redhat.

클라우드 네이티브 애플리케이션 구현을 위한 과정

8단계 여정 가이드

E-BOOK

목차

"디지털 측면에서 앞선 기업들은 시장 점유율 확대 가능성이 8배 높지만 디지털 네이티브 기업에 비해서는 여전히 뒤처집니다."

Bain 설문조사: 전통적인 기업을 위한 디지털 이전 과정과 컨테이너의 역할

1. 속도: 디지털 비즈니스를 위한 필수 조건

디지털 비즈니스라고 하면 모바일 기기, 지능형 센서, 웨어러블 기기, 가상 현실, 챗봇, 블록체인 및 머신 러닝 등의 혁신적인 기술을 떠올리게 마련입니다. 일부는 전통적인 비즈니스 모델에 구애받지 않고 기존의 기업 환경과 산업 전반을 뒤흔든 새로운 디지털 네이티브 기업들이 급격히 늘어났다는 사실을 떠올리기도 합니다. 대부분의 조직에서 디지털 비즈니스란 조직의 민첩한 문화에 중점을 두는 방식을 뜻합니다. 더 빠르고 유연한 개발 및 제공 모델을 통해서만 급격한 수요 증가에 대응할 수 있기 때문입니다. 대부분의 조직은 기술 기반을 완전히 재구축하거나 즉각적으로 새로운 사례와 관점을 채택할 만한 여유가 없기 때문에, 점진적으로 문화, 프로세스 및 기술의 근본적 변화를 수용해 속도와 민첩성을 높이고자 합니다.

사용자가 비즈니스에 참여하고 기업이 경쟁력을 유지하기 위해서는 소프트웨어가 점차 핵심 역할을 하게 되면서 애플리케이션 개발 및 배포 속도는 새로운 디지털 비즈니스의 필수 조건이 되었습니다.

클라우드 네이티브 접근 방식은 클라우드 컴퓨팅의 자동화와 민첩성을 위해 최적화된 프로세스를 채택하고 클라우드 원칙과 서비스를 사용함으로써 **기존 애플리케이션을 현대화하고 새로운 애플리케이션을 구축**하는 것을 의미합니다. 이 E-book에서는 현재 시점에서 클라우드 네이티브 애플리케이션 접근 방식을 채택하는 데 이르기까지의 성공적인 여정을 단계별로 자세하게 설명합니다.

*"'클라우드 네이티브'란 **변화에 신속히 대응하고 예측 불가능한 요소를 줄이는 역량을 높일 수 있도록** 애플리케이션, 아키텍처, 플랫폼/인프라 및 프로세스를 **효율적으로 제공**하는 방식을 가리킵니다."*

CHRISTIAN POSTA
RED HAT 수석 아키텍트 겸
JAVA 개발자를 위한 마이크로서비스 저자

출처: INFOQ, "DEFINING CLOUD NATIVE: A PANEL DISCUSSION (클라우드 네이티브 정의: 패널 토론)," 2017.

2. 클라우드 네이티브 애플리케이션이란?

클라우드 네이티브 애플리케이션은 애플리케이션 배포 위험을 낮추면서 속도와 유연성 및 품질을 높이기 위해 클라우드 컴퓨팅 모델을 활용하도록 설계된 애플리케이션을 뜻합니다. 클라우드 네이티브 방식은 이름에도 불구하고 이러한 방식은 애플리케이션이 배포되는 ***위치***가 아닌, 애플리케이션이 구축, 배포 및 관리되는 ***방식***에 중점을 두고 있습니다.

클라우드 네이티브 접근 방식은 마이크로서비스 아키텍처와 유사합니다. 하지만 마이크로서비스가 클라우드 네이티브 애플리케이션 구축의 한 결과일 수는 있지만, 프로덕션 환경에서 마이크로서비스를 관리하기 위해서는 몇 가지 단계를 거쳐 성숙한 수준에 도달해야 합니다. 클라우드 네이티브 애플리케이션이 제공하는 모든 이점을 활용하기 위해 마이크로서비스가 필수는 아닙니다. 동일한 원칙으로 더 나은 모듈식 모놀리스 구축에 집중하여 클라우드 네이티브 접근 방식의 이점을 누리는 조직이 많습니다.

클라우드 네이티브 애플리케이션 개발 및 배포로의 진화하는 과정은 문화와 프로세스, 아키텍처 및 기술에 영향을 주는 다차원적인 여정입니다. 다시 말해서 이는 결과보다는 과정 전체를 의미하며 그에 따른 주기적인 변화를 거쳐야 하는 과제를 수반합니다.

3. 전통적인 애플리케이션과 클라우드 네이티브 애플리케이션 비교

클라우드 네이티브 애플리케이션과 전통적인 애플리케이션 개발은 변화가 필요한 측면에서 차이가 있습니다.

표 1. 전통적인 애플리케이션과 클라우드 네이티브 애플리케이션 개발 비교

	전통적인 방식	클라우드 네이티브 방식
주안점	지속성 및 안정성	출시 속도
개발 방법론	워터폴(waterfall), 세미 애자일(semi-agile) 개발	애자일 개발, DevOps
팀	격리된 개발, 운영, QA 및 보안 팀	협업 DevOps 팀
제공 주기	장기	단기 및 지속적
애플리케이션 아키텍처	긴밀하게 결합(Tightly coupled) 모놀리식	탄력적으로 결합(Loosely coupled) 서비스 기반 API(애플리케이션 프로그래밍 인터페이스) 기반 통신
인프라	서버 중심 온프레미스용 설계 인프라 종속적 수직적 확장 피크 용량을 위해 사전 프로비저닝	컨테이너 중심 온프레미스 및 클라우드용 설계 인프라 전반에 이식 가능 수평적 확장 온디맨드 용량

3.1 전통적인 애플리케이션 개발 및 배포

비즈니스 운영에 필수적인 애플리케이션은 디지털 경험을 고려하지 않고 설계된 경우가 많았습니다. 이러한 애플리케이션은 수명(lifespans)이 길고, 장기간에 걸쳐 긴밀하게 결합된 모놀리스로 구축되며, 매우 구체적인 사양이 제공되기 한참 전에 결정됩니다.

이는 대부분 장기간에 걸쳐 이루어지는 순차적인 워터폴(waterfall) 개발 방식으로, 최근에 와서야 세미 애자일(semi-agile) 방식과 결합되었습니다. 애플리케이션 개발, 테스트, 보안 규정 준수, 배포 및 관리의 각 단계는 별도의 팀과 역할, 업무가 할당된 기능 영역으로 분리되었으며, 담당자 간 의사소통이 순차적으로 이루어집니다.

