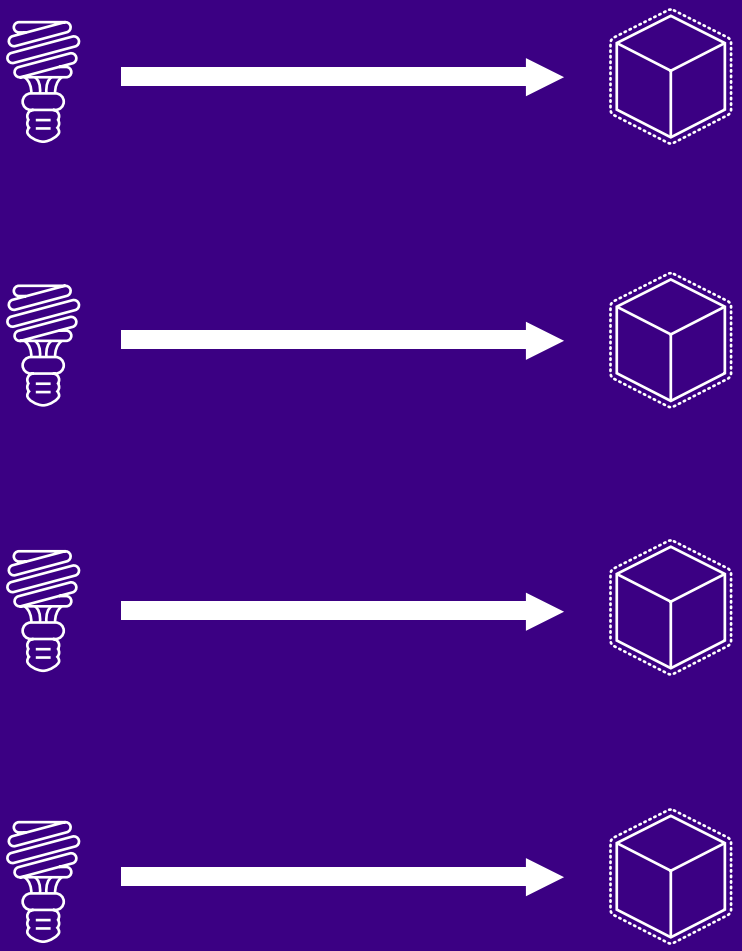


Ubiquitous, timely access to data

Enterprises



Smaller, innovative startups armed with cloud tech



Increased quality

Rapid delivery of product features and service

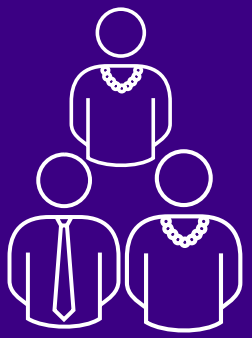
Better market fit

Doing more with less

TRADITIONAL IT ORGANIZATION

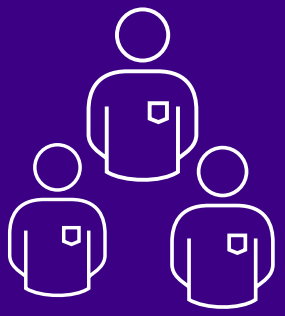
redhat.

