RED HAT® OPENSHIFT

redhat.

백서

클라우드 네이티브 개발 플랫폼 선택에 대한 이그제큐티브 가이드

소개

조직이 클라우드 네이티브 애플리케이션에 대한 계획을 시작할 때 개발 플랫폼 선택으로 시작하여 애플리케이션이 진정한 운영 환경 수준으로 구현되어 클라우드에서 제공이 가능하도록 준비될 때까지의 전체 기간을 고려해야 합니다. 이는 많은 의사결정이 필요한 장기적인 프로세스가 될 수 있으며 이로 인해 진행 속도가 빨라질 수도 느려질 수도 있게 됩니다.

예를 들어, 클라우드 네이티브 개발로 전환하는 과정의 초기에 개발자가 애플리케이션을 어디에 배포할지 모르는 상태에서 툴 및 프레임워크를 선택한다면 효율성이 떨어지기 쉽습니다. 엔터프라이즈 개발자는 런타임, 프레임워크, 언어에 대한 유연성을 원하며, 조직은 운영 비용을 줄이고 위험을 완화하며 규정 준수 요구 사항을 충족하기 위해 전체 애플리케이션 라이프 사이클에 대한 표준을 필요로 합니다. 또한 조직은 클라우드 인프라 제공업체이거나 최신 아키텍처 스타일이거나 모두 종속(Lock-in)을 피하기를 원합니다.

여기에 더해 클라우드 개발의 학습 곡선이 가파르다는 점을 감안한다면 강력하고 확장 가능하며 탄력적인 애플리케이션을 개발 주기에서 우선적으로 구축해야 합니다. 클라우드로 전환하는 주요 동인은 동적 확장을 위한 탄력적인 용량을 확보하는 것이므로 복원력은 차후로 미룰 수 있는 문제가 아닙니다.

효과적인 클라우드 네이티브 애플리케이션 전략을 위해서, 개발 툴과 클라우드 플랫폼에서 배포 자동화와 운영에 이르기까지 전체적인 그림을 고려해야 합니다. 클라우드 네이티브 개발에 대한 총체적인 접근 방식의 이점은 불필요하게 우회하지 않아도 되도록 일정한 경로를 제공한다는 것입니다.

클라우드 네이티브 개발의 이해

클라우드 네이티브 개발에서는 속도, 복원력, 민첩성으로 변화에 대응하는 것이 중요하며 배포 빈도를 높여 대응에 필요한 리드타임을 크게 줄임으로써 이를 실현할 수 있습니다.

클라우드 네이티브 애플리케이션에 관한 논의는 'The Twelve-Factor App'이라는 선언으로 거슬러 올라갑니다. 이 선언은 SaaS(서비스로서의 소프트웨어) 애플리케이션을 구축하고 운영하면서 얻은 경험으로부터 도출된 원칙입니다. 그 목표는 다음과 같습니다.

- 개발 학습 곡선 감소
- 클라우드 플랫폼 기반 배포에 적합한 애플리케이션 구축
- 연속 배포를 통한 민첩성 극대화
- 대규모의 변경 없이도 애플리케이션 스케일 업 수행

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

www.redhat.com/ko

Red Hat이 고객과 협업하면서 알아낸 클라우드 네이티브 애플리케이션 개발의 주요 요소는 다음과 같습니다.

- **서비스 기반 아키텍처:** 마이크로서비스 또는 모듈식의 탄력적으로 결합된 아키텍처 모델을 구현할 수 있습니다. 서비스는 상당히 독립적이고 쉽게 이해할 수 있는 비즈니스 활동을 나타냅니다. 쉽게 업데이트하거나 대체할 수 있는 서비스를 제공하는 것이 목표입니다. 잘 정의된 서비스는 전체 애플리케이션에 비해 더 철저하게 테스트하기가 더 쉽습니다.
- **컨테이너:** Docker 이미지를 사용하는 Linux® 컨테이너는 공통 패키지 모델이며 격리와 이식성을 제공하는 독립적인 실행 환경입니다. 컨테이너는 고도화된 자동화를 지원하여 클라우드 플랫폼을 더 강력하게 만듭니다.
- **DevOps 자동화:** 개발과 운영을 통합하는 것을 목표로 하는 협업적 프로세스와 프랙티스입니다. 목표는 배포 빈도를 높이고 더 우수한 품질의 릴리스를 제공하여 시장 출시 기간을 단축하고 위험을 줄이며 사용자 만족도를 높이는 것입니다. CI/CD(지속적인 통합 및 지속적인 제공)는 DevOps를 통해 얻은 개선과 매우 밀접하게 관련됩니다. 계측과 모니터링을 통해 성과를 파악하고 높은 수준의 최종 사용자 경험을 보장하는 것은 또 다른 주요 목표입니다.
- **API 기반 통신:** 프로세스 간 통신은 네트워크를 기반으로 간결한 계약 기반 인터페이스를 사용하여 API(애플리케이션 프로그래밍 인터페이스)를 통해서만 이루어집니다. 따라서 사라져 변경을 제한하고 작동 중단의 일반적인 원인이 되는 커플링이 의도치 않게 발생하지 않습니다. 애플리케이션이 더 이상 동일한 데이터센터에 상주하지 않는 하이브리드 환경에서 이 요소의 중요성이 더 분명하게 드러납니다.

