

소프트웨어프로젝트 I

가상화와 클라우드 컴퓨팅

2022학년도 1학기

국민대학교 소프트웨어학부

시뮬레이션 (Simulation)

- Simulation: 시스템 동작의 몇 가지 특성을 이용하여 그 결과를 산출하는 것
- Emulation: 대상 시스템의 동작을 그대로 모사하여 흉내내는 것



가상실험

- 가상실험이 가능해지는 이유
 - 컴퓨터 하드웨어의 성능 발전 (비용 절감)
 - 대상 시스템에 대한 이해도 증가
- 가상실험으로써 얻는 이득
 - 실패에 대한 위험 감소 (또는 제거)
 - 실험 대상인 시스템의 반응에 대한 세밀한 관측

추상화 (Abstraction)

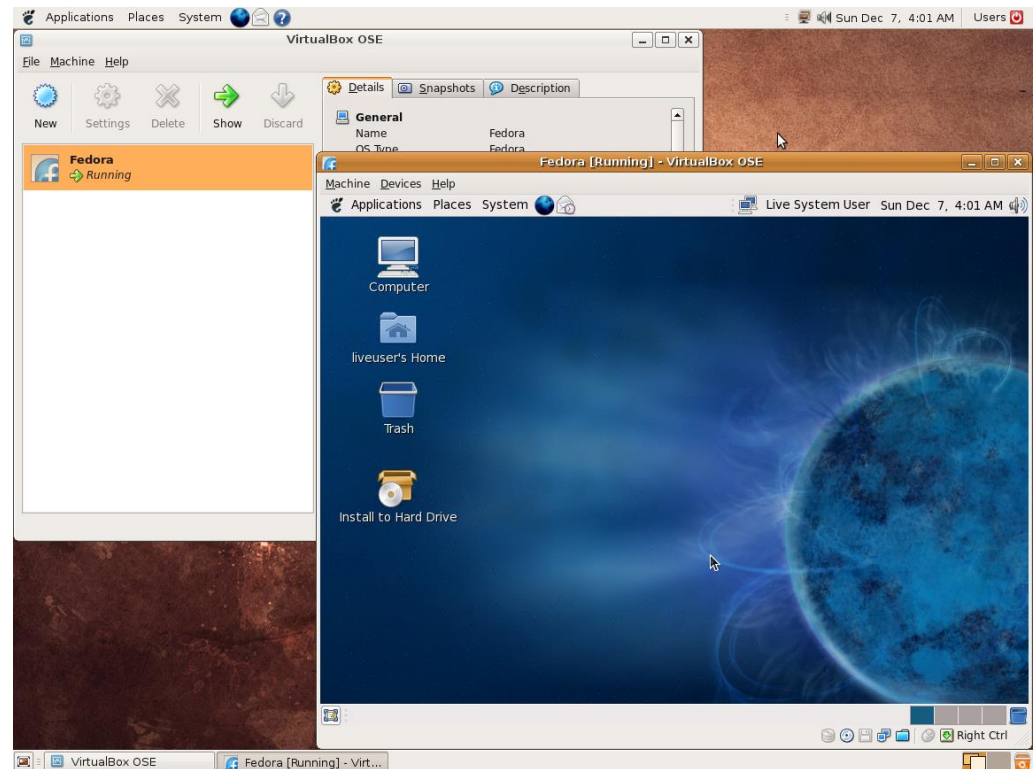
복잡한 자료, 모듈, 시스템 등으로부터
핵심적인 개념 또는 기능을 간추려 내는 것

- 예: 운영체제가 제공하는 추상화
 - 하드디스크에 대해서 “파일” 로 추상화
 - 메모리에 대해서 “주소” 로 추상화
 - 네트워크에 대해서 “포트” 로 추상화
- 컴퓨터과학은 곳곳에서 추상화의 계층을 다루고 있음

가상화 (Virtualization)

컴퓨터 리소스의 추상화를 일컫는 광범위한 용어

- (조금 딱딱하게) 물리적인 컴퓨터 리소스의 특징을 다른 시스템, 응용 프로그램, 최종 사용자들이 리소스와 상호 작용하는 방식으로부터 감추는 기술
- (대강 부드럽게) 컴퓨터 안에 또 다른, 즉 가상의 컴퓨터가 존재하도록 하는 기술



이미지 출처: Wikipedia

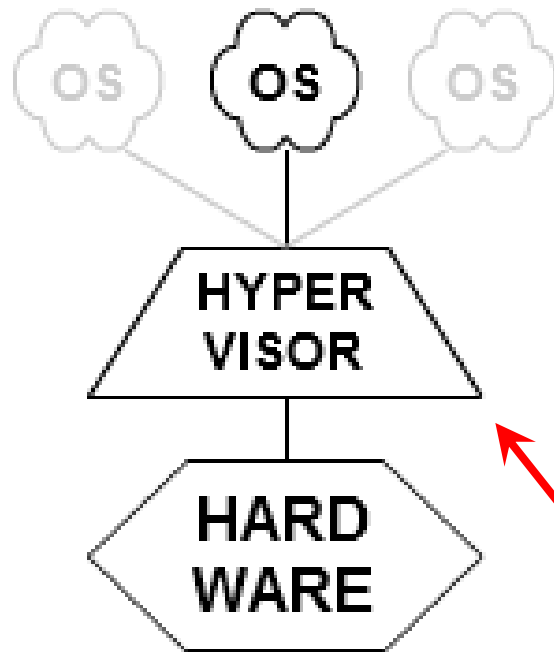
가상화 컴퓨팅의 이점

- 시스템 측면
 - 시스템 이용률 (utilization) 의 향상
 - 설정 (configuration) 구성과 복원이 용이
- 비즈니스 측면
 - 자본 및 운영 비용 절감
 - 다운타임 최소화 (또는 제거)
 - 비즈니스 연속성 및 재해 복구 향상
 - 데이터 센터 관리 간소화

