

# Cloud Transformation

# 클라우드?

구글이나 네이버에서 제공하는 서비스가 클라우드 아니야?

가상화 솔루션 도입하면 클라우드 아닌가요?

회사 내에서만 사용하는 시스템인데, 클라우드와 무슨 상관이 있어요?

클라우드가 실제로는 비용이 더 비싸요?

하이브리드 클라우드는 무슨 뜻이지?

클라우드 의미 그대로 뜯 구름 잡기처럼 사람들마다 얘기 하는게 달라요.



유튜브를 보고, 검색을 해봐도 대략은 알겠는데 정확히 뭔지 모르겠음!!



Cloud Transformation

IT Trends

# 비즈니스 측면에서 IT 과제들

다른 기업들은 **비용절감**을 위하여 **클라우드와 오픈 소스**를 적극 검토 중이라고 하던데..  
우리 IT 부서는 어떤 것을 준비하고 계신가요?



- 가상화/  
클라우드/  
오픈 소스

새로 오신 전무님은 **맥북**을 쓰시던데,  
우리 회사 IT는 **인터넷 익스플로러**만 되나요?  
**모바일**에서 결재할 수는 없나요?



- 모바일  
크로스 브라우저  
멀티 OS

우리 회사 쇼핑몰이 1등인데, **웹사이트 속도**는 왜 3 등이지?  
시스템 성능이 곧 **비즈니스 성능**이라고 하던데 ...  
시스템 장애가 곧 **매출 장애**라 매우 민감한데.....



- 지속적인 서비스  
품질과  
성능 개선

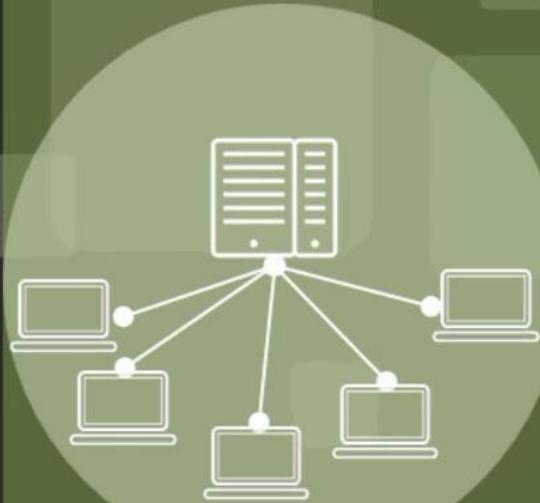
증권거래소와 증권사의 거래시스템이 모두 **리눅스**라고 하는데,  
아직도 안정성이라는 이유로 **고가의 유닉스**를 도입해야 하나요?  
X86 서버에 리눅스인데, **를** 구매해야 하나요?



- IT Commodity

1960

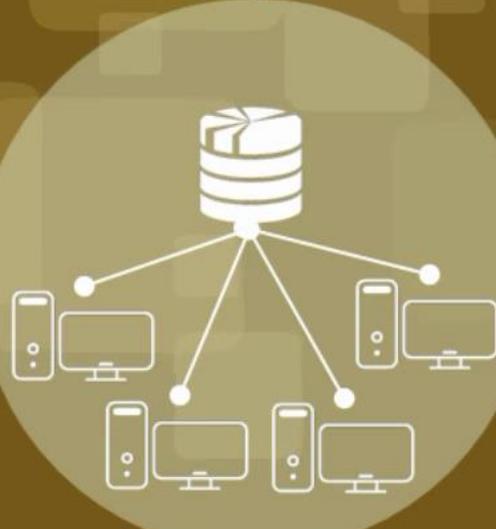
## Mainframe



Mainframe  
Vendor

1990

## Client/Server



2 Tier  
Unix (Vendor)  
Scale Up

2000

## Web



3 Tier  
Linux  
Scale out

2010

## Cloud



IOT/Big Data  
Mobile  
Elastic Scaling

## Development Process



WATERFALL



AGILE



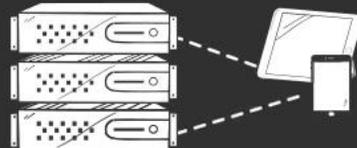
DEVOPS



## Application Architecture



MONOLITHIC



N-TIER



MICROSERVICES



## Deployment & Packaging



PHYSICAL SERVERS



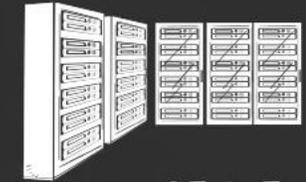
VIRTUAL SERVERS



CONTAINERS



## Application Infrastructure



DATA CENTER



HOSTED



CLOUD





Cloud Transformation

Cloud & Utility Computing

## Infrastructure as Code & DevOps Seminar

### Burden's Wheel

1851년 헨리 버든은  
미국에서 가장 크고  
강력한 공업용 수차를  
설치하여 공장을  
자가 발전했습니다.



엑스 4층  
룸 402호  
수) 13:00~18:00



“영화친구 김시선”

## Burden Wheel



1851년 헨리 버든은 미국에서 가장 크고 강력한 **공업용 수차**를 설치하여 **공장을 자가 발전**

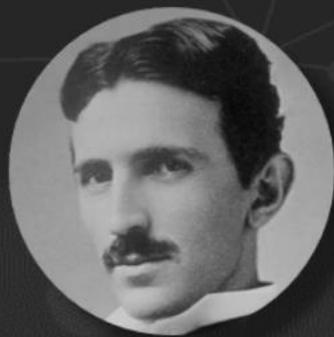
## Technology Inventor



토마스 에디슨은 수많은 전기 관련 기술을 발명하였지만 **직류** 기술을 옹호하고 기업에 **자체 발전소**를 건립하고 운영하는 데 필요한 많은 부품 판매에 열중

Thomas Edison

## Technology Disruptor



에디슨이 고용한 테슬라는 에디슨의 직류 전기의 한 계점(1마일 이상 송전이 안됨)을 지적했고 **교류를 발명**하여 오늘날의 **전기전송시스템** 발명

Nikola Tesla



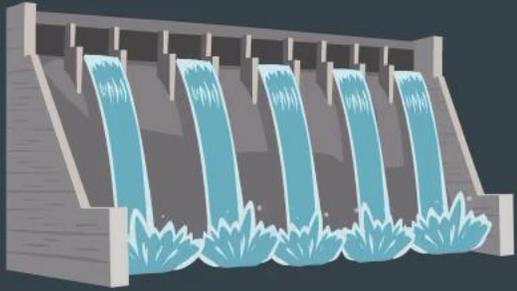
TESLA MOTORS

## Business Inventor



에디슨이 고용한 인설은 유틸리티의 운영이 결국엔 유틸리티 부품의 제조보다 더욱 더 중요한 비즈니스라고 판단. 오늘날의 **전기공급사업모델**을 개발

