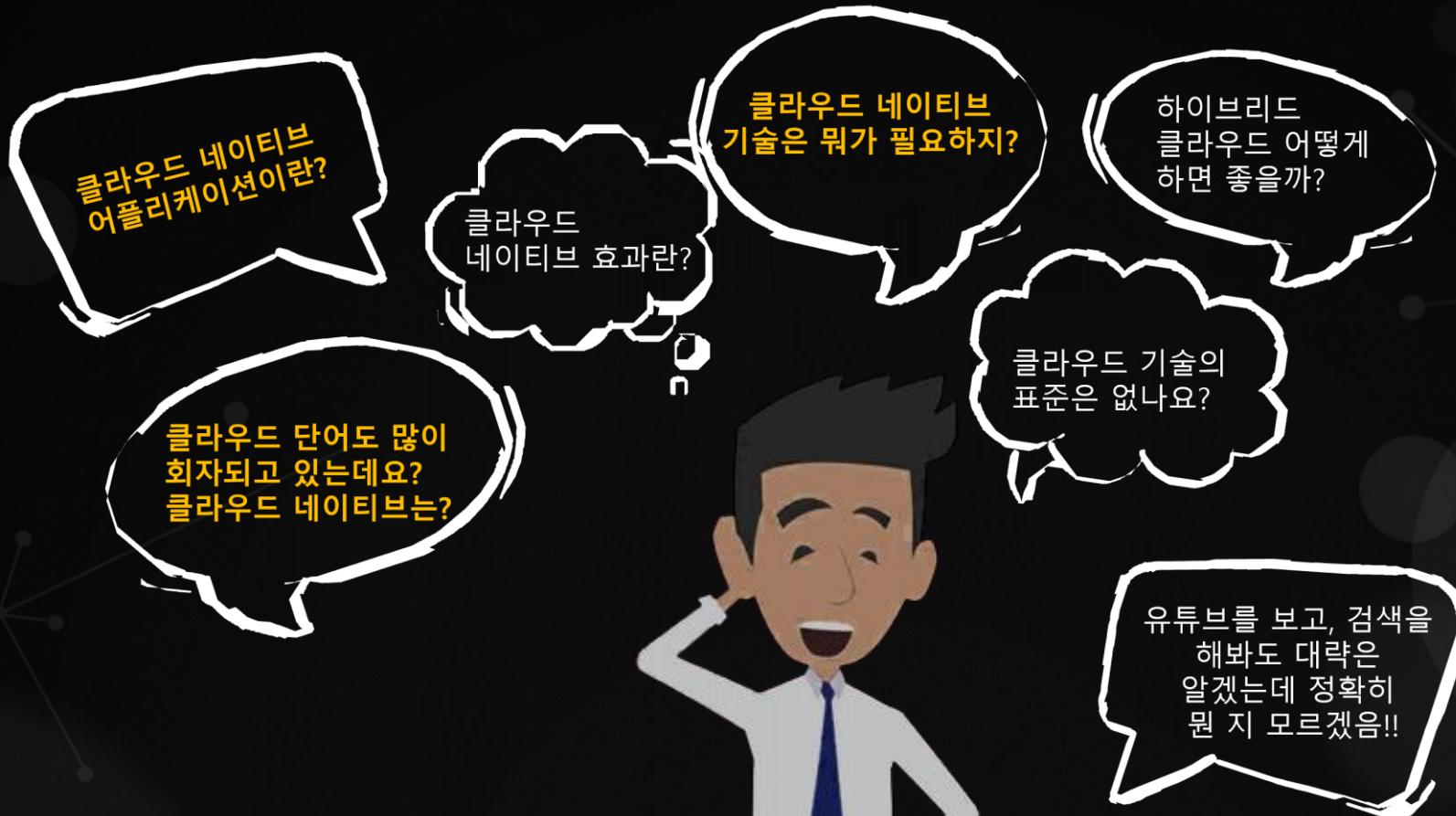


클라우드 네이티브 개념과 기술요소들

클라우드 네이티브?