이러한 애플리케이션은 긴밀히 결합된 대규모의 다목적 애플리케이션으로 구축되었으며, 사용자 인터페이스 및 다양한 애플리케이션 서비스, 데이터 액세스 코드 및 기타 구성 요소들이 기술 환경에 상관없이 단일 애플리케이션으로 결합되었습니다. 예를 들어, 긴밀히 결합된 모놀리스로 구축된 전자 상거래(e-commerce) 애플리케이션은 일반적으로 웹 사용자 인터페이스, 제품 카탈로그, 쇼핑 카트, 제품 추천, 제품 평점 및 후기, 결제 시스템 및 전자 상거래 웹사이트에서 구매 시 필요한 기타 구성 요소를 위한 모든 기능이 하나의 애플리케이션에 통합된 형태입니다.

대부분의 전통적인 애플리케이션의 경우, 인프라는 애플리케이션에 필요한 피크 용량을 위해 사전에 프로비저닝되었고, 수직적 확장을 통해 서버의 하드웨어 용량을 높여 확장이 이루어졌습니다.

2020년까지 프라이빗 데이터 센터에서 퍼블릭 클라우드로 마이그레이션된 모드 1* 애플리케이션의 50% 이상이 클라우드 네이티브 아키텍처 원칙에 따라 재작성될 예정이며, 이는 2017년의 10% 미만에 비해 증가한 수치입니다.

Gartner: Why You Must Begin Delivering Cloud-Native Offerings Today, Not Tomorrow(지금 클라우드 네이티브 솔루션을 제공해야 하는 이유), 2018년 1월

3.2 클라우드 네이티브 애플리케이션 개발 및 제공

출시 속도에 초점을 맞춘 클라우드 네이티브 애플리케이션 개발을 위해서는 보다 민첩한, 서비스 및 API 기반의 개발 및 지속적인 제공 방식이 필요합니다. 이러한 기능은 개발 및 제공팀, 한층 발전된 모듈식 아키텍처, 그리고 온디맨드로 수평적 확장이 가능하고 여러 환경을 지원할 수 있으며 애플리케이션 이식성을 제공할 수 있는 유연한 인프라 전반에서, DevOps 협업을 통해 지원됩니다.

조직은 현대적인 클라우드 기술로 제공되는 유연성과 민첩성을 통해, 전통적인 애플리케이션을 클라우드 환경으로 이전하여 민첩성 향상과 온디맨드 컴퓨팅 용량을 활용하고자 합니다.

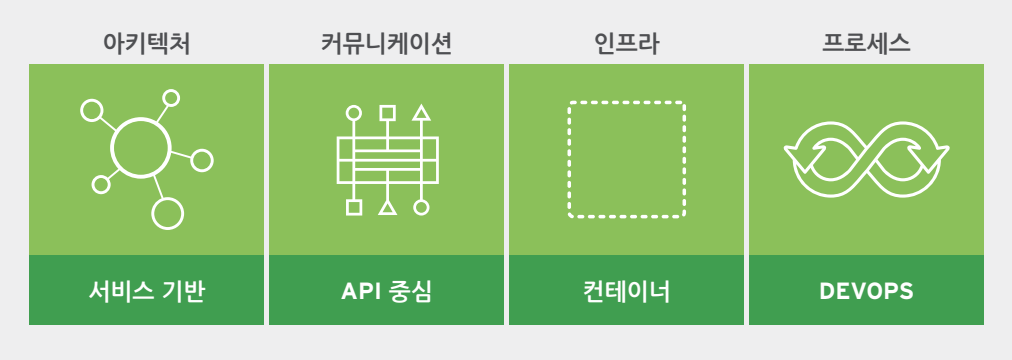
그러나 전통적인 플랫폼에 구축된 운영 기능의 상당수는 오래되어 클라우드 환경에서 불필요해지거나, 클라우드 환경에서 자체적으로 제공되어 사용할 수 있게 되었습니다. 클라우드 환경은 호스트의 라이프사이클 관리를 간소화할 뿐만 아니라 조직이 변경 불가능한 인프라 원칙을 활용하고 단일 애플리케이션 인스턴스의 필요에 따라 호스트를 튜닝하도록 지원합니다.

클라우드 네이티브 애플리케이션 구현을 위한 과정은 조직별로 달라질 수 있습니다. 마이크로서비스를 생성한다고 해서 디지털 비즈니스에 필요한 서비스 품질과 제공 빈도가 높아지는 것은 아닙니다. 마찬가지로 애자일 개발 방식 또는 IT 자동화를 지원하는 툴을 도입한다고 해서 클라우드 네이티브 접근 방식의 속도가 빨라지는 것은 아닙니다. 오히려, 사례, 기술, 프로세스 및 사고 방식이 결합될 때 성공적인 결과를 기대할 수 있습니다.

* ***Gartner가 정의한 바이모달(bimodal)****은 예측 가능성과 탐험성(exploration)이라는 별개의 일관된 2가지 업무를 관리하는 방식을 가리킵니다.* ***모드 1은 이해하기 쉽고 예측 가능성이 높은 영역에 최적화된 업무 방식입니다. 이는 디지털 세계에 적합한 상태로 레거시 환경을 개선하면서 이미 알려진 요소를 활용하는 데 중점을 둡니다.*** *모드 2는 실험적인 방식으로 새로운 문제를 해결하고 불확실성의 영역을 최적화하는 탐구적인 성격을 띱니다.*

4. 클라우드 네이티브 애플리케이션 개발 및 배포를 위한 4가지 원리

클라우드 네이티브 애플리케이션 개발은 4가지 핵심 원칙을 기반으로 클라우드 컴퓨팅 모델을 최대한 활용하는 애플리케이션을 구축하고 실행하는 방식입니다. 서비스 기반 아키텍처, API 기반 통신, 컨테이너 기반 인프라 및 DevOps 프로세스가 이에 해당합니다.



서비스 기반 아키텍처

마이크로서비스와 같은 서비스 기반 아키텍처는 탄력적으로 결합된 모듈식 서비스 구축을 지원합니다. 미니서비스와 같은 다른 모듈식 아키텍처 방식은 탄력적으로 결합된 서비스 기반 설계를 바탕으로 조직이 복잡도를 높이지 않고도 애플리케이션을 신속히 생성할 수 있도록 도와줍니다.

API 기반 통신

복잡성을 줄이고 배포, 확장성 및 유지관리 오버헤드를 낮춘 경량의, 기술에 얽매이지 않는 API를 통해 서비스가 노출됩니다. 기업은 노출된 API를 통해 내부 및 외부에서 새로운 역량과 기회를 발굴할 수 있습니다.