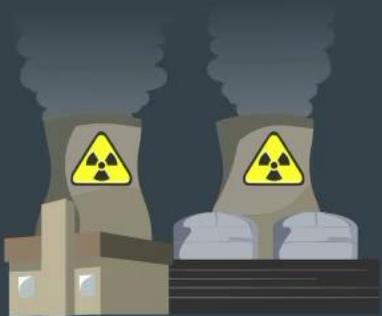
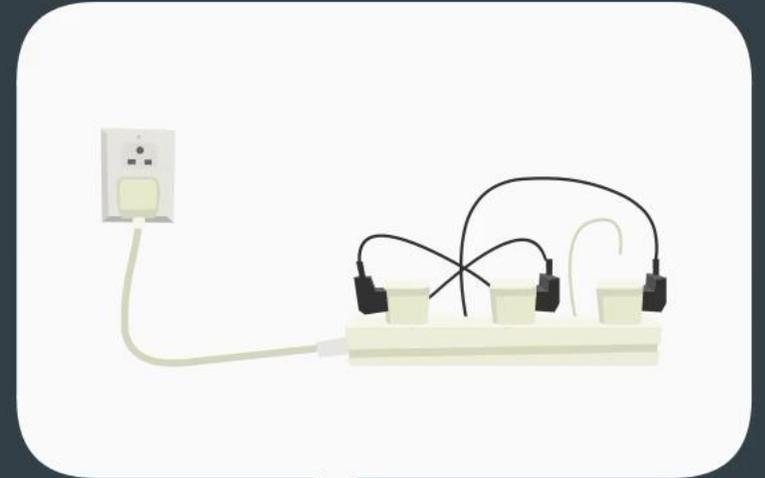
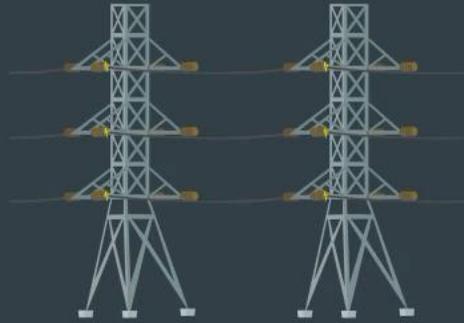
Samuel Insull



수력



풍력

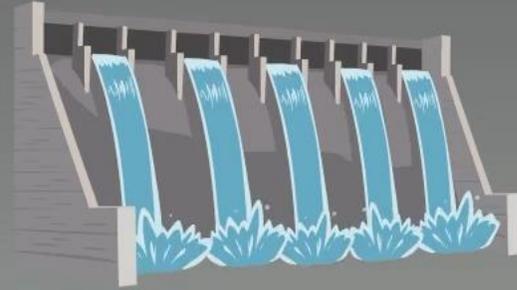


원자력



화력

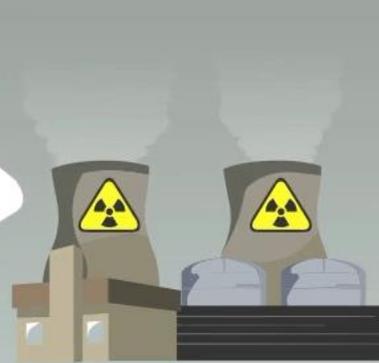




수력



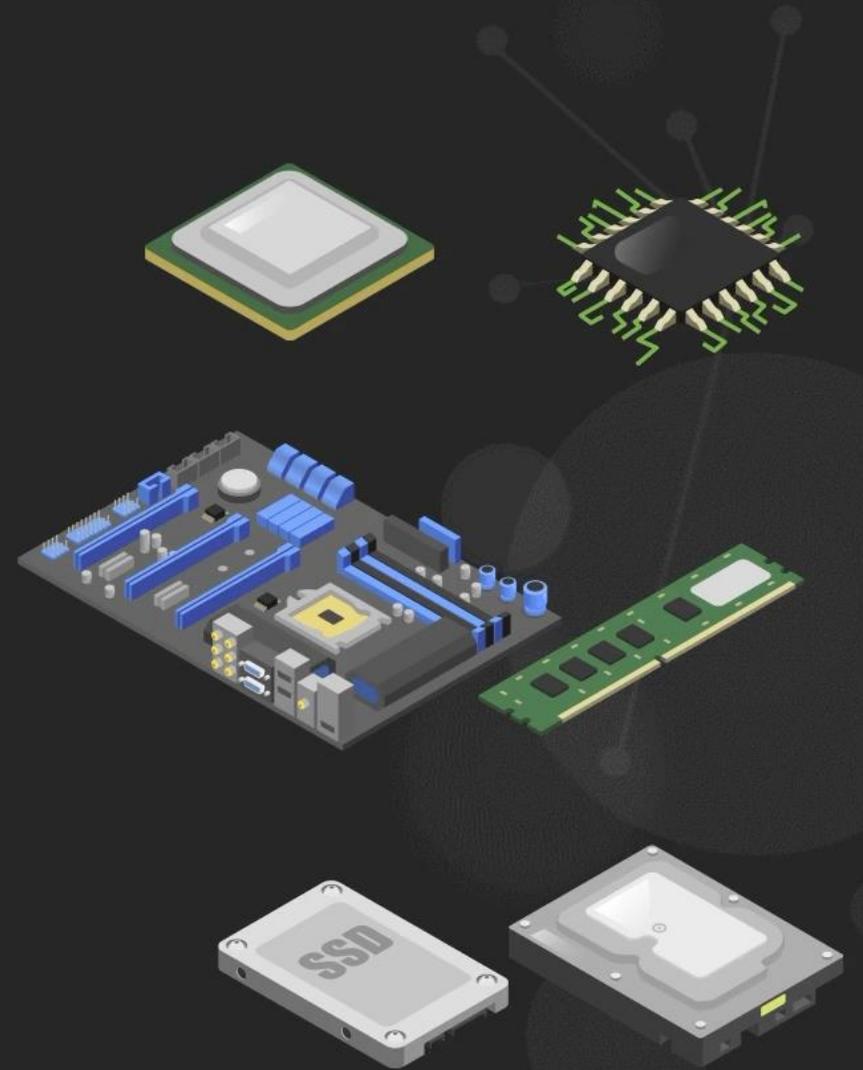
풍력



원자력



화력



클라우드 네이티브 개념과 기술요소들

클라우드 네이티브를 위한 준비

# 클라우드 핵심 개념 : Architecture & Model



클라우드 네이티브 (Container)

vs.



클라우드 이민 ( Virtualization )