- Docker , Kubernetes , containerd , cri-o
- 복잡하고 혼돈스런 컨테이너 표준

이것만은 알아야해 !

Development Process



Application Architecture

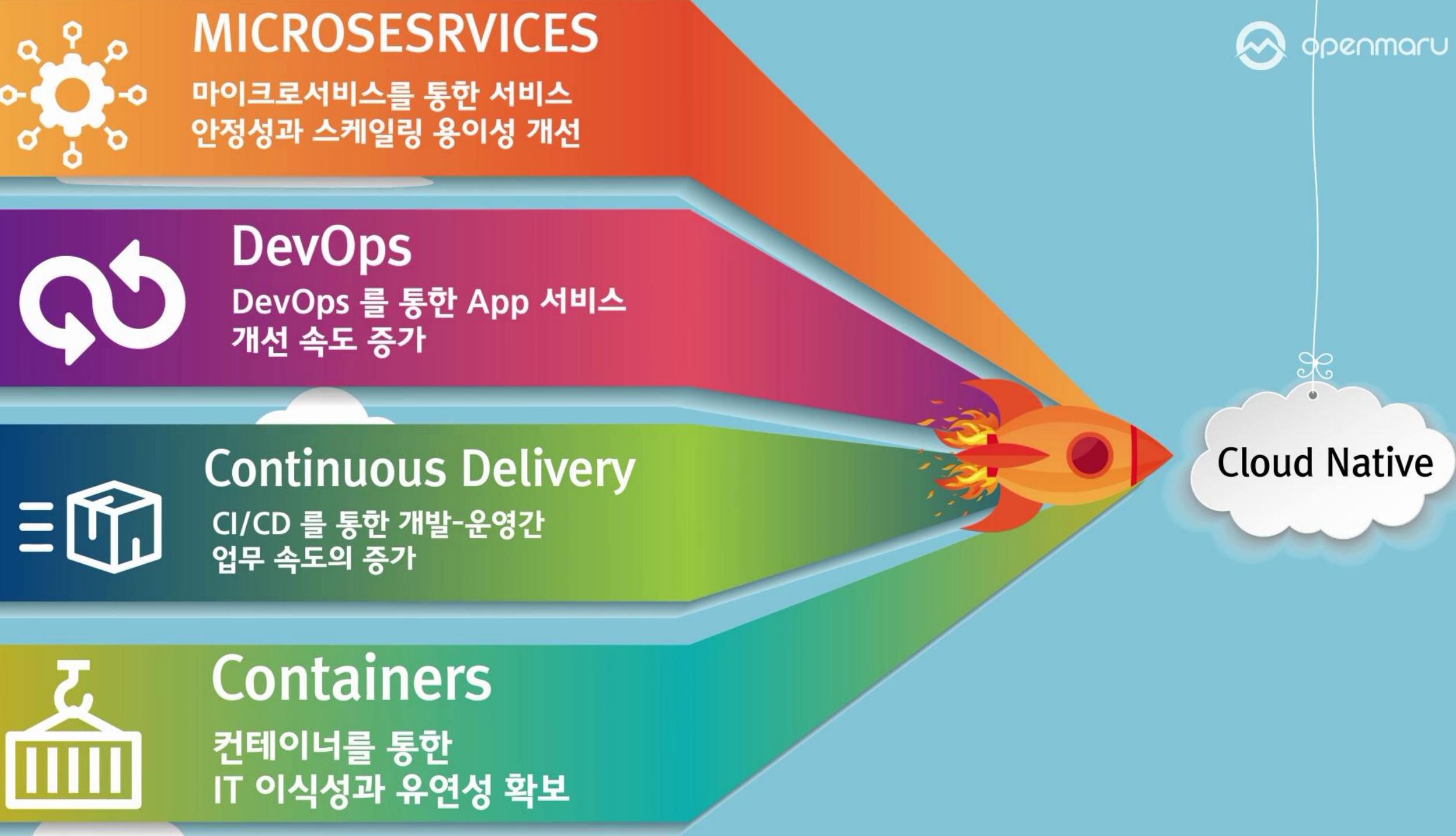


Deployment & Packaging



Application Infrastructure





물리서버 시대 ~ 2000년

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