서버 가상화 기술의 진화

- VM (Virtual Machine) 기반
 - 하이퍼바이저 (hypervisor) 를 이용
 - Type 1: 네이티브 (native) 또는 bare-metal 형
 - 전가상화 (full virtualization)
 - 반가상화 (para virtualization)
 - Type 2: 호스트형 (hosted)
- 컨테이너 (Container) 기반
 - 호스트 OS 의 컨테이너 기술을 이용

Hypervisor Type 1



TYPE 1

native
(bare metal)

하이퍼바이저가 제공하는
가상기계 (virtual machine) 위에
여러 (이종의) 게스트 (guest) OS 가 실행 가능

호스트 (가상화를 제공하는 컴퓨터) 의
하드웨어 위에서 직접 하이퍼바이저가 실행

→ 네이티브 (native) 또는 베어메탈 (bare metal) 타입이라고 불림

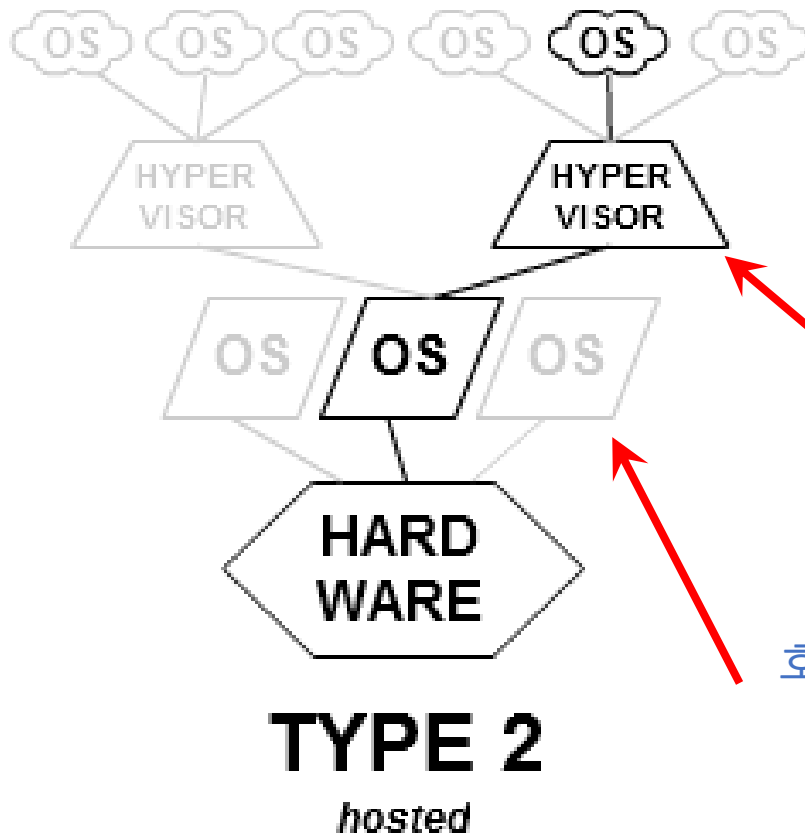
그림 출처: Wikipedia

전가상화와 반가상화

- 전가상화 (Full Virtualization)
 - 게스트 OS 는 호스트 시스템과 완전히 분리
 - 게스트 OS 는 하이퍼바이저의 특정한 기능을 통하여 호스트 시스템의 하드웨어 자원에 접근
 - 에뮬레이션에 준하는 중재의 부담
- 반가상화 (Para Virtualization)
 - 게스트 OS 를 일부 수정하여 직접 하드웨어 자원을 접근할 수 있도록 함 (hypercall)
 - 성능상의 이득이 있으나 게스트 OS 커널의 수정이 요구되는 번거로움도 있음



Hypervisor Type 2



Type 1 의 경우와 마찬가지로
게스트 (guest) OS 는 가상기계에서 실행

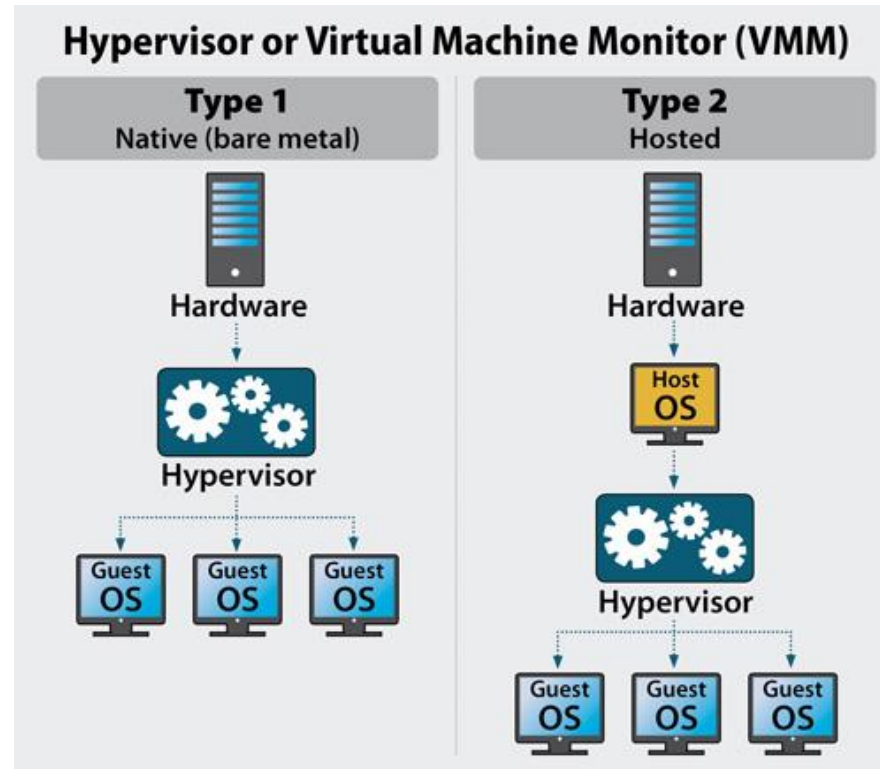
Host OS 위에서 하이퍼바이저가 실행하여
가상기계 (virtual machine) 추상화 제공