마이크로서비스의 이해

마이크로서비스 아키텍처는 많은 소프트웨어 개발 문제를 해결하는 강력한 아키텍처 스타일입니다. 애플리케이션은 느슨하게 연결되어 특정 비즈니스 기능을 구현하는 서비스로 나뉩니다. 개별 마이크로서비스는 모놀리식 애플리케이션보다 이해하고, 개선하고, 테스트하고, 배포하기가 훨씬 더 쉽다는 개념을 기반으로 혁신의 속도를 높이는 것이 주된 목표입니다. 각 마이크로서비스의 목적과 기능을 잘 정의해야 하므로 자동화된 테스트와 지속적인 제공을 구현하는 것이 훨씬 더 실용적인 방법이 됩니다.

클라우드 네이티브 애플리케이션 요구 사항

클라우드 네이티브 아키텍처는 애플리케이션에 대한 여러 기술 요구 사항을 제시합니다. 여기에는 다음 사항이 포함됩니다.

- **개발이 자동화되어야 합니다.** 릴리스 빈도를 높이기 위해, 승인 버튼을 클릭하기만 하면 프로세스가 진행될 수 있도록 배포 자동화를 통합해야 합니다. 필요한 경우 이전 릴리스로 빠르게 롤백할 수 있는 기능과 함께 자동화된 구축 및 배포 파이프라인이 필요합니다.
- **설정 정보를 애플리케이션에서 분리해야 합니다.** 특정 환경 내에서 애플리케이션을 실행하는 데 필요한 설정 상세 정보를 배포 환경에 저장해야 합니다. 그러면 배포, 테스트, 운영을 포함한 모든 환경에서 동일한 애플리케이션 이미지를 사용할 수 있습니다. 또한 애플리케이션이 네트워크를 통해 데이터 소스 및 기타 구성 요소와 안전하게 통신해야 하므로 자격을 안전하게 보관해야 합니다.
- **애플리케이션 서비스가 동적으로 배치되어야 합니다.** 이는 환경 전반의 이식성, 고가용성 그리고 로드 밸런싱을 사용하는 동적 스케일링에 필요합니다.
- **퍼시스턴트 데이터를 위한 별도의 데이터 저장소가 필요합니다.** 프로세스와 컨테이너가 스테이트리스여야 합니다. 프로세스가 충돌할 경우 다른 머신에서 몇 초 이내에 다시 시작될 수 있습니다. 데이터, 세션 정보, 로그가 모두 외부 데이터 저장소에 보관되어야 합니다.

클라우드 네이티브 개발을 시작할 때 다음 몇 가지 사항을 고려해야 합니다.

- 클라우드 플랫폼에서 실행할 애플리케이션 컨테이너를 구축하고 배포하는 데 있어서 어느 정도의 경험을 보유하고 있습니까? 개발자가 퍼시스턴트 데이터와 설정을 애플리케이션과 별도로 관리하는 데 무엇이 필요한지 알고 있습니까?
- 클라우드 네이티브 개발을 시작하기 위한 개발 환경을 구축하는 데 어느 정도의 시간이 걸리겠습니까? 개발 환경이 클라우드 운영 환경에 맞습니까?
- CI/CD를 구축하고 개발 및 배포 프로세스에 효과적으로 통합하는 데 어느 정도의 시간이 걸리겠습니까?

클라우드 네이티브 개발 플랫폼을 선택할 때 이러한 사항을 모두 고려해야 합니다.