~ 2000년
물리서버

2001년 ~ 2009년
가상화 기술 1세대

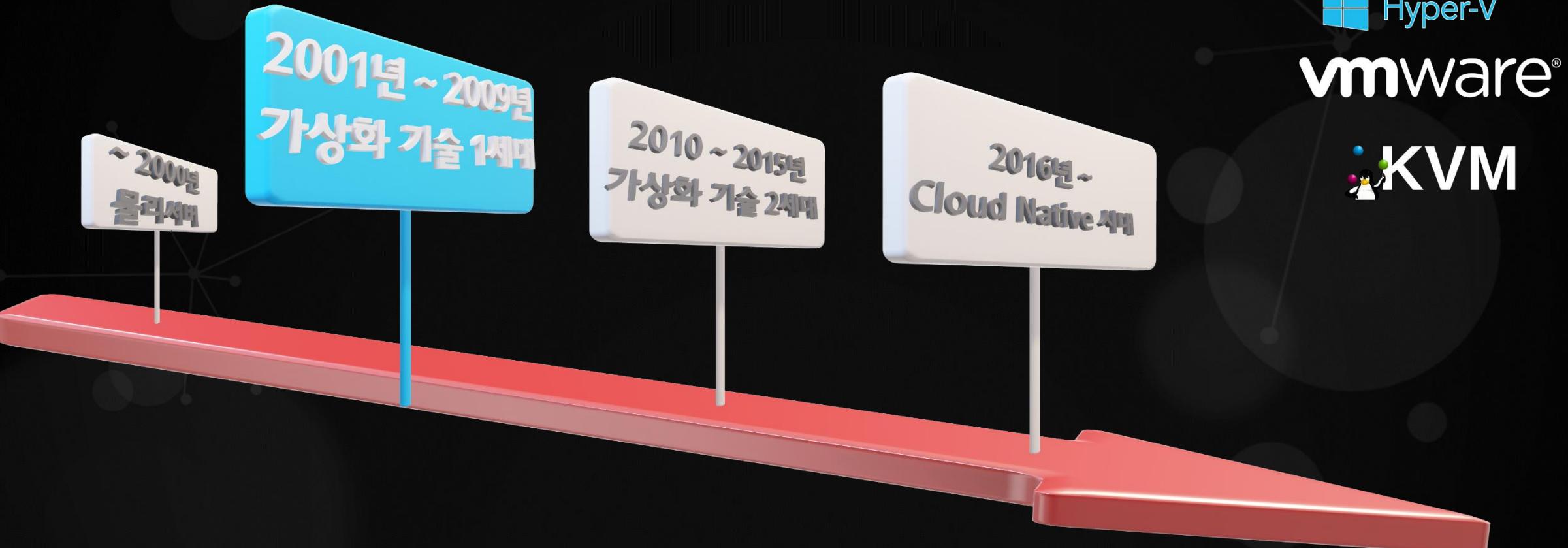
2010 ~ 2015년
가상화 기술 2세대

2016년 ~
Cloud Native 시대

가상화 기술 1세대 : 2001년 ~ 2009년



- 실제 시스템을 가상머신으로 대체, 가상머신을 애완 동물로 취급
- 통합 비율을 올리고 고효율화 하는 것이 목적 - 서버의 멀티 코어화와 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



openstack.

