표준 컨테이너와 Cloud-Native 개발환경을 활용한 신속한 애플리케이션 개발

Hyunsoo Kim(hykim@redhat.com)
Senior Solutions Architect

Agenda

1. 표준 컨테이너
2. 컨테이너 오케스트레이션
3. 마이크로서비스 아키텍처를 위한 **Cloud-Native** 개발환경
4. 데브옵스 환경 구성을 위한 **CI/CD**
5. 컨테이너 애플리케이션의 운영

표준 컨테이너

Container ?

- OS 가상화 기술 (vs. HW 가상화 기술 a.k.a. VM)
- HW 가상화 기술보다 더 경량화
- 컨테이너 기술의 종류
  - LXC
  - OpenVZ
  - Solaris/HPUX
  - Rocket
  - OCI Container - Open Industry Standard Container

Established in June 2015

Create Open Industry Standards
around Container Formats and Runtime

https://www.opencontainers.org/

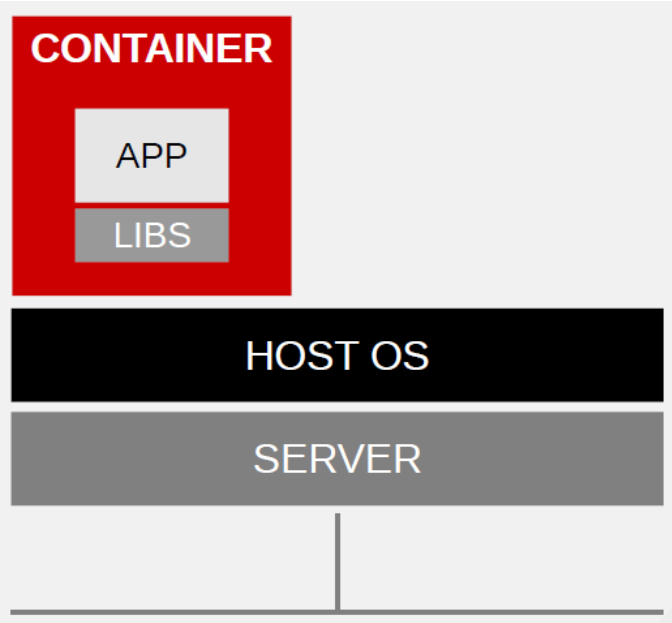
Managing/creating containers is

SIMPLE!

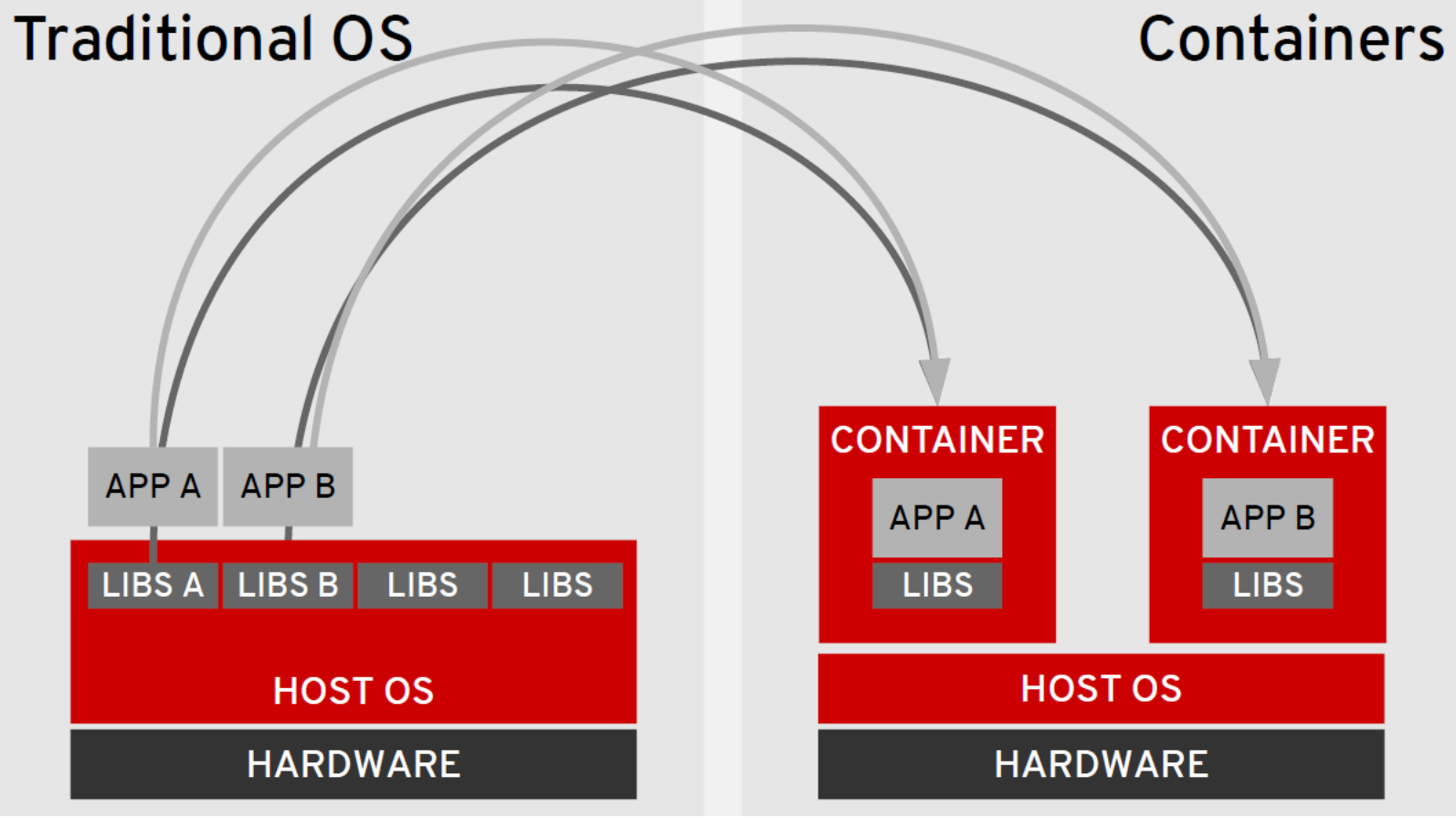
Software packaging concept

컨테이너 이미지 안에 필요한 소프트웨어와 의존성 라이브러리들을 같이 패키징하는 기술이 핵심

- 배포 및 라이프사이클 관리의 복잡성을 줄임
- 여러 대의 **Host**에 배포 및 포팅이 쉬움
- **Host OS** 상에서 컨테이너별로 격리 기능 제공
- In RHEL, this is done through:
  - Control Groups (cgroups)
  - Kernel namespaces
  - SELinux, sVirt, iptables
  - OCI Container

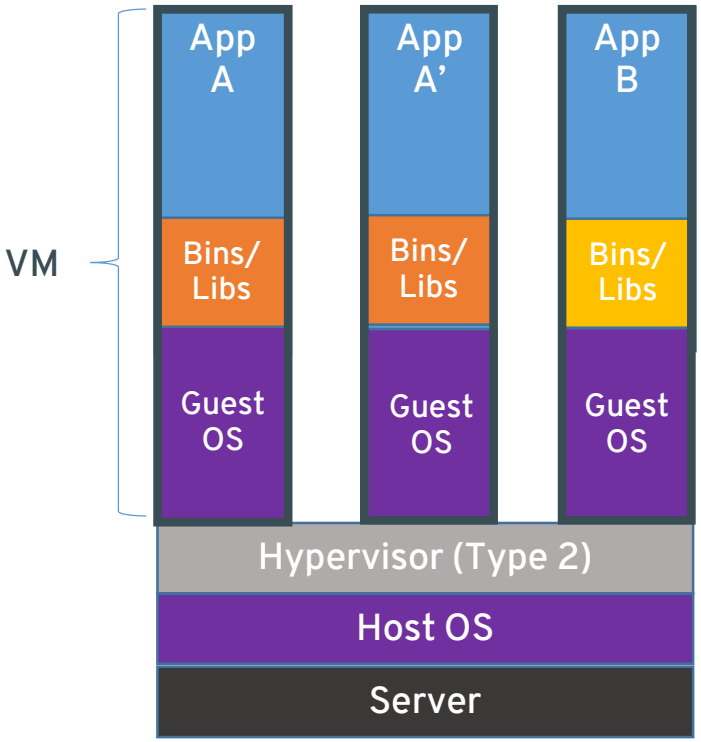


소프트웨어 패키징 = 보다 빠른 셋업 + 완벽한 이식성

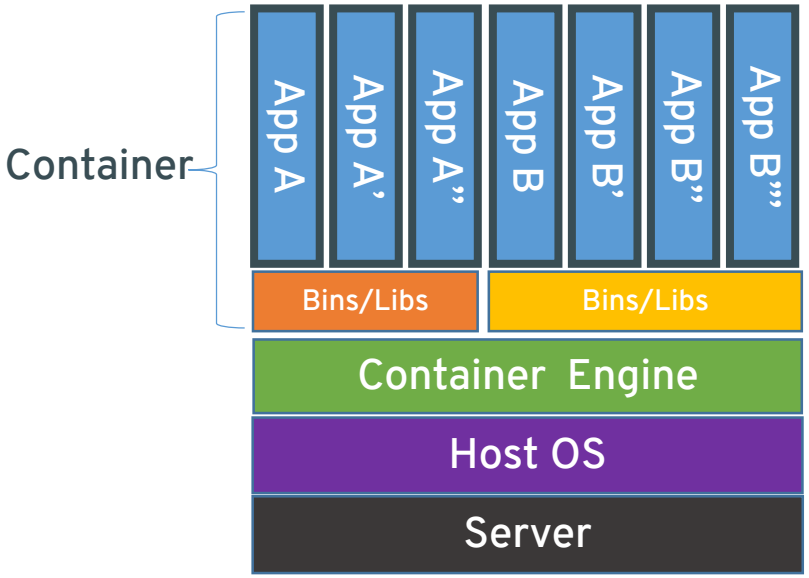


redhat.