EXPECTATIONS INSIDE IT ORGANIZATIONS



LINES OF BUSINESS
Responsive Delivery

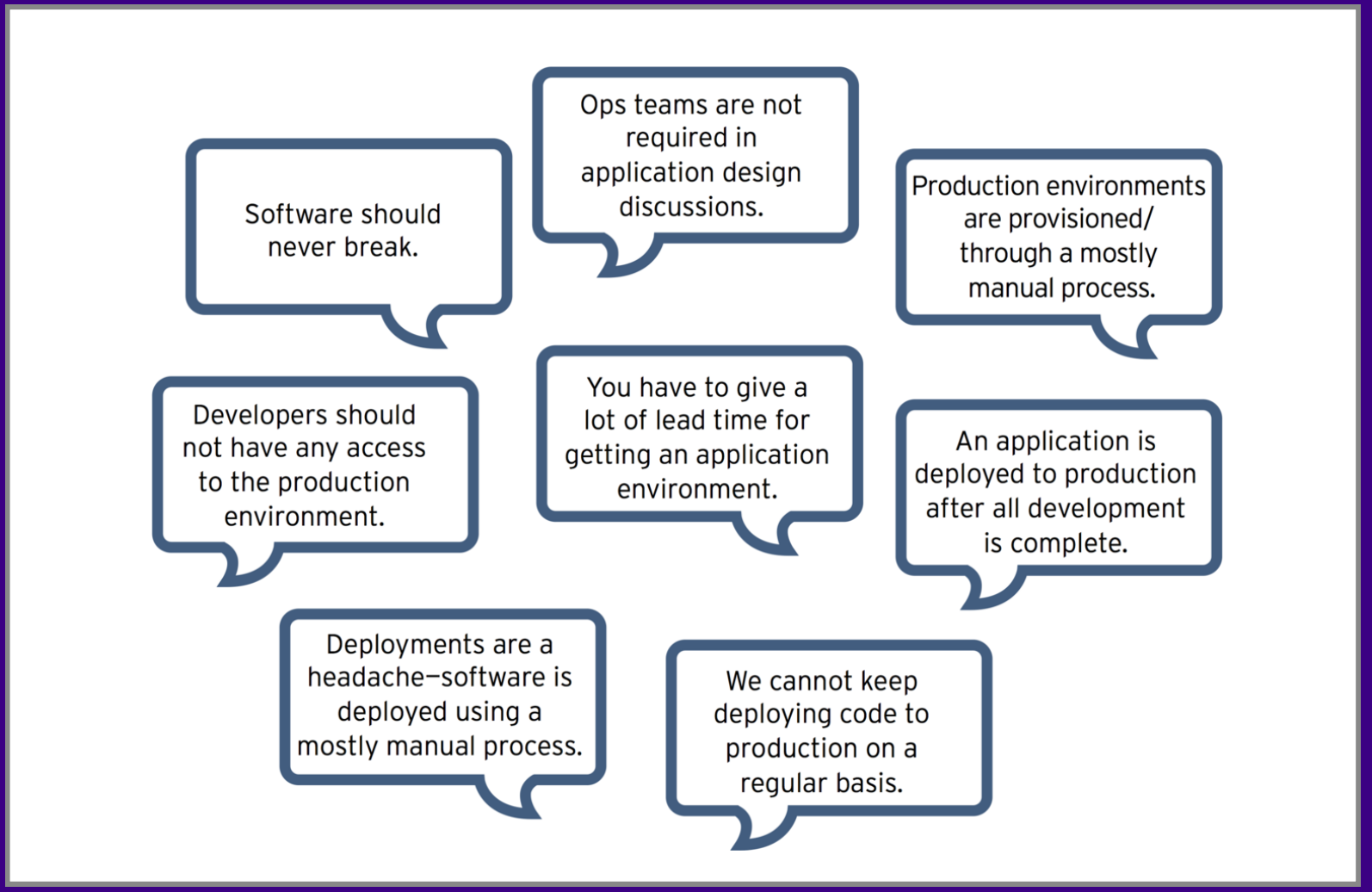
DEVELOPERS
Autonomy, Ability to Focus



QA
Testability

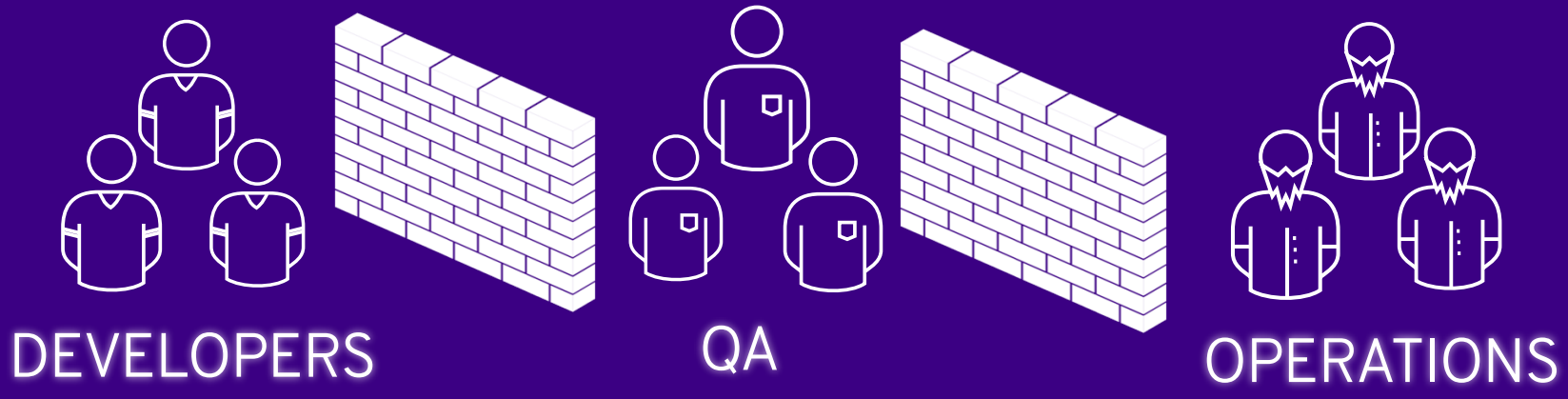
OPERATIONS
Stability

EXPECTATIONS INSIDE IT ORGANIZATIONS



"THROW IT OVER THE WALL"

Walled off people, processes, and technology



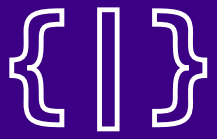
Opportunities to improve at the **system** level are potentially lost

STATE OF DEVOPS

redhat.

