

모놀리스로부터 마이크로서비스로의 이전을 위한 개발자 가이드

목차

1 소개	2
2 마이크로서비스로 리팩토링하기 위한 기능 분석	3
3 Red Hat Runtimes	4
4 마이크로서비스를 위한 오픈소스 기술 선택	4
4.1 Thorntail(Eclipse MicroProfile)	5
4.2 Spring Framework 및 Spring Boot	7
4.3 Eclipse Vert.x	8
5 요약	9

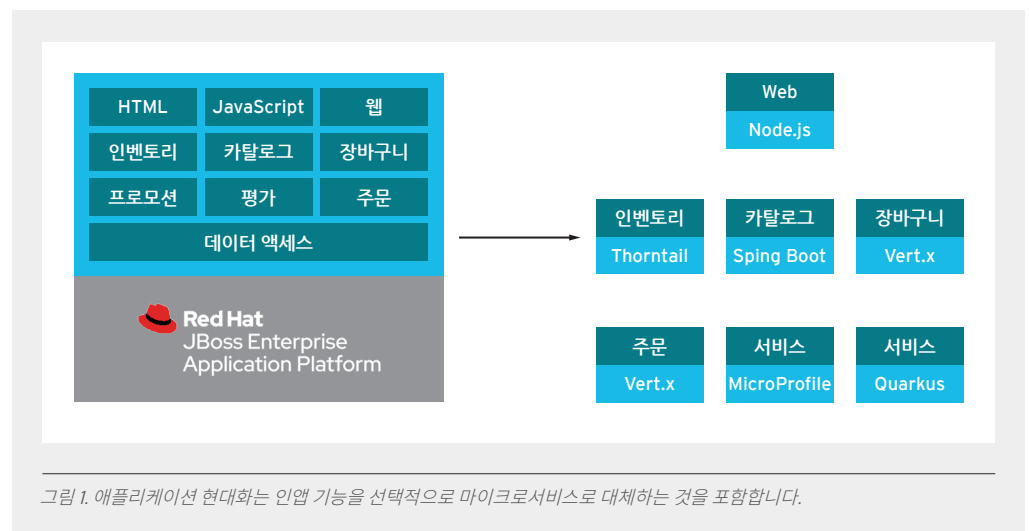
소개

클라우드 네이티브 개발은 컨테이너 및 오케스트레이션 기반의 클라우드 배포를 통해 마이크로서비스를 이용하려는 조직에게 그 중요성이 점차 커지고 있습니다. 하지만 시간 및 비용상 요건으로 인해 모든 레거시 애플리케이션을 완전히 재작업하기란 거의 불가능합니다. Red Hat의 단계별 클라우드 마이그레이션 접근 방식은 기존의 기능과 데이터를 최대한 다시 사용합니다. 이 프로세스는 기존 워크로드를 현대식 배포 플랫폼으로 이전하고, 궁극적으로 새로운 프로세스, 제품, 기술을 적용해 애플리케이션을 현대화합니다.

모놀리식 애플리케이션의 현대화는 클라우드 네이티브 개발의 첫 단계인 "리프트 앤 시프트(Lift and Shift)" 프로세스로 시작됩니다. [리프트 앤 시프트 클라우드 마이그레이션에 대한 개발자 가이드](#)에 설명된 이 프로세스에는 다음이 포함되어 있습니다.

- 기존 모놀리식 워크로드의 컨테이너화
- Red Hat의 쿠버네티스 애플리케이션 플랫폼 솔루션인 Red Hat® OpenShift® Container Platform에서 워크로드 배포
- 레거시 플랫폼에서 외부 통합 및 데이터 유지

이러한 단계를 완료한 후, 개발자는 “모놀리식을 작게 자를” 수 있습니다. 그림 1과 같이, 이 프로세스는 인앱(in-app) 기능을 더욱 경량화되고, 빠르며, 유지관리에 용이한 마이크로서비스로 점차 교체해 나가며, 그 과정에서 모놀리식의 비중을 선택적으로 제거하거나 종료할 수 있습니다. 또한 개발자는 이 프로세스를 진행하여 새로운 마이크로서비스를 통해 신규 기능을 도입해, 고객 또는 비즈니스 이해 관계자들의 관심을 크게 부르는 애플리케이션을 만들 수 있습니다.