~2000년
물리서버

2001년~2009년
가상화 기술 1세대

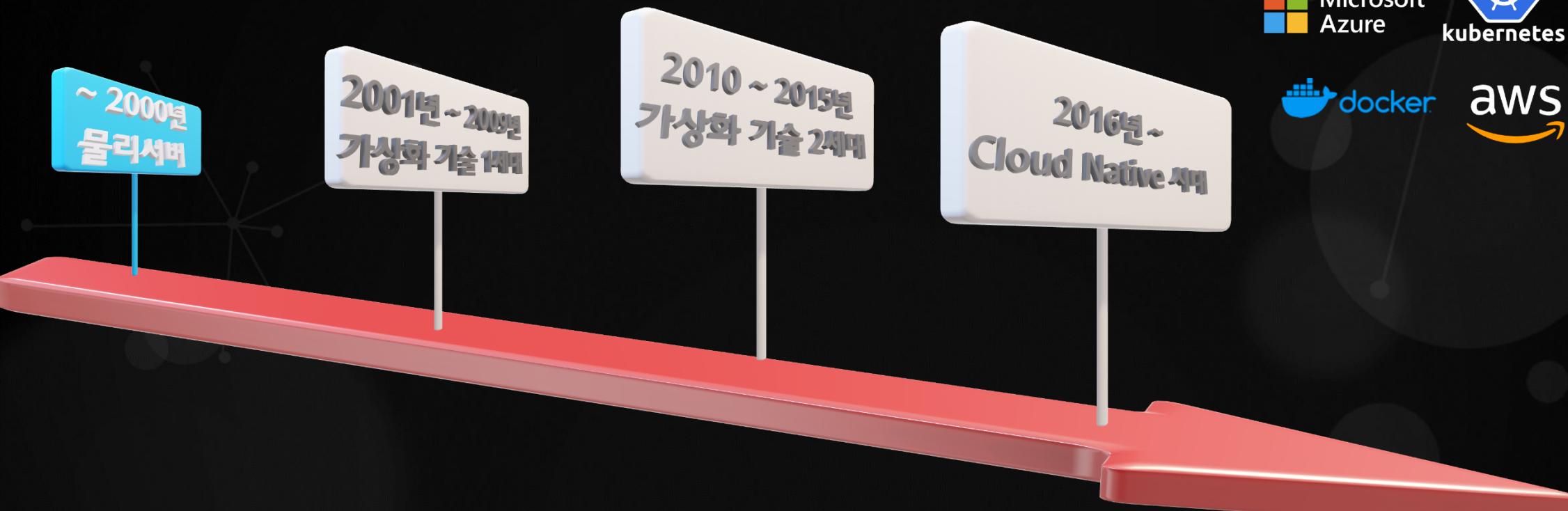
2010 ~ 2015년
가상화 기술 2세대

2016년~
Cloud Native 시대

Cloud Native 시대 2016년 ~

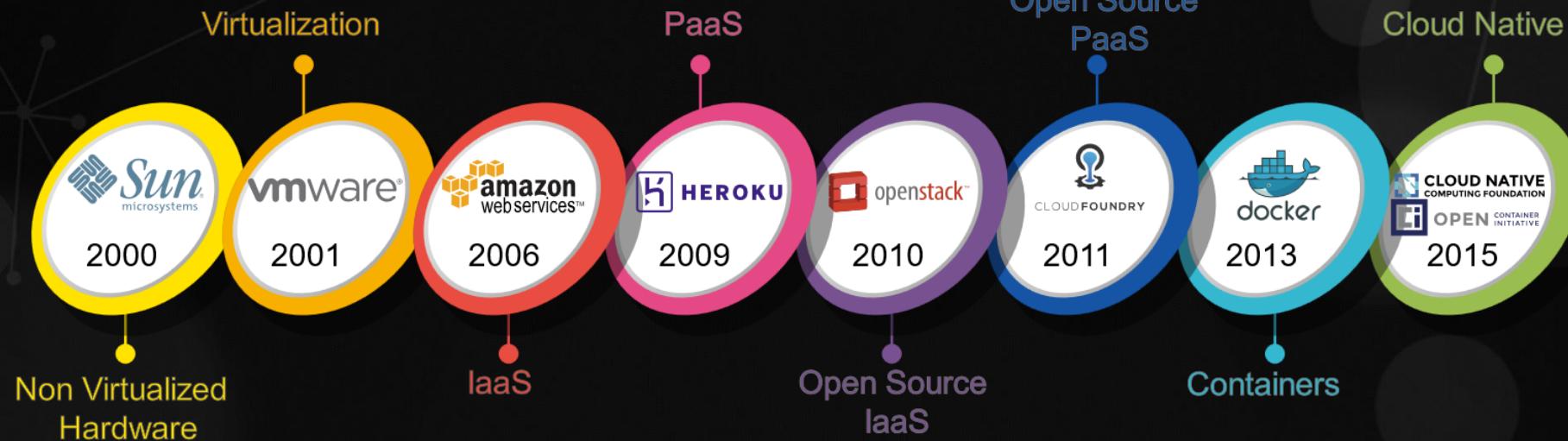


- 2016년 1월에 정식 출범 한 Cloud Native Computing Foundation (이하 CNCF)는 진짜 클라우드 기술을 오픈소스로 해결하는 하는 것을 목표
- 애플리케이션을 실행하는 데 필요한 최적의 인프라 제공
 - 개발 한 것을 "즉시" "안정적으로" 제공 (비즈니스 우위)



From Virtualization to Cloud Native

- Cloud native computing uses an open source software
- Standardization : <https://www.opencontainers.org/>



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 생태계를 육성하고 유지함으로써 해당 패러다임 채택을 촉진한다.

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



Cloud Native Computing Foundation