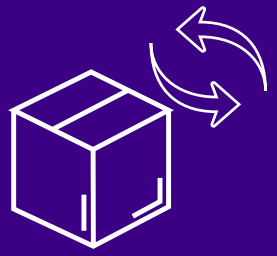
DEVOPS LINEAGE

Agile Development



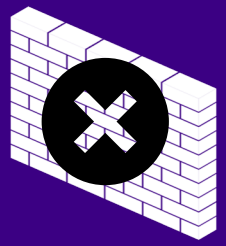
Lean

Continuous Delivery

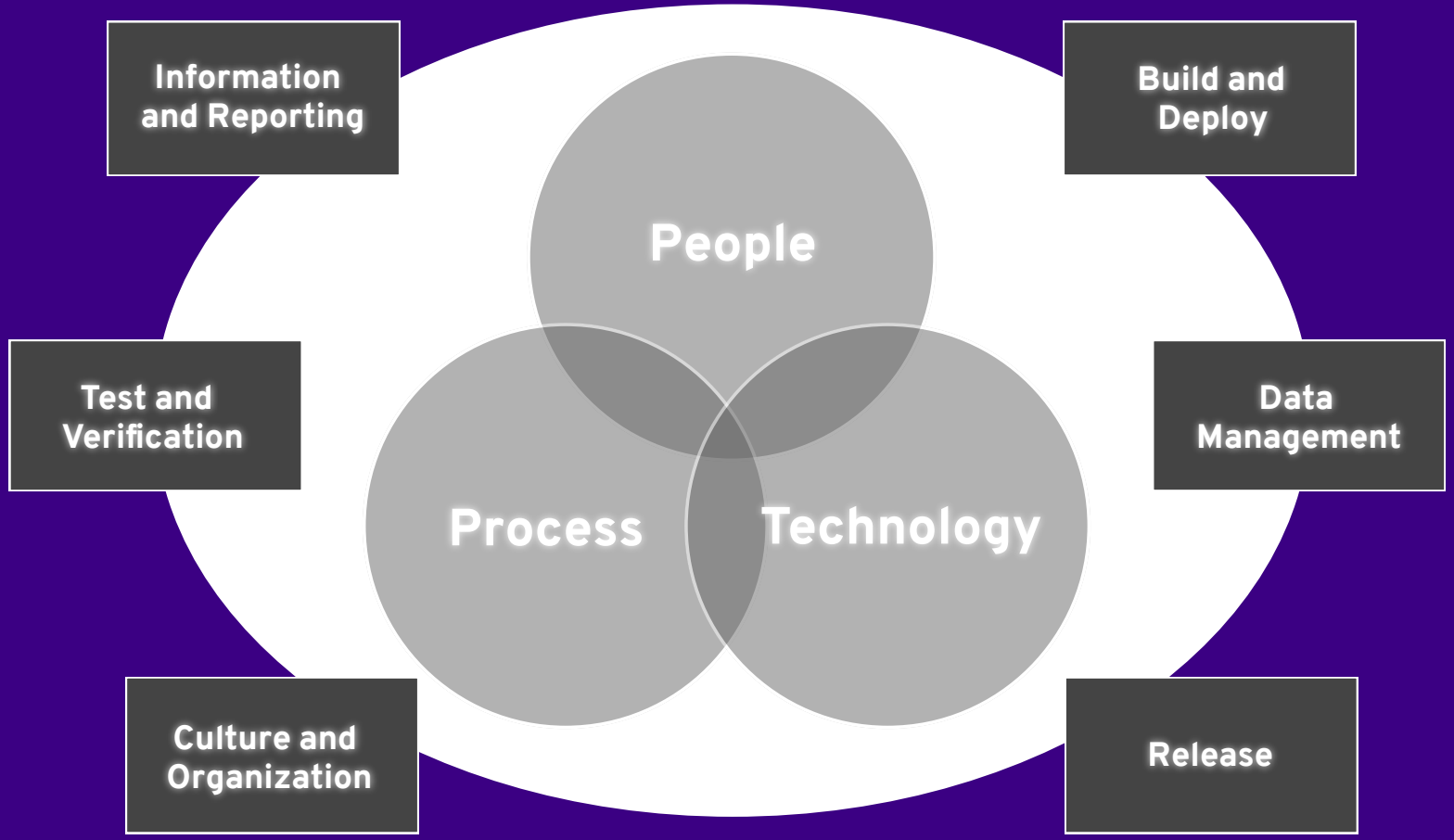


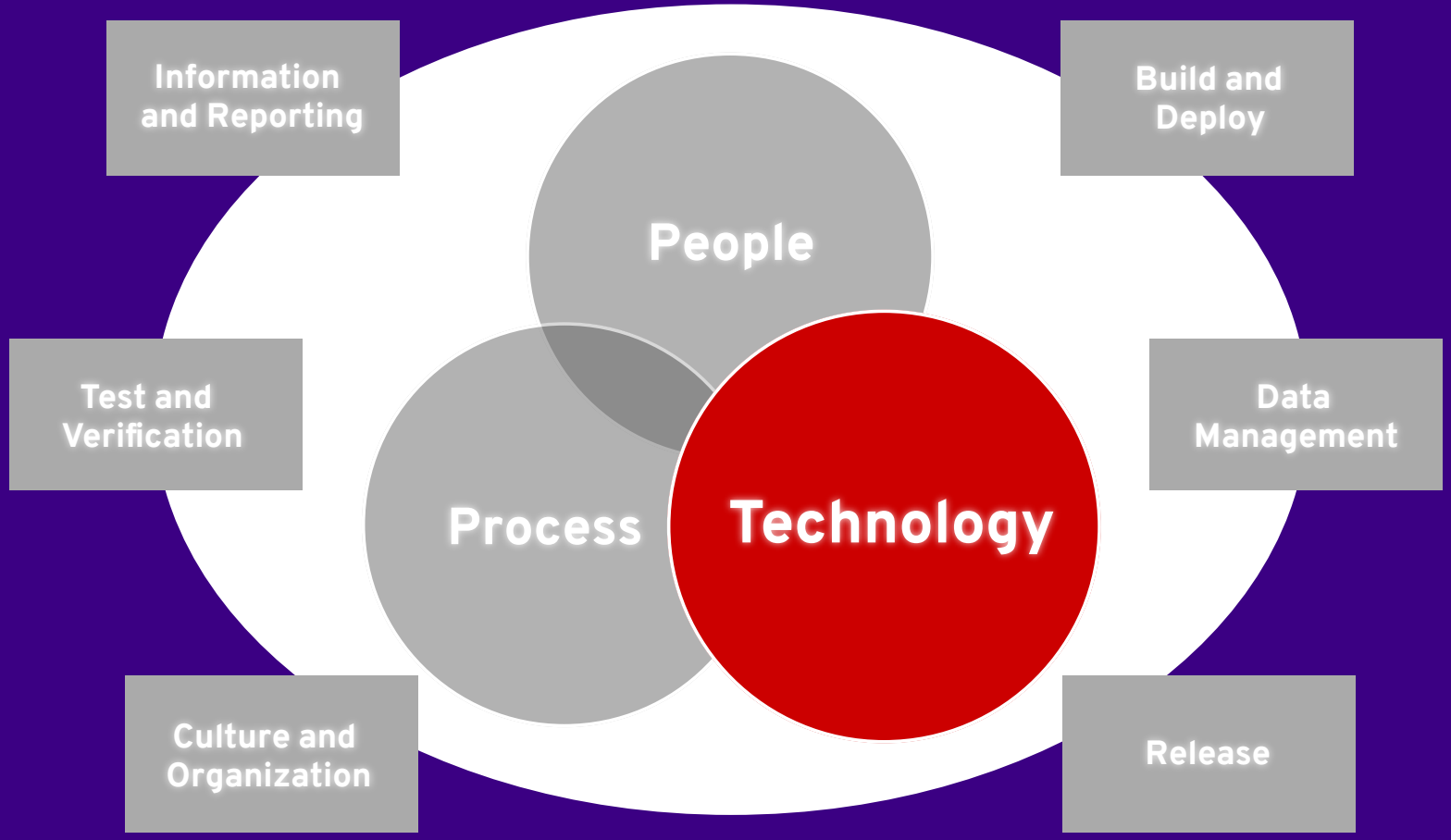
Cloud Automation

Cultural Change



DEVOPS


Information and Reporting
Build and Deploy
People
Test and Verification
Data Management
Process
Technology
Culture and Organization
Release
INITIAL
IMPROVED
OPTIMIZING


Information and Reporting
Build and Deploy
People
Test and Verification
Data Management
Process
Technology
Culture and Organization
Release
INITIAL
IMPROVED
OPTIMIZING

CLOUD AUTOMATION

redhat.

CLOUD COMPUTING

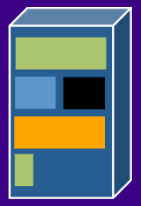
- on-demand self service
- broad network access
- resource pooling
- rapid elasticity
- measured service

(NIST Definition of Cloud Computing)

CLOUD SERVERS

Ephemeral

Anonymous



Multitenant

Cloud automation is
DevOps technology.

LEVELS OF AUTOMATION

APPLICATION LIFE CYCLE AUTOMATION
Application

APPLICATION PLATFORM AUTOMATION
Containers | Web/app servers | Libraries

INFRASTRUCTURE AUTOMATION
Container Host | Virtualization | OS | Bare metal

LEVELS OF AUTOMATION

APPLICATION LIFE CYCLE AUTOMATION

Application

APPLICATION PLATFORM AUTOMATION

Containers | Web/app servers | 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

LEVELS OF AUTOMATION

APPLICATION LIFE CYCLE AUTOMATION

Application

APPLICATION PLATFORM AUTOMATION

Provisioning middleware platforms

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

Typically provided by container orchestration and 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

Container Host | Virtualization | OS | Bare metal

LEVELS OF AUTOMATION

APPLICATION LIFE CYCLE AUTOMATION

Application life cycle

- Software features, enhancements, versions
- Version control, builds, IDE integration, continuous integration, release management
- Common frames of references for monitoring

Typical use cases

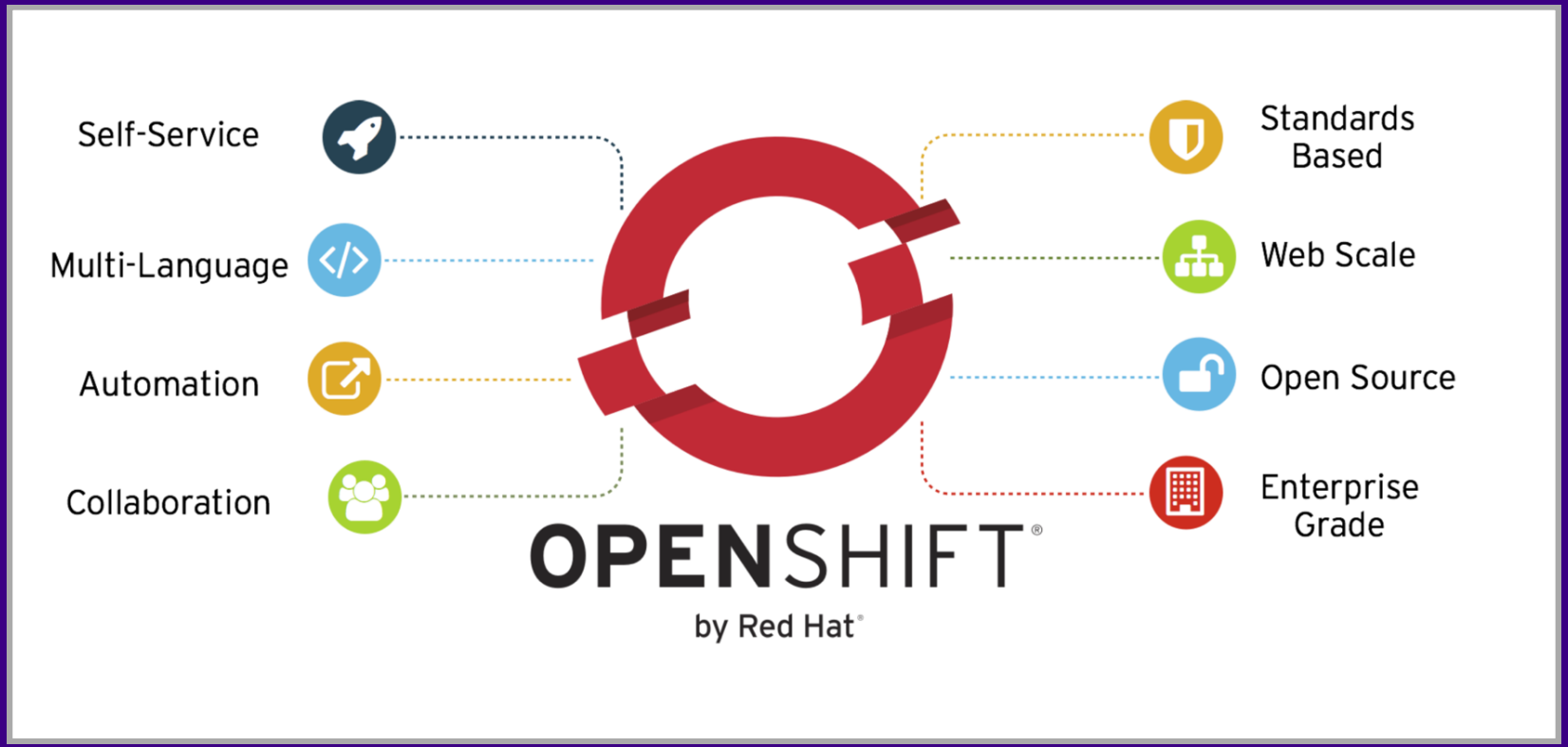
- Continuous integration
- Continuous delivery
- Automated testing