SaaS : Software As A Service



PaaS : Platform As A Service



IaaS : Infrastructure As A Service



Hybrid Cloud



Private Cloud



Public Cloud

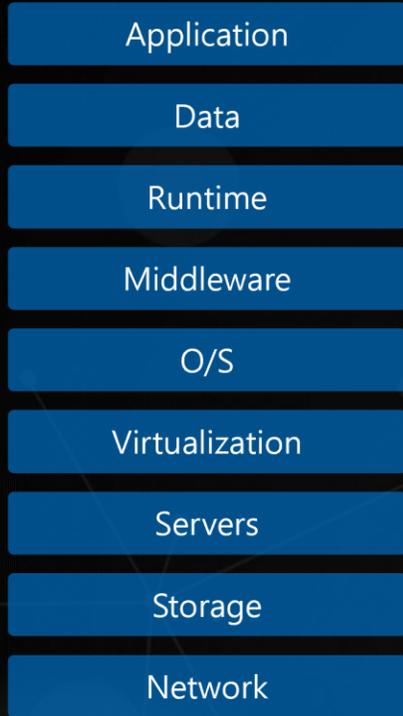
# Cloud Delivery Model

구분	클라우드 네이티브	구조	특징	벤더
	<b>Public Cloud</b>	클라우드 서비스 업체가 인터넷을 통해 컴퓨팅 리소스를 제공하고, 서버의 유지 관리	<ul style="list-style-type: none"> <li>• 여러 기업이 하나의 클라우드 인프라를 이용 (Multi-Tenant)</li> <li>• 더 적은 구축 비용</li> <li>• 더 적은 유지 보수</li> </ul>	<ul style="list-style-type: none"> <li>• AWS</li> <li>• Azure</li> <li>• Google Cloud</li> <li>• Oracle</li> <li>• Alibaba</li> </ul>
	<b>Private Cloud</b>	기업이 클라우드 서버를 독점 아키텍처를 이용하여 자사의 데이터 센터 운영	<ul style="list-style-type: none"> <li>• 하나의 기업에 하나의 인프라스트럭처 (싱글 테넌트)</li> <li>• On Premise 하드웨어</li> <li>• 고객이 인프라 관리</li> </ul>	<ul style="list-style-type: none"> <li>• HPE</li> <li>• VMWare</li> <li>• Dell EMC</li> <li>• IBM</li> <li>• Red Hat</li> </ul>
	<b>Hybrid Cloud</b>	On Premise 인프라, 프라이빗 클라우드 과 퍼블릭 클라우드의 혼합 컴퓨팅 환경	<ul style="list-style-type: none"> <li>• 데이터 처리를 더 • 개인 및 제어 가능</li> <li>• 베스트 기능</li> <li>• 기존 시스템을 함유 할 수있다</li> </ul>	<ul style="list-style-type: none"> <li>• Red Hat</li> <li>• AWS</li> <li>• Azure</li> <li>• Google Cloud</li> </ul>

# 클라우드 서비스 모델



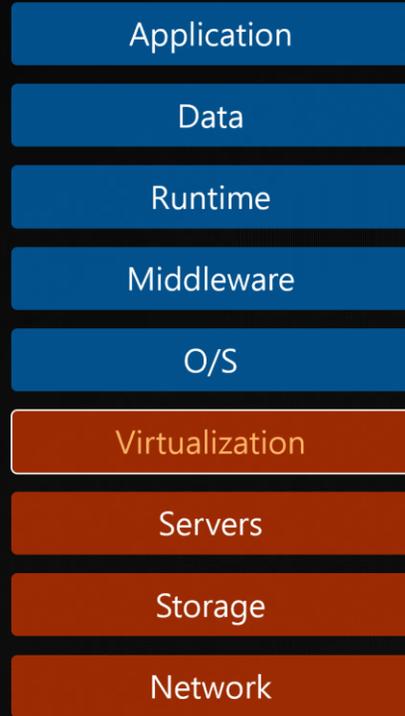
## On-Premise



- 고객은 인프라 제공, 유지 및 애플리케이션 호스팅 모두 책임



## Infrastructure-as-a-Service (IaaS)



- 공급 업체는 인터넷을 통해 컴퓨팅 인프라를 제공
- 예 : AWS EC2, MSFT Azure



## Platform-as-a-Service (PaaS)



- 애플리케이션 개발을 위한 플랫폼을 제공
- 공급 업체는 서버, 스토리지, 네트워크를 관리
- 고객은 애플리케이션 관리
- 예 : OpenShift , Heroku



## Software-as-a-Service (SaaS)



- 인터넷을 통해 제공되는 소프트웨어
- 공급 업체가 소프트웨어 구축, 유지, 운영
- 예 : G-suite, Microsoft 365

범례 :

기업 고객 관리

클라우드 공급자 관리

# 물리서버 시대 ~ 2000년

- 모놀리스 애플리케이션 운영
- 물리서버 대수가 많지 않고, 서버를 1:1로 관리



~ 2000년  
물리서버

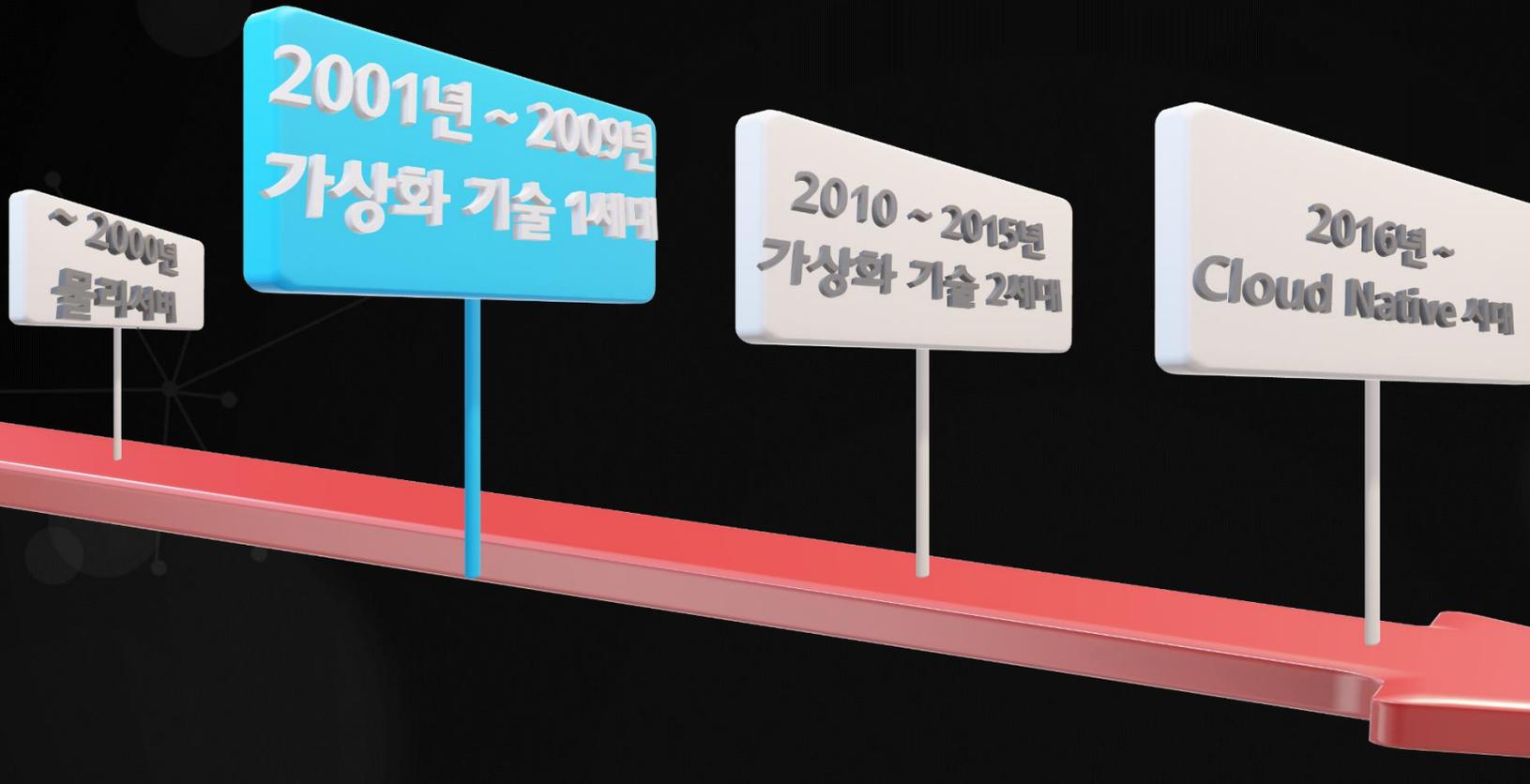
2001년 ~ 2009년  
가상화 기술 1세대

2010 ~ 2015년  
가상화 기술 2세대

2016년 ~  
Cloud Native 4대

# 가상화 기술 1세대 : 2001년 ~ 2009년

- 실제 시스템을 가상머신으로 대체, 가상머신을 애완 동물로 취급
- 통합 비율을 올리고 고효율화 하는 것이 목적 - 서버의 멀티 코어 화와 가상화 기술의 보급
- 가상화 기술에 대한 대안으로 컨테이너 기술이