- Non-profit, part of the Linux Foundation; founded Dec 2015



- Platinum members:



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

기존 애플리케이션과 Cloud Native 애플리케이션 비교



분류	기존 애플리케이션	Cloud Native 애플리케이션
실행 환경	물리 서버 중심	컨테이너 중심
확장	Scale Up (수직 확장)	Scale Out (수평 확장)
결합	크고 조밀 결합	느슨하게 & 서비스 기반
인프라 의존성	인프라 의존	인프라 독립적으로 이동성 보장
Delivery 방법	폭포수형으로 장기간 개발	Agile & Continuous Delivery
개발 도구	로컬 IDE 개발 도구	클라우드 기반의 지능형 개발 도구
조직구조	사일로화 된 개발, 운영, 보안 팀	DevSecOps, NoOps 또는 협업

클라우드 네이티브 개념과 기술요소들

Cloud Native 효과

Cloud Native Computing으로 전환 효과

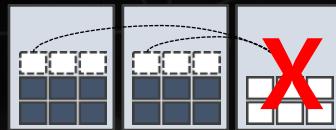


- Cloud Native Computing 환경은 클라우드가 제공하는 민첩성, 가용성, 확장성의 장점을 어플리케이션/서비스의 개발, 운영, 관리에 적용하여 기존 컴퓨팅 환경을 최적화 함