마이크로서비스로 리팩토링을 위한 기능 분석

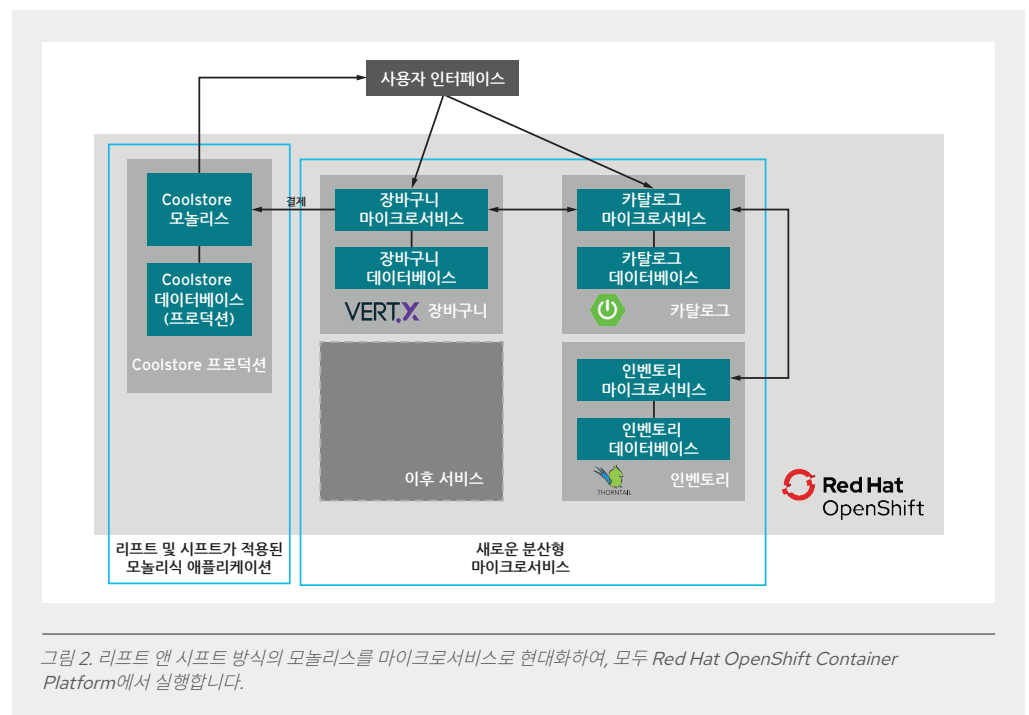
모놀리스에서 마이크로서비스로 전환하면 개발팀은 모놀리스 고유의 복잡성을 해소하면서 기존 기능을 개선하거나, 새로운 기능을 도입하는 등 코드를 리팩토링할 수 있습니다. 하지만 어떤 기능을 마이크로서비스로 추출해야 하는지 결정하는 프로세스는 복잡할 수 있습니다. 소프트웨어를 단일화된 통합 모델로 간주하는 것이 이상적이지만, 대규모 모델 및 팀에는 실용적이지 않은 경우가 많기 때문입니다.

입증된 도메인 중심 설계 개념은 마이크로서비스가 기반이 될 수 있는 발전적인 모델에 소프트웨어 구현을 연결하는 효율적인 전략을 제공합니다. 특히 **제한된 컨텍스트(BC)** 패턴은 애플리케이션 기능을 마이크로서비스로 리팩토링하는 데 적합한 구성 요소로 분할하는 데 이상적인 방식으로, 규모가 크고 복잡한 모델을 제한된 컨텍스트로 분할하며 컨텍스트간 특정한 상호관계성을 부여합니다.

이 가이드에서는 가상의 "Coolstore"라는 온라인 스토어를 예시로 들어 추가 마이크로서비스를 통해 모놀리스를 확장하고 보강하여 "모놀리스를 단계적으로 축소"하는 프로세스에 대한 내용을 다룹니다. 이미 리프트 앤 시프트 방식이 적용된 모놀리스는 Red Hat OpenShift Container Platform에서 계속 작동하며, 새로운 마이크로서비스가 원활하게 도입되어 인앱 기능을 교체하거나 보강합니다(그림 2). 이 예시에서는 오픈소스 런타임을 기반으로 모놀리스에 추가된 새로운 서비스는 다음과 같습니다.