API 기반 설계에서는 직접 연결, 공유 메모리 모델 또는 다른 팀의 데이터 저장소에 직접 연동되는 위험을 피해, 네트워크 상에서 서비스 인터페이스 호출을 통한 통신만이 가능합니다. 이러한 설계를 통해 애플리케이션 및 서비스의 범위가 다른 기기 및 형태로 확장됩니다.

컨테이너 기반 인프라

클라우드 네이티브 애플리케이션은 컨테이너에 의존하여 기술 환경 전반에서 공통된 운영 모델을 지원하고 퍼블릭, 프라이빗 및 하이브리드를 포함한 다양한 환경 및 인프라 전체에서 진정한 애플리케이션 이식성을 지원합니다. 컨테이너 기술은 운영 시스템 가상화 기능을 사용하여 다양한 애플리케이션에서 이용 가능한 컴퓨팅 리소스를 분할하고, 애플리케이션이 안전하게 서로 격리되도록 합니다.

클라우드 네이티브 애플리케이션은 수평적으로 확장되면서 애플리케이션 인스턴스를 간단히 추가해 용량을 늘립니다. 이는 컨테이너 인프라 내에서 자동화를 통해 이루어지는 경우가 많습니다.

컨테이너의 낮은 오버헤드와 높은 집적도는 동일한 가상 머신 또는 물리 서버 내에서 다수의 컨테이너가 호스팅되도록 지원하기 때문에 클라우드 네이티브 애플리케이션을 제공하기 위한 최적의 조건입니다.

DEVOPS 프로세스

클라우드 네이티브 방식의 애플리케이션 개발은 CD(Continuous Delivery) 및 DevOps 원칙을 통해 애자일 방식을 따르며, 그러한 CD 및 DevOps 원칙은 개발, 품질 보증, 보안, IT 운영팀 및 서비스 제공에 관련된 팀들이 협업을 통해 애플리케이션을 구축하고 제공하는 데 중점을 둡니다.

대규모 조직의 과반수(51%)가 이미 DevOps를 도입했습니다. 그러나 현재 이들 대부분은 10-40%의 애플리케이션(일반적으로 20%)에 대해서만 DevOps를 활용합니다.

개발자를 위한 PaaS 보기 IDC 설문조사, 2017년 11월

5. 클라우드 네이티브 애플리케이션 구현을 위한 과정: 8단계

1단계: DEVOPS 문화 및 사례의 발달

클라우드 네이티브 애플리케이션 구현을 위한 과정에서는 더 빠르고 효율적으로 애플리케이션을 구축하고 배포하기 위해 개발 및 IT 운영팀이 다양한 방식으로 발전해야 합니다. 업종이나 규모에 관계없이 모든 기업은 DevOps 문화를 함께 만들어 나가기 위한 다양한 활동과 기술, 팀 및 프로세스를 고려해야 합니다. 새로운 기술과 신속한 접근 방식 및 긴밀한 협업의 이점을 활용하려면 DevOps의 원칙과 문화적 가치를 진정으로 수용하고 이러한 가치를 중심으로 조직을 구성해야 합니다.

급격한 디지털 혁신의 시대에는 다양한 분산 환경과 고도의 맞춤형 레거시 애플리케이션 및 새로운 애플리케이션 워크로드 관리가 복잡하기 때문에 일부 조직에는 DevOps가 큰 과제가 될 수 있습니다. 애플리케이션 포트폴리오 전반에는 여전히 DevOps 적용을 확대할 수 있는 잠재력이 숨겨져 있습니다.

DevOps 문화를 도입하려면 툴과 기술이 필요할 뿐만 아니라, 애플리케이션 개발 및 제공에 대해 더 긴밀히 통합된 협업 방식을 도입하려는 의지와 신뢰가 바탕이 되어야 합니다. 오픈소스 소프트웨어 프로젝트 문화는 DevOps 문화를 구축하기 위한 길잡이가 될 수 있습니다.

Red Hat Open Innovation Labs에서 조직은 실험, 초기에 겪는 실패, 의사 결정의 투명성을 권장하는 DevOps 프로세스를 도입하고, 인식 및 보상을 활용하여 신뢰와 협력의 수준을 높이게 됩니다. 혁신을 촉진하도록 설계된 이러한 환경에서 각 팀은 혁신적인 오픈소스 기술을 사용하여 신속하게 프로토타입을 구축하고 DevOps를 경험하며 애자일 워크플로우를 도입합니다.

Red Hat Open Innovation Labs가 귀사의 DevOps 구현을 어떻게 지원하는지 알아보세요.
E-Book 다운로드

"잘 구조화된 모놀리스를 구축할 수 없다면, 잘 구조화된 마이크로서비스를 구축할 수 있을까요?"

SIMON BROWN
CODINGTHEARCHITECTURE.COM/PRESENTATIONS/SA2015-MODULAR-MONOLITHS

2단계: 신속한 모놀리스(MONOLITHS)를 통한 기존 애플리케이션 속도 향상

클라우드 네이티브 애플리케이션 구현을 시작할 때 조직은 새로운 개발에만 초점을 맞춰서는 안됩니다. 레거시 애플리케이션은 비즈니스 운영과 수익 창출에 있어 매우 중요하며 간단히 교체될 수 없는 경우가 많습니다. 오히려 이는 새로운 클라우드 네이티브 애플리케이션과 통합되어야 합니다. 그러면 기존 모놀리스를 어떻게 가속화할까요? 해답은 기존 모놀리식 아키텍처를 한층 발전된 모듈식 서비스 기반 아키텍처 및 API 기반 통신으로 이전함으로써 신속한 모놀리식 방식을 채택하는 것입니다.

모놀리식 애플리케이션을 마이크로서비스로 리팩토링하는 까다로운 작업을 시작하기 전에 조직은 먼저 모놀리식 아키텍처를 위한 확고한 기반을 구축해야 합니다. 모놀리식 애플리케이션은 민첩성이 부족하지만, 모놀리식 애플리케이션에 대한 부정적인 평가는 대부분 구축 방식 때문입니다. 그렇지만 신속한 모놀리스는 복잡성과 비용이 가중되지 않고도 마이크로서비스와 관련된 애자일 방식의 많은 이점을 실현할 수 있습니다.

신속한 모놀리식 접근 방식을 평가하여 애플리케이션이 확고한 설계 원칙과 적절하게 정의된 도메인 영역에 따라 구축되도록 할 수 있습니다. 이러한 방식은 필요한 경우 보다 점진적이고 위험이 낮은 마이크로서비스 아키텍처로의 전환을 지원합니다. 이처럼 신속한 모놀리스를 진화하면 성공적인 마이크로서비스 아키텍처의 기반을 다질 수 있습니다.