# 가상화 기술 2세대 ≙ Cloud 시대: 2010년 ~ 2015년

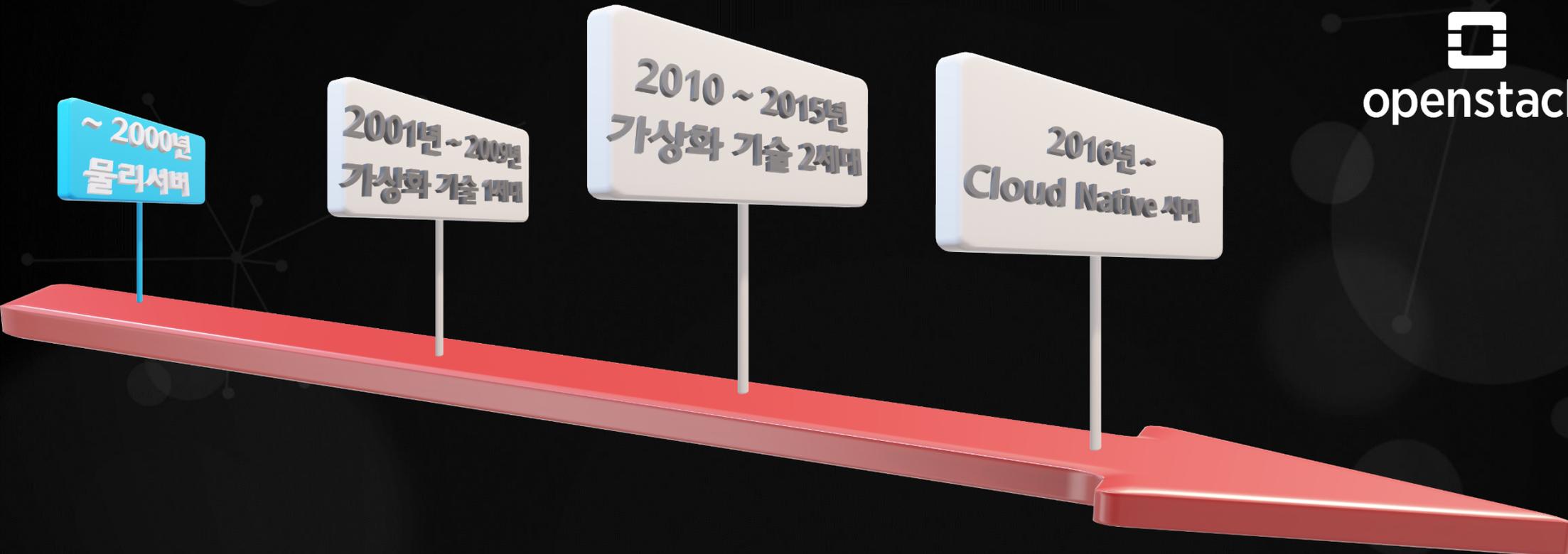
- 클라우드가 대규모 서비스에 적합하고 확장 가능성을 입증
- 안정적으로 대규모 인프라를 관리하기 위한 기술도 보급
  - Immutable Infrastructure/ Infrastructure as Code



IBM Cloud



openstack.



# Cloud Native 시대 2016 년 ~

- 2016 년 1 월에 정식 출범 한 Cloud Native Computing Foundation (이하 CNCF)는 진짜 클라우드 기술을 오픈소스로 해결하는 하는 것을 목표
- 애플리케이션을 실행하는 데 필요한 최적의 인프라 제공
  - 개발 한 것을 "즉시" "안정적으로" 제공 (비즈니스 우위)

~ 2000년  
물리서버

2001년~2009년  
가상화 기술 1세대

2010 ~ 2015년  
가상화 기술 2세대

2016년~  
Cloud Native 시대



# 클라우드 전략은 기술 그 이상입니다.

## 가치 전략

•비즈니스 사례를 개발하여 비즈니스에 더 큰 가치를 제공하기 위해 클라우드를 구현하고 실현하는 방법을 결정합니다.

## 애플리케이션 전략

•기존 애플리케이션 및 대상 플랫폼의 준비 상태를 검사합니다. 가치있는 전환을 달성하기위한 전략을 만듭니다.

## DevOps 전략

•클라우드 채택으로의 전환으로 인해 도구, 프로세스 및 개발 팀과 운영 팀 간의 상호 작용에 미치는 영향을 식별합니다.

## 운영 모델 전략

•클라우드 지원 제품군으로 전환 한 후 IT 조직이 원활하게 작동 할 수 있도록 향후 운영 모델을 정의합니다.

## 보안 전략

•클라우드에서 리소스를 안전하게 사용하고 거버넌스, 위험 및 규정 준수 요구 사항을 준수하도록 선도적 인 관행을 채택하십시오.

## 하이브리드 전략

•비즈니스 요구 사항에 따라 퍼블릭 및 프라이빗 클라우드를 지능적이고 원활하게 활용하고 레거시와 통합되도록 인프라를 설계합니다.

## 클라우드 서비스 전략

•서비스가 운영상 어떻게 사용되고 지원 파이프 라인에 통합되는지 정의합니다. 퍼블릭 클라우드 서비스로 애플리케이션 설계 원칙을 구축하십시오.

## 클라우드 네트워킹

•클라우드에 대한 기본 인터넷 지원, 인터넷 전용을 결합하여 기업 백본 및 제로 트러스트 경계없는 네트워크를 최소화합니다.