1. Hypervisor나 GuestOS로 인한 overhead 없음
2. Guest OS 관리 부분의 부담이 없음
3. 빠른 startup
4. 인스턴스 증가시, 추가적인 설정작업 없음

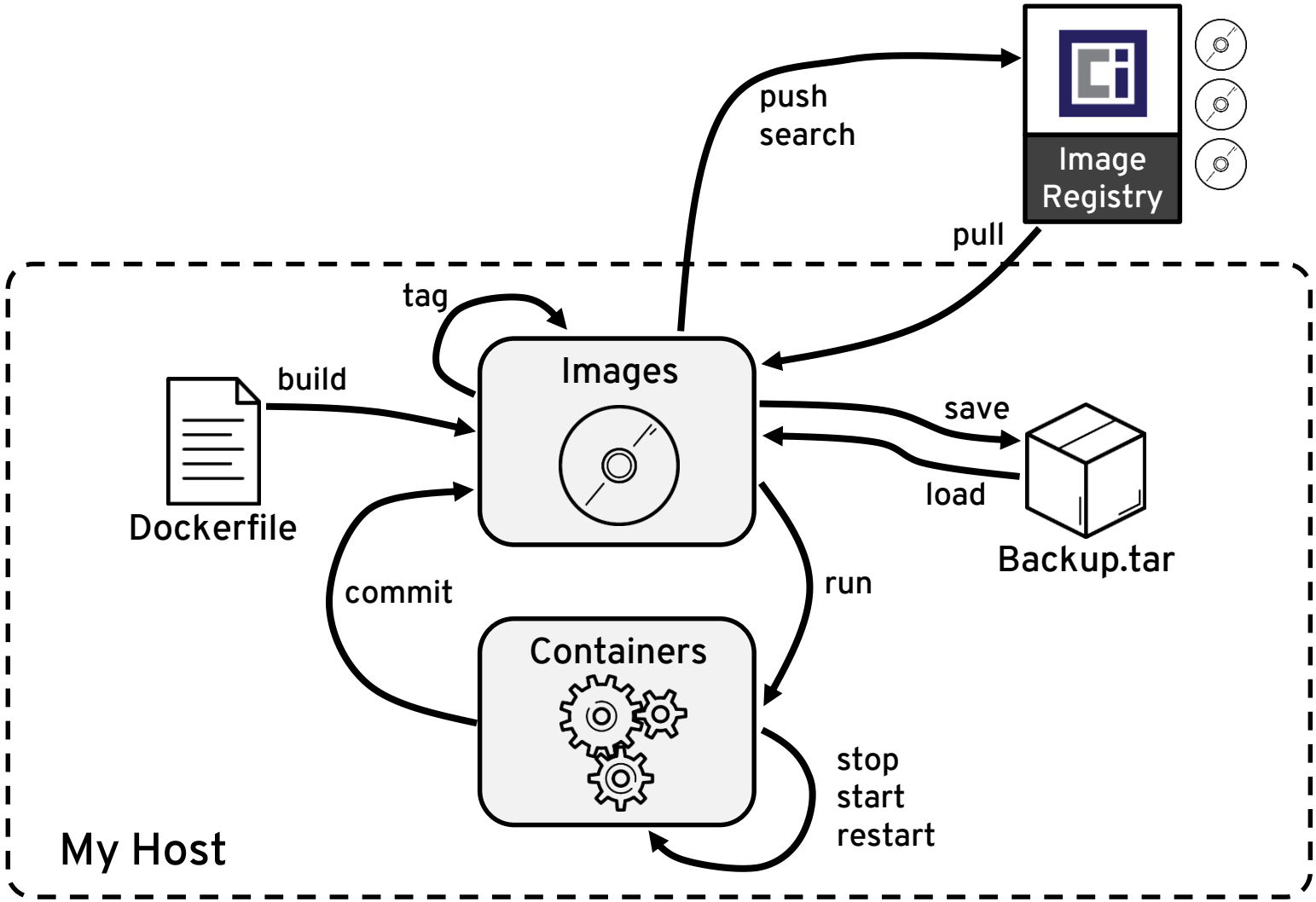


HW Virtualization



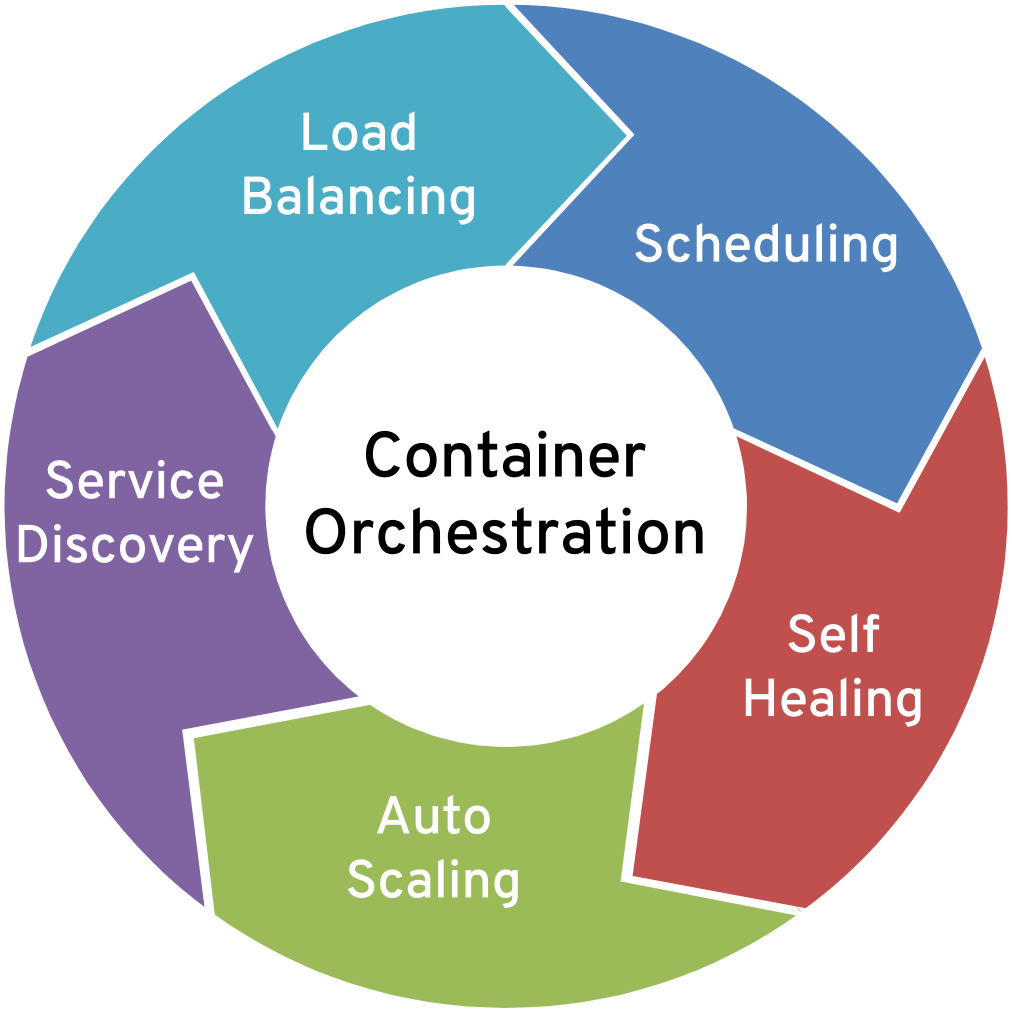
OS Virtualization

redhat.

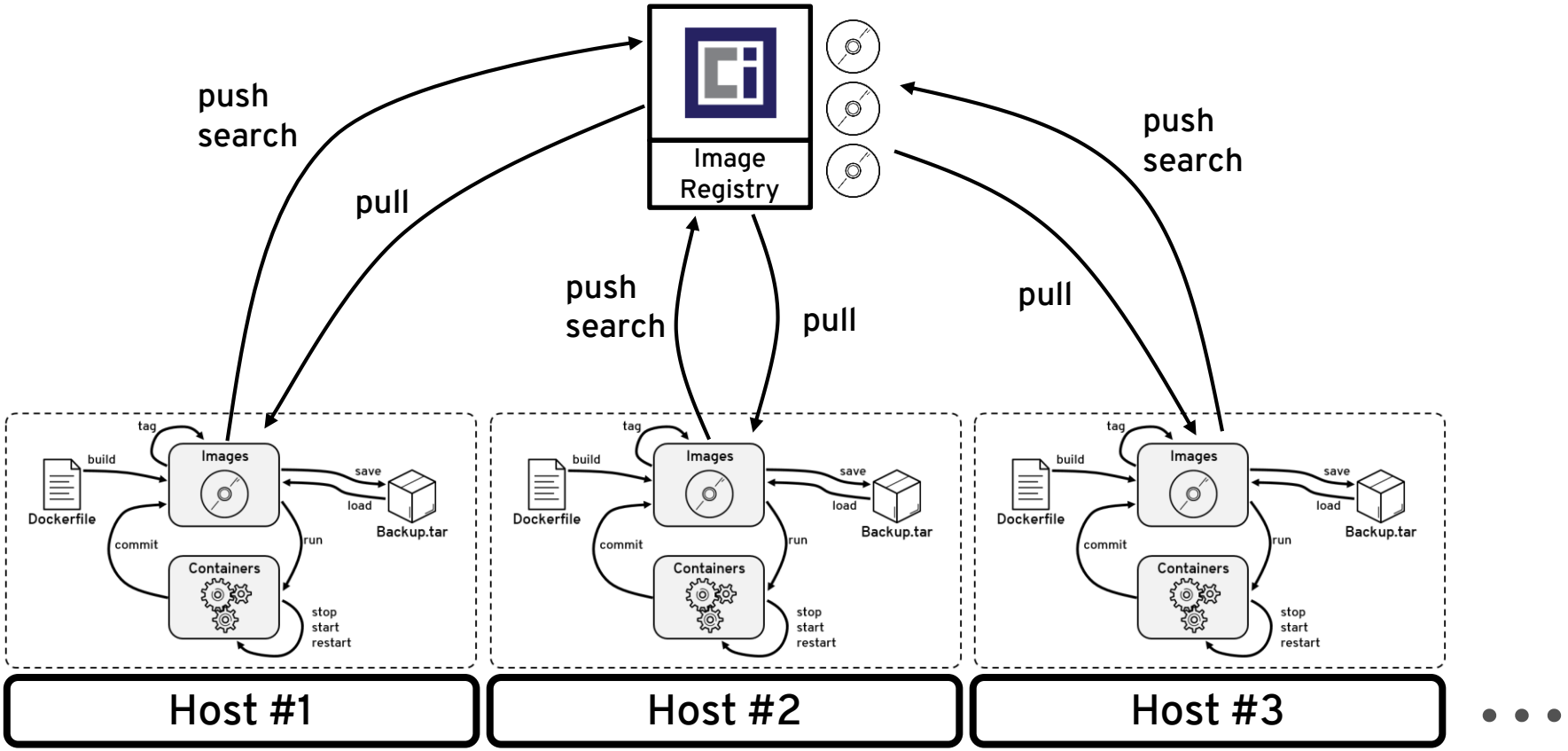

push
search
Image
Registry
pull
tag
build
Images
save
load
Dockerfile
Backup.tar
commit
run
Containers
stop
start
restart
My Host

컨테이너 오케스트레이션

멀티 호스트상의 멀티 컨테이너를 조직화하고 연결

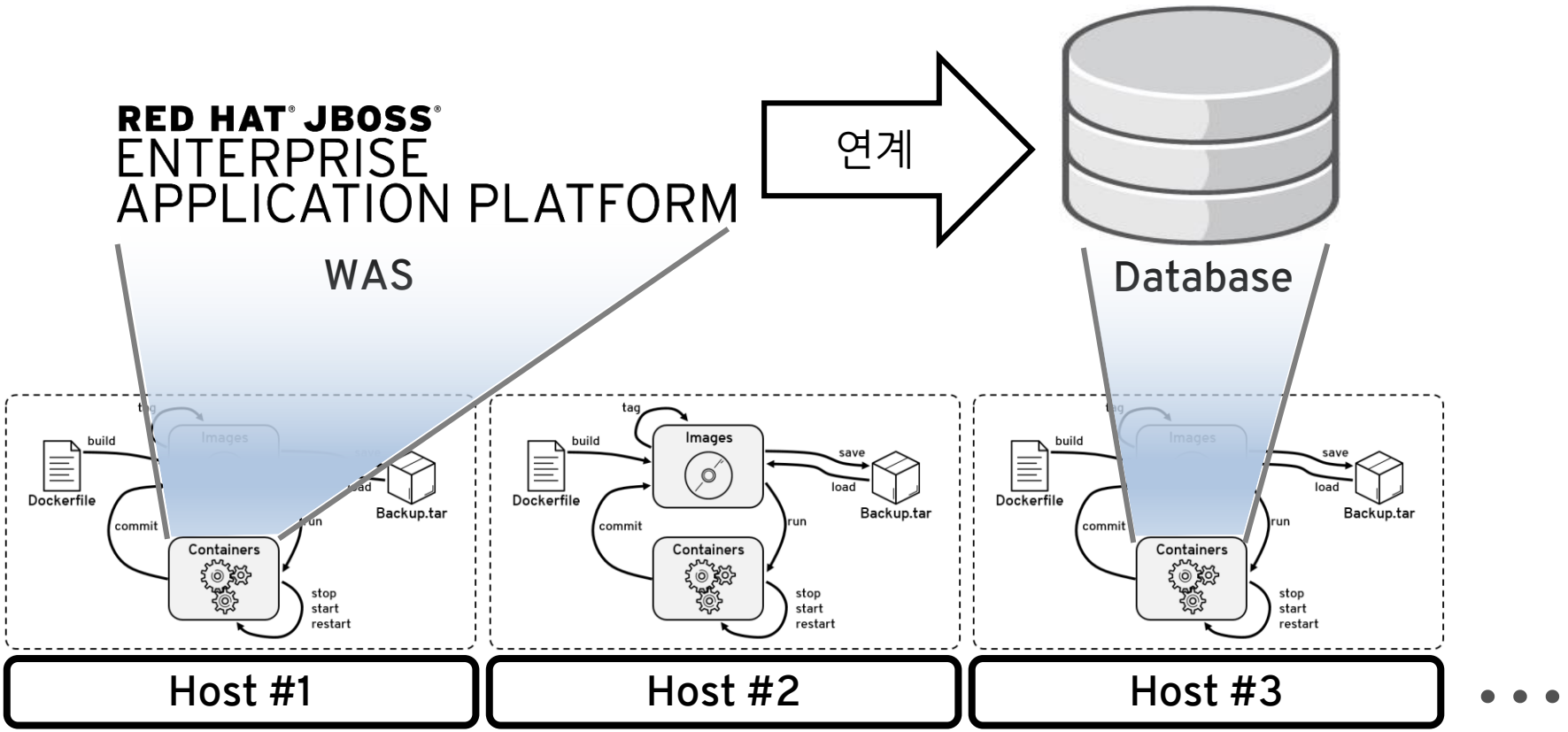


멀티 호스트상의 멀티 컨테이너를 배치



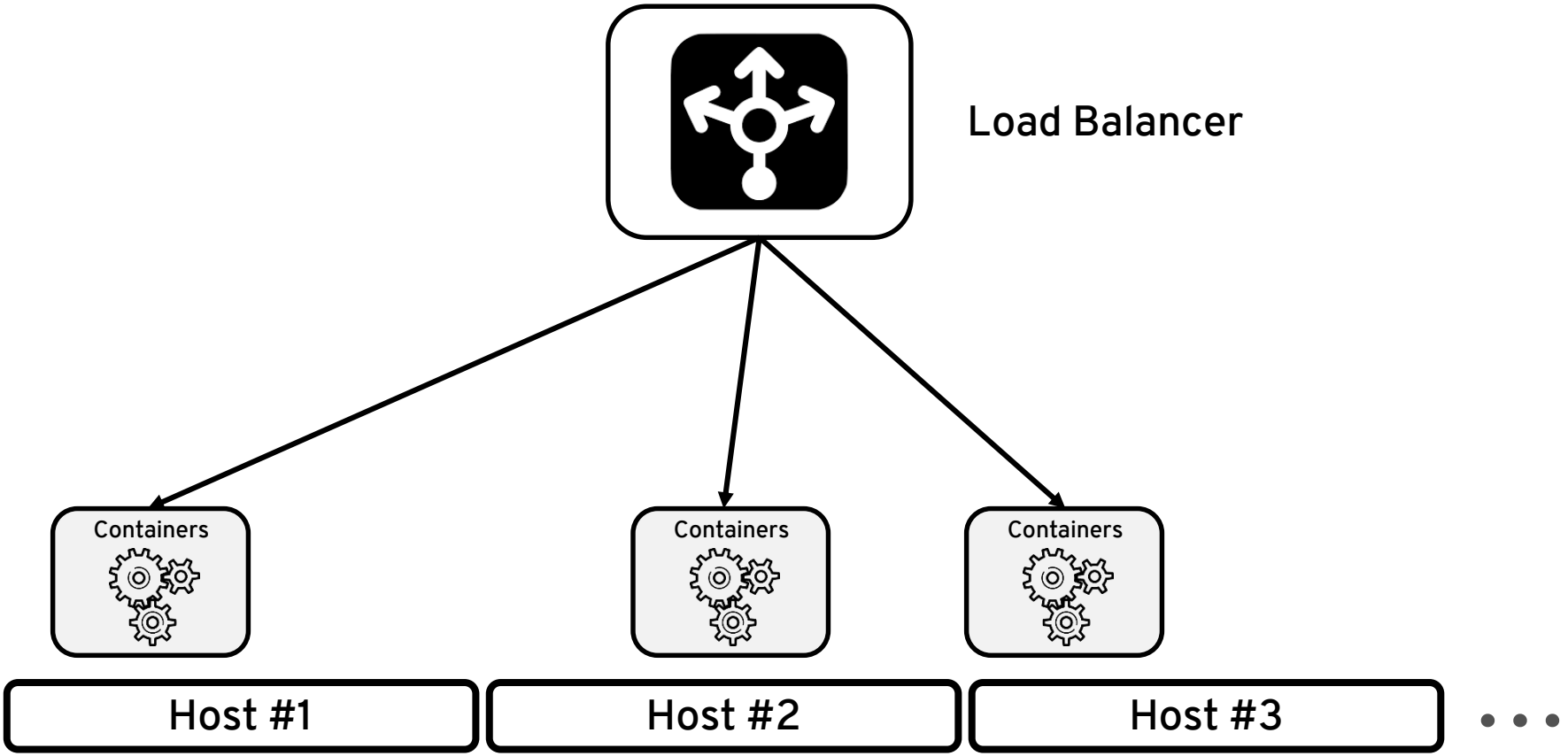
redhat.