APPLICATION PLATFORM AUTOMATION

Containers | Web/app servers | Libraries

INFRASTRUCTURE AUTOMATION

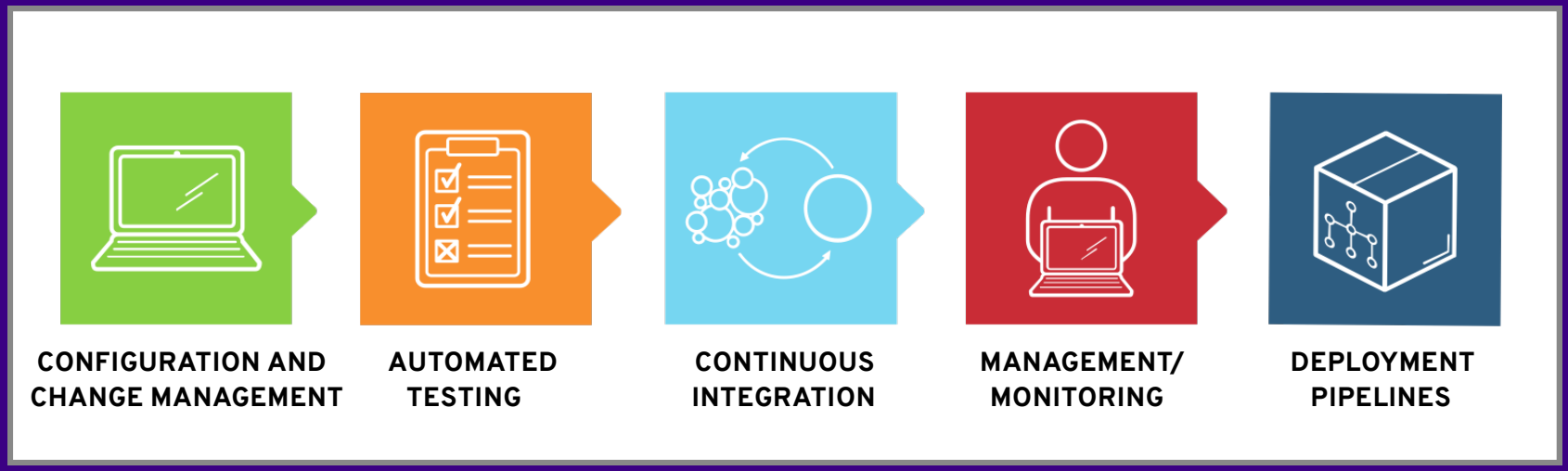
Container Host | Virtualization | OS | Bare metal


Self-Service
Multi-Language
Automation
Collaboration
Standards Based
Web Scale
Open Source
Enterprise Grade
OPENSHIFT®
by Red Hat®

APPLICATION LIFECYCLE MANAGEMENT

redhat.

APPLICATION LIFECYCLE MANAGEMENT



CONFIGURATION MANAGEMENT

Definition

All artifacts relevant to the project, and the relationships between them, are stored, retrieved, uniquely identified, and modified. (Humble and Farley, 2011)

Benefits

Allows you to exactly reproduce an entire environment (OS, system configuration, application server, server configuration, application, etc.)

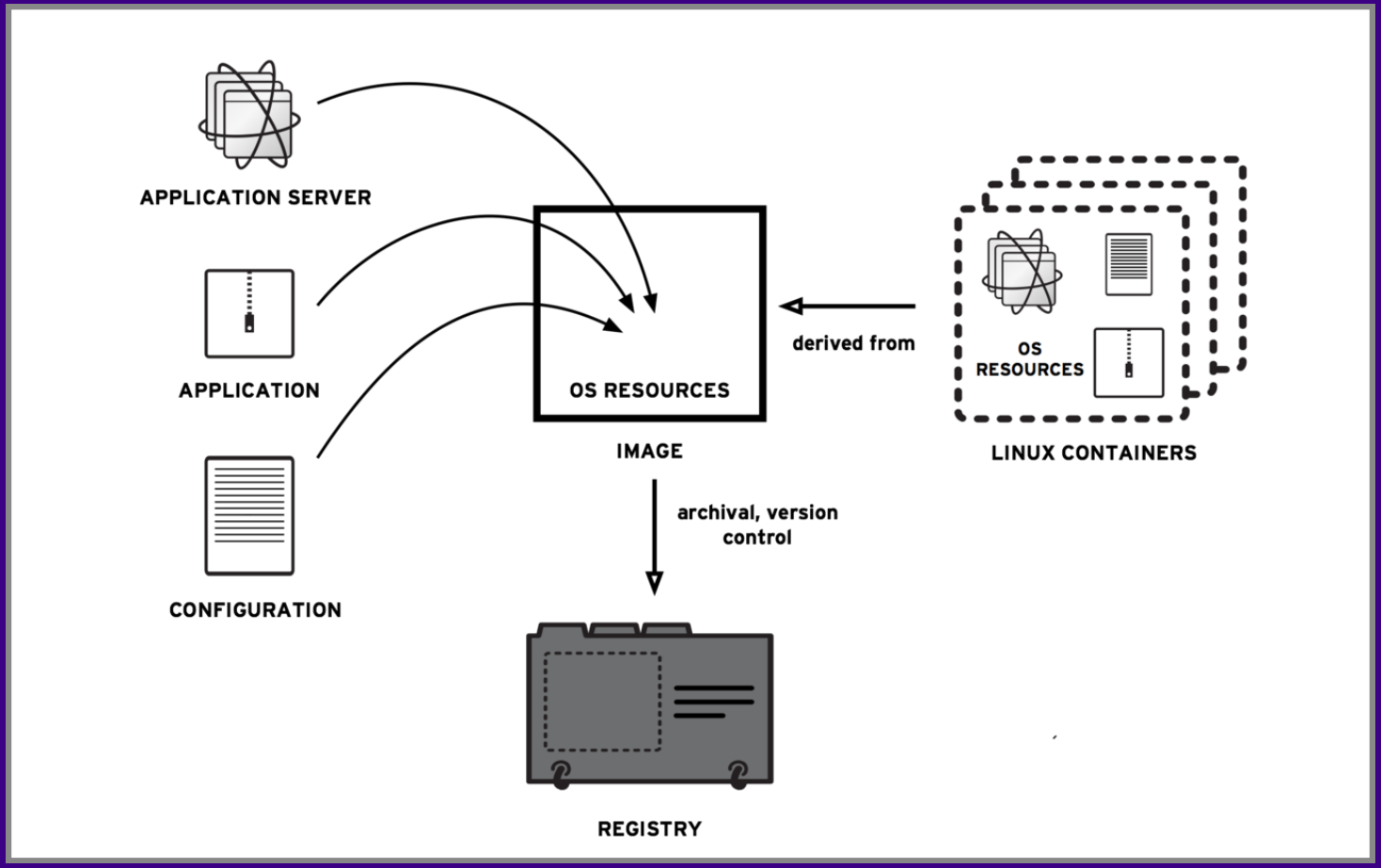
- Trace changes
- Rollback an environment to earlier working state