호스트 (가상화를 제공하는 컴퓨터) 의
하드웨어 위에는 host OS 가 실행

그림 출처: Wikipedia

(거의) 모든 하드웨어 자원을 에뮬레이션으로 제공
→ 오버헤드가 크다는 단점 있으나 유연성이 높음

하이퍼바이저의 두 가지 형태



한 세트의 서버 하드웨어를 이용하여
여러 개의 VM 이 실행될 수 있도록 하여
자원 이용의 효율성을 높임

유연성이 높아
데스크톱 (랩톱) 또는
작은 규모의 서버 환경에서 널리 이용

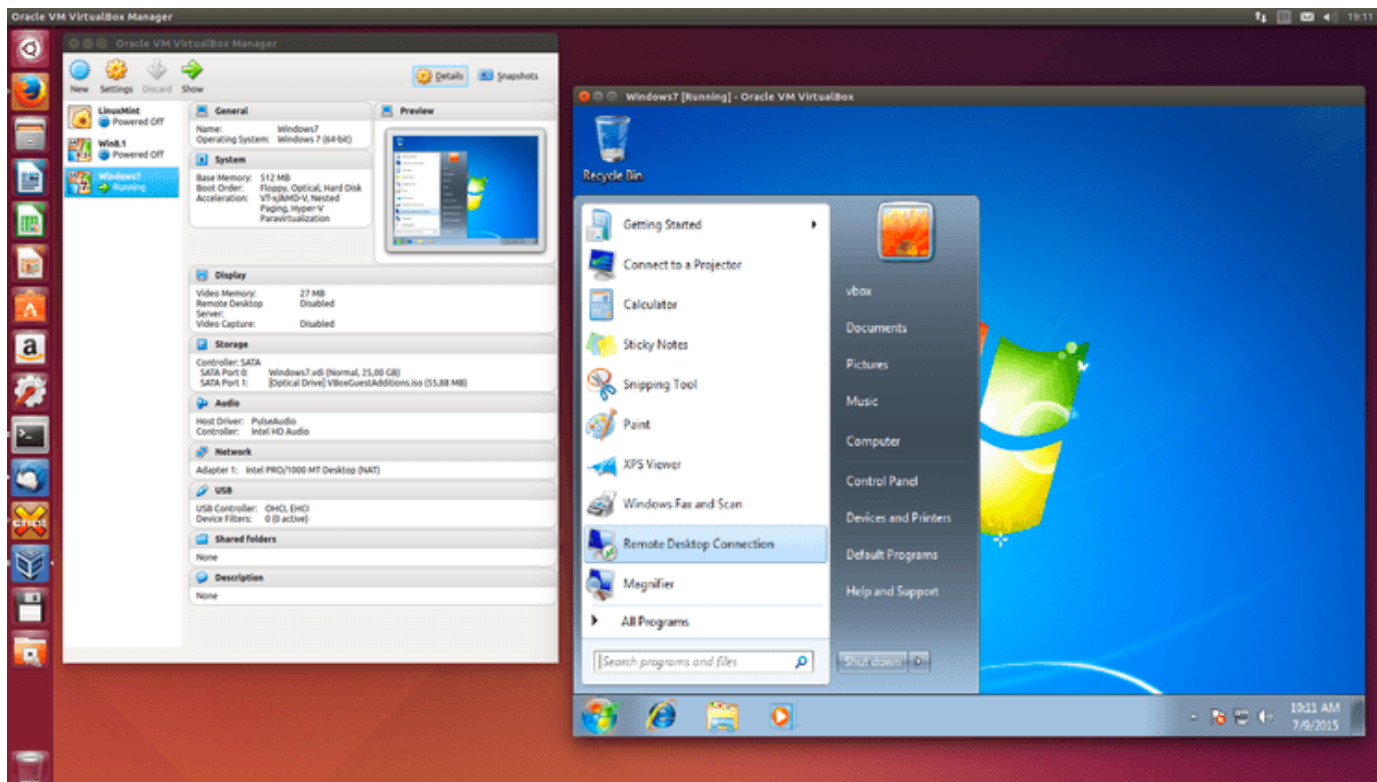
Type 2 하이퍼바이저의 예



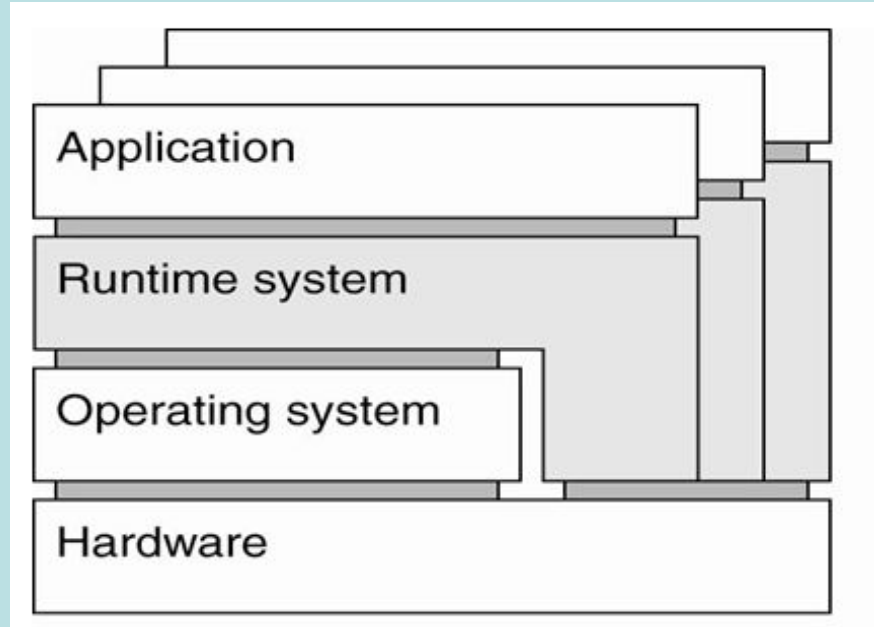
Oracle Virtualbox on Linux (Ubuntu)