마이크로서비스 고려 사항

마이크로서비스와 클라우드 네이티브 애플리케이션은 밀접하게 연관되어 있습니다. 실제로 마이크로서비스는 클라우드 플랫폼, 컨테이너, DevOps 자동화 없이는 성공적으로 구현하기가 어렵습니다. 마이크로서비스의 주된 동인은 민첩성으로서, 작은 애플리케이션 구성 요소를 사용하여 얻을 수 있으며 위험은 낮추면서 출시 빈도를 높일 수 있습니다. 그러나 한때는 단일 애플리케이션이었던 것이 독립적인 릴리즈 주기를 가진 10개 이상의 마이크로서비스가 될 수 있습니다. 다음은 마이크로서비스로 전환할 때 고려해야 할 몇 가지 사항입니다.

- 각각의 새로운 마이크로서비스를 위해 새로운 개발, 테스트, 운영 환경을 프로비저닝하는 데 어느 정도의 시간이 걸립니까?
- 여러 필수 릴리스를 지원할 수 있을 만큼 구축과 배포가 충분히 자동화되어 있습니까?
- 애플리케이션이 마이크로서비스의 실행 위치를 어떻게 알 수 있습니까? 가용성 또는 확장성을 위해 필요한 만큼 마이크로서비스를 재배치할 수 있는 유연성이 있습니까?
- 최종 사용자에게 영향을 미치는 다운타임을 유발하지 않고 마이크로서비스를 릴리스할 수 있습니까?
- 문제를 진단하는 데 있어 어느 정도의 어려움이 예상됩니까? 다양한 서비스와 하이브리드 환경 전체에서 이벤트를 추적하는 데 있어 어느 정도의 어려움이 예상됩니까?
- 하나의 서비스에서 발생한 문제가 제대로 격리됩니까, 아니면 연속적인 오류로 이어집니까?
- 마이크로서비스는 분산성을 가지고 있기 때문에 서비스 검색, 클라이언트 측 로드 밸런싱, 퍼시스턴트 상태 관리, 분산 추적, 복원력 등의 기능을 수행하려면 추가 소프트웨어 인프라 구성 요소가 필요합니다.
- 개발자가 이러한 서비스를 구현해야 합니까, 아니면 인터넷에서 구성 요소를 찾아냅니까?
- 배포된 모든 마이크로서비스에 이러한 구성 요소가 속한다면 유지관리 부담이 어느 정도 되겠습니까? 버그와 보안 취약점을 해결하기 위해 구성 요소를 어떻게 업데이트하겠습니까?
- 이러한 상용 서비스를 클라우드 런타임 또는 플랫폼 중 어디에 구축해야 합니까?

마이크로서비스 구축을 시작하려는 개발자가 가장 먼저 떠올리는 질문은 어떤 종류의 런타임, 프레임워크, 언어의 필요성 여부입니다. 이 질문에 답변하기 위해 고려해야 할 사항은 다음과 같습니다.

- 개발자가 구축을 시작하기 위해 새로운 프레임워크를 익혀야 합니까?
- 마이크로서비스로 이동할 때 조직의 기존 Java™ EE 코드와 전문성을 활용할 수 있습니까?
- 리액티브(Reactive) 프로그래밍과 같은 새로운 개발 스타일을 사용할 수 있습니까? 플랫폼에서 모바일과 IoT(사물인터넷)에서 생성된 이벤트 중심의 워크로드와 같은 새로운 요구를 충족할 수 있는 유연성이 제공됩니까?
- 작업에 가장 적합한 툴을 사용하기 위해 서로 다른 런타임, 프레임워크 또는 언어를 사용하는 여러 마이크로서비스를 구현하는 것이 합리적입니까? 마이크로서비스가 각자 다른 방식으로 설정, 배포 또는 보호됩니까?
- 실제 퍼블릭 클라우드 환경에서 새로운 마이크로서비스를 테스트할 수 있기까지 어느 정도의 시간이 걸립니까?

마이크로서비스는 매우 강력한 기능을 제공합니다. 그러나 마이크로서비스와 분산된 애플리케이션 아키텍처를 제대로 관리하는 것은 어려울 수 있습니다. 복잡성이 없으며 개발을 간소화해주는 플랫폼을 선택하는 것이 중요합니다.