Tools

Version control and library repositories

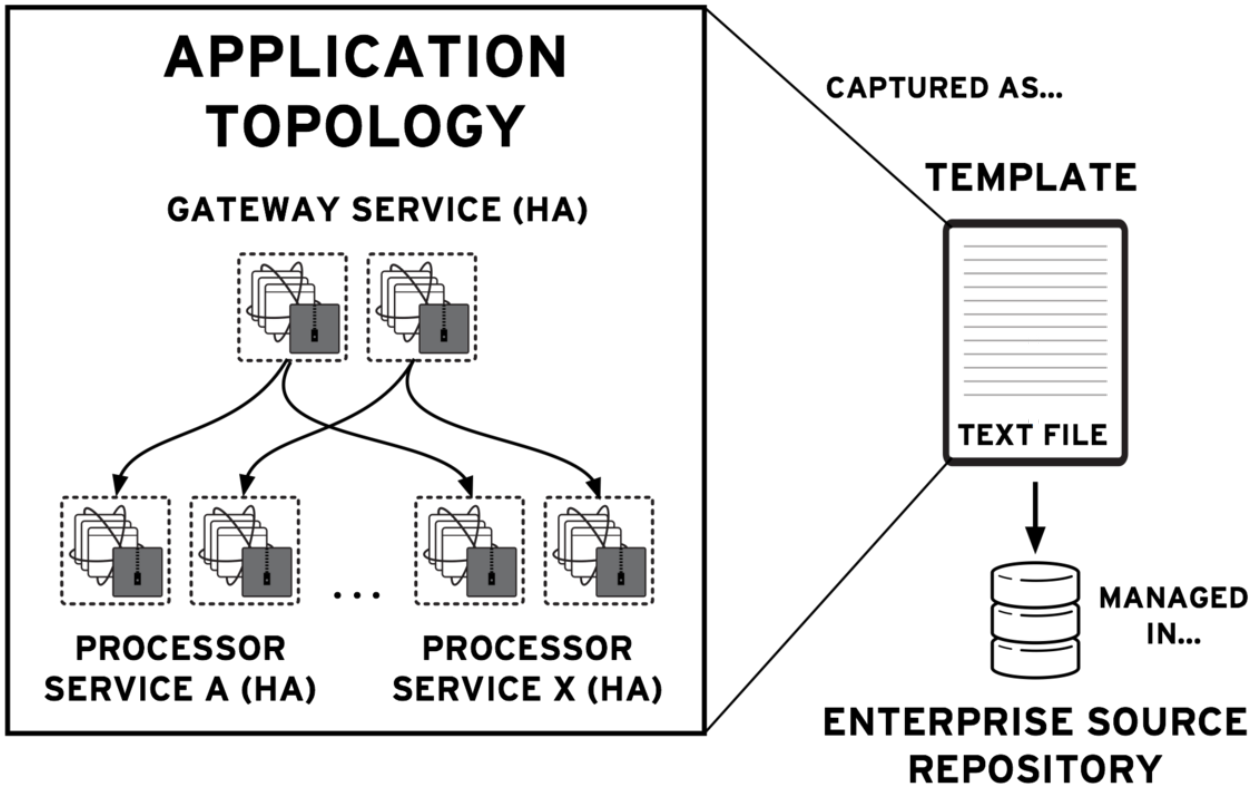
CONFIGURATION MANAGEMENT WITH OPENSHIFT

Images as Managed Artifacts



CONFIGURATION MANAGEMENT WITH OPENSHIFT

Application Topology as Managed Artifact



AUTOMATED TESTING

Definition

Automate tests beyond unit, including integration, system, functional, and even some non-functional acceptance tests (performance, security, etc.)

Trigger tests from the continuous integration process

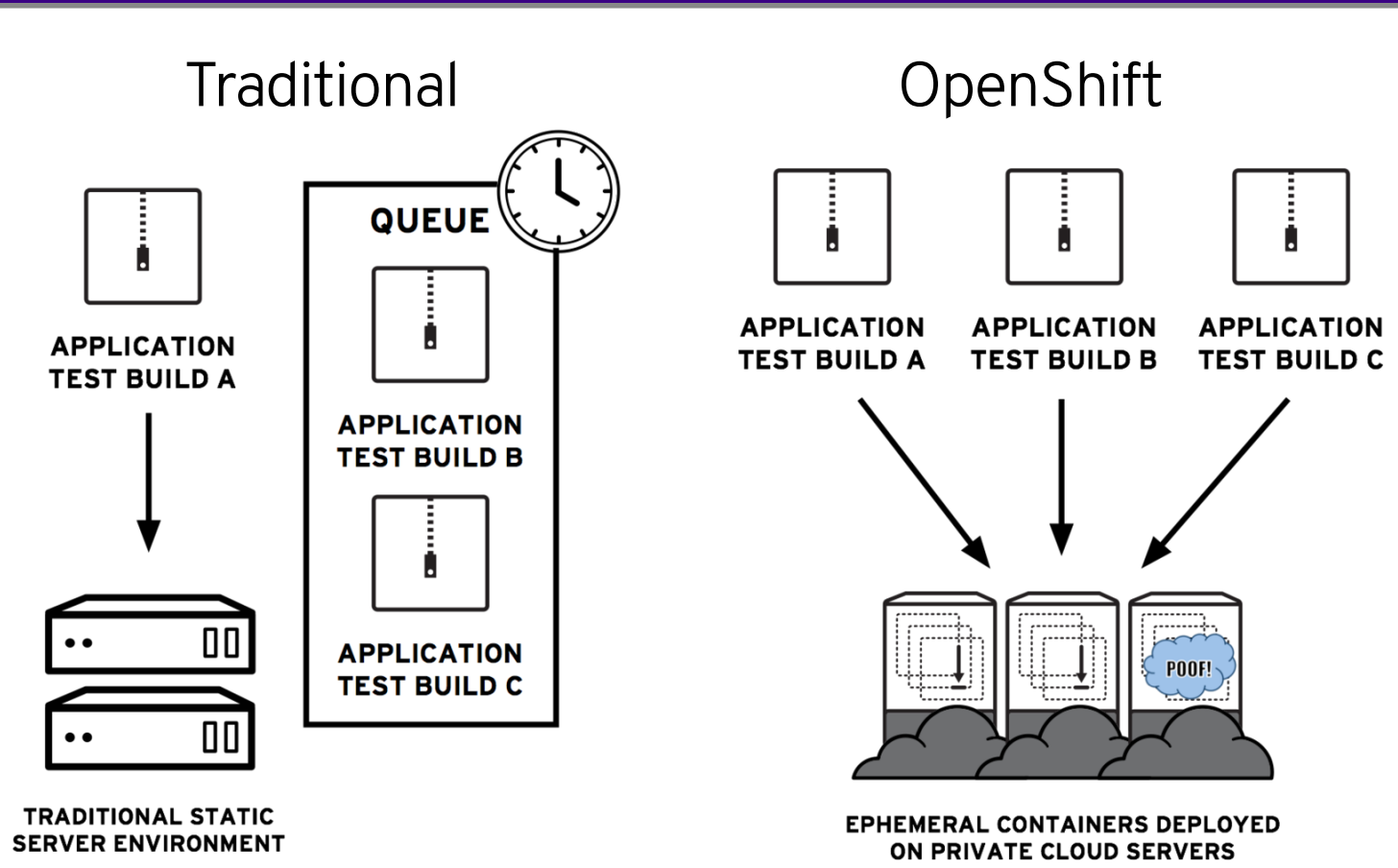
Benefits

- Supports **rapid development** by providing quick feedback (through CI process) on functional breaks, performance degradation
- Safeguards against regression when refactoring