컨테이너간 연계 자동화

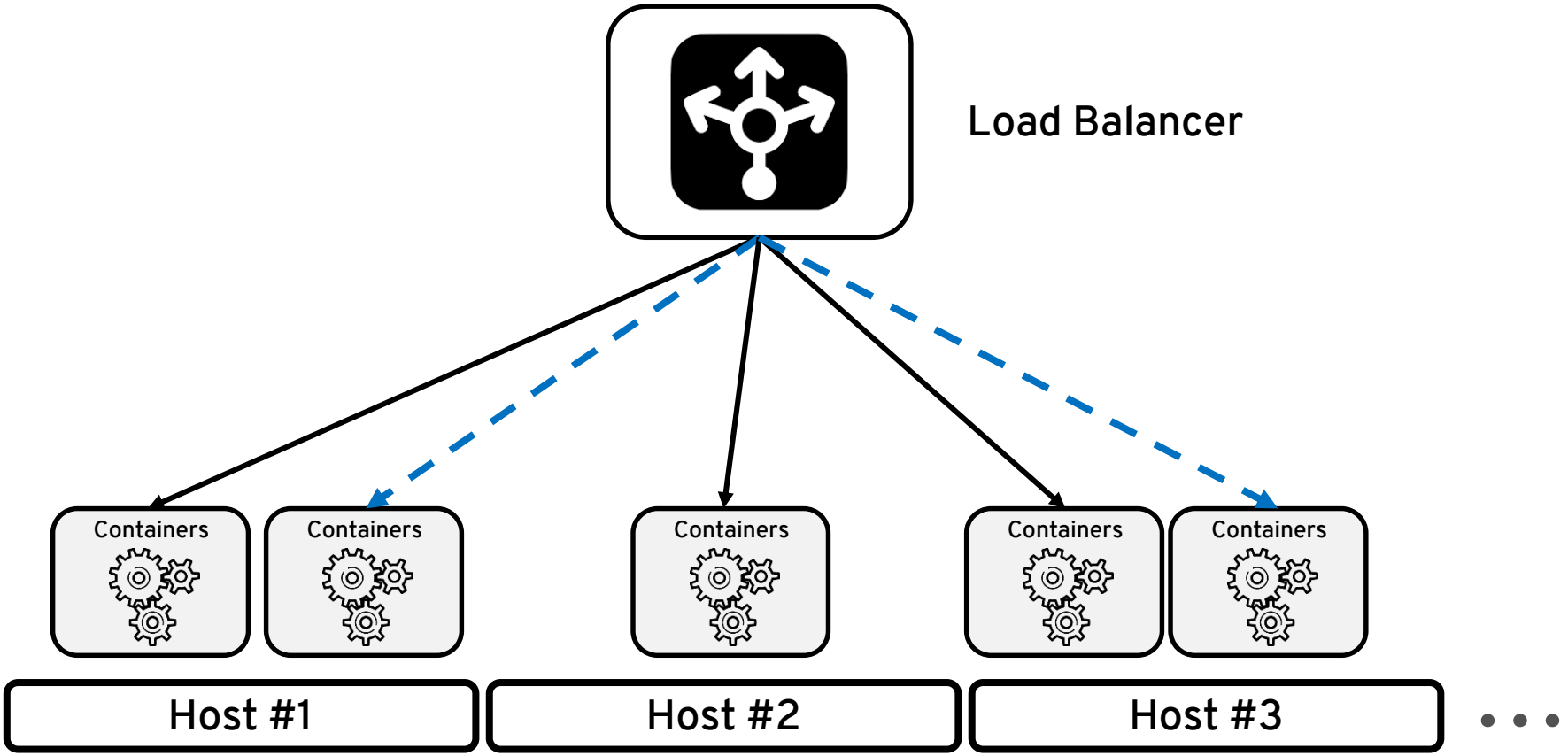


redhat.

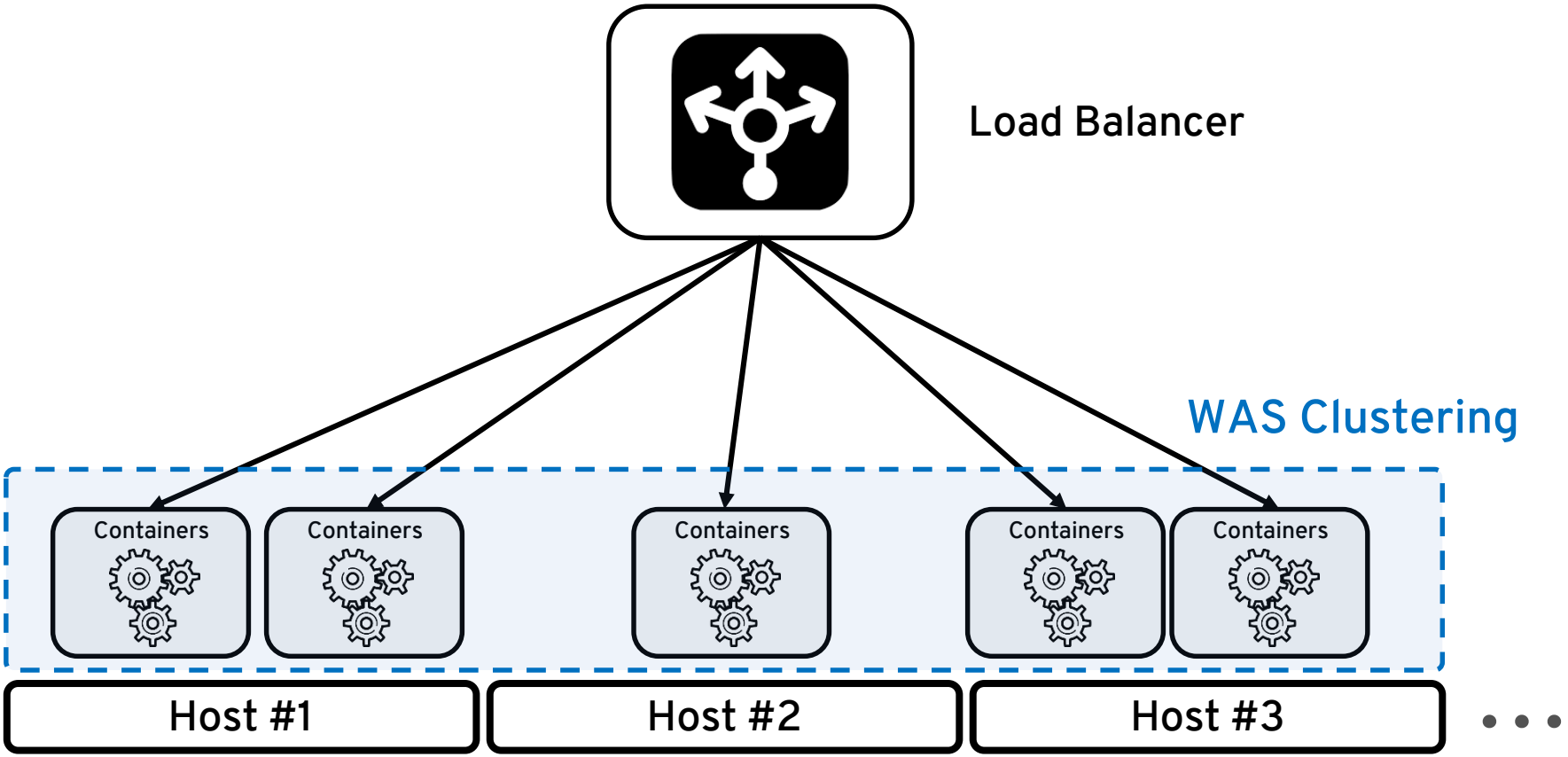
컨테이너간 부하분산 자동화



컨테이너 확장/축소 자동화



WAS 컨테이너간 Session Clustering 자동화



2014.6 구글 컨테이너 관리 프로젝트를 오픈소스화

Red Hat 은 kubernetes 프로젝트에 참여하고 협업 진행 지속

de facto standard

kubernetes 고려 사항

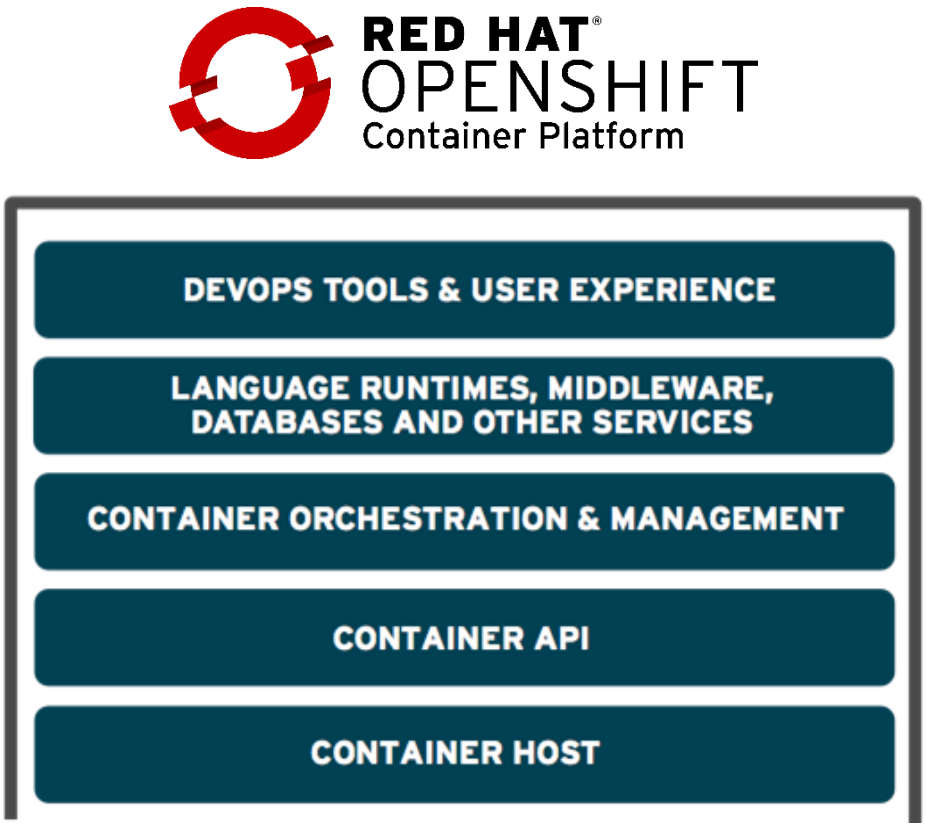
- 소스 빌드, 컨테이너 이미지 생성 자동화 별도 구성
- 컨테이너간 네트워킹을 위한 **Network** 별도 구성
- 컨테이너 이미지 버전 관리용 **Registry** 별도 구성
- **Load Balancer** 별도 구성
- **WAS Clustering(Session Clustering)** 기능 미지원
- **H/W** 확장시 수동 설치/구성

OCI 표준 컨테이너 기반 컨테이너 플랫폼

엔터프라이즈급 **kubernetes**

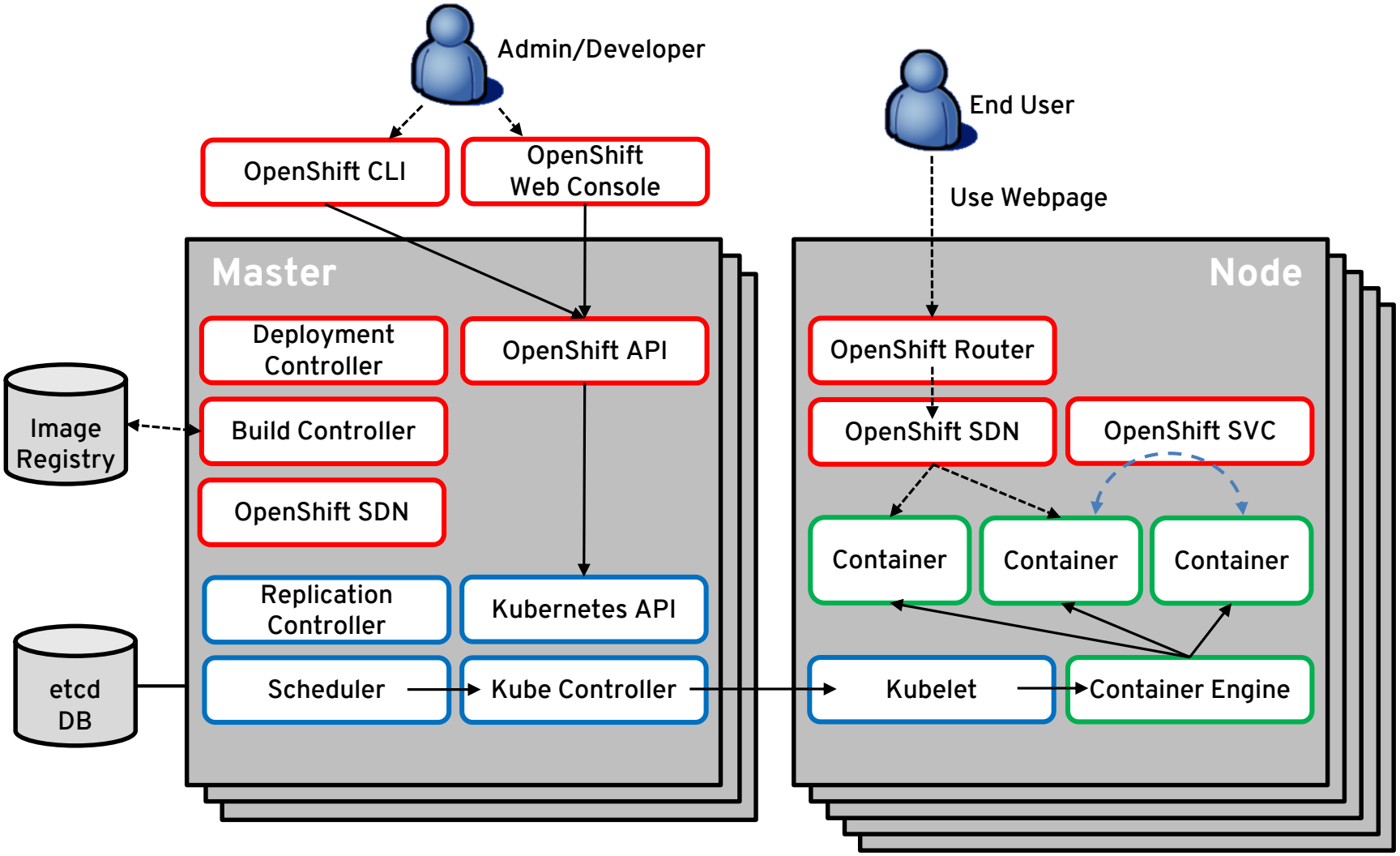
Red Hat OpenShift Products

redhat.