애플리케이션이 신속한 모놀리식 접근방식을 사용하여 설계되지 않은 경우라 하더라도, 기존 모놀리스를 컨테이너 기반 플랫폼으로 전환하면 속도를 높일 수 있습니다. 이러한 전환은 배포 속도 및 ROI(투자수익률)를 높여줍니다. 이후 통합이나 모놀리스 기능 구현은 클라우드 네이티브 기술 및 접근 방식을 사용하여 구축할 수 있습니다.

또한 단계적 접근 방식을 사용하여 원하는 속도로 모놀리스를 더 작은 구성 요소로 세분할 수도 있습니다.

3단계: 애플리케이션 서비스를 사용하여 개발 속도 촉진

재사용성은 언제나 소프트웨어 개발 속도를 높이는 데 있어서 핵심 요소였으며, 클라우드 네이티브 애플리케이션 또한 마찬가지입니다. 그러나 이점을 최대한 활용하기 위해선 클라우드 네이티브 애플리케이션의 재사용 가능 구성 요소를 기본 클라우드 네이티브 인프라로 최적화하고 통합해야 합니다.

기본 컨테이너 기반 인프라에 최적화되어 통합된 기존 구성 요소를 사용할 수 있는데, 캐싱 서비스, 룰 또는 워크플로우 엔진, 통합 커넥터, 모바일 및 API 관리 기능, 데이터 가상화 서비스, 메시징 브로커, 또는 서버리스(serverless) 프레임워크를 재생성하는 이유는 무엇일까요? 이러한 애플리케이션 서비스는 SaaS(서비스로서의 소프트웨어), PaaS(서비스로서의 플랫폼) 또는 iPaaS 서비스 등에 상관없이 효과적으로 즉시 사용할 수 있는(ready-to-use) 개발자 툴입니다.

개발자들이 개발 속도를 높이고 새로운 애플리케이션을 보다 빨리 출시할 수 있도록, 클라우드 네이티브 애플리케이션에는 이러한 종류의 서비스가 하나 이상 필요할 수 있습니다. DevOps 및 컨테이너가 클라우드 네이티브 애플리케이션 제공 및 배포 속도를 높이는 반면, 애플리케이션 서비스는 해당 개발을 가속화합니다.

예를 들어, 클라우드 네이티브 애플리케이션 개발자는 컨테이너 기반 인프라에서 작업을 원활히 수행하기 위해서 뿐만 아니라, CI/CD 파이프라인, 롤링 및 blue/green 배포, 자동 확장, 내결함성 등과 같은 플랫폼 기능을 활용하기 위해, 특별히 구축한 애플리케이션 서비스를 이용할 수 있습니다.

4단계: 올바른 작업에 적합한 툴 선택

IoT(사물 인터넷), 머신 러닝, AI(인공 지능), 데이터 마이닝, 이미지 인식, 자율 주행 차량 등 소프트웨어 연구 분야가 확장되면서 소프트웨어 개발을 위한 다양한 프레임워크, 언어 및 접근 방식이 늘어났습니다.

언어나 프레임워크 선택이 특정 비즈니스 애플리케이션 요구에 점점 특화되면서 클라우드 네이티브 애플리케이션 구축이 다양한 모습으로 나타나고 있습니다. 그 결과 복잡성이 증가하면서, 클라우드 네이티브 개발을 지원하는 프레임워크와 언어, 아키텍처가 적절히 결합된 컨테이너 기반 애플리케이션 플랫폼 활용의 장점이 부각되고 있습니다.

클라우드 네이티브 개발을 위해서는 해당 작업에 적합한 툴을 선택해야 합니다. 12 팩터 접근 방식, 도메인 기반 설계, 테스트 기반 설계 및 개발, MonolithFirst, 신속한 모놀리스, 미니서비스, 또는 마이크로서비스를 사용한 클라우드 네이티브 애플리케이션의 구현 여부에 상관없이, 클라우드 네이티브 플랫폼은 선택한 개발 요구사항을 지원하기 위해 프레임워크와 언어 및 아키텍처를 적절히 결합해야 합니다. 뿐만 아니라 기본 컨테이너 기반 플랫폼은 기술 변화에 따라 지속적인 업데이트를 통해 관리되는(curated) 런타임 및 프레임워크를 지원해야 합니다.

그림 1. 더욱 다양해진 클라우드 네이티브 애플리케이션 개발

5단계: 셀프 서비스 온디맨드 인프라 제공

개발자들은 애자일 방식으로 소프트웨어를 신속하게 구축해 업데이트했으나, 필요한 시기와 위치에 인프라 액세스를 적시 제공하기 위한 효율적인 메커니즘은 마련하지 못한 상태입니다. 애플리케이션이 프로덕션 환경으로 릴리스될 때 전반적인 출시 속도는 영향을 받습니다. 인프라는 저렴하고 엔지니어링 인력은 비싼 시대에서 IT 운영팀이 리소스를 릴리스하기 위해 티켓을 제출하고 몇 주간 기다리는 것은 더 이상 지속가능한 모델이 아닙니다.

셀프 서비스와 온디맨드 인프라 프로비저닝은 개발자들이 원할 때 필요로 하는 인프라에 액세스하도록 허용하여, 승인되지 않은 섀도우 IT(Shadow IT)에 대한 확실한 대안을 제공합니다. 그러나 이러한 모델은 IT 운영팀이 자주 다이나믹하고 복잡한 환경 전반에서 제어력과 가시성을 확보하는 경우에만 효과가 있습니다.

컨테이너와 컨테이너 오케스트레이션 기술은 데이터 센터, 프라이빗 클라우드 및 퍼블릭 클라우드와 같은 다양한 인프라 환경 전반에서 기본 인프라에 대한 액세스를 추상화하고 간소화하며 강력한 애플리케이션 라이프사이클 관리를 제공합니다. 컨테이너 플랫폼은 추가적인 셀프 서비스, 자동화 및 애플리케이션 라이프사이클 관리 기능을 제공합니다. 개발자와 운영팀은 이 모델을 활용하여 일관된 환경을 신속하게 가동하고, 인프라 프로비저닝 관련 장애나 지연없이 애플리케이션 구축에 집중할 수 있습니다.

표준화 또한 셀프 서비스 모델의 중요한 부분입니다. 이는 조직이 일관된 자동화와 제공을 통해 비즈니스 목표를 달성하도록 지원합니다. 프로세스 표준화에는 새로운 환경으로 애플리케이션을 릴리스하는 등의 작업 수행에 필요한 정확한 이벤트 및 활동 순서를 매핑하는 과정이 포함됩니다.