1. Red Hat에서 Eclipse MicroProfile을 구현한 **Thorntail**로 구축된 인벤토리 마이크로서비스
2. **Spring Boot**로 구축된 카탈로그 마이크로서비스
3. **Eclipse Vert.x**로 구축된 장바구니 마이크로서비스 및 데이터베이스

이후에도 애플리케이션의 기능을 교체하거나 확장하기 위해 원하는 대로 서비스를 추가할 수 있습니다.

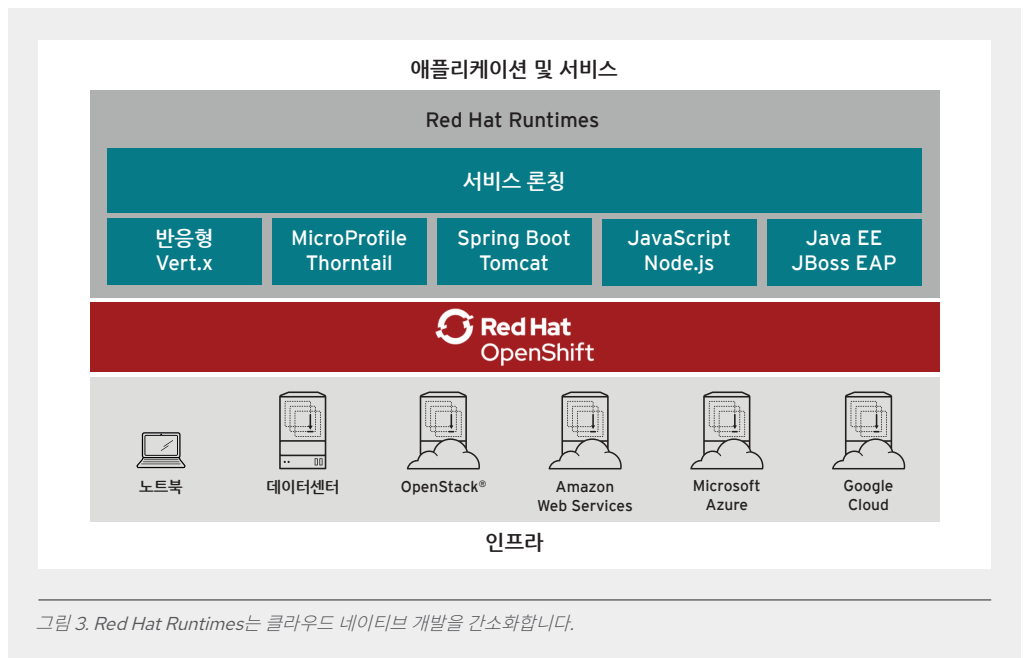


Red Hat Runtimes

Red Hat Runtimes는 개발자에게 여러 현대화 옵션을 제공해 기존 애플리케이션을 원활하게 클라우드로 전환할 수 있게 해줍니다. 개발팀은 단일 클라우드 네이티브 개발 플랫폼에서 신규 애플리케이션을 개발하고, 모놀리스에서 마이크로서비스로의 전환을 원하는 속도로 진행할 수 있습니다. Red Hat Runtimes는 다수의 런타임과 프레임워크를 지원합니다.

- Runtimes는 Red Hat JBoss Enterprise Application Platform, Eclipse Vert.x, Thorntail, Node.js를 포함합니다.
- 프레임워크는 Spring Boot, Netflix Ribbon, Netflix Hystrix를 포함합니다.

제품의 런칭 서비스는 개발자가 클라우드에서 즉시 실행 가능한 다양한 예제 또는 미션 부스터를 통해 신속하게 실행할 수 있도록 지원하며 Red Hat Runtimes(그림 3)의 강력한 성능을 입증합니다.