- 개발 및 운영을 위한 다양한 툴 제공(IDE, UX 등)
- Autoscaling 기능 제공
- 다양한 application runtimes & services 제공
- kubernetes 기반의 컨테이너 오케스트레이션 및 관리
- OCI 표준 컨테이너
- 컨테이너 최적화 OS 기반
  - Red Hat Enterprise Linux
    (CC EAL4+ 인증)

OpenShift Container Platform Architecture



redhat.

OpenShift Container Platform 특장점

- Bare-Metal, VM(vSphere, Hyper-V, RHEV 등), Public/Private IaaS 제약없이 설치
- Git 연동을 통한 소스 maven 자동 컴파일 및 배포
- Open vSwitch를 통한 컨테이너 Network 자동 구성
- 관리 및 모니터링을 위한 Web Console, REST API 및 CLI 제공
- 소스 버전 관리용 Git, Docker 이미지 버전 관리용 Registry 제공
- Router를 통한 자동 부하 분산
- H/W 확장시 ANSIBLE을 통한 쉬운 설치/구성
- 컨테이너 자동 확장/축소(Autoscaling)
- JBoss EAP를 통한 WAS 클러스터링(Session Clustering) 기능 제공

redhat.

마이크로서비스 아키텍처를 위한 **Cloud-Native** 개발환경

소프트웨어 개발은 경쟁을 위한 핵심

모든 회사는 소프트웨어 회사이기도 합니다

- 지난 30년간 30대 혁신 가운데 20건은 소프트웨어와 관련이 있었습니다

소프트웨어 릴리즈에 성공할수록 경쟁 능력이 증가됩니다

- 소비자는 소프트웨어의 지속적인 제공을 기대합니다
- 품질을 희생하지 않으면서, 변화하는 애플리케이션 요구에 신속하게 대응할수록, 경쟁 우위를 확보할 수 있습니다

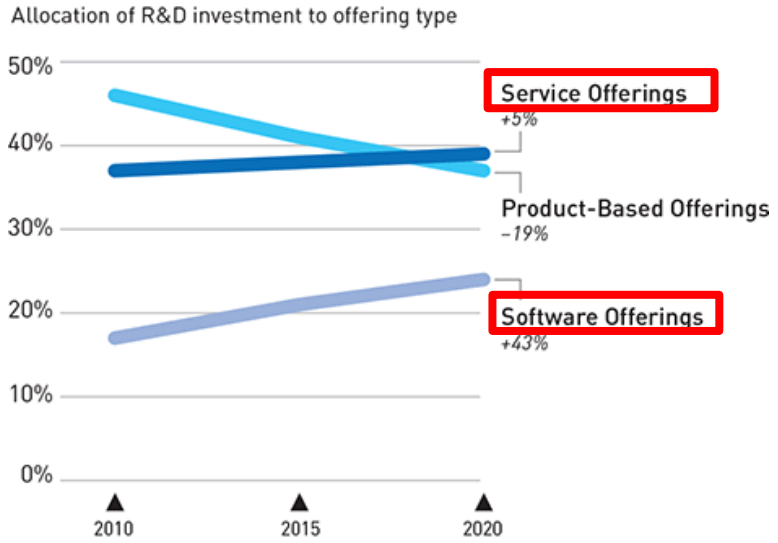
redhat.

You already know that software is eating the world,

this is how fast it's happening

Exhibit 3: **Shifting Priorities**

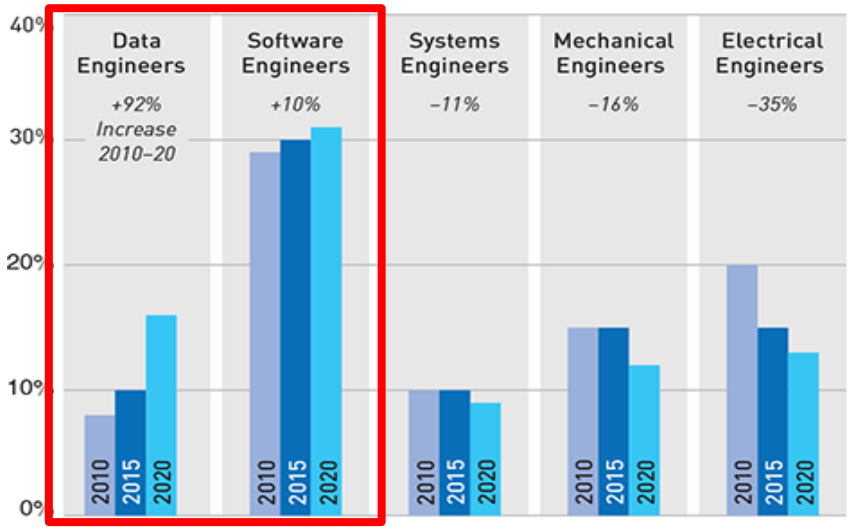
Companies are allocating higher proportions of their R&D budgets to software and service offerings.



Source: Strategy& analysis

Exhibit 4: **Allocating Talent**

Companies are focusing less on traditional disciplines like mechanical and electrical engineering and more on data and software engineering.



Source: Strategy& analysis

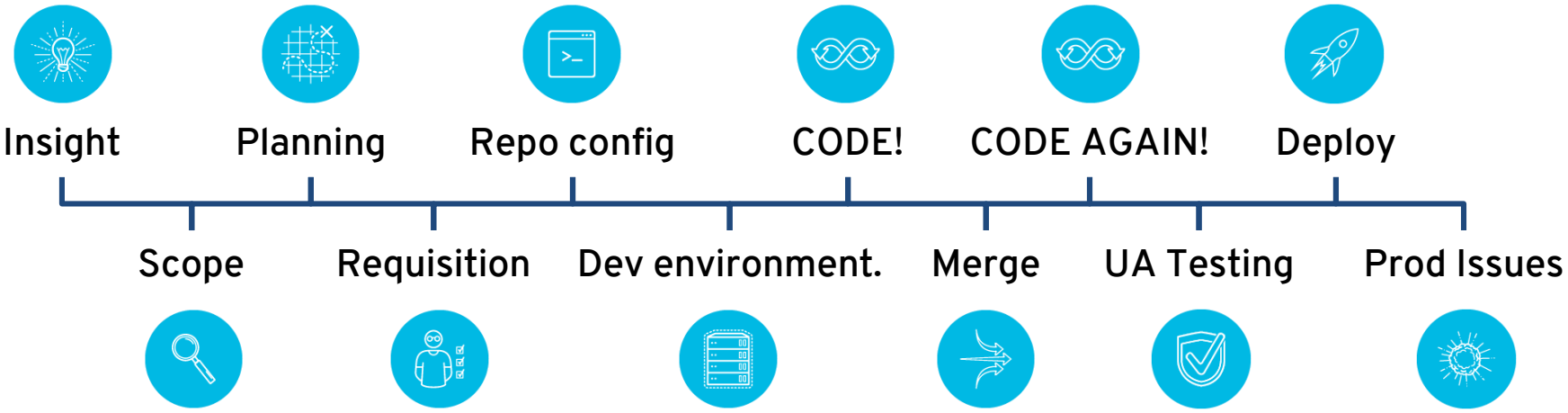
Source: http://www.strategy-business.com/feature/Software-as-a-Catalyst?gko=7a1ae

소프트웨어 개발 방식이 변화하고 있습니다

개발에 대한 전통적인 접근법이 무너지고 있습니다

- 도구와 환경을 설정하는 것은 너무 어렵고 시간과 비용이 많이 듭니다
- 개발자와 운영자간에 협업이 없으면, 막대한 비용이 소요되는 운영환경 상의 버그가 발생합니다
- 전체 프로젝트 팀은 가치와 품질을 지속적으로 제공해야합니다

통상적인 개발 타임라인



새롭고 더 나은 방법이 필요

통합된 Cloud-Native 개발 환경으로 이동

검증 가능한 환경을 사용하여, App를 평가하고 배포 제어

분석적으로 생성된 Insight를 사용하여 위험 완화

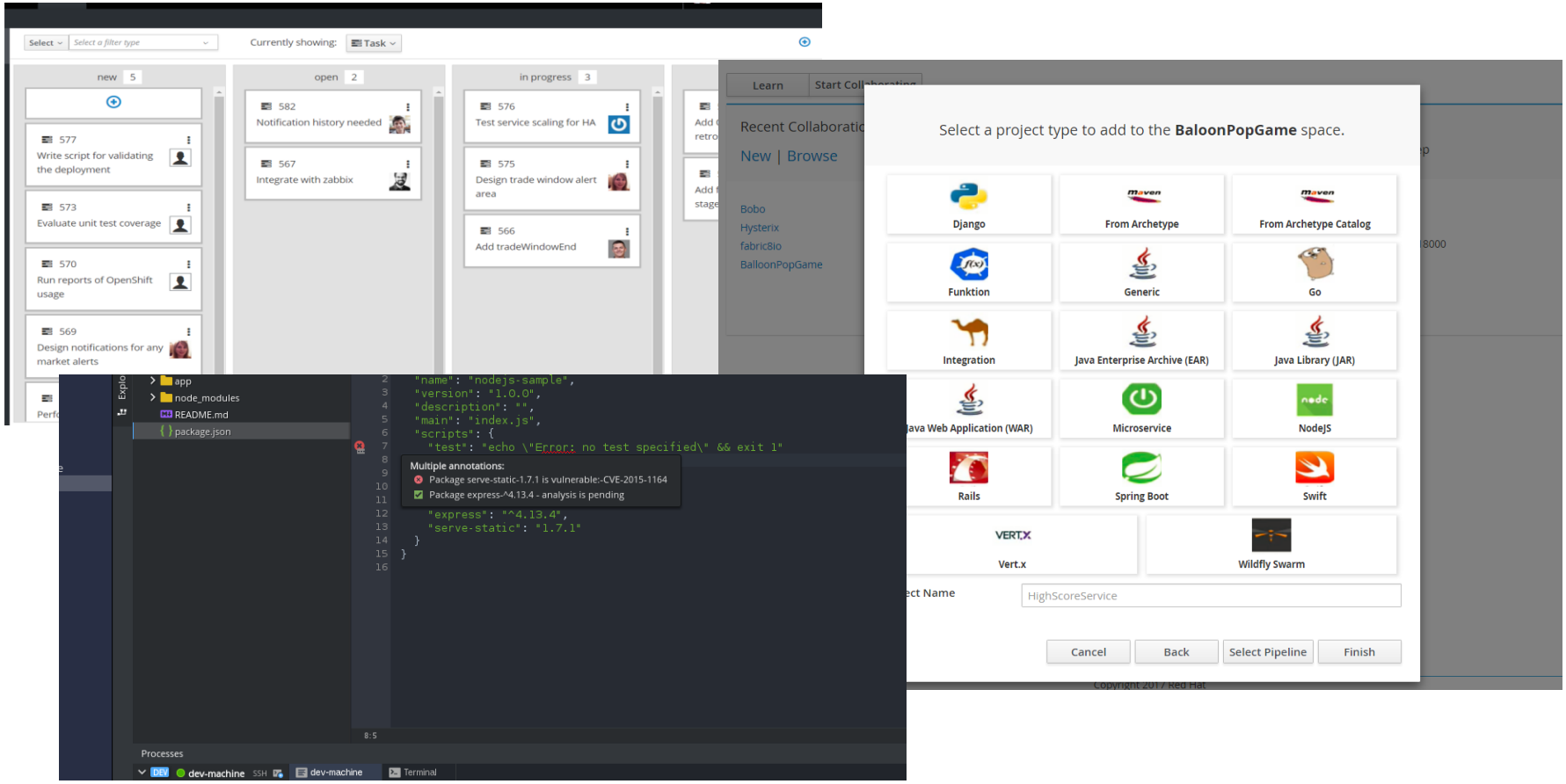
- 통합된 **Cloud-Native** 개발환경으로 비즈니스 민첩성 및 출시 시간 단축
- 값 비싼 프로덕션 전용 버그를 예방하기 위한 개발 환경과 프로덕션 환경 간의 차이점 제거
- 개발자는 필요할 때 필요한 리소스를 사용하고 사용하지 않을 때는 비용을 절감
- 개발의 모든 단계에서 **insight**를 통해 위험을 줄이고 자신감을 높임

JBoss Developer Suite
Red Hat Container Development Kit
OpenJDK

OpenShift.io

The world is rapidly moving to a cloud+container+service model

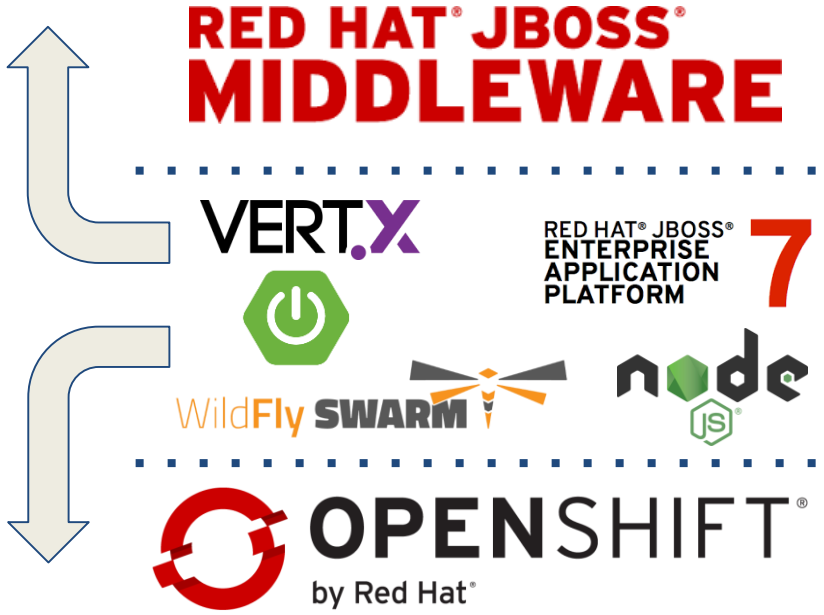
What is OpenShift.io?