기존의 모놀리식 애플리케이션 현대화

기존의 모놀리식 애플리케이션을 마이크로서비스에 다시 쓰는 방법은 개발자에게는 유용하게 들릴 수 있지만 항상 실용적이거나 비용 효율적인 것은 아닙니다. 그러나 이러한 애플리케이션 역시 유지관리해야 하며 필요한 개선 사항에 대한 백로그가 있어야 합니다. 릴리스 주기가 길어 백로그를 줄이는 것이 힘들 수 있습니다. 이러한 애플리케이션을 클라우드 플랫폼에 재호스팅하면 CI/CD를 블루/그린 또는 카나리(Canary) 배포 등의 롤링 릴리스와 함께 훨씬 쉽게 구현할 수 있습니다. 그러면 개발자가 위험은 줄이면서 출시 빈도를 높일 수 있습니다. 클라우드 호스티드 애플리케이션은 '고속 모놀리스'가 될 수 있으며 이를 통해 개발자가 개선 사항의 백로그를 해결할 수 있습니다.

클라우드에서 애플리케이션을 재호스팅하는 것 역시 마이크로서비스를 구현하는 효과적인 첫 단계일 수 있습니다. 클라우드의 유연성 덕분에 컨테이너에서 새 마이크로서비스와 기존 애플리케이션을 더 쉽게 배포할 수 있습니다. 그런 다음 개발자가 모놀리식 애플리케이션의 기능을 마이그레이션하는 마이크로서비스를 구현할 수 있습니다.

기존 애플리케이션을 클라우드로 이전함으로써 얻게 되는 이점을 감안하면, 클라우드 네이티브 개발 플랫폼을 선택할 때 다음 사항을 고려해야 합니다.

- 개발 플랫폼에 기존 애플리케이션을 현대화하면서 새 애플리케이션을 구축할 수 있는 옵션이 있습니까?
- 클라우드 플랫폼에서 아직 클라우드 네이티브가 전체적으로 구현되지 않은 애플리케이션을 지원할 수 있습니까?
- 기존의 Java EE 애플리케이션을 마이그레이션할 수 있도록 어떤 옵션이 제공됩니까?
- 개발 플랫폼이 모놀리스 스타일의 애플리케이션을 빠르게 제공하는 기능을 지원합니까?

애플리케이션 서버는 클라우드 네이티브 애플리케이션을 위해 실행 가능한 런타임 환경이 될 수 있지만 클라우드 네이티브 플랫폼으로 전환하면 전통적인 애플리케이션 서버의 역할이 바뀝니다. 전통적으로 애플리케이션 서버는 런타임과 배포 환경은 물론, 시스템 클러스터에서 실행되는 중앙 도메인 관리 서비스도 제공했습니다. 그러나 관리 콘솔을 비롯하여 다수의 내장 운영 툴은 더 이상 필요하지 않으며 생산성에 방해가 됩니다. 클라우드 플랫폼은 자동 관리 기능을 통해 동적 스케일링과 지속적인 제공을 지원하는 강력한 기능을 제공합니다. 전통적인 애플리케이션 서버를 그 관리 구성 요소와 함께 컨테이너에 패키징하면 개발자가 그러한 기능을 활용하지 못하게 됩니다. 다음 사항을 고려해야 합니다.

- 클라우드 플랫폼의 관리 기능과 통합된 Java EE 런타임을 위한 옵션이 플랫폼에 포함되어 있습니까?
- 레거시 애플리케이션 서버에서 실행되는 애플리케이션을 현대적인 클라우드 기반 Java EE 환경으로 마이그레이션할 수 있는 툴을 보유하고 있습니까?
- 웹 애플리케이션 등 Java EE의 모든 기능을 사용하지 않는 애플리케이션의 경우 클라우드를 위해 더 적합한 대안이 있습니까?

올바른 플랫폼과 툴을 사용함으로써 조직은 기존 애플리케이션에서 더 큰 가치를 얻으면서 클라우드 네이티브와 마이크로서비스 모델로 마이그레이션할 수 있습니다.

온프레미스 클라우드와 여러 클라우드 지원

대부분의 IT 조직은 퍼블릭 클라우드 인프라를 제공하는 단일 업체로 한정되는 것을 원치 않습니다. 또한, 대부분의 조직은 몇 년이 지나도 애플리케이션의 절반 이상이 온사이트로 실행될 것이라고 생각합니다. 대부분의 조직은 이러한 두 가지 요구 사항을 갖고 있으므로 다음 사항을 고려해야 합니다.