클라우드 네이티브 개념과 기술요소들

# IaaS with Virtualization vs. PaaS with Container

# 가상화 기반 IaaS vs. 컨테이너 기반 PaaS

## 기존 IaaS (가상화 기술 기반)

가상화 기반 전용 클라우드

상용 SW 기반

규모의 경제

기술 및 공급자

## 향후 PaaS (컨테이너 기술 기반)

하이브리드  
(전용+공용, 물리+가상)

오픈소스 SW 기반

범위의 경제

비즈니스 및 수요자

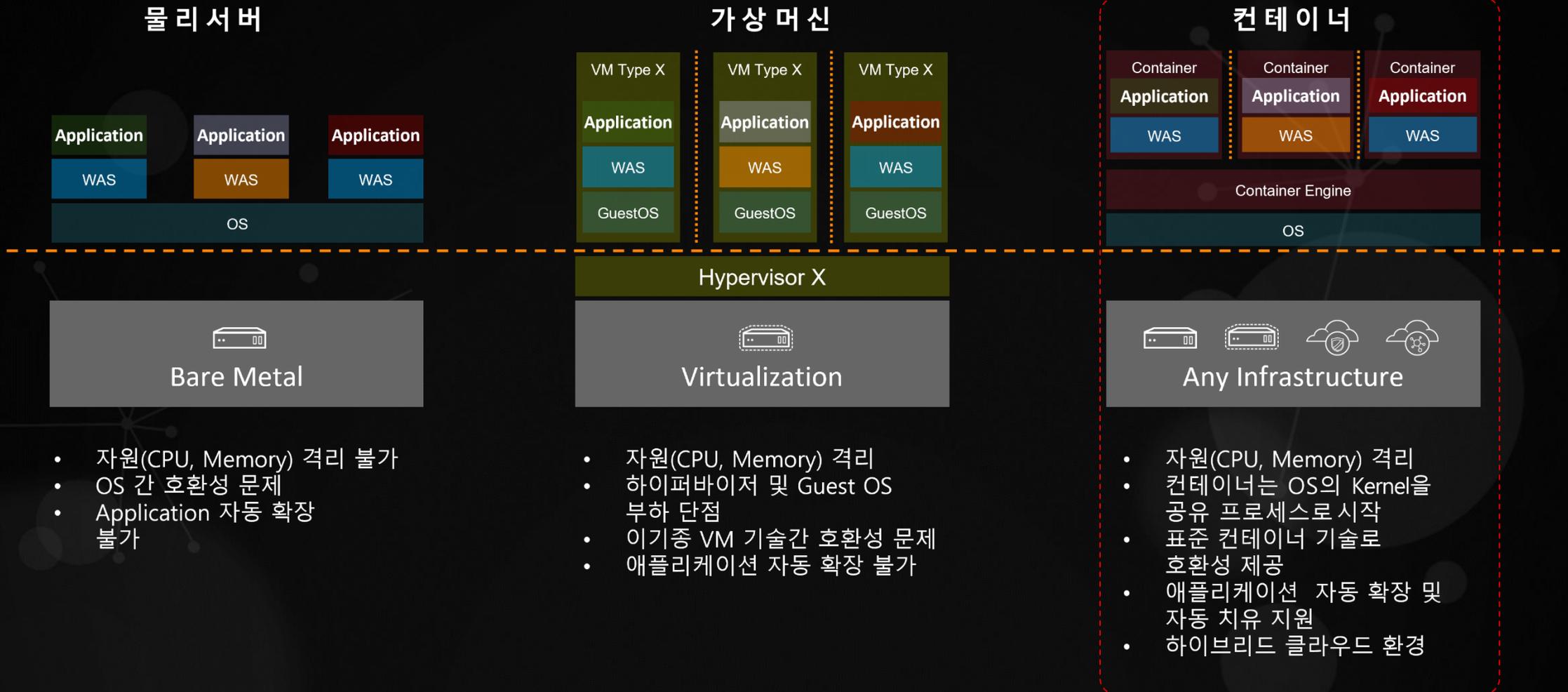
# Cloud Immigrant vs. Cloud Native



구분	Cloud Immigrant	Cloud Native
서비스 모델	가상화 기반 IaaS ( Infrastructure As A Service)	컨테이너 기반 PaaS ( Platform As A Service)
디자인	On Premise 에 구축된 시스템을 클라우드로 이전하여 운영	시작 단계부터 클라우드의 장점인 민첩성, 확장성 그리고 이동성을 최대한 활용할 수 있도록 설계
구현	특정 클라우드 벤더에 의존적인 설정이 있어 구축에 시간이 걸림	어떤 클라우드 환경에서도 빠르고 효율적으로 전환 ( Portability )
확장성	애플리케이션 업데이트가 수작업이기 때문에 장시간의 다운타임일 필요하고 Scale In/Out 이 어려움	컨테이너와 MSA 기반으로 서비스에 영향을 주지 않고, 업데이트가 필요한 서비스만 변경할 수 있으며, 서비스 단위의 Scale In/out 지원
비용	애플리케이션이 커질 수록 인프라 비용이 상승	인프라 부분의 종속성이 없어 비용이 저렴
유지보수	버전관리, 설치 그리고 구성관리가 수작업이며 복잡함	CI (Continuous Integration) / CD (Continuous Delivery )

# WHY CONTAINER ?

- 자원 효율성, 자원 격리, 호환성, Auto Scaling, DevOps, MSA, 관리 편의성



- 자원(CPU, Memory) 격리 불가
- OS 간 호환성 문제
- Application 자동 확장 불가

- 자원(CPU, Memory) 격리
- 하이퍼바이저 및 Guest OS 부하 단점
- 이기종 VM 기술간 호환성 문제
- 애플리케이션 자동 확장 불가

- 자원(CPU, Memory) 격리
- 컨테이너는 OS의 Kernel을 공유 프로세스로 시작
- 표준 컨테이너 기술로 호환성 제공
- 애플리케이션 자동 확장 및 자동 치유 지원
- 하이브리드 클라우드 환경