- Free online end-to-end development environment at openshift.io
- Provides planning tools, web-based IDE, integrated CI/CD and reporting

redhat.

- 지원 가능한, 검증된 Microservice Architecture Application 런타임 제공
  - Spring Boot
  - WildFly Swarm (MicroProfile)
  - Eclipse Vert.x
  - Node.js
- 단순화된 클라우드 네이티브 App 개발
- OpenShift 및 Middleware와 응용 프로그램 런타임 통합
- 손 쉬운 사용법 제공

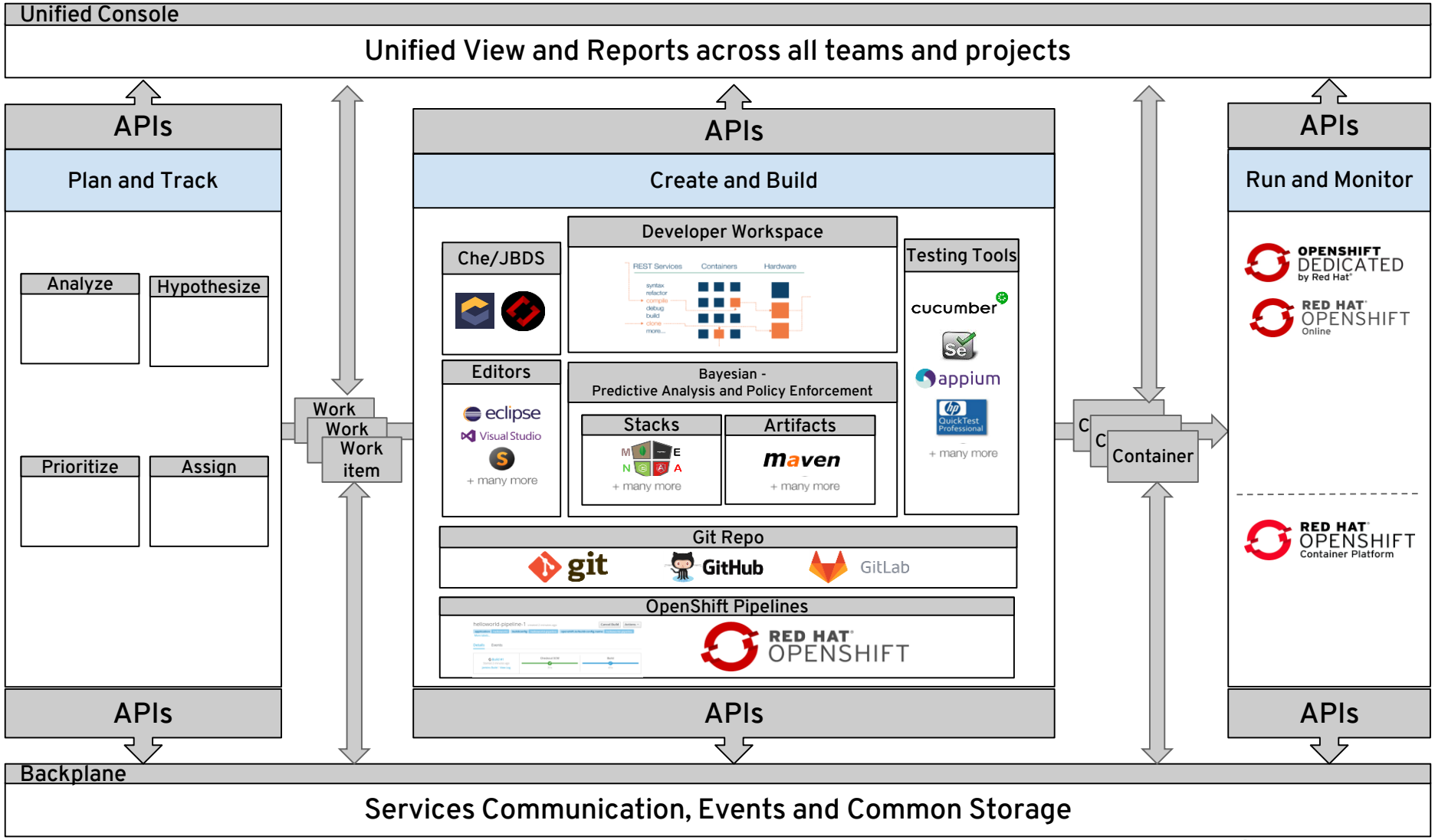


OpenShift.io 데모 동영상

redhat.