컨테이너 또한 어떠한 클라우드 제공업체에서도 배포 후 실행할 수 있는 클라우드 네이티브 애플리케이션 생성을 포함한 애플리케이션 이식성을 지원합니다. 이식성을 통해 자유롭게 언제든 원하는 클라우드 제공업체를 선택하고, 손쉽게 다른 클라우드 제공업체로 마이그레이션할 수 있으며, 관련 비용을 최적화하고 특정 클라우드 제공업체 API에 맞춰 코딩할 필요 없이 멀티클라우드 애플리케이션을 개발할 수 있습니다.

클라우드 네이티브로의 여정을 지원하는 다양한 사례와 기술에 대해 자세히 알아보세요.
오픈 프랙티스 라이브러리 탐색

6단계: IT 자동화를 통한 애플리케이션 제공 가속화

IT 또는 인프라 자동화는 수동 IT 태스크를 줄여서 클라우드 네이티브 애플리케이션 제공 속도를 높이기 위한 핵심 요소입니다. 자동화는 네트워크 및 인프라 프로비저닝에서 애플리케이션 배포 및 설정 관리에 이르기까지, 모든 작업이나 구성 요소와 통합되고 이에 적용될 수 있습니다.

IT 관리 및 자동화 툴은 많은 인력을 필요로 하여 출시 속도를 지연시키는 수작업을 대체하거나 줄일 수 있는 반복가능한 프로세스, 룰 및 프레임워크를 생성합니다. 이는 컨테이너와 같은 특정 기술, DevOps와 같은 방법론, 또는 클라우드 컴퓨팅, 보안, 테스트, 모니터링 및 알림과 같이 더 넓은 영역으로 확장될 수 있습니다. 그 결과, 자동화는 전체적으로 가치 창출을 앞당겨 IT 최적화 및 디지털 트랜스포메이션을 위한 핵심 역할을 수행합니다.

IT 자동화 가이드

1. IT 운영에 대한 전사적인 프로그래밍 기반 자동화 접근 방식을 채택하세요. 서비스 요구사항을 설계하기 위해 조직 전반에서 협업을 위한 대화 채널을 도입하세요.
2. 자동화 언어 및 프로세스를 학습하기 위한 기반으로 자동화 샌드박스(automation sandboxes)를 고려하세요.
3. 자동화를 신중히 검토하세요. 수동 제어 기능을 유지하는 것이 안심이 되더라도, 불필요한 수동 단계를 제거하세요.
4. 체계적인 방법으로 실현 가능한 작은 단계부터 자동화 문제를 점진적으로 해결하는 방식을 고려하세요. 각 단계는 폭넓은 자동화 사례를 생성하기 위한 이전 단계를 기반으로 구축합니다.
5. 컴퓨팅, 네트워크, 스토리지 또는 프로비저닝 등 한 가지 태스크 또는 서비스를 자동화하여 시작하세요. 해당 자동화 작업을 다른 영역과 공유하고 이를 체계적으로 구축하세요.
6. 사용자의 권한을 보강하고 서비스 제공 속도를 높이는 셀프 서비스 카탈로그를 구현하세요.
7. 미터링, 모니터링 및 차지백 정책과 프로세스를 구현하세요.

점차 통합된 풀스케일 자동화가 실현되어 효율성이 높아지고 DevOps 및 혁신이 가속화됩니다.

"자동화된 엔터프라이즈"에서 IT 자동화의 중요성에 대해 자세히 알아보세요.
E-Book 다운로드

CD는 여러 팀이 짧은 주기 내에 유용한 소프트웨어를 계속 개발하면서 해당 소프트웨어가 언제든지 안정적으로 릴리스될 수 있도록 하는 소프트웨어 엔지니어링 접근 방식입니다. 위험 부담이 적은 안정적인 릴리스를 통해, CD는 사용자 피드백과 시장 변화 및 비즈니스 전략 변경을 통합하도록 소프트웨어의 지속적 개선을 지원합니다.

Gartner의 정의

7단계: 지속적 제공(CD) 및 고급 배포 기술 구현

릴리스 주기가 길어지면 소프트웨어 버그를 발견해서 해결하기까지 오랜 시간이 걸리고, 고객 및 시장의 수요 변화에 대한 시기적절한 대응을 방해할 가능성이 있습니다. 모바일, 웹 또는 IoT 애플리케이션과 같이 트래픽이 높은 애플리케이션의 경우, 버그가 해결되지 않으면 여러 사용자가 영향을 받게 되므로 고객 경험이 저하되고 보안이나 안전 문제를 유발해 생산성이나 수익이 낮아집니다. 기타 내부 비즈니스 애플리케이션의 경우에도 운영이 중단(outages)되거나 소프트웨어 버그 해결이 지연되면 상당한 비용이 발생할 수 있습니다.

애자일 개발 방식이 진화하면서 더욱 일찍 그리고 자주 릴리스하는 모델이 생성되었습니다. DevOps 및 CD 방식은 개발자, 운영, 품질 보증 및 보안팀이 긴밀한 협업을 통해 소프트웨어 제공 프로세스를 개선할 수 있도록 지원하면서 이러한 방식을 확장합니다. 그 결과 코드 변경을 프로덕션 환경에 신속하고 안정적으로 푸시해 개발자들에게 빠르게 피드백을 제공할 수 있습니다. 이처럼 반복적이고 빠른 피드백 루프는 CI/CD를 통해 지원되어, 자동화된 테스트, 취약점 검사, 보안 규정 준수 및 규정 확인을 포함해 애플리케이션 제공의 모든 측면을 포괄하는 엔드 투 엔드 자동화 서비스 제공 시스템까지 인프라 자동화를 확장합니다. 자동화된 제공 파이프라인의 목표는 운영 능력 저하 없이 서비스 제공 위험을 낮추면서 업데이트를 제공하는 것입니다.

CD를 실현하는 첫 단계는 CI(Continuous Integration)대한 지원입니다. CI 시스템은 변경 사항에 대해 다양한 소스 컨트롤 리포지토리를 관리하고 해당 테스트를 실행하며 Jenkins와 같이 각 소스 컨트롤 변경으로부터 최신 버전의 애플리케이션을 자동으로 구축하는 빌드 시스템입니다.

Red Hat Ansible® Automation과 같은 현대적인 자동화 기술이 CI/CD를 어떻게 지원하는지 확인하세요.
백서 다운로드

"고급 배포 기법은 혁신에 구조적인 토대와 명확성을 부여합니다. 성숙한 배포 방법론은 진정한 실험과 피드백 및 분석을 가능하게 하는 환경을 만듭니다. 향상된 실험 환경을 통해 더 나은 혁신을 실현할 수 있습니다."

BURR SUTTER
RED HAT 개발자 경험 담당 디렉터
REDHAT.COM/KO/ENGAGE/TEACHING-AN-ELEPHANT-TO-DANCE

고급 배포 패턴은 소프트웨어 릴리스의 위험을 줄이고, 고객에게 의도치 않은 부정적인 영향을 주지 않으면서 결과가 통제된 실험을 위한 환경을 구축하는 것을 목표로 합니다. 이러한 목표는 조직 전체에서 혁신을 강화하는 데 필수적입니다.