Guest OS: Windows

<https://www.virtualbox.org/>

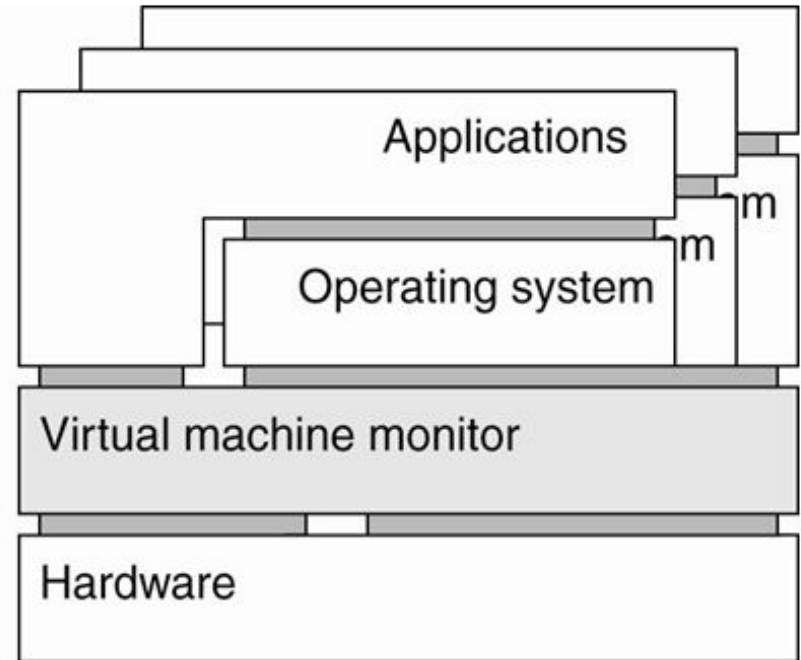


프로세스 가상기계 (Process VM)



Process Virtual Machine:

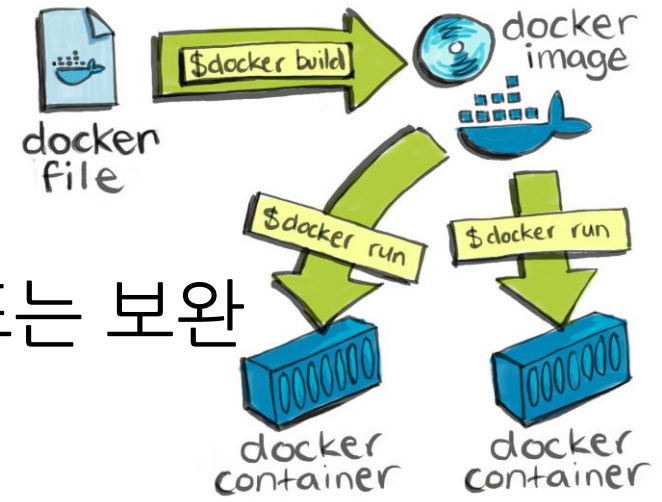
응용 프로그램은 가상기계 명령어로 번역되고
가상기계인 런타임 시스템이 이것을 해석, 실행
(예: Java Virtual Machine)



Virtual Machine Monitor (Hypervisor):

VMM 이 컴퓨터 하드웨어를 가상화하여
이 위에 운영체제와 응용 소프트웨어가 실행

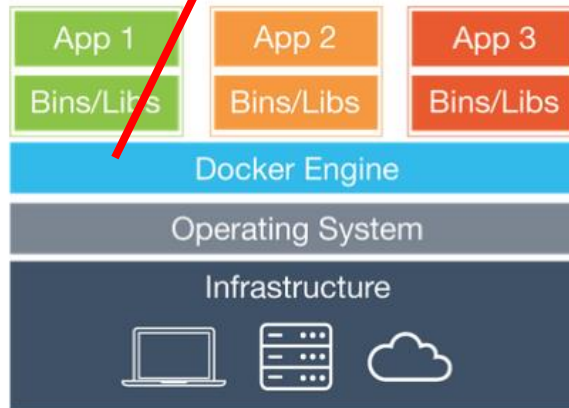
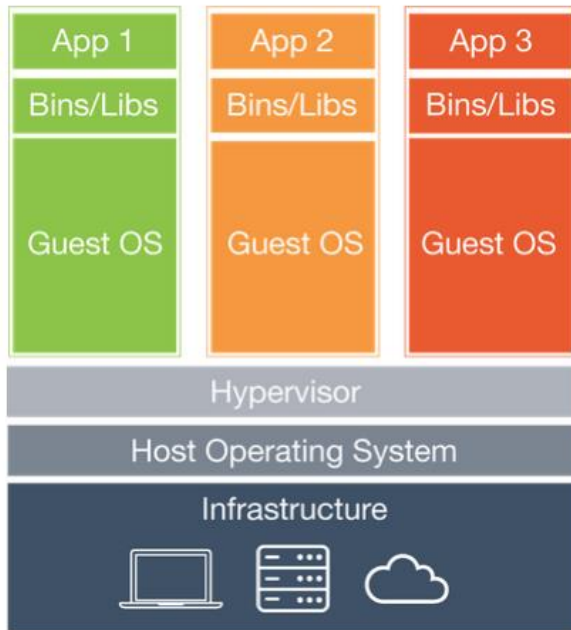
컨테이너 (Containers)



- 가상기계 (virtual machine) 의 대체 또는 보완 방식으로서 각광받고 있음
- 소프트웨어 개발 및 배포의 효율과 안정성을 향상시킴
 - 응용 프로그램, 설정 (configuration) 파일, 라이브러리, 그리고 이들 사이의 의존성 관계를 한데 묶어 관리
 - 이 “묶음” 을 컨테이너라고 부르고,
 - 컨테이너 엔진의 도움으로 시스템 의존성이 최소화되어 소프트웨어 시스템의 이식이 용이해짐

Containerization

컨테이너 엔진이
가상의 런타임 환경을 제공



각각의 응용 소프트웨어는
격리된 환경에서 실행

그림 출처: <https://medium.com/@darkrasid/docker%EC%99%80-vm-d95d60e56fdd>

컨테이너가 에뮬레이트하는 대상은
하드웨어가 아니고 운영체제

→ 하이퍼바이저를 이용한 가상화에 비하여
가볍고 빠르다는 장점을 가짐

컨테이너 사용의 이점

Gmail 에서 YouTube, Google 검색에 이르기까지
Google 의 모든 제품은 컨테이너에서 실행됩니다.
개발팀은 컨테이너화를 통해 더욱 신속하게 움직이고,
효율적으로 소프트웨어를 배포하며
전례 없는 수준의 확장성을 확보할 수 있습니다.