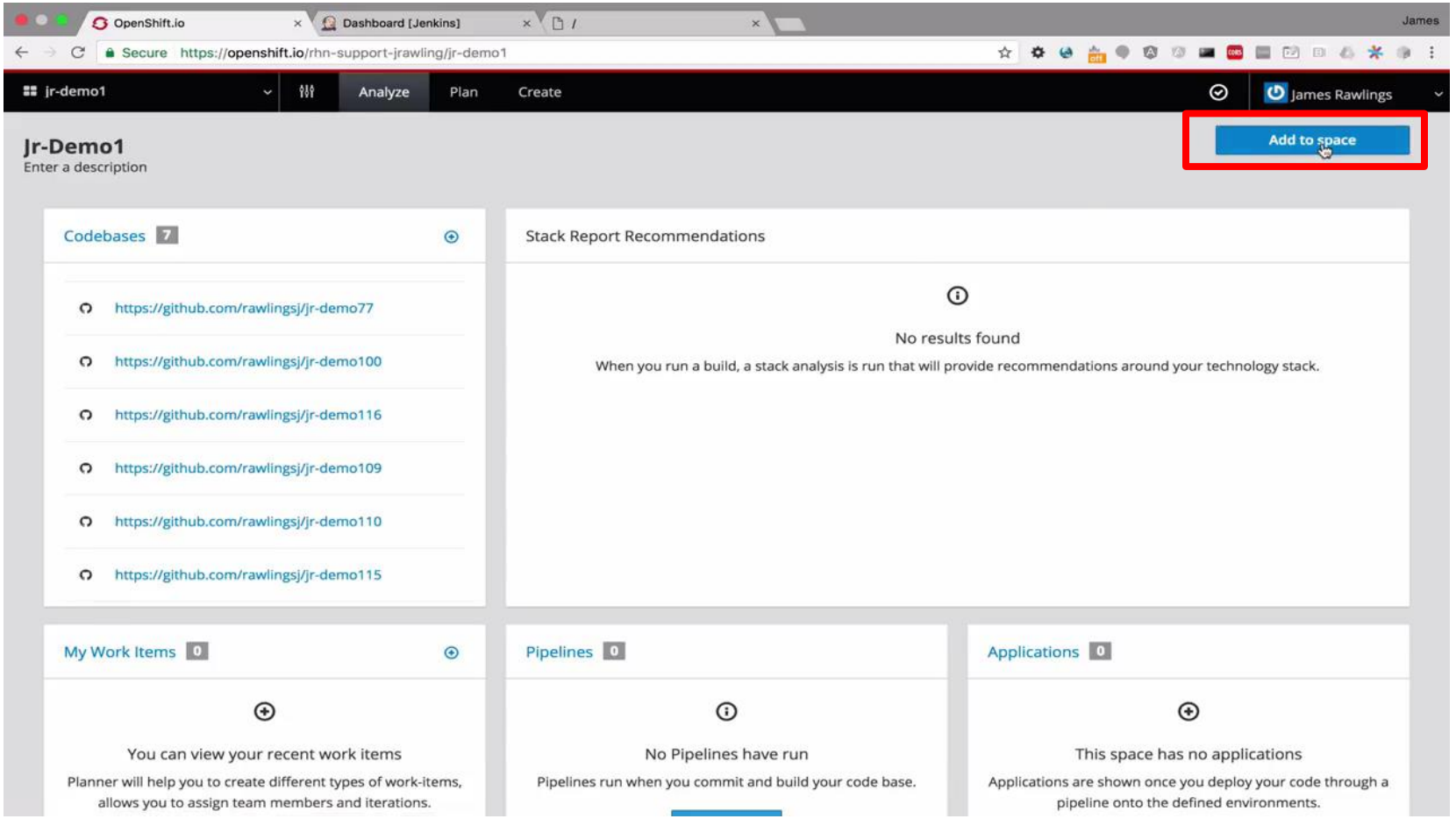
데브옵스 환경 구성을 위한 CI/CD

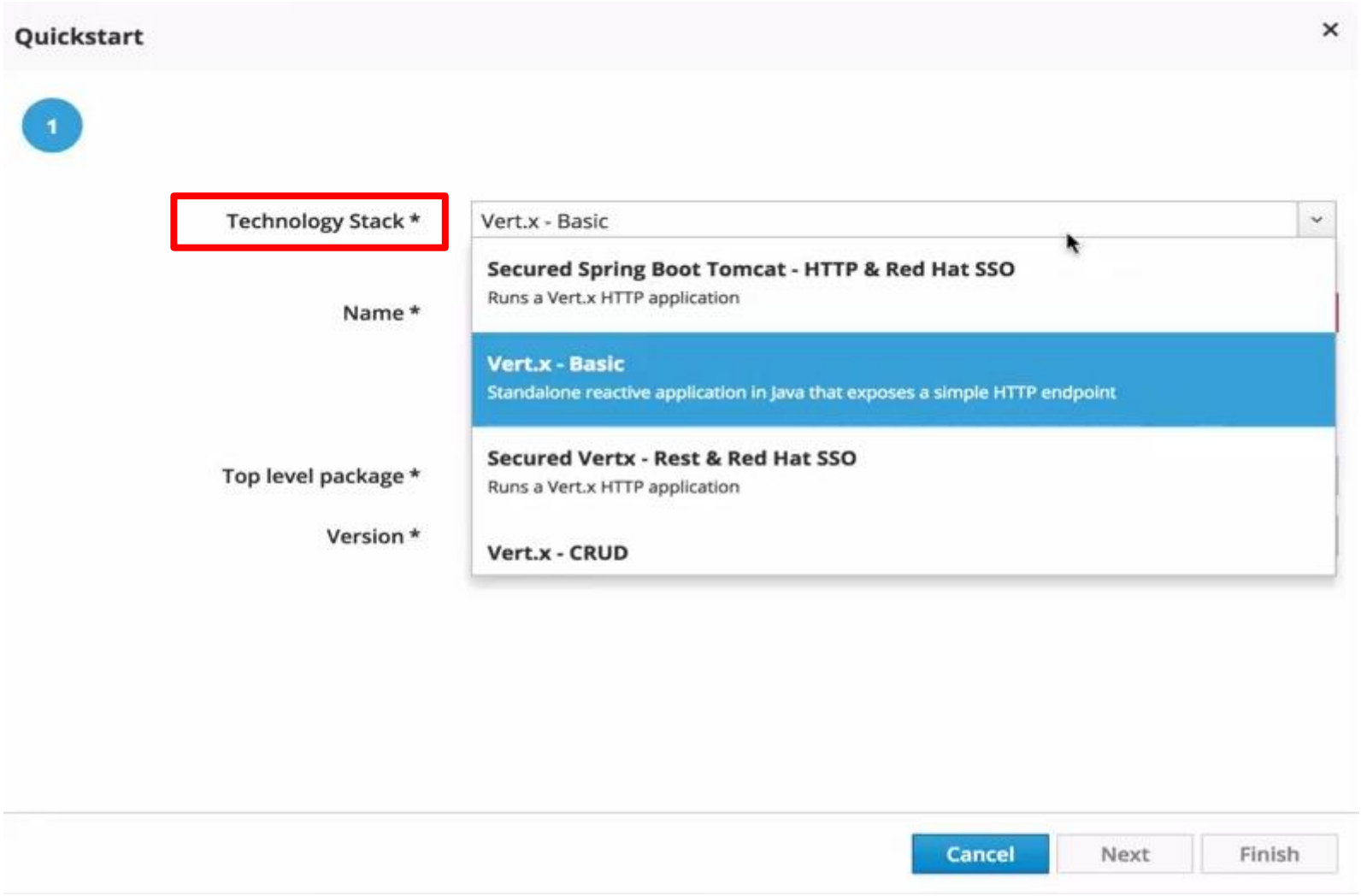
OpenShift.io 아키텍처



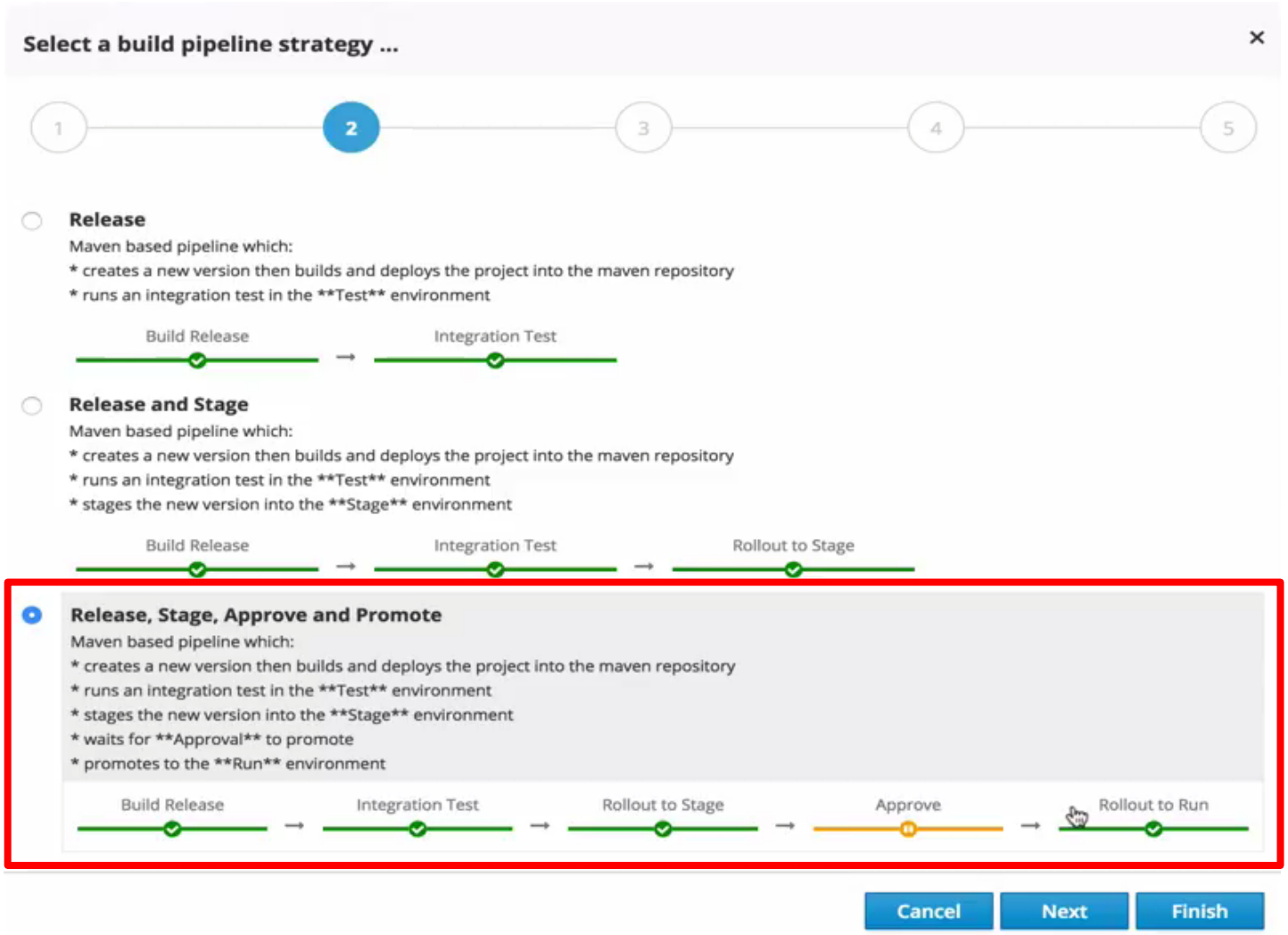
redhat.

Add to space



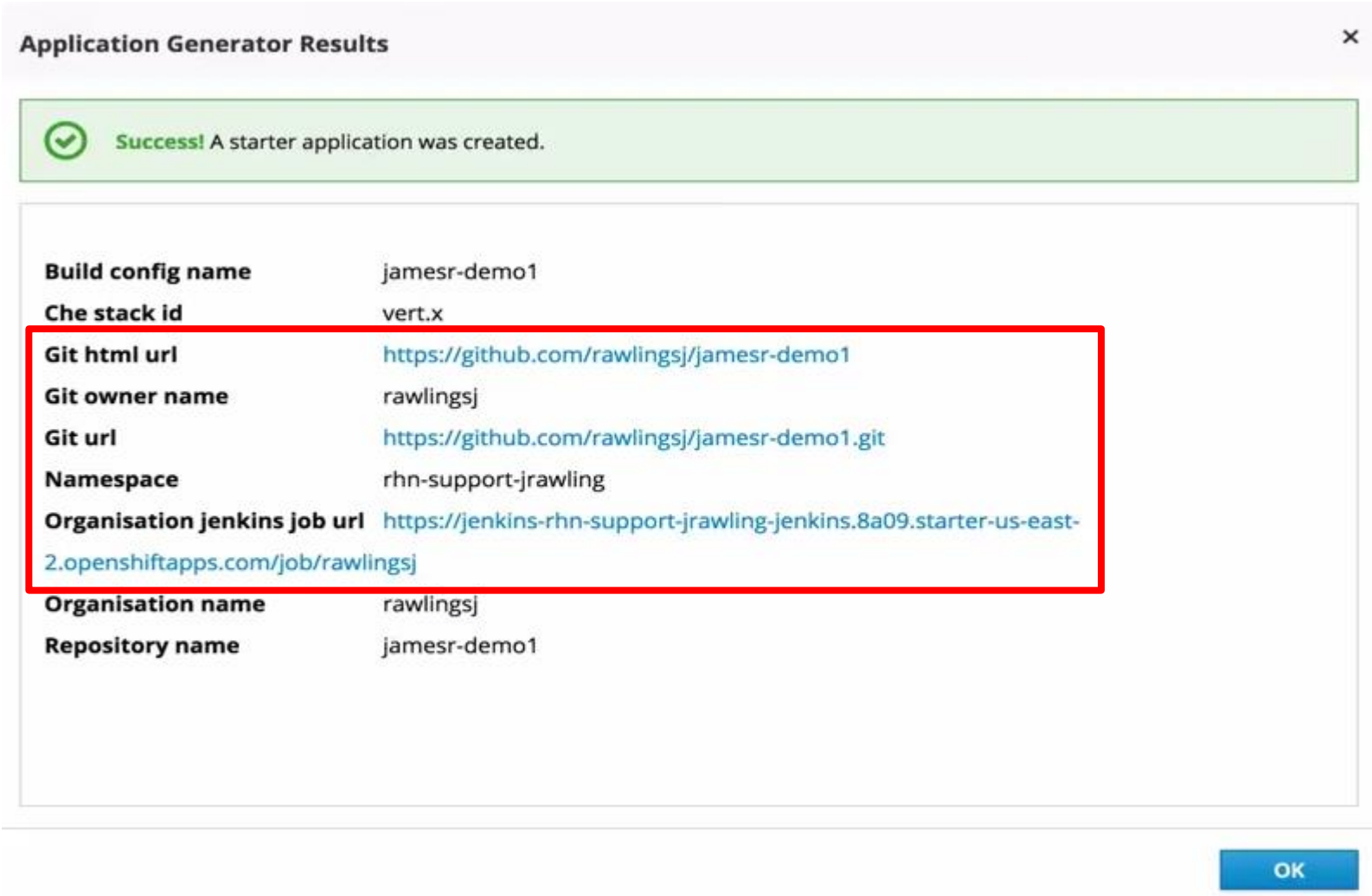

Quickstart
1
Technology Stack *
Name *
Top level package *
Version *
Vert.x - Basic
Secured Spring Boot Tomcat - HTTP & Red Hat SSO
Runs a Vert.x HTTP application
Vert.x - Basic
Standalone reactive application in java that exposes a simple HTTP endpoint
Secured Vertx - Rest & Red Hat SSO
Runs a Vert.x HTTP application
Vert.x - CRUD
Cancel
Next
Finish

Build Pipeline Strategy

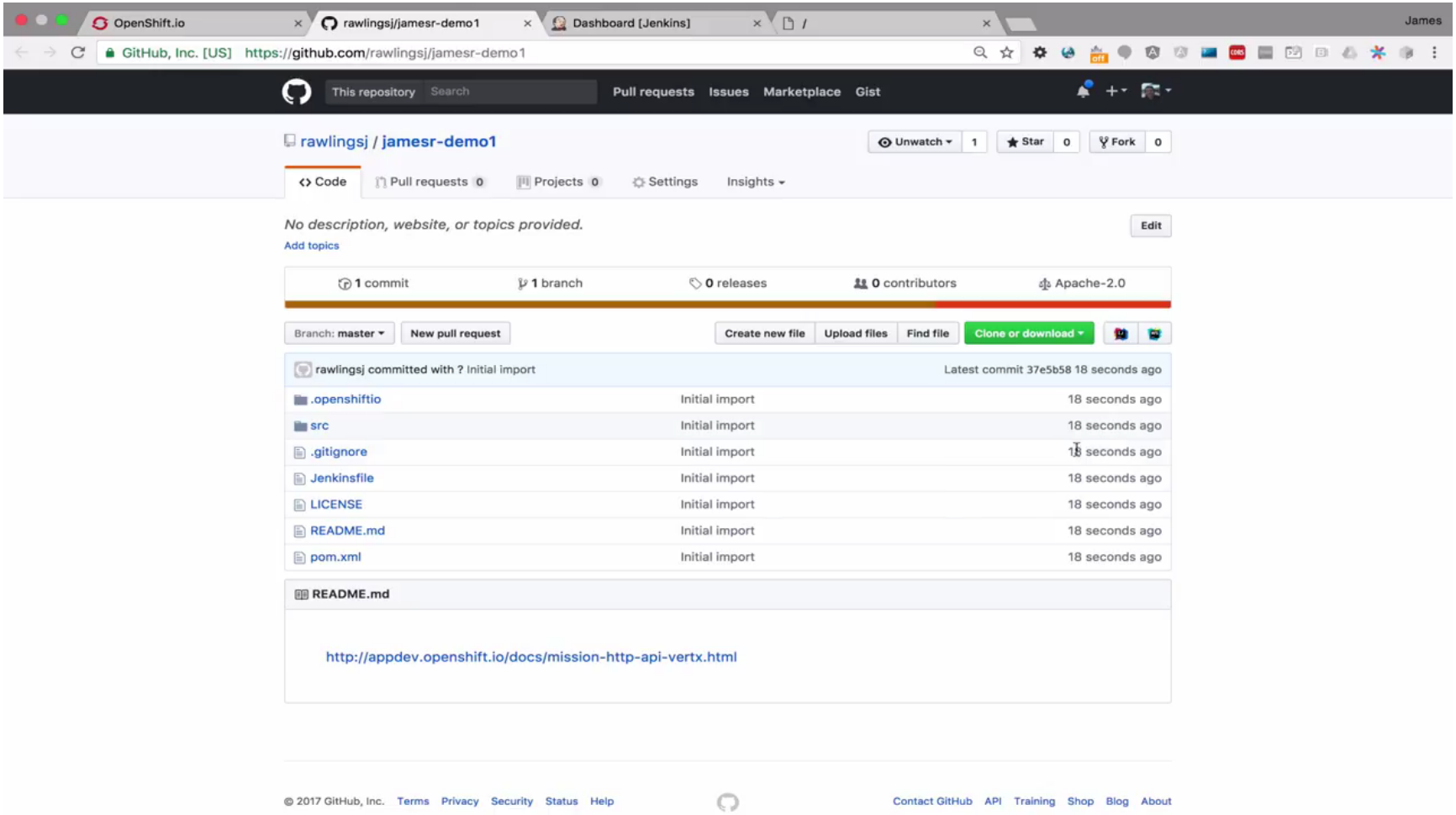


redhat.