- 다른 제공업체의 클라우드 인프라를 사용하기 위해 애플리케이션을 변경해야 합니까? 배포, 운영, 모니터링은 어떻습니까? 각 클라우드 플랫폼에서 차이가 있습니까?
- 퍼블릭 클라우드 인프라와 온사이트 시스템 간의 차이를 어떻게 최소화할 수 있습니까? 퍼블릭 클라우드에 사용되는 클라우드 플랫폼을 온사이트로 실행하거나 조직에서 선호하는 IaaS(서비스로서의 인프라)에서 실행할 수도 있습니까?

종합적인 접근

클라우드 네이티브 개발 플랫폼을 평가할 때는 다음과 같이 그 선택의 결과가 다른 영역에 어떤 영향을 미칠지 고려해야 합니다.

- 클라우드 개발 및 배포 플랫폼과 연동하여 별도의 수정이 필요 없이 사용할 수 있는 개발 툴을 보유하고 있습니까? 그렇지 않다면, 개발자가 툴을 설정하고 통합하는 데 어느 정도의 시간이 걸리겠습니까?
- 플랫폼이 생산성 증대를 위한 사전 규정된 접근 방식이나 가이드형 접근 방식을 개발자에게 제공합니까?
- 메시징, 데이터 스토리지, 비즈니스 프로세스 또는 룰 관리를 위한 클라우드 기반 미들웨어 서비스가 제공되며 클라우드 플랫폼에서 실행할 준비가 되었습니까?
- 사전 구축된 타사 컨테이너를 애플리케이션과 통합할 수 있습니까?
- 교육 및 컨설팅 서비스를 사용할 수 있습니까? 사내의 가용 리소스가 애플리케이션을 현대화한 경험을 보유하고 있습니까?

올바른 클라우드 개발 플랫폼을 선택하는 것의 중요성

클라우드 개발을 시작하는 개발자에게 있어서 튜토리얼을 따르고 인터넷에서 소프트웨어 구성 요소를 선택하여 한 번에 하나씩 문제를 해결하는 것은 익숙한 일입니다. 이들은 먼저 런타임 환경을 통합하여 컨테이너에 애플리케이션 코드를 구축하는 방법을 익혀야 합니다. 그런 다음 전체 애플리케이션을 구축하는 데 필요한 기타 소프트웨어 구성 요소를 결정해야 합니다. 이 프로세스는 시간이 많이 걸릴 수 있으며 다음과 같은 여러 가지 문제점을 가지고 있습니다.

- 학습 곡선과 광범위한 통합의 필요성으로 인해 회사에서 수용하기 힘들 정도로 개발자의 생산성이 저하될 수 있습니다. 신규 채용된 모든 개발자가 사내에서 생성된 스택을 익혀야 합니다.
- 선택한 소프트웨어 구성 요소 전체를 유지관리하여 알려진 버그와 보안 취약점이 발생하지 않도록 해야 합니다. 오픈소스 구성 요소의 경우 엔터프라이즈 사용에 적합한지도 살펴보아야 합니다.
- 대다수 조직은 개발자가 직접적인 비즈니스 가치를 제공하지 않는 설정 및 배포 등의 기능을 위한 코드 작성을 마무리할 수 있다고 전합니다. 게다가 이러한 기능은 대부분 배포 플랫폼의 유연성과 관련이 있으며 배포 플랫폼에 포함되어야 하는 상용 서비스입니다.

구성 요소 컬렉션을 통합하는 데 있어 가장 큰 문제는 개발과 운영의 전체 학습 곡선을 줄이는 데 도움이 되지 않는다는 것입니다. 대안은 런타임 선택에서 개발 시작, 운영 배포에 이르는 모든 단계를 처리할 수 있도록 설계된 클라우드 개발 플랫폼을 선택하는 것입니다.

클라우드 네이티브 애플리케이션으로의 전환은 단기간에 완료할 수 없는 프로세스입니다. 많은 조직이 경험을 축적하면서 이 프로세스를 개선해 나갑니다. 클라우드 네이티브 개발에 수반되는 모든 사항을 감안할 때, 이것이 개발에서 배포를 아우르는 포괄적인 접근 방식이고 성공할 가능성이 높다는 것을 분명히 알 수 있습니다. Red Hat은 Red Hat® OpenShift Application Runtimes 및 Red Hat OpenShift를 통한 플랫폼을 제공합니다.