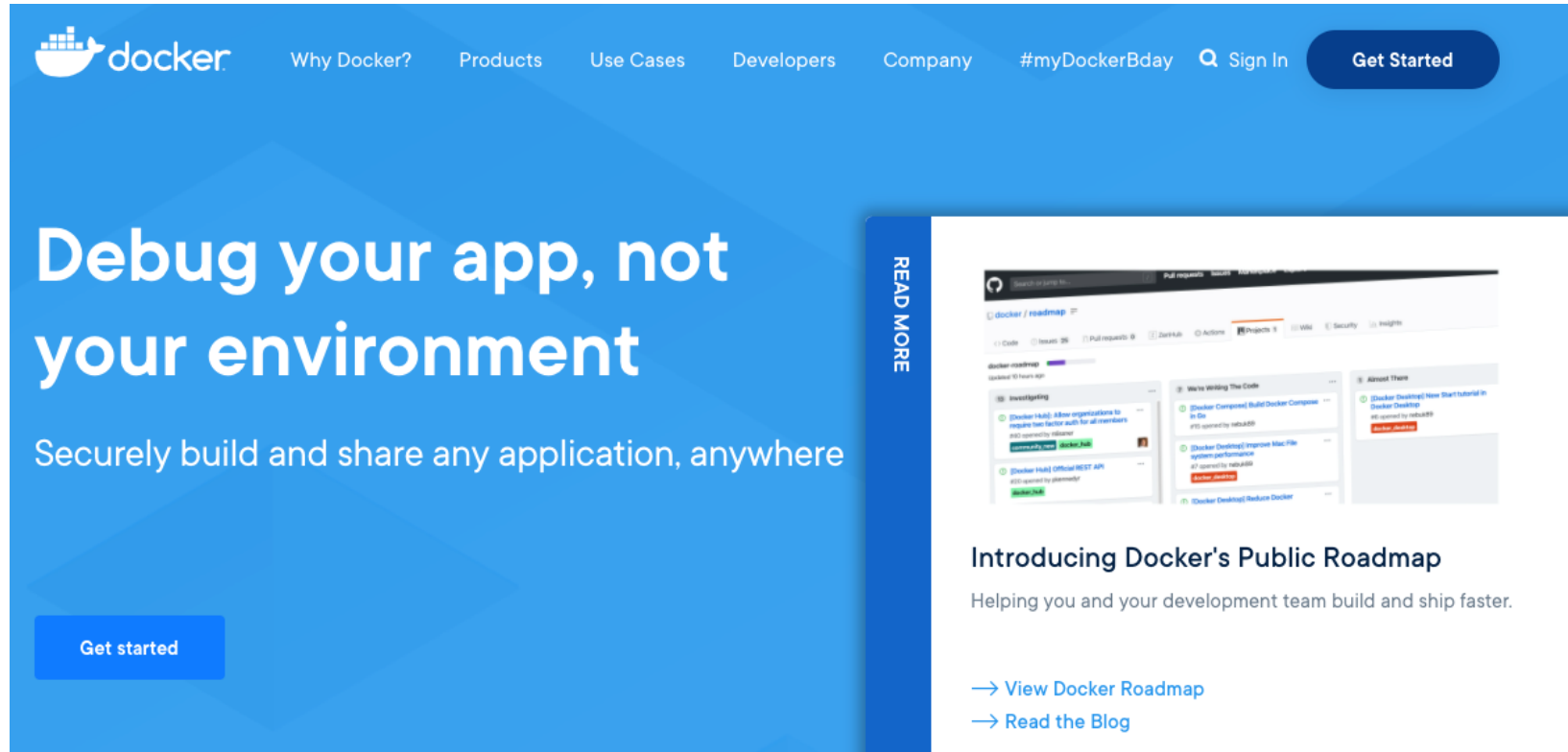
텍스트와 이미지 출처:

<https://cloud.google.com/containers?hl=ko>

	컨테이너의 이점	가상 머신의 이점
일관성 있는 런타임 환경	✓	✓
애플리케이션 샌드박스화	✓	✓
디스크 용량 절감	✓	
낮은 오버헤드	✓	

도커 (Docker)

<https://www.docker.com/>



The image shows a screenshot of the Docker website. The top navigation bar includes links for 'Why Docker?', 'Products', 'Use Cases', 'Developers', 'Company', '#myDockerBday', 'Sign In', and a 'Get Started' button. The main headline reads 'Debug your app, not your environment' with the subtext 'Securely build and share any application, anywhere' and a 'Get started' button. To the right, there is a 'READ MORE' section titled 'Introducing Docker's Public Roadmap' which includes a list of roadmap items and links to 'View Docker Roadmap' and 'Read the Blog'.

docker

Why Docker? Products Use Cases Developers Company #myDockerBday Sign In Get Started