마이크로서비스를 위한 오픈소스 기술 선택

리프트 앤 시프트 클라우드 마이그레이션을 위한 개발자 가이드에 제시된 프로세스는 기존 모놀리식 Java™ EE 애플리케이션을 Red Hat JBoss Enterprise Application Platform과 Red Hat OpenShift Container Platform을 사용해 클라우드로 이전하는 방법을 설명하며 기존 애플리케이션에 대한 Red Hat OpenShift의 효율성을 입증합니다. 마이크로서비스로 이전하면 애플리케이션을 더 작은 부분으로 분할하여 병렬 처리 및 자율성을 확대할 수 있습니다. 이러한 프로세스를 통해 궁극적으로 전체 애플리케이션을 분산된 마이크로서비스 세트로 재설계하기 위한 목표를 달성하고, 가치 창출을 가속화할 수 있습니다. 각 마이크로서비스 구현에 이상적인 기술은 각기 다릅니다. 예를 들어 보겠습니다.

- MicroProfile 사양을 구현하는 Thorntail은 Red Hat JBoss Enterprise Application Platform을 사용하는 애플리케이션 뿐만 아니라 기존 Java EE 애플리케이션을 마이크로서비스로 리팩토링하는 데 이상적입니다.
- Spring Boot는 Spring 에코시스템을 사용하는 웹 애플리케이션을 구축하도록 지원해, 프로덕션에 이르는 과정을 단축합니다.

- Eclipse Vert.x는 모바일 또는 웹 애플리케이션, 애플리케이션 프로그래밍 인터페이스(API), 사물 인터넷(IoT) 애플리케이션, 실시간 애플리케이션을 위한 반응형 마이크로서비스를 구축하도록 지원합니다.
- Node.js는 API 및 실시간 애플리케이션 구축은 물론 모바일 및 웹 애플리케이션 구축에 이상적인 서버-사이드 JavaScript를 제공합니다.

Thorntail(Eclipse Microprofile)

Thorntail은 기존 Java EE 애플리케이션을 시작하는 데 적합하며 기존 Java EE 개발자 역시 자연스럽게 Thorntail을 활용할 수 있습니다. MicroProfile이 구현된 Thorntail은 개발자를 위한 전문 툴링을 사용하여 경량화된 구현을 제공합니다. Thorntail의 구성적인 측면에서 지원하는 Thorntail "조각(pieces)"은 Apache Maven 아티팩트로 구현된 특정 기능을 포함하며 다음을 제공합니다.

- API 종속성(dependencies) 추가 방식(예: JAX-RS)
- 적절한 기본값으로 시스템을 설정하는 기능
- 토폴로지의 일부로서 다른 구성 요소 탐색
- 배포 변경 수단

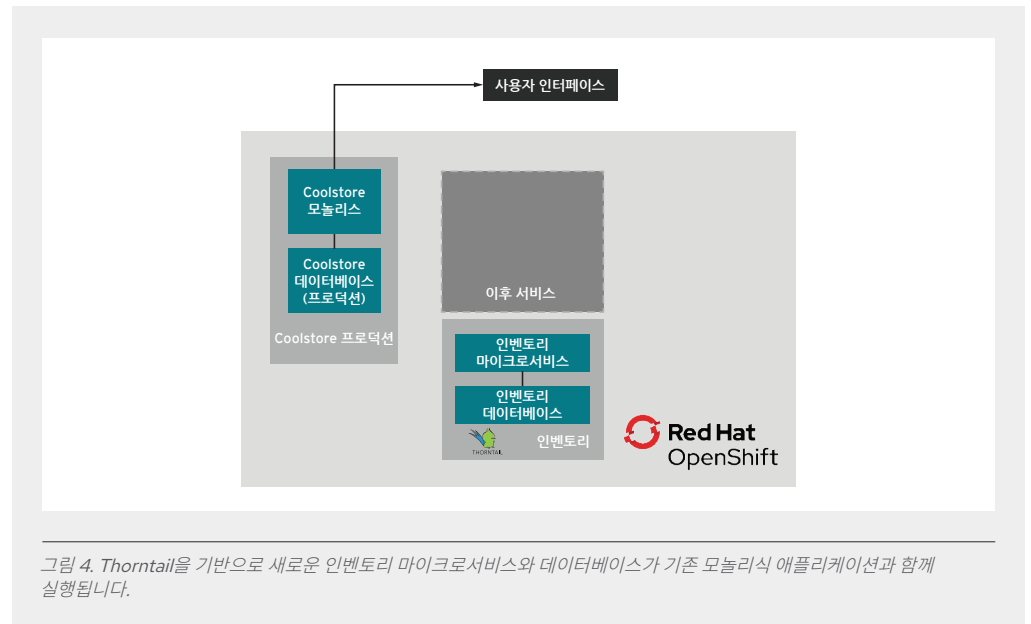
조각은 자동 탐지되거나 명시적으로 선언될 수 있습니다.