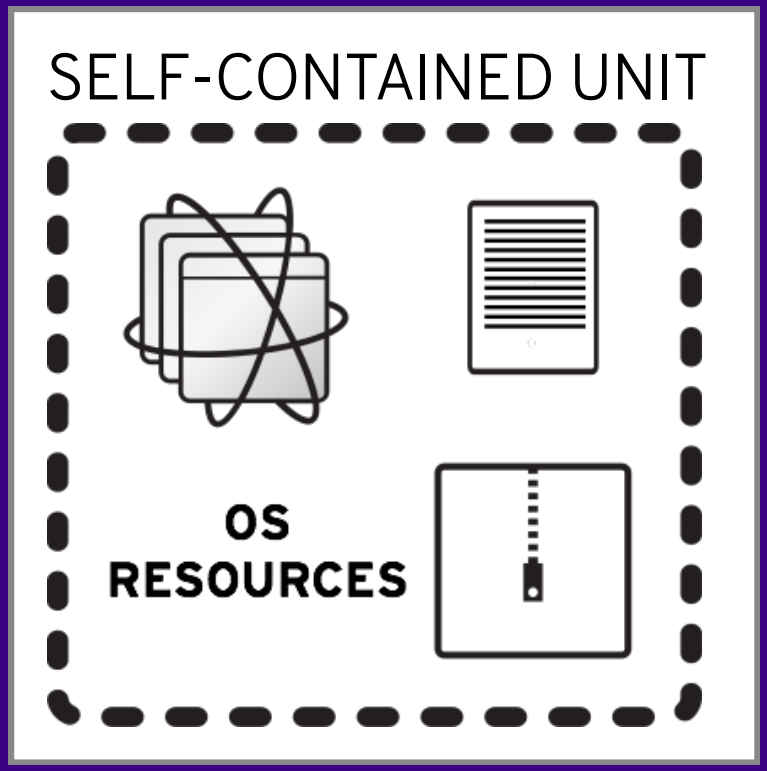
Tools

Automated functional and behavior-driven development test suites

AUTOMATED TESTING WITH OPENSHIFT



AUTOMATED TESTING WITH OPENSHIFT



TESTS RUN ON ALL COMPONENTS:

- OS resources
- Application server
- Configuration
- Application
- Libraries
- And the **interactions of all of these**

CONTINUOUS INTEGRATION

Definition

Every time somebody commits a change, the entire application is built and a comprehensive set of automated tests are run against it. (Humble and Farley, 2011)

Requires frequent code check ins, good test coverage, preferably a CI server

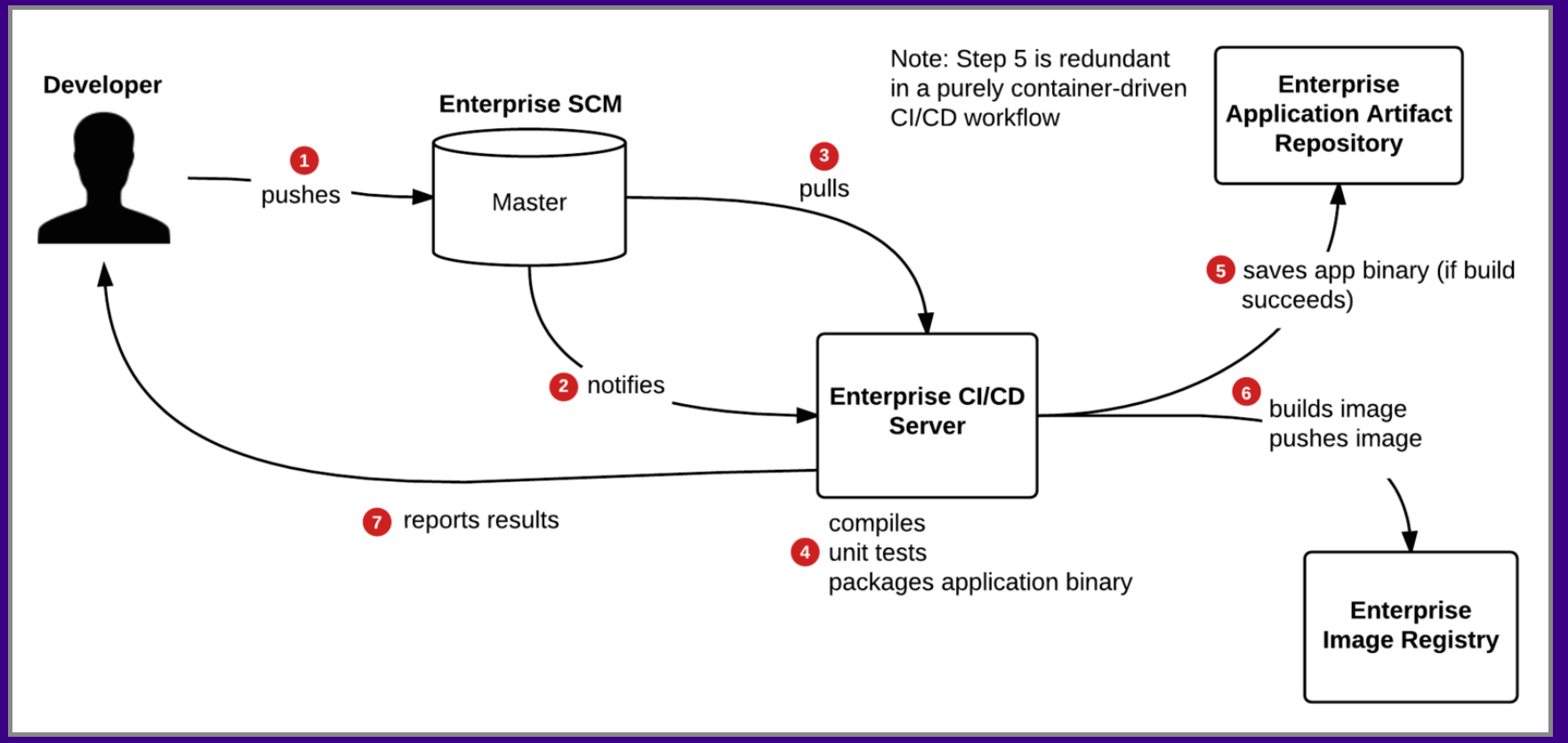
Benefits

- Normal state of the application is working, functional
- If the application is broken, it is treated as abnormal and requiring immediate attention

Tools

- Version control
- CI server

CONTINUOUS INTEGRATION WITH OPENSHIFT



MANAGEMENT AND MONITORING

Definition

Having tools that provide fine grained detail on all aspects of the application lifecycle:

- Design-time API governance
- Build-time quality metrics
- Feature traceability
- Run-time application and platform behavior