Debug your app, not your environment

Securely build and share any application, anywhere

Get started

READ MORE

Introducing Docker's Public Roadmap

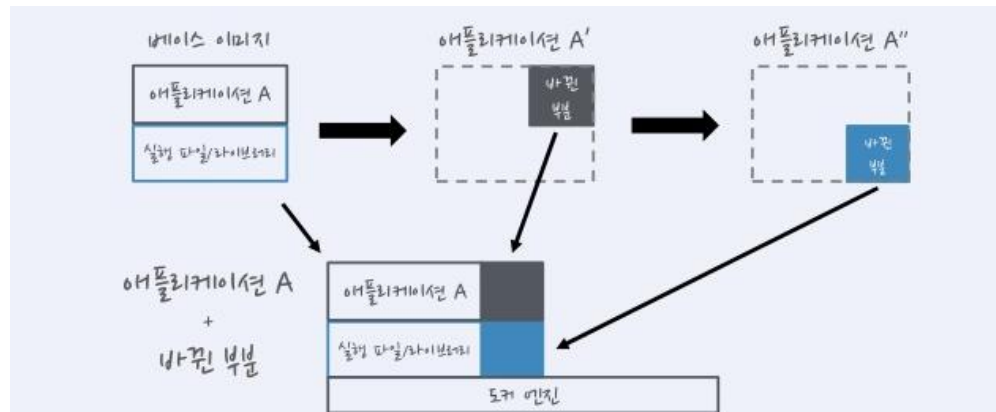
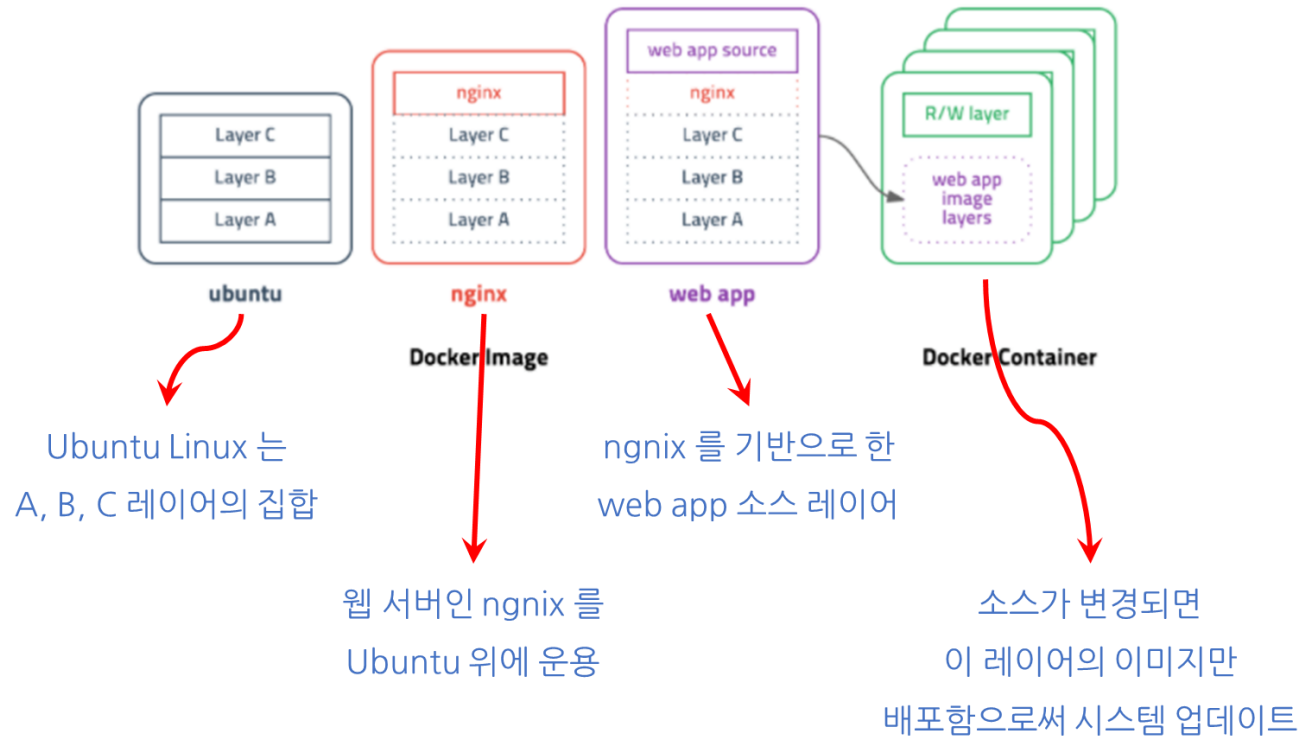
Helping you and your development team build and ship faster.

- View Docker Roadmap
- Read the Blog

널리 이용되고 있는 컨테이너 기반 가상화 플랫폼

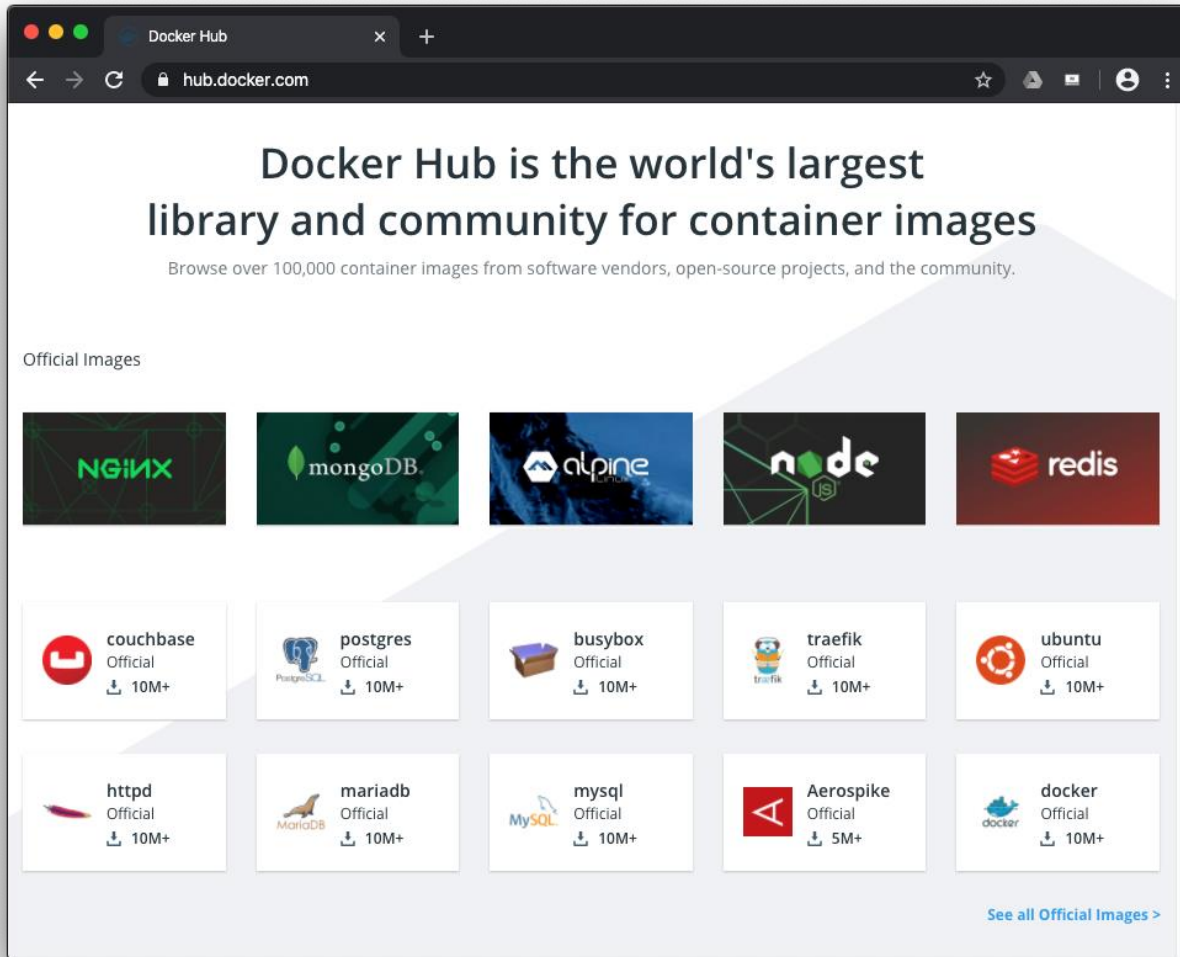
- Linux, Windows, MacOS 상에서 실행되는 컨테이너 엔진을 제공
- Azure, AWS 등의 클라우드 컴퓨팅 환경에서의 실행도 지원