GitHub/Jenkins 자동 연계

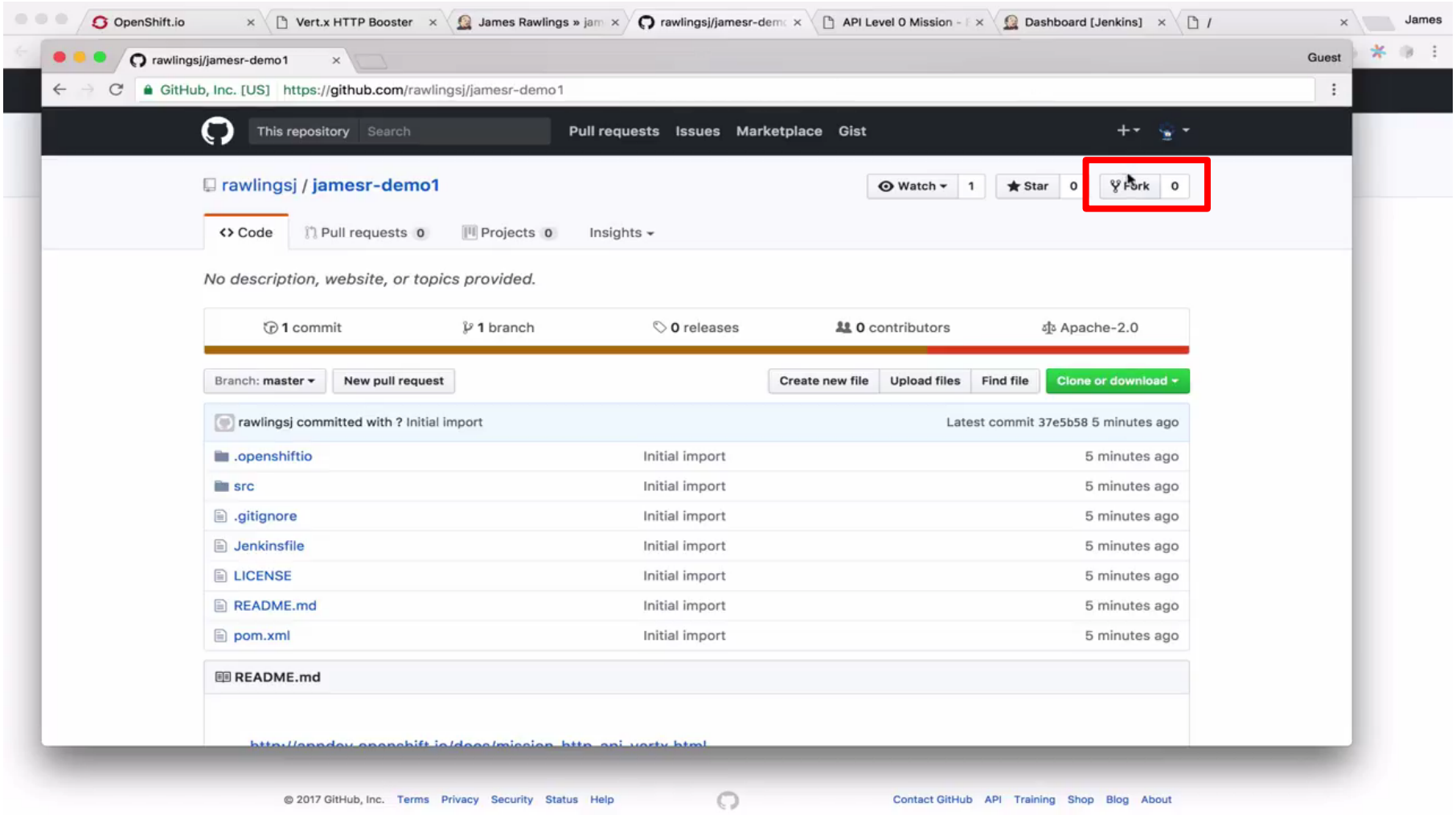


자동 생성된 Git Repository



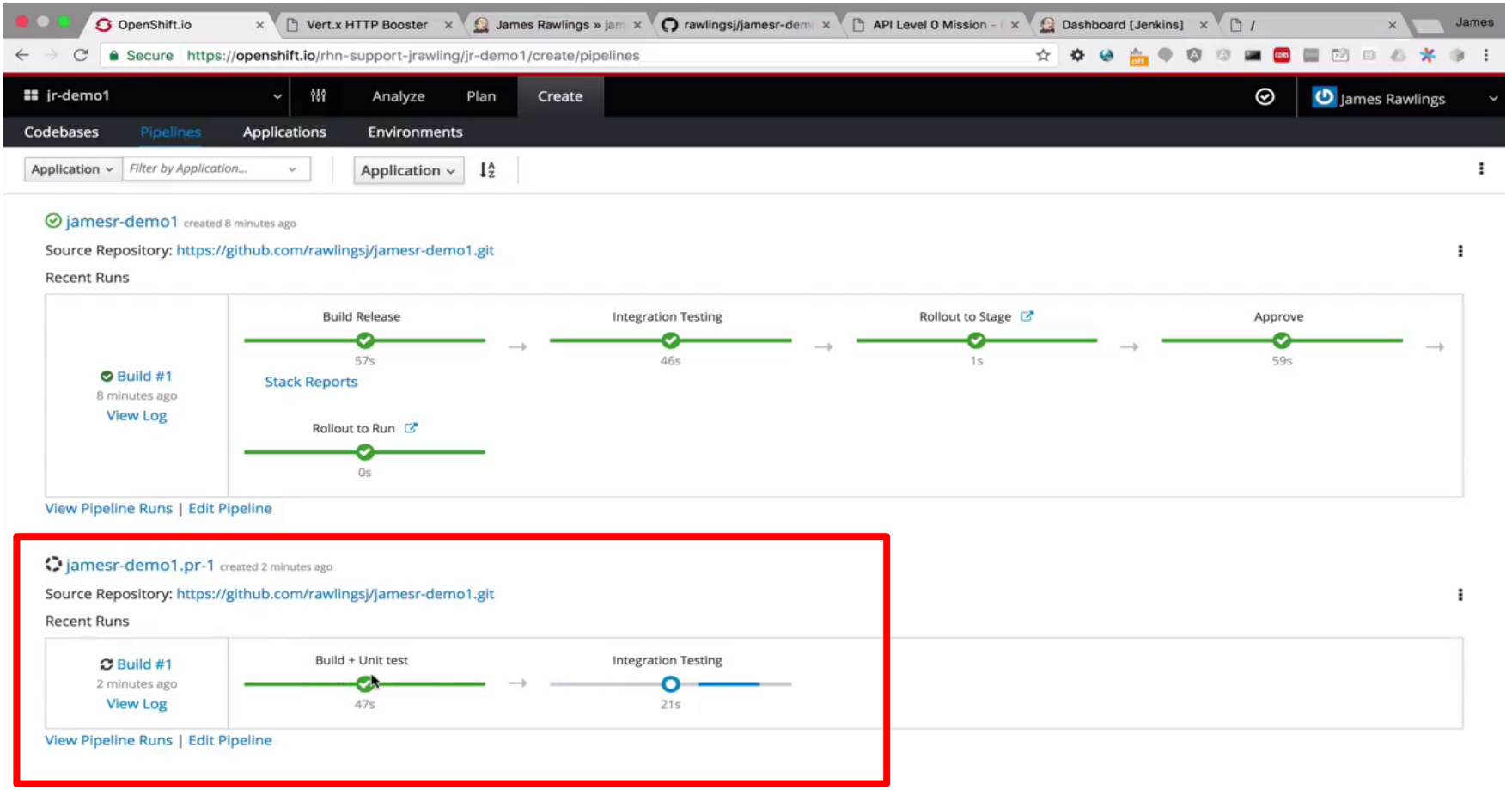
redhat.

Team 개발을 위한 Fork



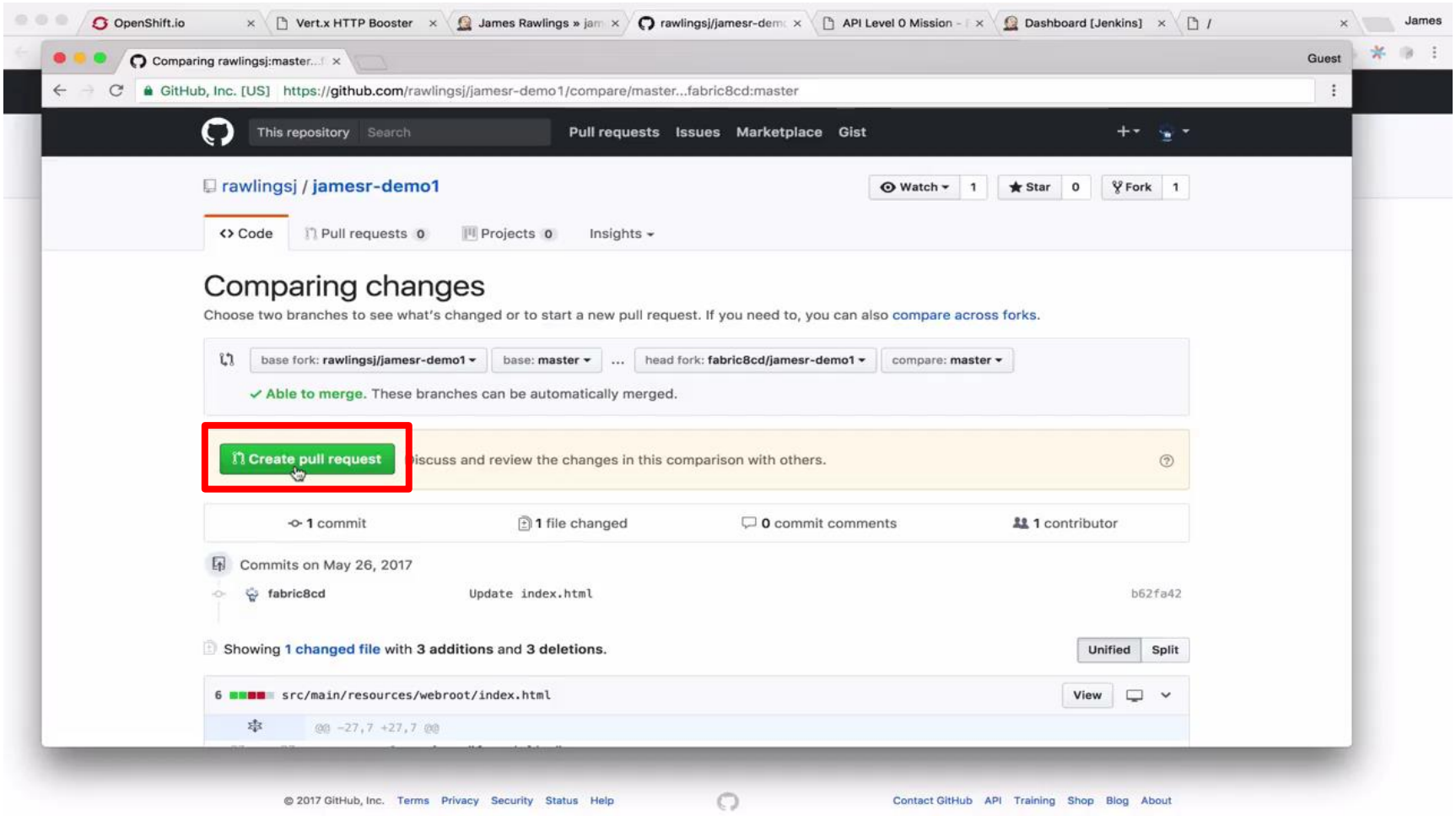
redhat.

별도의 Pipeline 으로 테스트



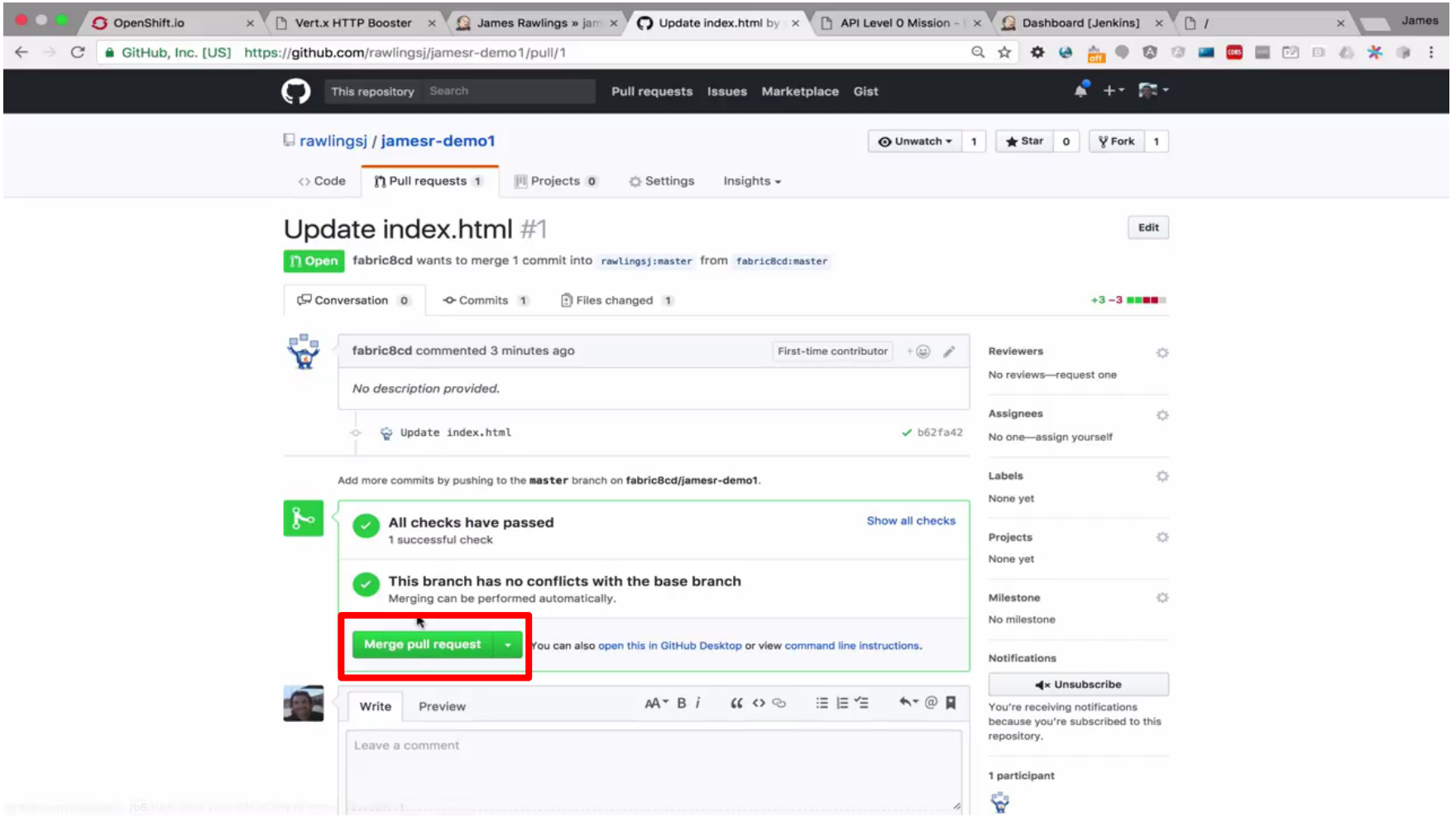
redhat.

CI를 위한 Pull 요청



redhat.