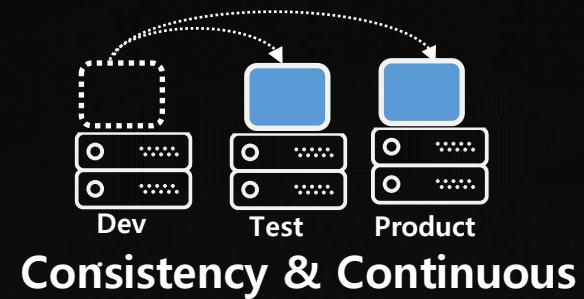
On Demand Delivery

필요한 컴퓨팅 자원을 즉시 제공



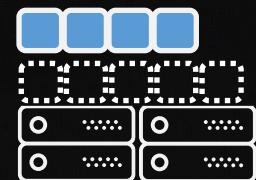
Self Recovery

- 비정상 애플리케이션 재시작
- 노드의 장애 발생시
정상 서버 노드로 자동 재배치



Consistency & Continuous

이미지 기반으로 구성, 배포 효율화
개발과 운영 환경의 일관성



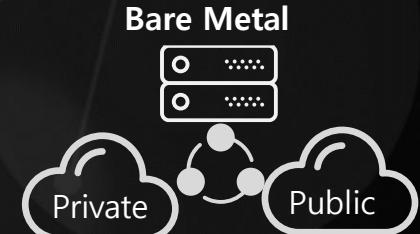
Application Scaling

VM 단위가 아닌 어플리케이
단위의 오토스케일링



Rolling Update

업그레이드 또는 패치 시
다운 타임은 제로 또는 최소화



Portable

멀티/하이브리드 클라우드 기반
애플리케이션/서비스 운영

Cloud Native Computing



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



클라우드 네이티브 vs. 클라우드 기반



- 클라우드 네이티브 애플리케이션은 클라우드의 모든 장점을 활용할 수 있도록 구축
- 클라우드 기반 애플리케이션은 On Premise 시스템을 클라우드에서 호스팅하여 운영

구분	클라우드 네이티브	클라우드 기반
서비스 모델	컨테이너 기반 PaaS (Platform As A Service)	가상화 기반 IaaS (Infrastructure As A Service)
디자인	시작 단계부터 클라우드의 장점인 민첩성, 확장성 그리고 이동성을 최대한 활용할 수 있도록 설계	On Premise에 구축된 시스템을 클라우드로 이전하여 운영
구현	하드웨어나 소프트웨어의 의존없이 표준 기반의 소프트웨어들로 빠르고 효율적으로 구축	특정 하드웨어와 소프트웨어에 의존적인 설정이 있어 구축에 시간이 걸림
확장성	마이크로 서비스 기반으로 전체서비스에 영향을 주지 않고, 업데이트가 필요한 서비스만 변경할 수 있으며, 서비스 단위의 Scale In/out 지원	애플리케이션 업데이트가 수작업이기 때문에 장시간의 다운타임일 필요하고 Scale In/Out이 어려움
비용	인프라 부분의 종속성이 없어 비용이 저렴	애플리케이션이 커질 수록 인프라 비용이 상승
유지보수	CI (Continuous Integration) / CD (Continuous Delivery)	버전관리, 설치 그리고 구성관리가 수작업이며 복잡함

클라우드의 핵심 개념 : 클라우드 네이티브 아키텍처



마이크로 서비스

유연하고 더 쉽게 구축할 수 있어
독립적으로 배포하고 스케일링 가능

API 지원

확장성 개발에 유연하고
합리적인 접근

분산 데이터 저장소

분산 시스템을 위한
데이터 사일로 새로운 접근 방식



CI/CD

높은 서비스 품질 유지와
혁신을 위한 지속적이고
빠른 업데이트

동적 스케일

탄력 있는 동적 스케일 -
스케일 아웃, 스케일 인을 쉽게
할 수 있는 기능

다양한 인터페이스 지원

웹, 모바일, 다양한 UI 를
지원하여 사용자에 보다 나은
경험을 제공

향상된 자동화

프로비저닝, 배포 스케일 주기가
고도로 자동화된 수요와 자원 제한
에 대응 가능

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マイ그레이션하거나 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.