도커를 이용한 소프트웨어 개발



Docker Hub

<https://hub.docker.com/>



컨테이너 이미지들에 대해
원격 저장, 유지관리,
공유, 권한 관리 등을
효율적으로 행할 수 있는
온라인 서비스 제공

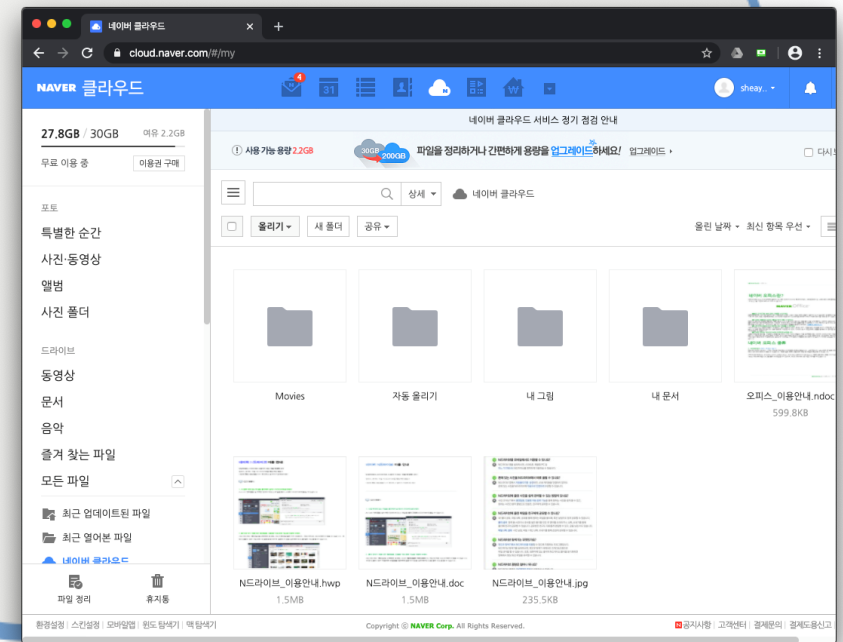
(이후에 배우게 될
GitHub 와 유사해요!)

소프트웨어의
개발 및 배포 프로세스의
효율성이 크게 증대

클라우드 컴퓨팅 (Cloud Computing)

IT 자원을 인터넷을 통해 온디맨드 (on-demand) 로 제공하고
사용한 만큼만 비용을 지불할 수 있도록 하는 기술

구입, 소유, 유지관리로부터
필요에 따라 “접근” 하는
인프라스트럭처로 중심 이동



클라우드 컴퓨팅의 이점

- 민첩성 (Agility)
 - IT 리소스의 구동이 빨라짐 (일 단위 → 분 단위)
- 탄력성 (Elasticity)
 - 비즈니스 요구에 따라 축소/확장이 가능
- 비용 절감 (Low Cost)
 - TCO (total cost of ownership) vs 사용 요금
- 가용성 (Availability)
 - 빠른 배포와 무중단 교체

IaaS (Infrastructure as a Service)



그림 출처: AWS (Amazon Web Service)
<https://aws.amazon.com/ko/what-is-cloud-computing/>

- IT 자원을 서비스로서 제공
 - 네트워킹 기능, 컴퓨터 (가상 또는 전용), 데이터 스토리지
- 유연성과 관리 제어 자유도가 높으며, 기존의 IT 부서 및 개발자 집단에 익숙한 기능을 제공

PaaS (Platform as a Service)



그림 출처: AWS (Amazon Web Service)
<https://aws.amazon.com/ko/what-is-cloud-computing/>

- 소프트웨어가 실행될 플랫폼을 서비스로서 제공
 - 하드웨어와 운영체제 등은 서비스 제공자가 모두 관리
 - 응용 소프트웨어의 배포와 유지 관리에 집중할 수 있음
- IT 자원의 구매, 용량 계획, 유지 관리 등의 부담을 경감