Thorntail은 다음을 비롯한 클라우드 네이티브를 지원합니다.

- 상태 점검(Health checks)
- 외부 설정
- 클라이언트측 탐색 및 로드 밸런싱
- Circuit Breaker 및 Bulkhead 적용
- 로깅, 모니터링, 추적, 메트릭
- Keycloak을 사용한 엔터프라이즈급 보안 배포

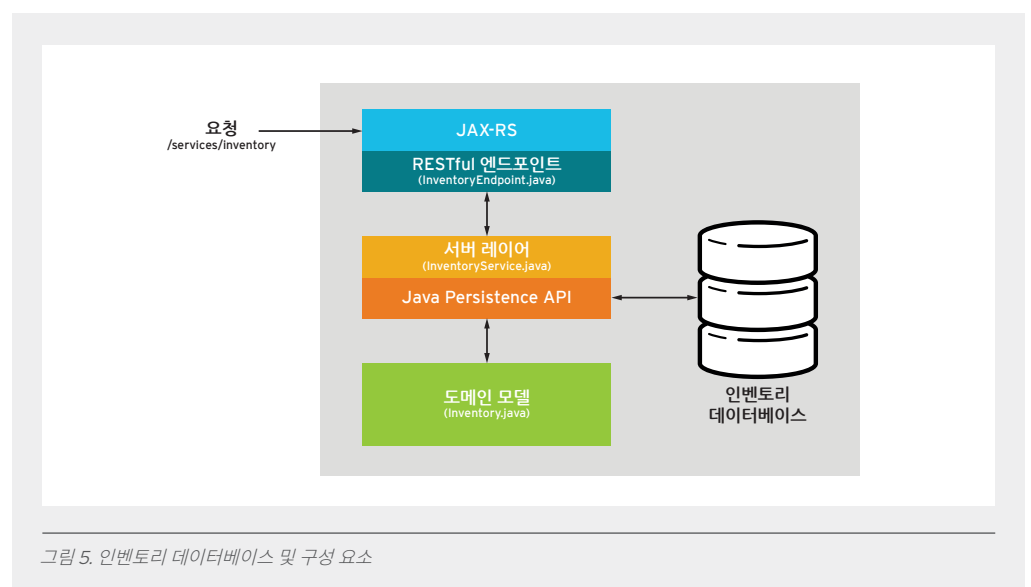
일반적인 시나리오에서 개발자는 모놀리스의 한 구성 요소를 Thorntail 마이크로서비스로 구현해 Red Hat OpenShift Container Platform에 배포합니다. 새로운 마이크로서비스는 기존 모놀리스와 함께 배포되어 나란히 실행됩니다. 예를 들어 Thorntail을 인벤토리 마이크로서비스와 데이터베이스를 구현하는 데 사용할 수 있습니다(그림 4).

¹ Eclipse MicroProfile은 마이크로서비스 아키텍처용 엔터프라이즈 Java를 최적화하도록 설계되었습니다.
<https://microprofile.io>



서비스 구축에는 인벤토리에 액세스할 도메인 모델, 서비스 인터페이스, RESTful 엔드포인트를 생성하기 위한 코드 작성 과정이 포함됩니다(그림 5). 이를 완료하면 REST 서비스는 두 개의 엔드포인트를 정의합니다.