고급 배포 기술은 서비스 시간대와 다운타임을 동반한 업무 시간 외 주말에 이루어지는 활동에서, 고객의 애플리케이션 이용에 지장을 주지 않으면서도 프로덕션 환경 내 다운타임이 전혀 없는 일상 업무 활동으로 서비스 제공의 속성에 변화를 가져옵니다.

이러한 기술은 새로운 배포 작업으로 인해 고객이 불편을 겪지 않도록 하고, 비즈니스 요구에 적합한 빈도로 업데이트 및 릴리스를 제공합니다. 다음은 애플리케이션 활용 사례에 따라 제로 다운타임 배포를 실현하는 데 사용할 수 있는 일반적인 배포 기술의 예입니다.

롤링 배포는 애플리케이션의 모든 인스턴스를 한 번에 업데이트하는 대신 각 인스턴스가 트래픽을 수신하지 않도록 로드 밸런서에서 이를 제거해 개별적으로 업데이트되는 패턴입니다. 인스턴스가 업데이트된 후에는 다시 로드 밸런서에 포함됩니다. 이 프로세스는 모든 인스턴스가 업데이트될 때까지 계속됩니다.

Blue/Green 배포는 하나는 활성 상태, 다른 하나는 유휴(idle) 상태인 두 가지 동일한 환경을 실행하는 방식을 말합니다. 변경 사항은 유휴 환경으로 롤아웃되며, 일단 변경 사항이 프로덕션 환경에서 검증되면 라이브 트래픽이 업데이트된 환경으로 전환됩니다. 데이터 전환을 고려하면, 이전 버전으로 롤백하는 것은 트래픽을 다시 전환하는 것만큼 간단합니다.

카나리아(Canary) 배포는 두 가지 동일한 환경을 사용한다는 점에서 blue/green 배포와 유사합니다. 그러나 롤아웃이 제어되는 방식에서 차이가 있습니다. 새로운 릴리스를 배포한 후 일부 고객들은 새로운 릴리스를 프로덕션 환경에서 테스트하게 됩니다. 새로운 릴리스 검증이 완료되면 트래픽이 점차 새로운 버전으로 이동하며 모든 사용자가 새로운 릴리스를 적용할 때까지 결과를 모니터링하고 검증합니다.

코끼리에게 춤추는 방법 가르치기
E-Book 다운로드

8단계: 한층 발전된 모듈식 아키텍처 구현

소프트웨어 작성에 대한 마이크로서비스 기반 아키텍처 접근 방식에서는 애플리케이션이 상호 독립적인 최소 구성 요소로 세분화됩니다. 모든 요소를 하나의 애플리케이션에 구축하는 전통적인 모놀리식 접근 방식 대신, 마이크로서비스에서는 모든 요소가 분리되고 연동되어 동일한 작업을 완수합니다. 소프트웨어 개발에 대한 이러한 접근 방식은 세분화, 경량화 및 다수의 애플리케이션 간 유사한 프로세스를 공유하는 기능을 중시합니다. 마이크로서비스 아키텍처에 특정 인프라를 기본으로 도입해야 하는 것은 아니지만, 컨테이너 기반 플랫폼은 최적화된 기반이라 할 수 있습니다.

마이크로서비스 기반 아키텍처의 진화를 통해 규모가 매우 큰 팀들은 추가로 이점을 누리거나 하루에 여러 번 프로덕션 배포를 제공할 수 있습니다. 아키텍처 관점에서 볼 때, 마이크로서비스에서는 각 서비스가 자체 배포 단위로 세분화되어야 합니다. 그러면 각 마이크로서비스는 독립적으로 관리 및 배포되며, 각각의 라이프사이클을 담당하는 서로 다른 팀이 이 작업을 수행할 수 있습니다.

그러나 마이크로서비스 아키텍처를 구현하려면 투자와 기술력이 필요하며 조직에 상당한 혼란을 야기할 수도 있습니다. 애널리스트와 해당 분야 전문가(subject-matter experts, SME)들은 마이크로서비스에 MonolithFirst 접근 방식을 권고합니다. 이는 마이크로서비스 아키텍처 생성을 고려 중이라 하더라도 모놀리스를 우선으로 애플리케이션을 구축하는 것을 의미합니다. 이런 방식을 선택하는 것은 우선 애플리케이션 도메인을 먼저 파악하고 난 후, 도메인 내에서 마이크로서비스로 전환될 수 있는 제한된 컨텍스트(BC)를 정확히 인식하기 위해서입니다. 그렇게 함으로써, 애플리케이션의 도메인 및 제한된 컨텍스트를 파악하기 이전에 마이크로서비스를 구축할 때 이러한 접근 방식을 통해 복구 비용과 같은 기술적 채무를 피하게 됩니다.

마이크로서비스의 또 다른 대안은 미니서비스입니다. 미니서비스는 도메인별로 분할되어 일반적으로 애플리케이션 서버에서 실행되는 서비스 컬렉션입니다. 미니서비스는 마이크로서비스 기반 설계 및 인프라의 복잡성 없이 민첩성(agility)과 규모(scale)를 향상시킵니다. 컨테이너 기반 인프라와 결합된 현대적인 애플리케이션 서버 또는 멀티 프레임워크, 멀티 아키텍처 및 다중 언어를 이상적인 솔루션으로 제시하는 미니서비스는 애자일, DevOps 및 CI/CD 접근 방식에 대한 투자를 요구합니다.

클라우드 네이티브 애플리케이션 개발에 대한 다양한 프레임워크와 언어 및 접근 방식을 지원하는 플랫폼(예: 마이크로서비스, 미니서비스 또는 MonolithFirst)은 클라우드 네이티브 애플리케이션 성공의 핵심 요소입니다.

6. 클라우드 네이티브 애플리케이션에 대한 비즈니스 사례

비즈니스마다 디지털 트랜스포메이션을 위한 우선순위는 다릅니다. 기존 애플리케이션 아키텍처와 인프라를 현대화하면서 현대적인 클라우드 네이티브 원칙으로 진화하는 기업도 있고, 새로운 비즈니스 모델과 애플리케이션으로 혁신을 꾀하는 기업도 있습니다. 비즈니스 계획과 활용 사례에 차이는 있겠지만, 이들은 모두 속도와 유연성, 디지털 준비 상태(digital readiness) 향상이라는 공통 목표를 갖고 있습니다. 클라우드 네이티브 애플리케이션의 일반적인 활용 사례는 대체로 다음 4가지 비즈니스 과제에 따라 분류될 수 있습니다.