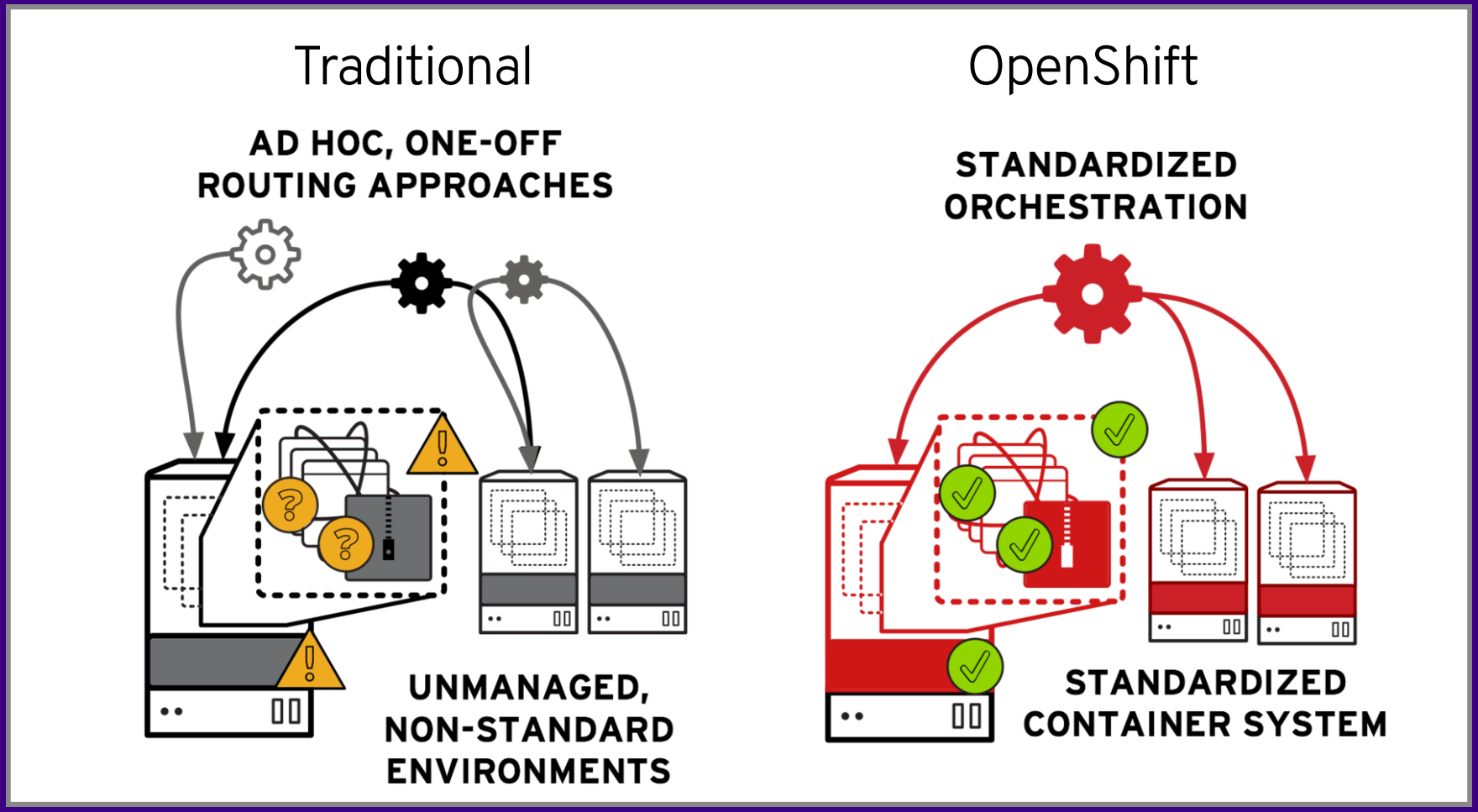
Benefits

- Ability to gauge software quality and infrastructure performance
- Ability to gauge DevOps program improvement

Tools

- Application monitoring suites
- Cloud management tools

MANAGEMENT AND MONITORING



Standardized container system and orchestration leads to standardized management and monitoring, driving down MTTR.

DEPLOYMENT PIPELINES

Definition

Well-described, automated, measured, and continually optimized process for moving an application through the life cycle from idea to production

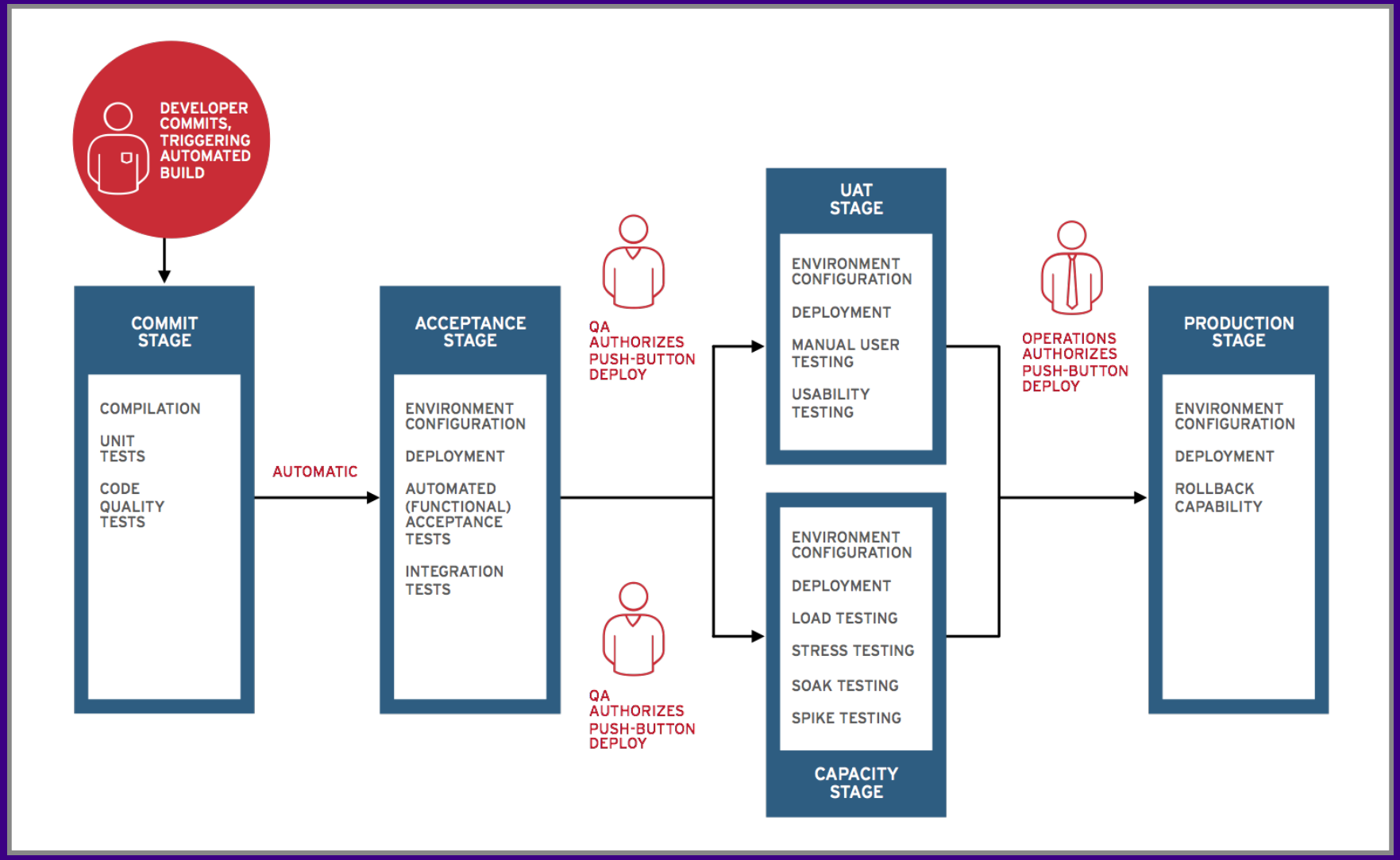
Benefits

- Process control over releases: Releases **cannot** go to production without passing through all prior stages of validation
- Optimization of the entire delivery process: Understanding where bottlenecks are and means to reduce them