- `/services/inventory`는 HTTP GET를 통해 액세스 가능하며, 모든 알려진 제품 인벤토리 엔티티를 JSON으로 반환합니다.
- `/Services/inventory/<id>`는 `<id>`는 HTTP GET를 통해 액세스 가능하며 마지막 경로 매개 변수가 인벤토리 상태 확인을 위한 제품 ID가 됩니다(예: `/services/inventory/329299`).



Spring Framework 및 Spring Boot

이전 시나리오에서 인벤토리 서비스와 데이터베이스를 생성했지만, 지금까지 모놀리스의 *세분화*를 시작했다고 볼 수는 없습니다. 사용자 인터페이스(UI)는 다른 백엔드 서비스에서만 사용되는 백엔드 서비스이기 때문에 절대 인벤토리 서비스를 직접 호출하지 않습니다. 이 섹션에서는 인벤토리 서비스를 호출하는 카탈로그 서비스를 추가하는 방법을 설명하며 이 방법으로 결국에는 UI 호출이 새로운 서비스로 리라우팅됩니다.

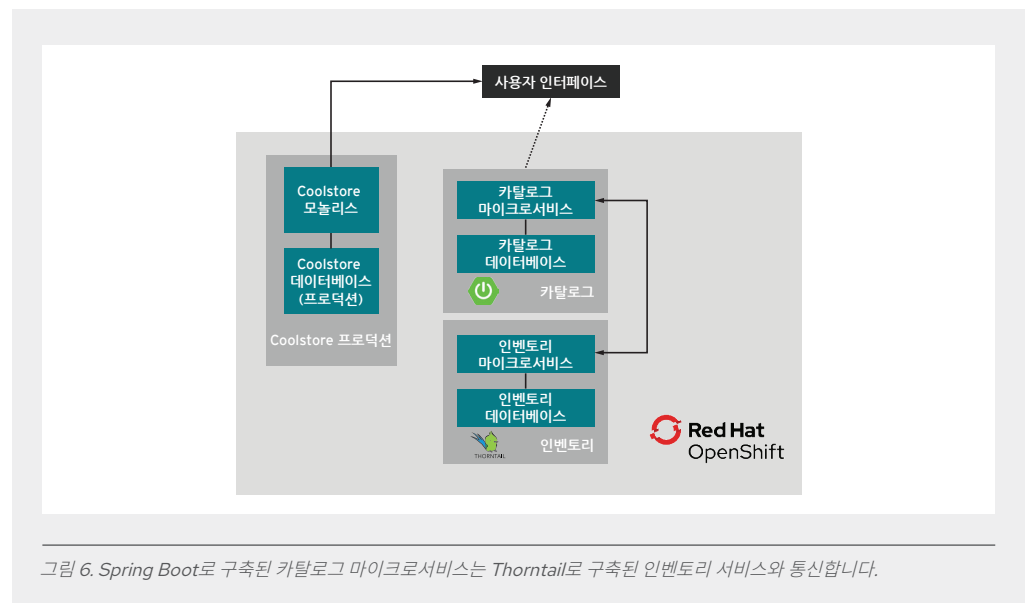
이 예시에서는 Red Hat Runtimes의 일부로도 제공되는 Spring Framework를 사용합니다. Spring은 널리 사용되는 Java 프레임워크이며 Java EE 프로그래밍 모델 대신 제공됩니다. 대부분 개인적 선호 또는 기존 지식에 따라 Spring 또는 기타 플랫폼을 선택하며, Spring과 Java EE의 핵심적인 부분은 유사합니다.

Spring은 주로 마이크로서비스 아키텍처를 기반으로 한 애플리케이션을 구축하는 데 사용됩니다. Spring Boot는 Spring과 타사 라이브러리를 함께 구성하거나 사용할 수 있도록 지원하는 Spring 에코시스템의 인기 툴로, Apache Tomcat과 같은 임베드 가능한 런타임을 부트스트랩하는 메커니즘도 제공합니다. 부팅 가능한 애플리케이션(*fat JAR*라고 불리기도 함)은 컨테이너 모델에 이상적입니다. Red Hat OpenShift Container Platform에서 애플리케이션 시작, 중단, 모니터링과 같은 작업은 애플리케이션 서버 대신 컨테이너 플랫폼에서 처리합니다. Thorntail과 마찬가지로 Spring은 클라우드 네이티브 전체 기능을 제공합니다.