“

**2022년까지** 글로벌 조직의 **75%** 이상이 프로덕션 환경에서 **컨테이너화** 된 애플리케이션을 실행할 것으로 예상합니다.

— Gartner

“

**컨테이너** 채택은 프라이빗 과 퍼블릭 클라우드 인프라 간의 격차를 해소하여 **하이브리드 클라우드 아키텍처**를 **재정의** 할 것입니다.

— Forbes

“

**컨테이너**가 가상머신을 대체하기 시작했습니다. Kubernetes 와 마이크로 서비스의 사용이 가속화되면서 **클라우드 네이티브가 프로덕션에 적용**되고 있습니다.

— 451 Research

# 컨테이너는 클라우드에서 Java 와 같이 벤더 종속성 해제

## 2000 년 – Java 를 통한 Vendor Lock-In 해제



## 2020 년 – 컨테이너와 Kubernetes 를 통한 Vendor Lock-In 해제





**Cloud Native**

## Development Process



WATERFALL



AGILE



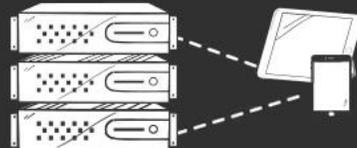
DEVOPS



## Application Architecture



MONOLITHIC



N-TIER



MICROSERVICES



## Deployment & Packaging



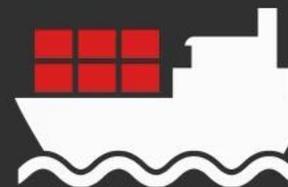
PHYSICAL SERVERS



VIRTUAL SERVERS



CONTAINERS



## Application Infrastructure



DATA CENTER

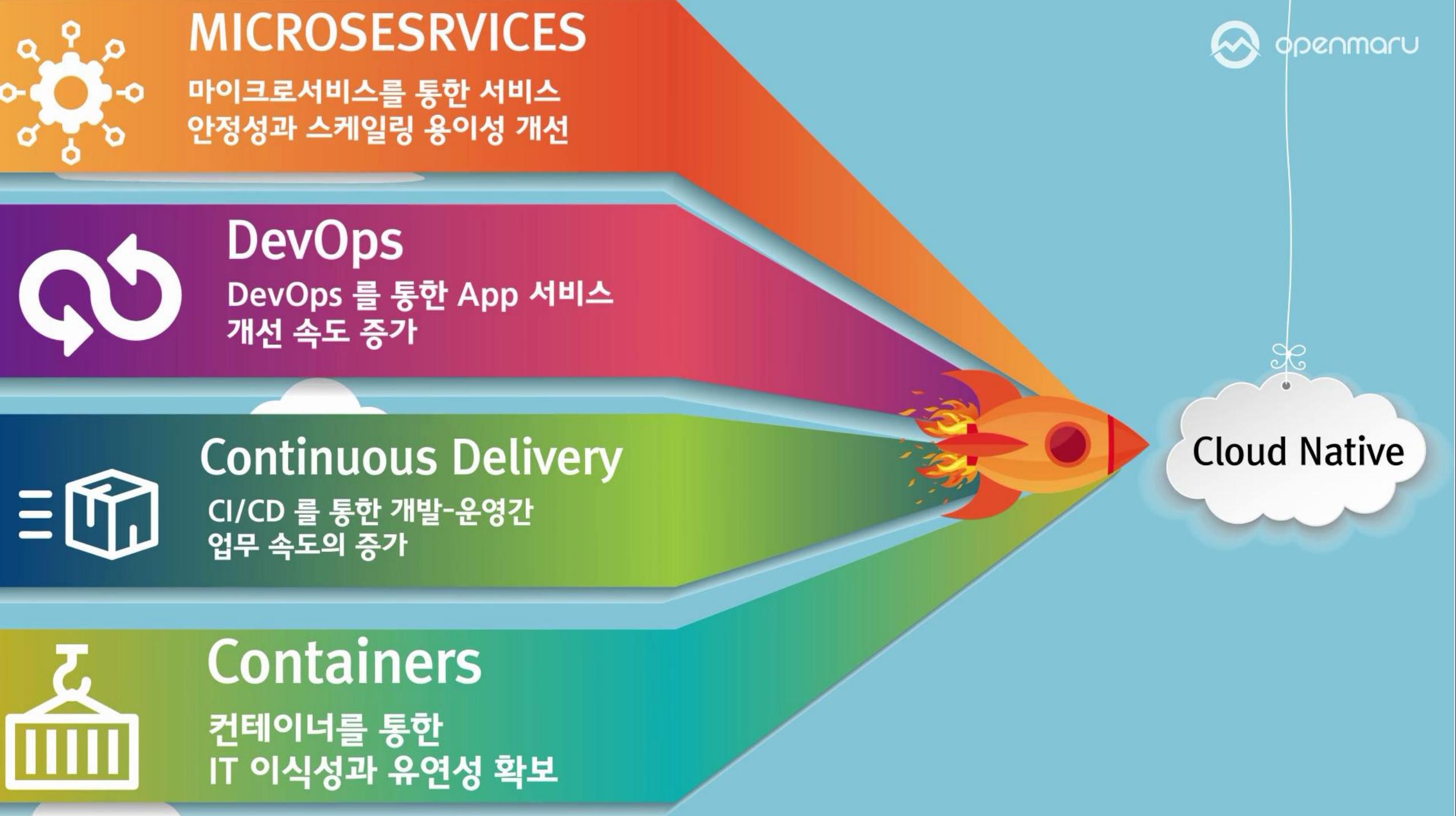


HOSTED



CLOUD





# MICROSERVICES

마이크로서비스를 통한 서비스  
안정성과 스케일링 용이성 개선



# DevOps

DevOps 를 통한 App 서비스  
개선 속도 증가



# Continuous Delivery

CI/CD 를 통한 개발-운영간  
업무 속도의 증가



# Containers

컨테이너를 통한  
IT 이식성과 유연성 확보



Cloud Native

# How does one build apps for the cloud?

- 가상화 기반 IaaS Cloud



# CNCF Cloud Native Definition v1.0

클라우드 네이티브 기술을 사용하는 조직은 현대적인 퍼블릭, 프라이빗, 그리고 하이브리드 클라우드와 같이 동적인 환경에서 확장성 있는 애플리케이션을 만들고 운영할 수 있다.

컨테이너, 서비스 메시, 마이크로서비스, 불변의 인프라스트럭처, 그리고 선언적 API가 전형적인 접근 방식에 해당한다.

이 기술은 회복성이 있고, 관리 편의성을 제공하며, 가시성을 갖는 느슨하게 결합된 시스템을 가능하게 한다.

견고한 자동화와 함께 사용하면, 엔지니어는 영향이 큰 변경을 최소한의 노력으로 자주, 예측 가능하게 수행할 수 있다.

Cloud Native Computing Foundation은 **벤더 중립적인 오픈소스 프로젝트 생태계**를 육성하고 유지함으로써 해당 패러다임 채택을 촉진한다.

우리 재단은 최신 기술 수준의 패턴을 대중화하여 이런 혁신을 누구나 접근 가능하도록 한다.



# Cloud Native Computing

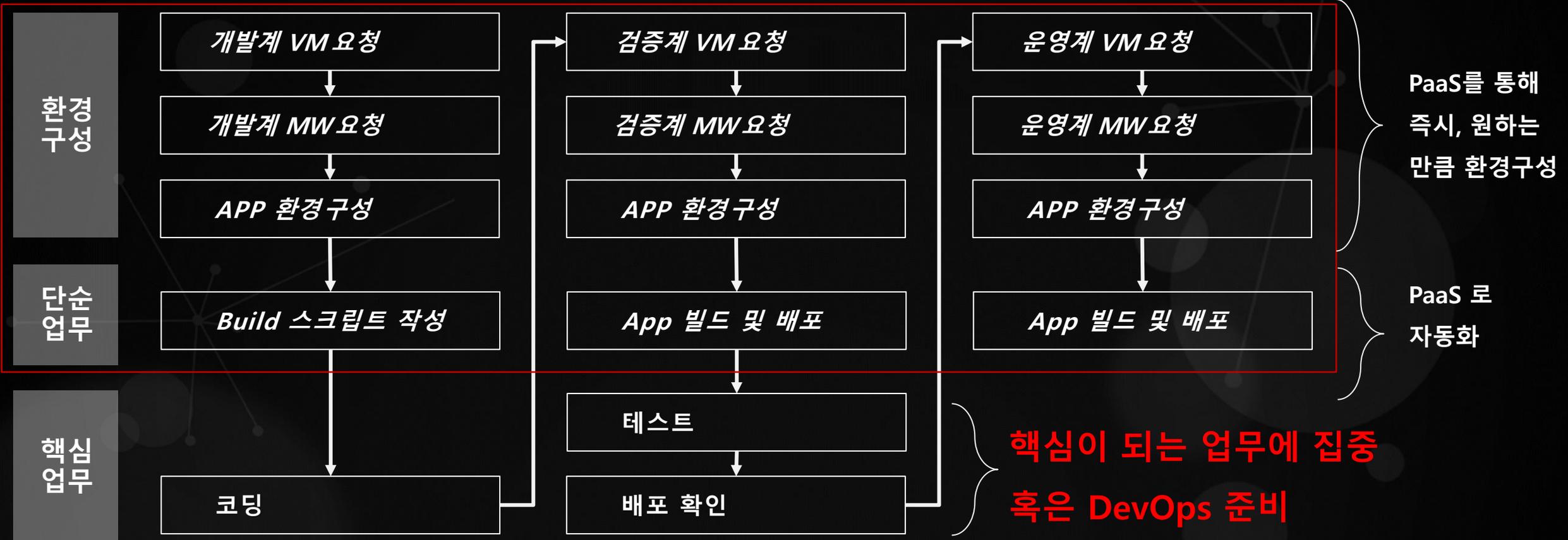
- Cloud Native Computing은 클라우드의 특성과 장점을 적용하여 구성된 컴퓨팅 환경으로, 인프라, 플랫폼, 어플리케이션/서비스와 개발, 운영, 관리의 전체 영역을 대상으로 함