"신규 은행을 인수한 첫 날부터, 변경해야 하는 10가지 항목을 결함 없이 프로덕션 환경에 적용했습니다."

JOHN RZESZOTARSKI
KEYBANK, DEVOPS 디렉터

배포 시간: **12주에서 1주로 단축**

2023년까지 현재 사용 중인 애플리케이션의 90%가 계속 사용될 전망이지만, 현대화를 위한 투자는 대부분의 경우 불충분할 것으로 예상됩니다.

Gartner: Application Modernization should be business-centric, continuous, and multiplatform (애플리케이션 현대화는 비즈니스 중심의 지속적인 멀티플랫폼을 통해 실현), 2018년 1월.

"Red Hat JBoss Enterprise Application Platform에 전문가의 지원이 제공되므로 일상 업무에 대해 걱정할 필요가 없습니다."

(전) 총괄 관리자
기업 기술 부문,
ASX

60배 빠른 애플리케이션 재시작 속도

플랫폼 지원 비용 및 시간 **절약**, 혁신적인 서비스 개발을 위한 리소스 확보

비즈니스 과제 1: 애플리케이션 제공 가속화

목표:
고객의 기존 애플리케이션과 신규 애플리케이션 제공 속도를 향상합니다.

접근 방식:
컨테이너는 공통 플랫폼을 제공하여 개발, 운영, 보안, QA 및 기타 팀들이 협업을 통해 인프라 및 애플리케이션 기술과 별도로 DevOps를 도입하도록 합니다. DevOps 접근 방식으로, 각 팀은 자동화와 CI/CD 적용 사례를 활용해 빠르고 안정적으로 소프트웨어를 릴리스합니다. 컨테이너 기반 자동화를 통해 배포 문제를 해결하여, IT 팀의 제공 속도에 비즈니스를 맞추는 것이 아니라 비즈니스 속도에 맞게 애플리케이션 제공 주기를 가속화할 수 있습니다.

KeyBank

고객 소개:
미국 내 15대 은행 중 하나인 KeyBank는 웹 경험을 업데이트하고 새로운 모바일 웹 애플리케이션을 생성하기 위해 디지털 채널 현대화 계획을 시작했습니다. KeyBank는 모놀리식 애플리케이션에서 마이크로서비스로의 마이그레이션을 지원하는 Red Hat OpenShift®를 통해, 자동화된 CD 파이프라인을 구축하고 매분기 배포에서 매주 배포로 전환할 수 있었습니다.

비즈니스 과제 2: 기존 애플리케이션 현대화

목표:
기존 애플리케이션 현대화를 통해 시장 및 고객에 맞게 변화의 속도를 높입니다.

접근 방식:
유용한 비즈니스 애플리케이션은 디지털 시대에 맞게 설계되지 않은 레거시 애플리케이션인 경우가 많습니다. 하지만 전면적인 교체 방식이 늘 가능한 것은 아니며 경제성도 떨어질 뿐만 아니라 모든 레거시 애플리케이션이 현대화에 적합한 것도 아닙니다.

전통적인 애플리케이션을 클라우드로 마이그레이션하는 것이 실행 가능하다고 간주되는 경우, 컨테이너가 이러한 접근 방식을 지원하며, 그 경우 기본 인프라에 대한 종속성이 제거될 수 있습니다. 그 결과 애플리케이션을 온프레미스 인프라에서 클라우드로 이식 가능하며, 필요한 경우 클라우드 네이티브로 리팩토링하고 재설계할 수 있습니다. 컨테이너 플랫폼 접근 방식은 또한 플랫폼의 자동화 기능 및 DevOps 방식을 활용해 기존 애플리케이션의 마이그레이션을 간소화할 수 있습니다.

ASX

고객 소개:
ASX(Australian Securities Exchange)는 매일 전 세계에서 가장 먼저 열리는 금융 시장으로, 글로벌 금융 서비스 부문에서 주요한 역할을 합니다. ASX는 높은 안정성과 보안 및 고성능 환경에서 운영되어야 하지만, 레거시 애플리케이션 서버 플랫폼은 점점 불일치 및 불안정성이 높아지면서 비용도 증가하고 있었습니다. ASX는 새로운 기술로 디지털 플랫폼 현대화 계획을 세우고 Red Hat JBoss® Enterprise Application Platform을 구축해 애플리케이션 서버에 강력한 기반을 마련하기로 결정했습니다. 초기 배포에는 조직의 주요 B2B(Business-to-Business) 웹 애플리케이션 중 하나인 ASX Online이 포함되어 있었습니다. 이는 규정 요건을 준수하면서 가격과 회사 공지 및 중요한 시장 보고 등을 제공하는 웹 애플리케이션입니다.

"Red Hat OpenShift Container Platform은 매우 이상적이었습니다. 혁신적이며 컨테이너를 빠르게 배포하고 손쉽게 제어할 수 있습니다."

MICHAEL AALBERS
선임 기술 애플리케이션 코디네이터, 스키폴 국제공항

50%의 속도가 가속화된 새로운 API 개발

비즈니스 과제 3: 새로운 클라우드 네이티브 애플리케이션 개발

목표:
신규 애플리케이션 개발 속도를 높여 새로운 비즈니스 기회를 생성합니다.

접근 방식:
비즈니스 및 고객 요구 변화는 조직에 기회를 제시하여 아이디어를 서비스 및 제품으로 신속히 전환하고 새로운 환경에서 결과를 평가하며 이를 개선할 수 있도록 합니다. 새로운 애플리케이션 구축에 대한 클라우드 네이티브 접근 방식은 서비스 기반 아키텍처, API 통합, 컨테이너화된 서비스 및 오케스트레이션, DevOps 사례, 자동화 및 툴 지원을 통해 아이디어를 혁신적인 애플리케이션으로 전환하는 과정을 가속화하는 단계입니다.

Schiphol Amsterdam Airport

고객 소개:
스키폴 국제공항은 유럽에서 3번째로 이용 여객 수가 많은 공항으로, 연간 6,400만 명이 이 공항을 이용합니다. 스키폴 국제공항은 2018년까지 전 세계 최고의 디지털 공항이 되는 것을 목표로 하고 있습니다. 이러한 목표를 달성하기 위해 클라우드 유형에 구애받지 않는 플랫폼을 통해 애플리케이션 개발을 가속화해야 했습니다. 스키폴 국제공항의 디지털 전략 핵심 부분 중 하나는 승객에게 게이트, 터미널 및 체크인 시간 등의 정보를 알려주는 Flight API를 비롯해 API를 통해 제공되는 서비스입니다. 스키폴 국제공항은 Red Hat OpenShift Container Platform을 활용하여 새로운 서비스를 위한 개발 시간을 단축하며 내부 IT 팀 및 비즈니스 파트너를 위해 셀프 서비스, 멀티클라우드 플랫폼을 구축하고 있습니다.