Red Hat Runtimes의 일부로 Spring 지원에는 다음이 포함됩니다.

- Spring Boot, Spring Cloud Kubernetes, Ribbon, Hystrix에 대한 Red Hat 테스트 및 검증
- Tomcat, Hibernate, Apache CXF, Keycloak으로 구축한 SSO(Single Sign-On) 및 Red Hat AMQ를 통한 메시징에 대해 전체 지원
- ConfigMap을 통한 쿠버네티스, Ribbon, Spring Cloud Config를 사용해 서비스 검색을 위한 네이티브 쿠버네티스 및 OpenShift 통합(Spring Cloud)
- launch.openshift.io 및 스타터를 통한 개발자 툴링

예를 들면 카탈로그 마이크로서비스와 데이터베이스를 Coolstore 온라인 스토어 애플리케이션에 추가하는 데 Spring Boot를 사용할 수 있습니다. 이 카탈로그 마이크로서비스는 차례로 이전 단계에서 개발된 인벤토리 마이크로서비스를 사용합니다(그림 6).



Eclipse Vert.x

반응형 마이크로서비스는 자극에 응답합니다. Eclipse Vert.x는 분산형 및 반응형 시스템을 구축하는 데 사용하는 툴킷입니다. 비동기식 논블로킹(Nonblocking) 개발 모델과 핵심 아키텍처의 클러스터링에 대한 기본 이해를 기반으로 Vert.x는 다음을 제공합니다.

- 동시 실행(concurrency) 간소화(이벤트 루프)
- 반응형 마이크로서비스, 웹 애플리케이션, IoT
- 지연 시간이 짧은 대규모 애플리케이션

Vert.x는 이벤트 기반의 논블로킹 방식으로 작동하기 때문에 Vert.x의 애플리케이션은 소수의 커널 스레드만으로 다수의 요청을 동시에 처리할 수 있습니다. 네트워크 유틸리티, 정교한 현대식 웹 애플리케이션, HTTP/REST 마이크로서비스, 대규모 이벤트 처리, 완전한 백엔드 메시지-버스 애플리케이션까지 모든 것을 지원합니다. Vert.x는 다음과 같은 특징을 지닙니다.

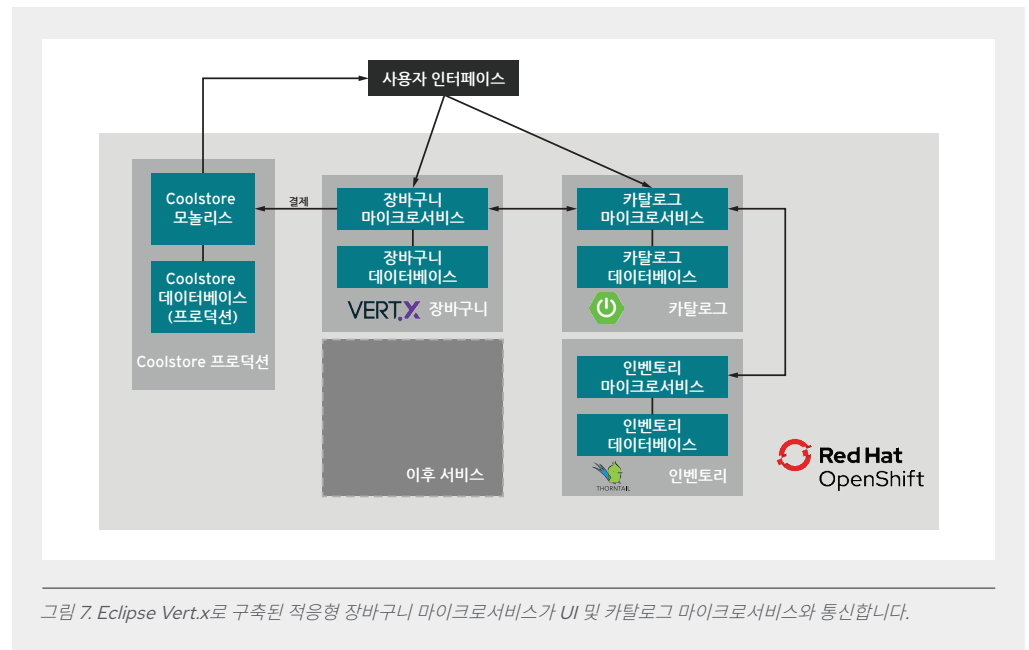
- 신뢰성: 실시간 게임 앱에서 बैं킹 앱까지, [다양한 기업](#)에서 Vert.x를 사용합니다.
- 경량화: Vert.x 코어의 용량은 약 650KB입니다.
- 빠른 속도: 개별 [성능 결과](#)로 확인할 수 있습니다.
- 모듈식: 더 많은 기능이 필요한 경우에 필요한 구성 요소만 추가할 수 있습니다.