CI를 위한 Merge

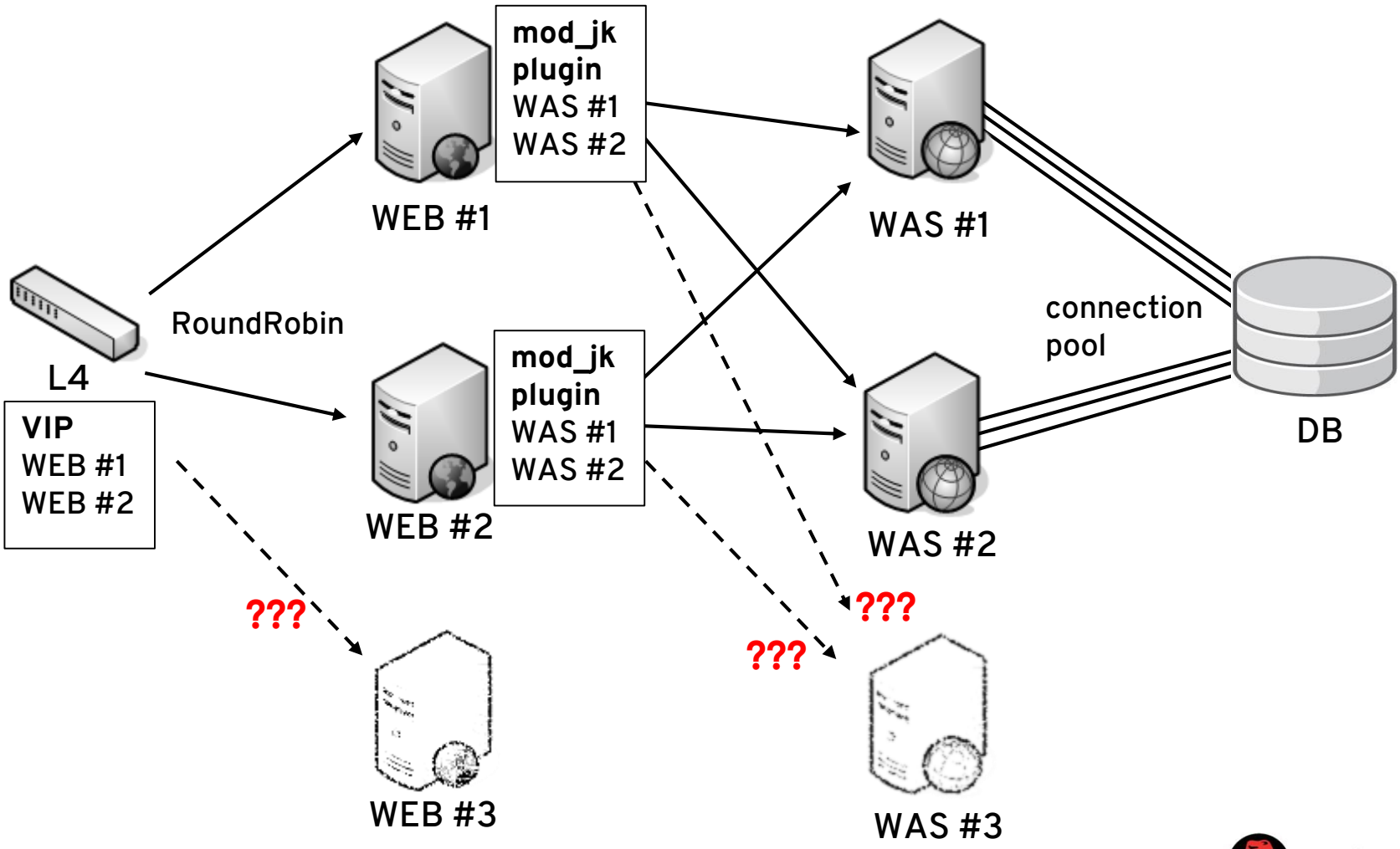


redhat.

CI/CD 데모 동영상

컨테이너 애플리케이션의 운영

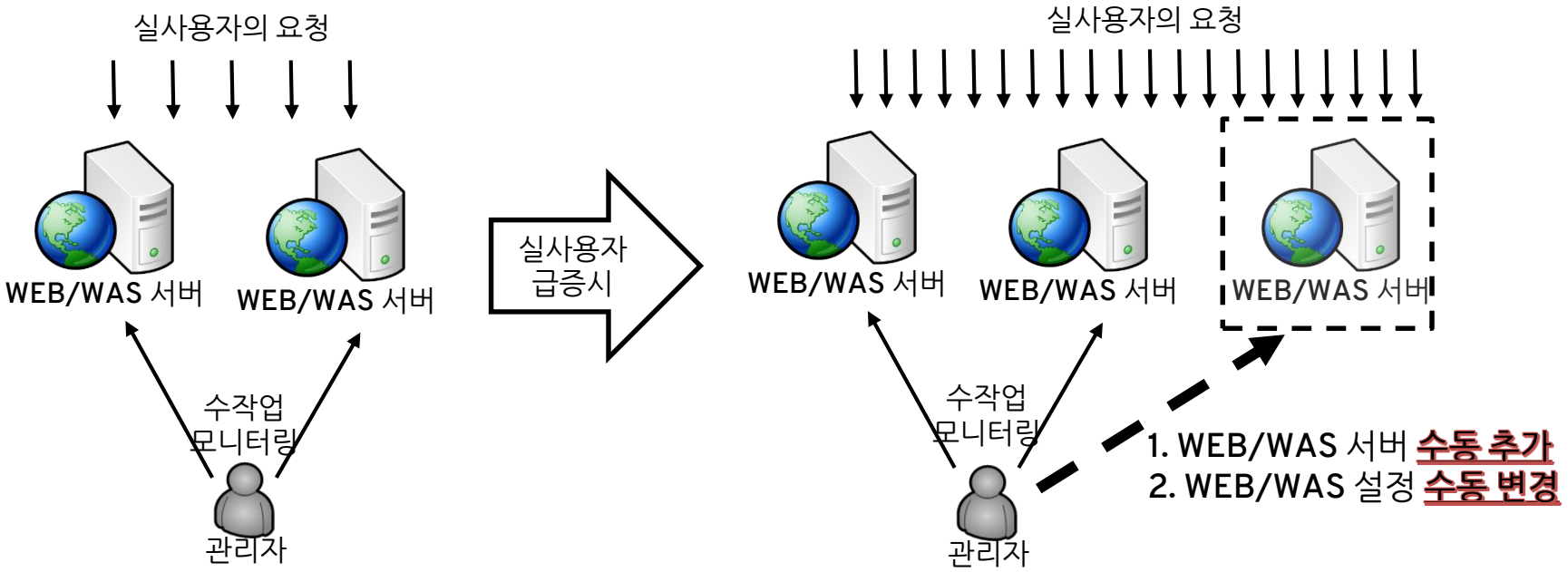
통상의 WEB-WAS-DB 아키텍처



redhat.

자동 확장(Autoscaling) 적용 전

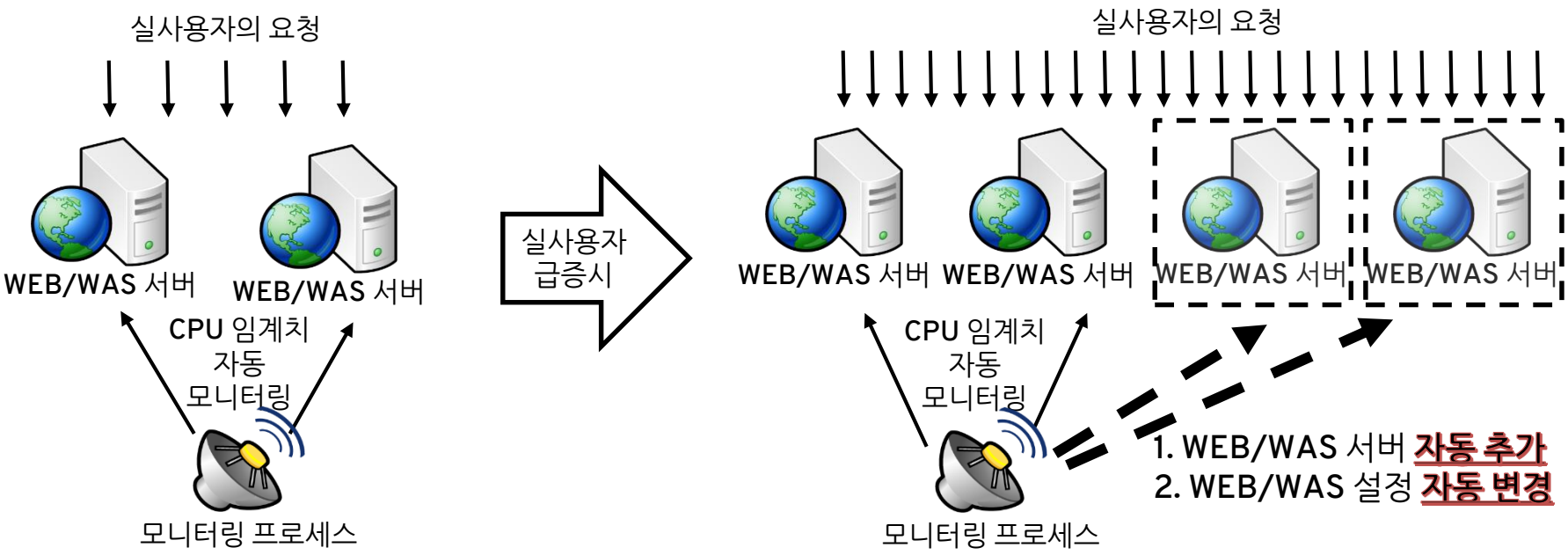
관리자가 서비스 상태를 모니터링한 후, 수작업으로 WEB/WAS 서버를 추가하고, 수작업으로 WEB/WAS 설정을 변경하여, 처리량을 수작업으로 조절하는 방식



모니터링, WEB/WAS 서버 추가 및 WEB/WAS 설정 변경이 모두 수작업이기 때문에,
서비스되기까지 시간이 오래 걸리며, 이로 인해 서비스 중단이 발생할 확률이 높습니다.

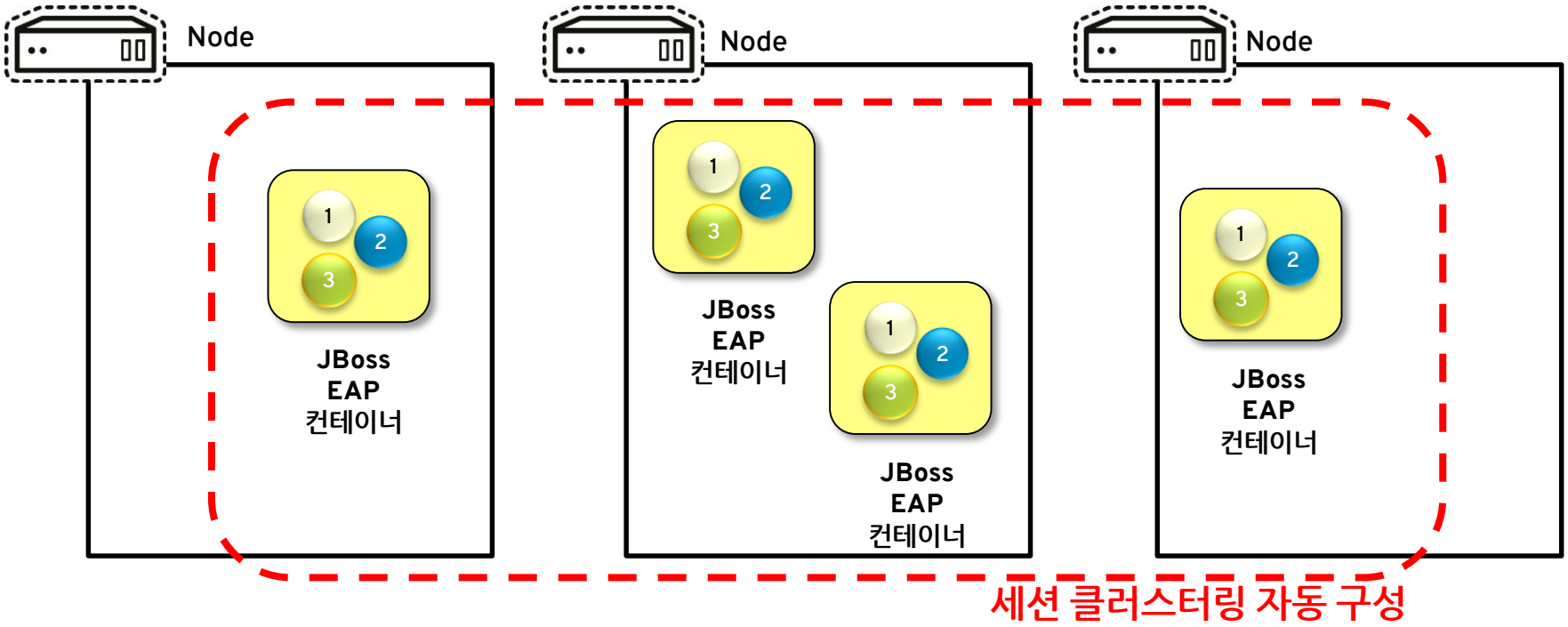
자동 확장(Autoscaling) 적용 후

CPU 사용률 임계치 기준으로, 모니터링된 CPU 값에 따라, 자동 알고리즘에 의해서, 자동으로 필요한 개수만큼 WEB/WAS 개수를 늘이거나 자동으로 줄여서, 처리량을 자동으로 조절하는 방식



모니터링, WEB/WAS 서버 추가 및 WEB/WAS 설정 변경이 모두 자동이기 때문에,
서비스되기까지 시간이 짧으며, 중단 없이 서비스 가능합니다.

JBoss EAP 컨테이너의 기동 위치에 상관없이, 하나의 서비스를 제공하는 JBoss EAP 컨테이너는, 간단한 옵션 설정만으로 **Session Clustering**이 자동으로 구성



redhat.

Autoscaling 데모 동영상

redhat.

감사합니다