"가장 큰 장점은 IT 운영 방식을 재설계한다는 점입니다. 우리 회사는 비즈니스 운영 방식을 완전히 변화시키고 은행 전체의 업무 방식 변경 프로세스를 곧 시작할 예정입니다."

WAYNE MARCHANT
CIO(최고 정보 책임자), HERITAGE BANK

비즈니스 과제 4: 비즈니스 혁신 추진

목표:
조직 전체에서 비즈니스 요구에 따라 유연하게 **혁신의 속도**를 높입니다.

접근 방식:
빠르게 변화하는 세계에서 현상 유지는 도태를 의미합니다. IT 팀은 앞다퉈 고객 만족과 직원의 효율적인 업무를 위해 새로운 기능 및 서비스를 신속히 도입하고 있습니다. 성공의 비결은 끊임없는 혁신에 있으며, 새로운 툴과 기술 이상의 것이 필요합니다. 획기적인 관점에서 조직 전체에 혁신과 실험을 지원할 수 있는 새로운 문화, 툴 및 프로세스가 필요합니다.

Heritage Bank

고객 소개:
Heritage Bank는 호주에서 가장 오래된 금융 기관 중 하나로 142년의 역사를 자랑합니다. 시장에서 경쟁이 격화되고 새로운 시장 수요가 나타나면서, Heritage Bank는 소프트웨어를 빠르게 제공하기 위한 새로운 방법을 모색해야 했습니다. Heritage Bank 팀은 Red Hat Open Innovation Labs에 적극 참여해 혁신적인 뱅킹 솔루션을 개발했습니다. 이 팀은 높은 성과를 내면서, 향후 더 나은 소프트웨어를 보다 신속히 개발할 수 있는 역량을 갖추게 되었습니다.

Heritage Bank 동영상

RED HAT의 지원 방식

디지털 및 클라우드 네이티브로의 전환 과정에서 어느 단계에 있는지와 우선 순위가 무엇인지에 따라 Red Hat은 그에 맞는 기술과 서비스를 지원합니다.

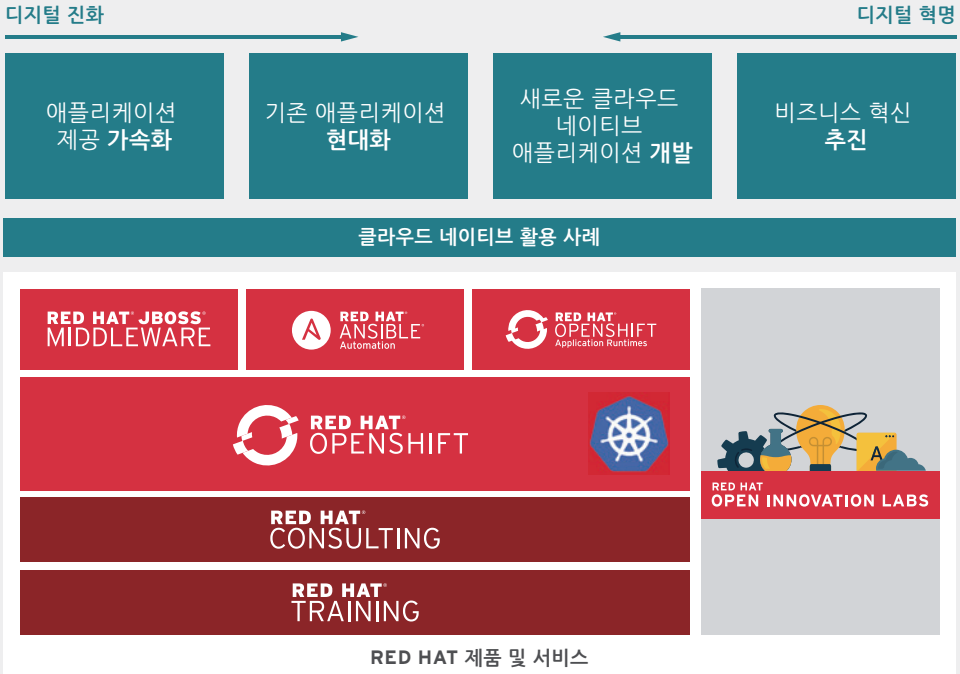


그림 2. Red Hat이 지원하는 클라우드 네이티브 전환 과정

클라우드 네이티브의 한 가지 활용 사례에만 주목하는 조직도 있고, 동시에 여러 가지 활용 사례에 우선 순위를 부여하는 조직도 있습니다. 발전적이거나 혁신적인 접근 방식을 택하는 경우, 그 과정은 매우 독자적이며 항상 연속적인 것은 아닙니다. 그 과정이 어떠하든, 애플리케이션 출시 속도를 앞당기려면 적합한 기술과 DevOps 사례 및 문화가 필요합니다.

Red Hat은 클라우드 네이티브 컨테이너 개발 플랫폼인 Red Hat OpenShift를 통해 이러한 과정을 지원합니다. Red Hat OpenShift Application Runtimes는 OpenShift에서 컨테이너화된 런타임을 통해 개발 시간을 단축하면서, 클라우드 네이티브 애플리케이션을 구축하기 위한 오픈소스 런타임 및 프레임워크를 제공합니다. Ansible 자동화 및 관리 기술을 포함한 다양한 Red Hat Middleware 기술을 OpenShift에 배포할 수 있습니다.

디지털 트랜스포메이션으로 인한 복잡성을 해결하도록 지원하면서, Red Hat Consulting은 전략적 조언과 심층적인 기술 전문성을 제공합니다. Red Hat Open Innovation Labs에서 디스커버리 세션 및 프로젝트 구현 계획에 이르기까지, Red Hat의 컨설턴트가 클라우드 네이티브 전환 과정의 모든 단계를 지원합니다.

E-BOOK 클라우드 네이티브 애플리케이션 구현을 위한 과정

클라우드 네이티브로 전환하시나요?

Red Hat이 클라우드 네이티브 애플리케이션 전환 과정을 지원하는 방식에 대해 자세히 알아보세요.

- Red Hat Consulting의 지원 방식을 보려면 컨설팅 디스커버리 세션을 통해 모범 사례와 플래닝 가이드를 확인하세요.
- Red Hat의 Services Speak 블로그에서 다양한 인사이트와 팁을 확인하세요.
- 귀사의 DevOps 성숙도와 클라우드 네이티브 전환을 위한 준비 상태를 알아 보려면, Ready To Innovate 평가를 수행해 보세요.

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
#F12255_0518