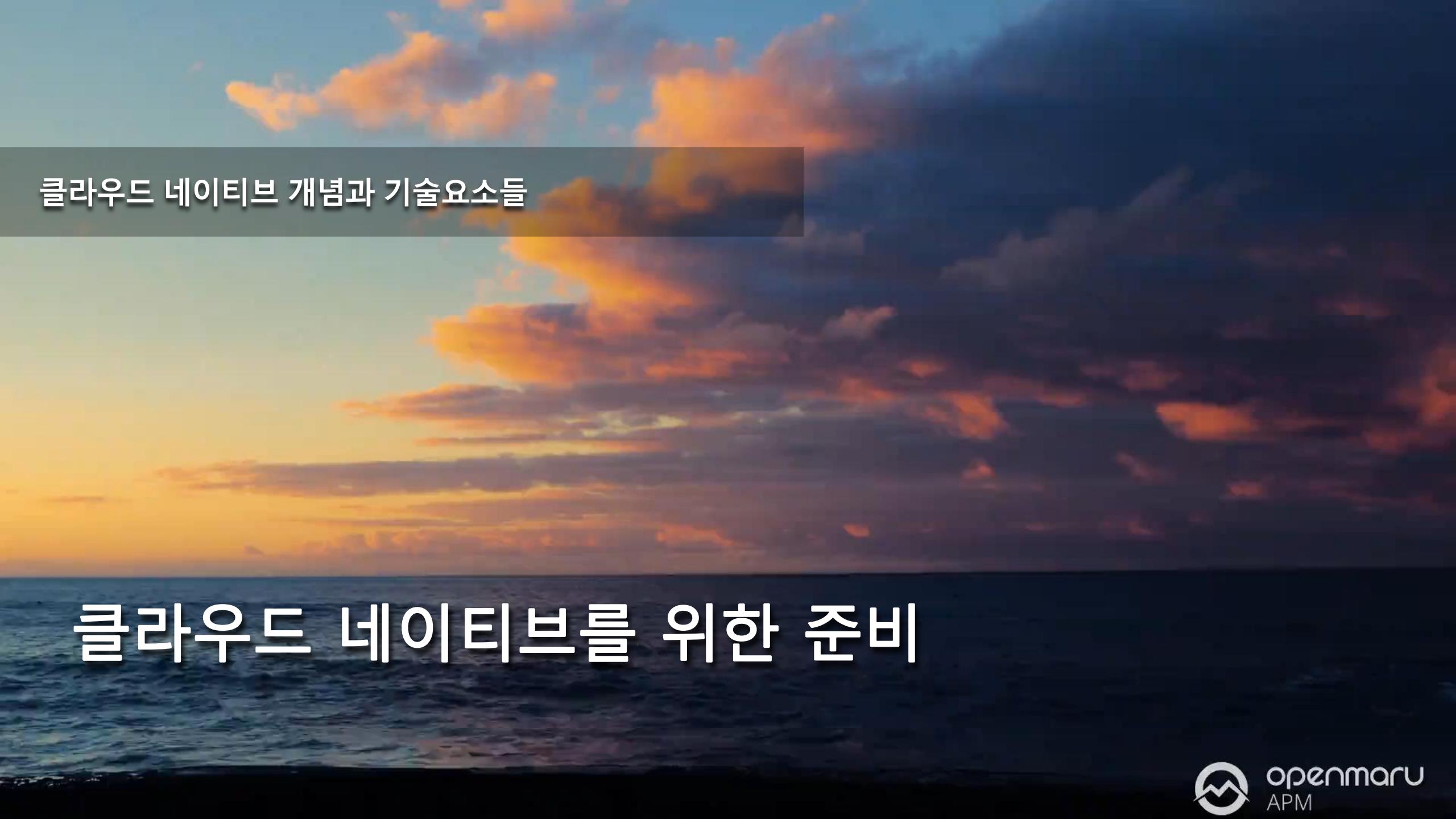
마이크로서비스 아키텍처

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.



클라우드 네이티브 개념과 기술요소들

클라우드 네이티브를 위한 준비

Different Levels of Cloud Native Application Maturity.



Maturity

Level 3: Cloud Native (클라우드 네이티브 단계)

- 마이크로 서비스 구조 사용과 원칙 준수
- API 기반의 소프트웨어 아키텍처

Level 2: Cloud Resilient (클라우드 심화 단계)

- 장애를 고려한 IT 디자인
- 개별 애플리케이션 장애가 전체 서비스에 영향을 주지 않음
- 적극적인 장애 테스트- 커버리지 90% 이상을 추구
- 중앙 집중화된 모니터링, 메트릭
- 퍼블릭/프라이빗 클라우드를 아우르는 스케일 전략

Level 1: Cloud Friendly (클라우드 친화 단계)