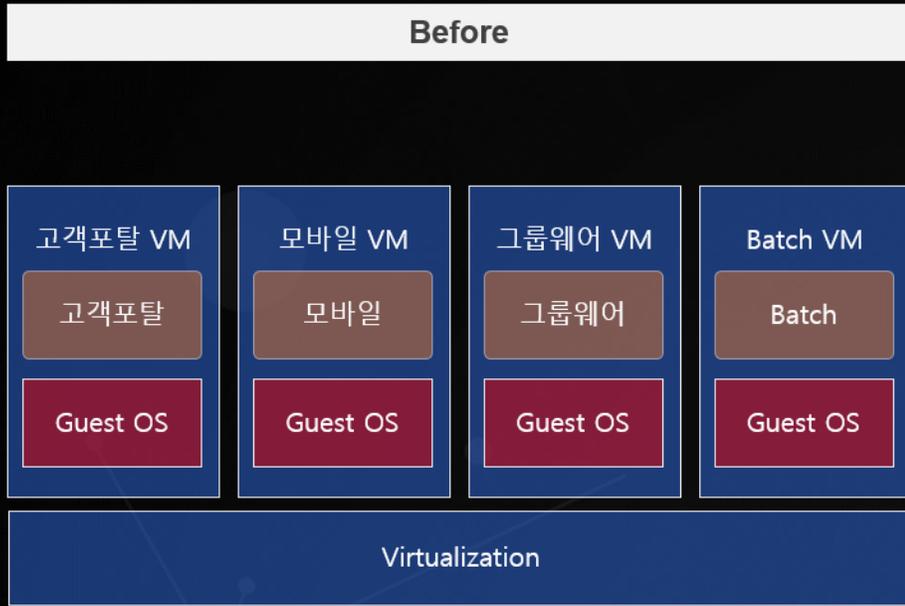
# IT개발 측면 > 개발프로세스 대폭 축소



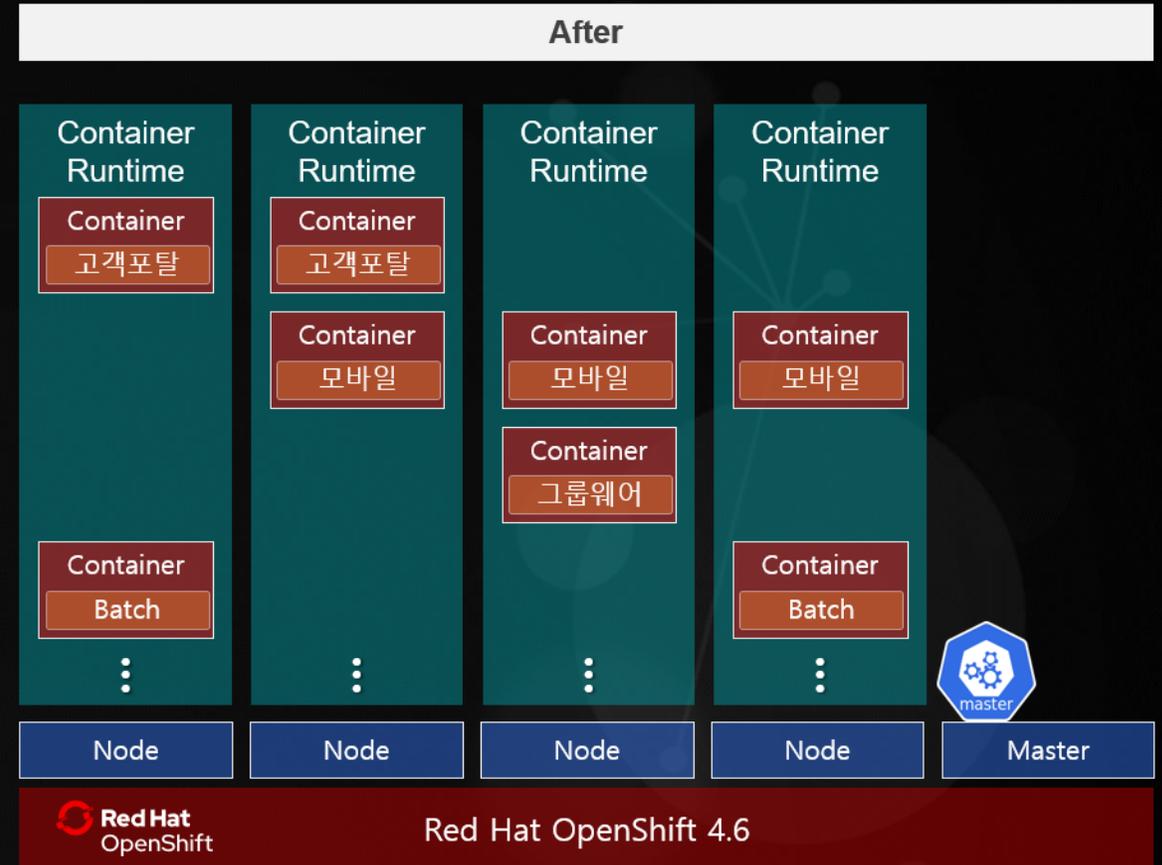
OpenShift 도입을 통해 사라지거나 극도로 축소될 업무



# Kubernetes 의한 컨테이너 오케스트레이션의 실현



- 용도 당 VM 불필요한 자원 소비
- 환경 복제가 어려움



- Kubernetes이 Node의 자원 상황을 보고 적절한 컨테이너를 배치
- 손쉬운 개발환경 구축 과 관리자의 개입을 최소화 하여 자동 확장/복구 실현

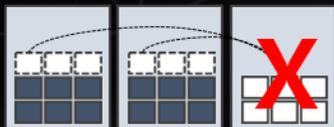
# Cloud Native Computing으로 전환 효과

- Cloud Native Computing 환경은 클라우드가 제공하는 민첩성, 가용성, 확장성의 장점을 어플리케이션/서비스의 개발, 운영, 관리에 적용하여 기존 컴퓨팅 환경을 최적화 함