RED HAT® OPENSHIFT

redhat.

백서 클라우드 네이티브 개발 플랫폼 선택에 대한 이그제큐티브 가이드

RED HAT OPENSHIFT APPLICATION RUNTIMES 및 RED HAT OPENSHIFT

Red Hat OpenShift Application Runtimes는 클라우드 네이티브 애플리케이션 개발을 단순화하도록 설계되었습니다. 관리되고 통합된 런타임과 프레임워크는 사전 규정된 환경을 통해 경로 가이드를 제공하므로 개발을 바로 시작할 수 있습니다. OpenShift Application Runtimes는 하이브리드 클라우드용으로 최적화된 컨테이너 애플리케이션 플랫폼인 Red Hat OpenShift에 기반을 두므로 개발 및 배포 플랫폼 전체가 간소화됩니다.

Red Hat OpenShift는 애플리케이션 컨테이너를 구축하고 실행할 수 있는 셀프 서비스 플랫폼을 개발자와 운영팀에게 제공합니다. OpenShift를 사용하면 새로운 마이크로서비스 또는 애플리케이션을 위한 환경을 몇 분 이내에 프로비저닝할 수 있습니다. OpenShift에서는 위험을 훨씬 더 완화하고 배포 주기를 크게 단축할 수 있습니다. 몇 번만 클릭하면 강력한 자동 CI/CD 구축 및 배포 파이프라인이 제공됩니다. Red Hat OpenShift Application Runtimes와 Red Hat OpenShift를 Red Hat 개발 툴 및 Red Hat Consulting과 함께 사용하면 조직은 더 빠르고 안전하게 클라우드 네이티브 애플리케이션으로 전환할 수 있습니다.

OpenShift Application Runtimes의 런타임은 개발자의 업무에 적합한 툴을 제공합니다. 마이크로서비스 개발에서 개발자가 Java EE를 사용하면 기존의 전문성과 함께 마이크로서비스에 대한 요구를 충족할 수 있도록 기능이 향상된 최신 Java MicroProfile 표준 또는 이벤트 기반의 프레임워크를 활용해 동시 실행률이 높고 대기 시간이 짧은 워크로드에 맞춰 스케일링할 수 있는 반응형 마이크로서비스를 구축할 수 있습니다. Node.js 런타임은 모바일 및 웹 애플리케이션에서 자주 사용되는 JavaScript 백엔드 서비스를 위해 제공됩니다. 기존 애플리케이션을 마이그레이션할 경우 현대적인 모듈식 클라우드 레디 아키텍처 기반의 Java EE 7 인증 애플리케이션 서버인 Red Hat JBoss® Enterprise Application Platform에 기반한 런타임이 제공됩니다. 모든 런타임은 Red Hat에 의해 테스트되고 검증되었습니다.

개발자가 빠르게 시작할 수 있도록 Red Hat OpenShift Online에서 실행되는 클라우드 기반의 무료 SaaS 툴이 Red Hat Developers 웹사이트를 통해 제공됩니다. 개발자가 샘플 애플리케이션과 런타임을 선택하면 OpenShift Online의 클라우드에서 바로 구축하고 실행할 수 있는 전체 코드 베이스가 제공됩니다. 교육 툴로, 모든 런타임에 동일한 샘플 애플리케이션이 제공되므로 개발자가 사용 가능한 아키텍처 스타일의 장점을 쉽게 비교할 수 있습니다.

자세히 알아보려면 https://www.redhat.com/ko/technologies/cloud-computing/openshift/application-runtimes를 참조하세요.

한국레드햇 홈페이지 https://www.redhat.com/korea

RED HAT 소개

Red Hat은 세계적인 오픈소스 솔루션 공급업체로서 커뮤니티 기반의 접근 방식을 통해 신뢰도 높은 고성능 클라우드, Linux, 미들웨어, 스토리지, 가상화 기술을 제공합니다. 또한, 전세계 고객에게 높은 수준의 지원과 교육 및 컨설팅 서비스를 제공하여 권위있는 어워드를 다수 수상한 바 있습니다. Red Hat은 기업, 파트너, 오픈소스 커뮤니티로 구성된 글로벌 네트워크의 허브 역할을 하며 고객들이 IT의 미래를 준비하고 개발할 수 있도록 리소스를 공개하여 혁신적인 기술 발전에 기여하고 있습니다.

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

www.redhat.com/ko
F10657_V1_0118_KVM