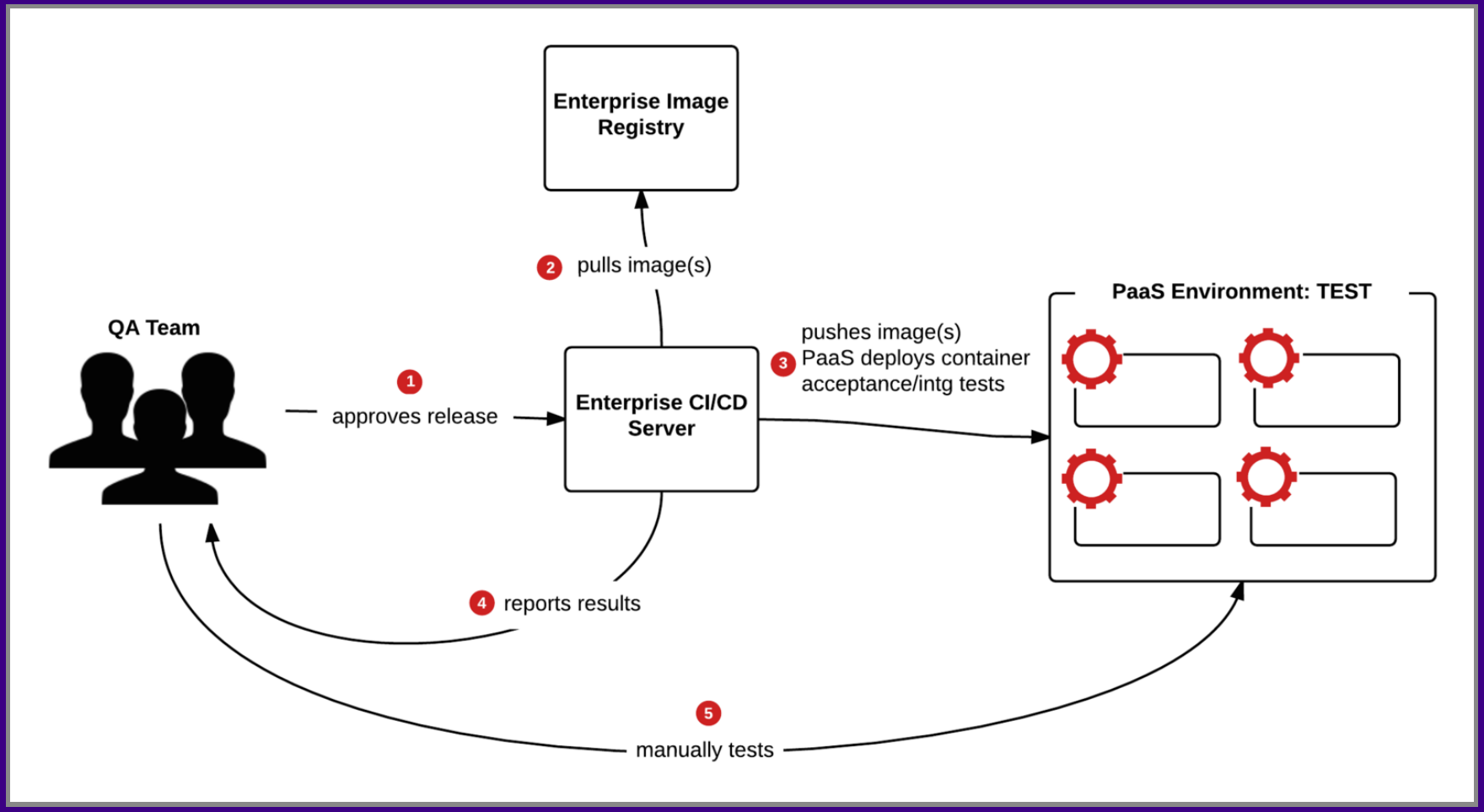
Tools

- Self-service requirements of deployment pipelines require mature automation of builds and deployments (including environment provisioning)
- Version control, binary management (e.g. Maven), CI/CD server

DEPLOYMENT PIPELINE EXAMPLE

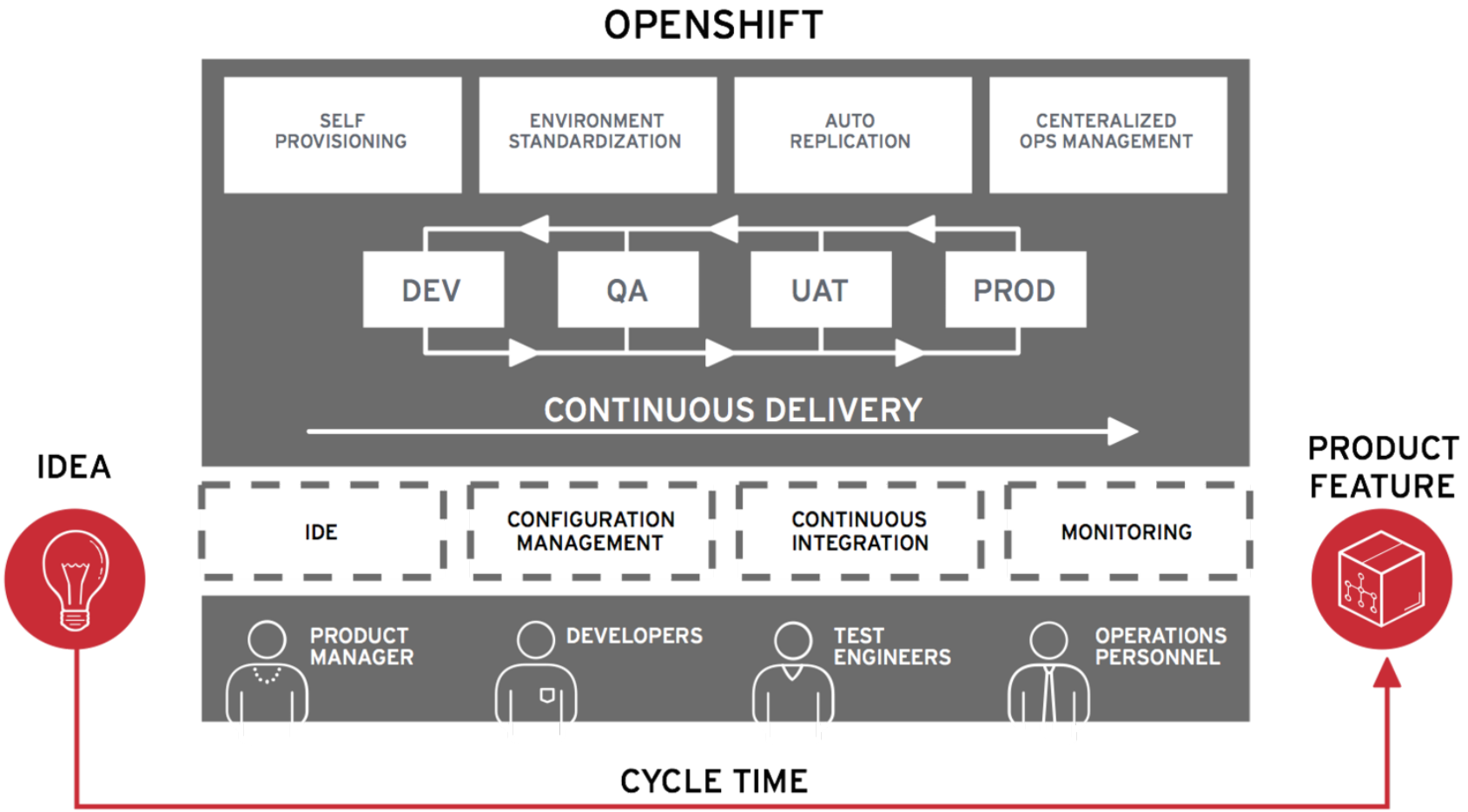


DEPLOYMENT PIPELINE WITH OPENSHIFT UAT STAGE EXAMPLE



DEMO (OSE 2)

OPENSHIFT AS DEVOPS ACCELERATOR



QUESTIONS

redhat.