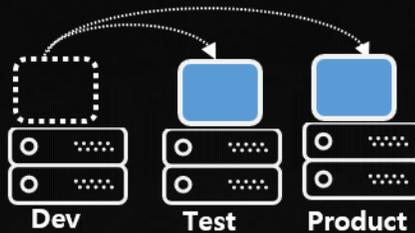
## On Demand Delivery

필요한 컴퓨팅 자원을 즉시 제공



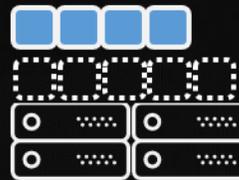
## Self Recovery

- 비정상 어플리케이션 재시작
- 노드의 장애 발생시 정상 서버 노드로 자동 재배포



## Consistency & Continuous

이미지 기반으로 구성, 배포 효율화  
개발과 운영 환경의 일관성



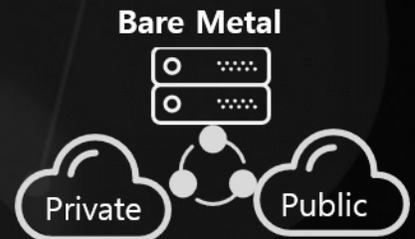
## Application Scaling

VM 단위가 아닌 어플리케이션 단위의 오토스케일링



## Rolling Update

업그레이드 또는 패치 시  
다운 타임은 제로 또는 최소화



## Portable

멀티/하이브리드 클라우드 기반  
어플리케이션/서비스 운영

# 클라우드 네이티브 를 도입해야 하는 이유



## 경쟁 우위 확보

Cloud Native 란 클라우드 목표를 IT 비용 절감에서 비즈니스 성장 엔진으로 바꾼다는 의미입니다. 소프트웨어 시대에는 고객의 요구에 부응해 신속하게 애플리케이션을 구현하고 전달할 수 있어야 비즈니스에 경쟁력을 확보할 수 있습니다. 애플리케이션은 규모의 탄력성을 보장할 수 있는 클라우드에 운영 가능해야 합니다.



## 유연성

기업은 개발한 애플리케이션을 어떠한 클라우드 환경에서도 수정없이 실행할 수 있습니다. 하이브리드 클라우드는 비즈니스 우선 순위를 맞추고 클라우드 가격을 최적화하기 위해 다양한 프라이빗과 퍼블릭 클라우드에 마이그레이션하거나 배포할 수 있는 능력을 유지할 수 있습니다.



## 개발 생산성 향상

많은 조직이 실용적인 서비스 기반 아키텍처를 이용하여 계속해서 레거시 애플리케이션을 최적화할 수 있습니다. 이러한 최적화는 지속적인 통합(Continuous Integration, CI)과 지속적인 제공(Continuous Delivery, CD) 및 완전히 자동화된 배포 운영 같은 DevOps 워크플로우를 통해 지원됩니다.



## 개발 기간 단축

디지털화가 진행되는 세계에서 기업이 더 많은 고객을 확보하고 유지하려면 신속한 대응이 필수적입니다. 클라우드 네이티브 애플리케이션을 개발 및 배포함으로써 신속한 업데이트 및 개선 할 수 있습니다. 결과적으로 제품과 서비스를 신속하게 제공 할 수 있게 되어 기업의 경쟁력을 높일 수 있습니다.



## 마이크로서비스 아키텍처

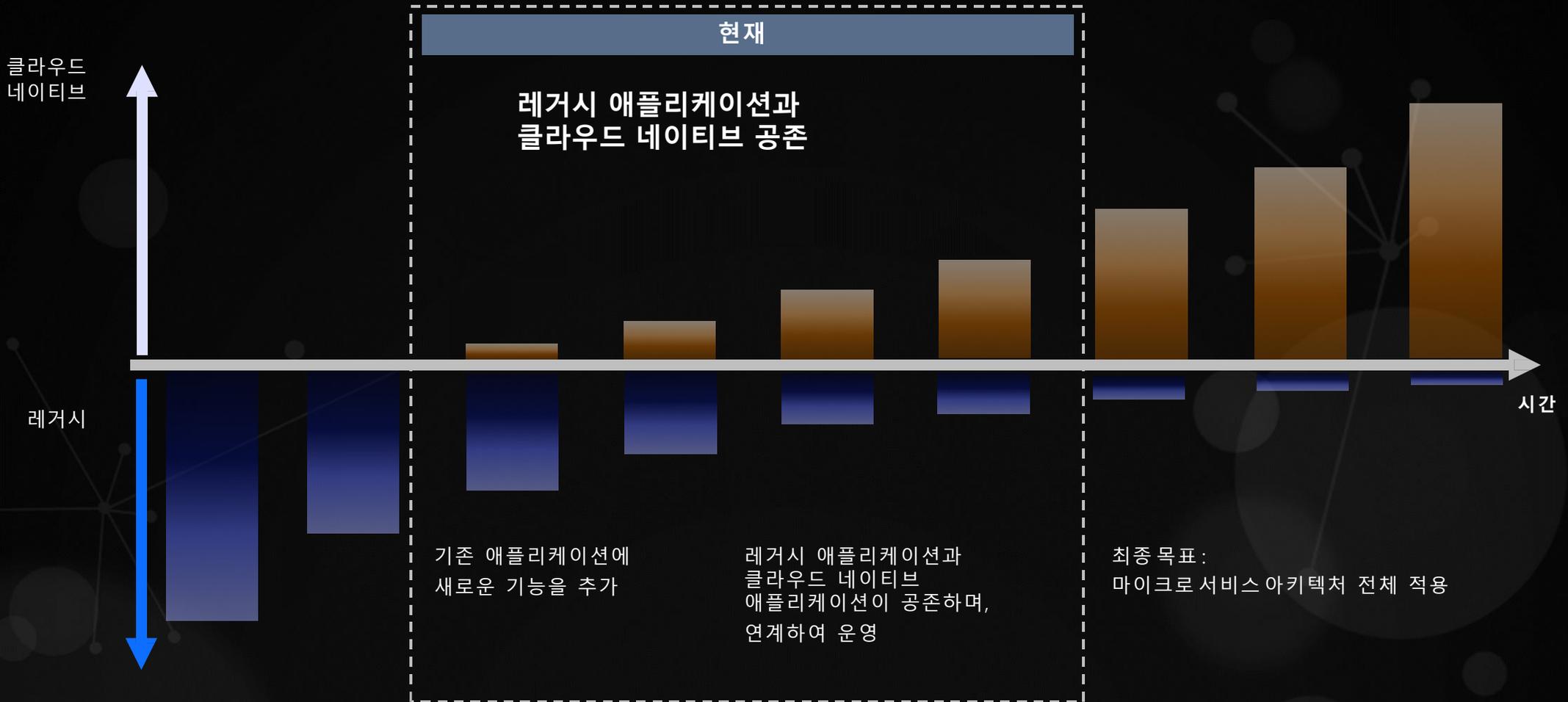
Cloud Native 기술은 마이크로 서비스를 사용하여 탄력성을 높이고 자동 스케일링을 제공합니다. 마이크로 서비스는 시스템이나 사용자에게 영향을 주지 않고 배포, 업데이트, 확장/축소, 재시작 할 수 있어 뛰어난 고객 경험을 24 시간 365 일 제공 할 수 있습니다.



## 운영 효율성 향상

사업이 성장하고 있더라도 변화하는 시장 환경에 신속하게 대응하는 것은 쉽지 않습니다. Cloud Native 애플리케이션으로 개발하면 자동화, 셀프 서비스, 원격 측정, 분석 등의 기능을 필요에 따라 확장 또는 확장 할 수 있기 때문에 비즈니스 운영 효율성이 높아집니다.

# 레거시와 클라우드 네이티브 애플리케이션 전환



Source : IBM

# Different Levels of Cloud Native Application Maturity.



[https://www.opendatacenteralliance.org/docs/architecting\\_cloud\\_aware\\_applications.pdf](https://www.opendatacenteralliance.org/docs/architecting_cloud_aware_applications.pdf)



openmaru