Vert.x를 사용하면 다음이 가능합니다.

- 최소 하드웨어로 애플리케이션을 확장합니다.
- 원하는 언어를 사용해 원하는 방식으로 강력한 애플리케이션을 제작합니다. Vert.x는 제한적 프레임워크 또는 컨테이너가 아니기 때문에, 애플리케이션 작성 방식에 구애를 받지 않습니다.
- 어디에서든 애플리케이션을 실행합니다. Vert.x는 애플리케이션 서버가 아니기 때문에 애플리케이션을 배포해야 하는 모놀리식 Vert.x 인스턴스가 없습니다.
- 경량화된 고성능 마이크로서비스를 구축합니다.

예시 애플리케이션에서 Vert.x는 장바구니 마이크로서비스를 구현하는 데 사용되었습니다. 그림 7과 같이, 이 반응형 마이크로서비스는 다음과 같은 작업을 수행합니다.

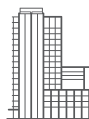
- 사용자 상호 작용을 위해 UI와 통신합니다.
- 카탈로그 마이크로서비스와 통신하여 사용자의 쇼핑을 지원합니다.
- 결제 시 오리지널 Coolstore 모놀리스와 통신합니다.



요약

Red Hat Runtimes는 개발자에게 클라우드 네이티브 개발 툴과 플랫폼을 제공하여 조직이 마이크로서비스로 이전할 수 있도록 지원합니다. 모놀리스를 Red Hat JBoss Enterprise Application Platform 및 Red Hat OpenShift Container Platform을 통해 클라우드로 리프트 앤 시프트한 후, 모놀리스 작동을 그대로 유지하며 마이크로서비스를 추가해 모놀리스 측면을 교체 또는 보강할 수 있습니다. MicroProfile 사양을 구현하는 Thorntail, Spring Boot, Node.js 및 Eclipse Vert.x 등의 플랫폼으로 개발자가 제어하고 선택하여 클라우드에 마이크로서비스를 구현할 수 있습니다.

한국레드햇 홈페이지 <https://www.redhat.com/ko>



RED HAT 정보

Red Hat은 세계적인 엔터프라이즈 오픈소스 솔루션 공급업체로서 커뮤니티 기반 접근 방식을 통해 신뢰도 높은 고성능 Linux, 하이브리드 클라우드, 컨테이너, 쿠버네티스 기술을 제공합니다. 또한 고객으로 하여금 신규 및 기존 IT 애플리케이션을 통합하고, 클라우드 네이티브 애플리케이션을 개발하며, 업계를 선도하는 Red Hat의 운영 체제를 기반으로 표준화하는 동시에 복잡한 환경의 자동화, 보안 및 관리를 실현할 수 있도록 지원합니다. Red Hat은 전세계 고객에게 높은 수준의 지원과 교육 및 컨설팅 서비스를 제공하여 권위있는 어워드를 다수 수상한 바 있으며, Fortune 선정 500대 기업의 신뢰를 받는 어드바이저로 인정받고 있습니다. 또한 기업, 파트너, 오픈소스 커뮤니티의 전략적인 파트너로서 고객들이 디지털 미래에 대비할 수 있도록 지원하고 있습니다.



www.facebook.com/redhatkorea
구매문의 080 708 0880
buy-kr@redhat.com