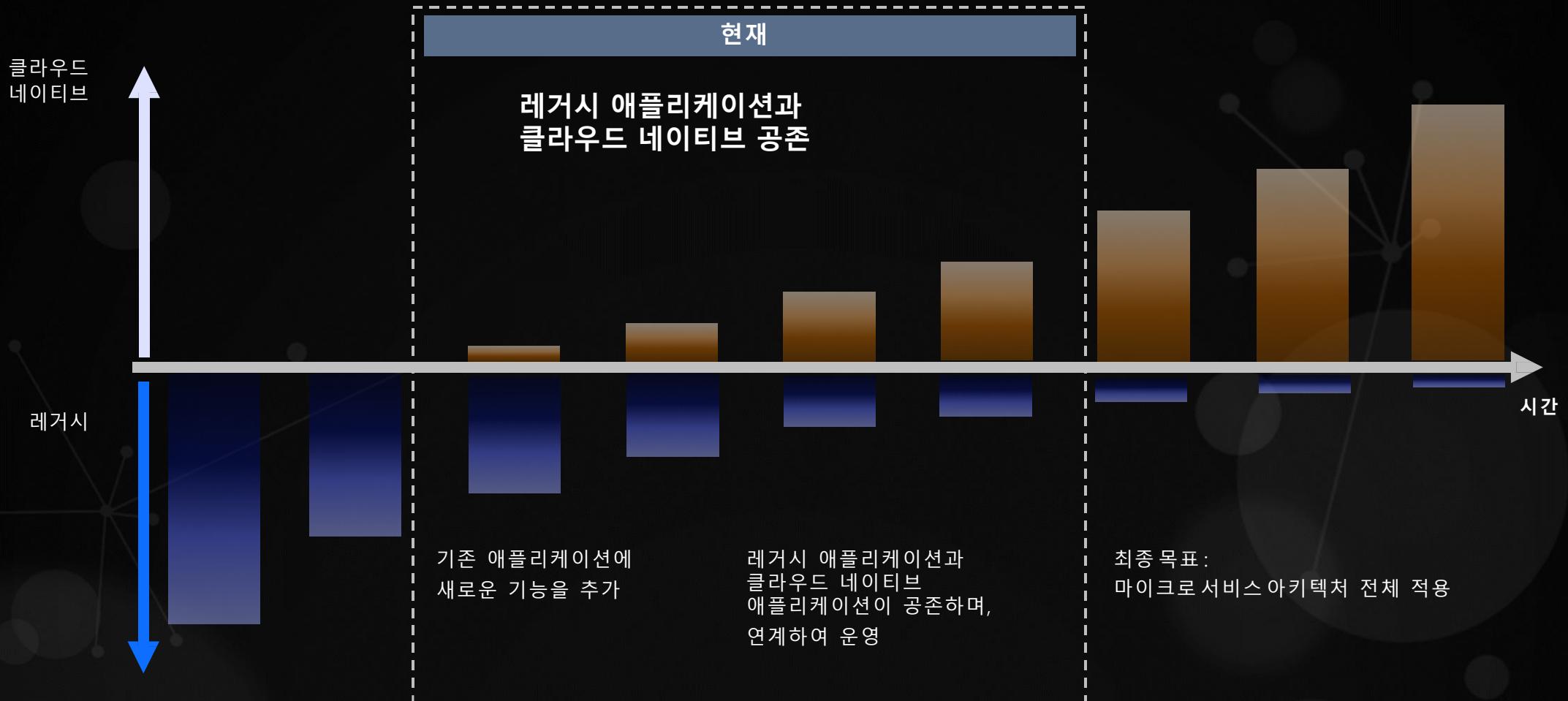
- 느슨한 시스템 결합 구조
- 이름으로 찾는 서비스
- 12-factor app 원칙 준수

Level 0: Cloud Ready (클라우드 준비 단계)

- 내부 파일시스템이 없는 환경
- 하드웨어 가상화
- 컨테이너화 된 이미지를 통한 실행 환경

https://www.opendatacenteralliance.org/docs/architecting_cloud_aware_applications.pdf

레거시와 클라우드 네이티브 애플리케이션 전환



Source : IBM



openmaru



클라우드 네이티브 개념과 기술요소들

How Organization?