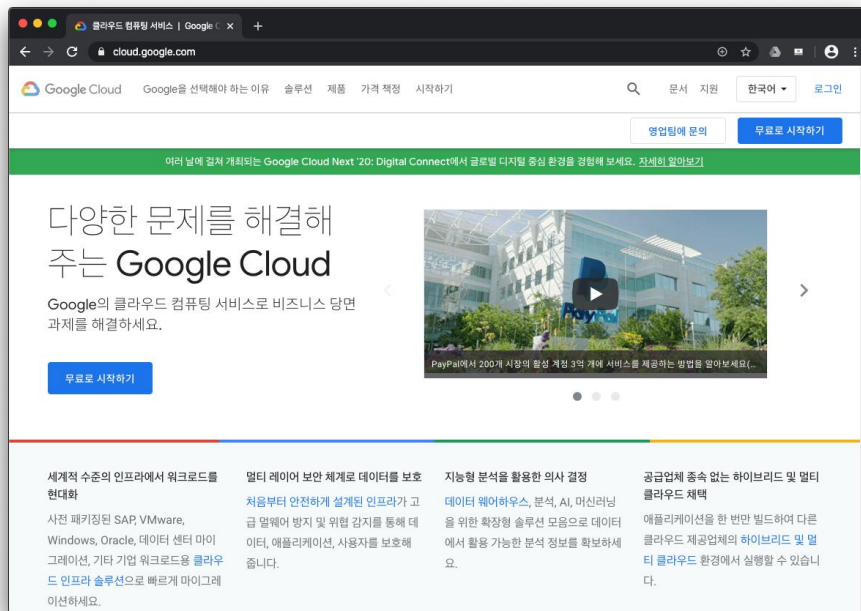
SaaS (Software as a Service)



그림 출처: AWS (Amazon Web Service)
<https://aws.amazon.com/ko/what-is-cloud-computing/>

- 서비스 공급자에 의해 실행되고 관리되는 완전한 제품을 제공
 - 최종 사용자 응용 서비스 (예: 웹 기반 이메일 등)
- 서비스 유지 관리 방법 및 시스템 인프라의 운용에 대하여 고려할 필요 없이 소프트웨어를 이용할 수 있음

Google Cloud Platform

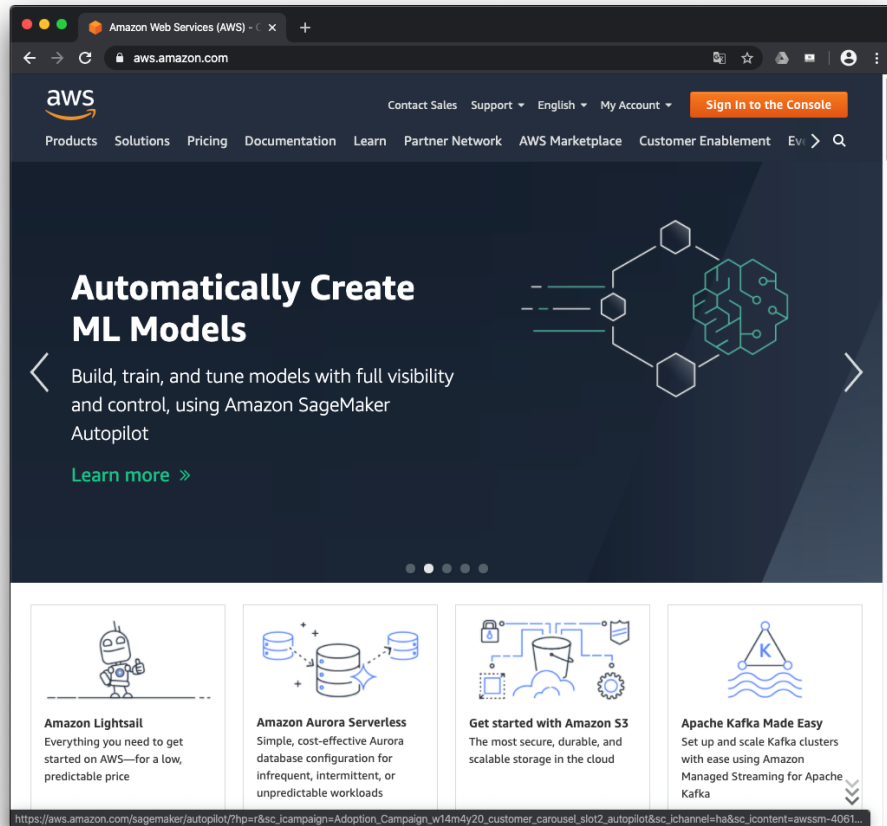


- GAE (Google App Engine) 을 이용하여 빠른 개발 및 배포가 가능
- Google 이 자사의 서비스 (검색, Gmail, Youtube 등) 을 구성하는 데 이용
- 데이터 분석 및 기계학습 플랫폼을 강조

<https://cloud.google.com/>

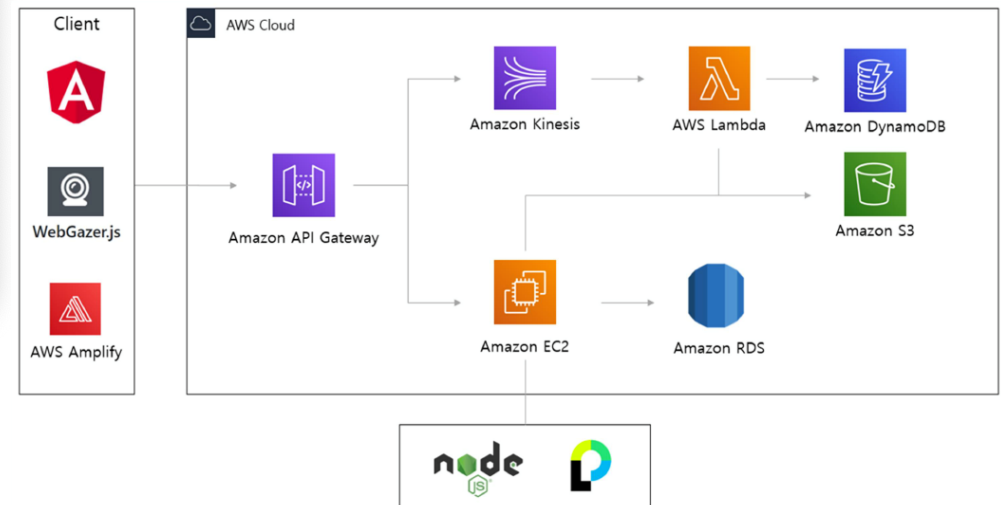


AWS (Amazon Web Services)

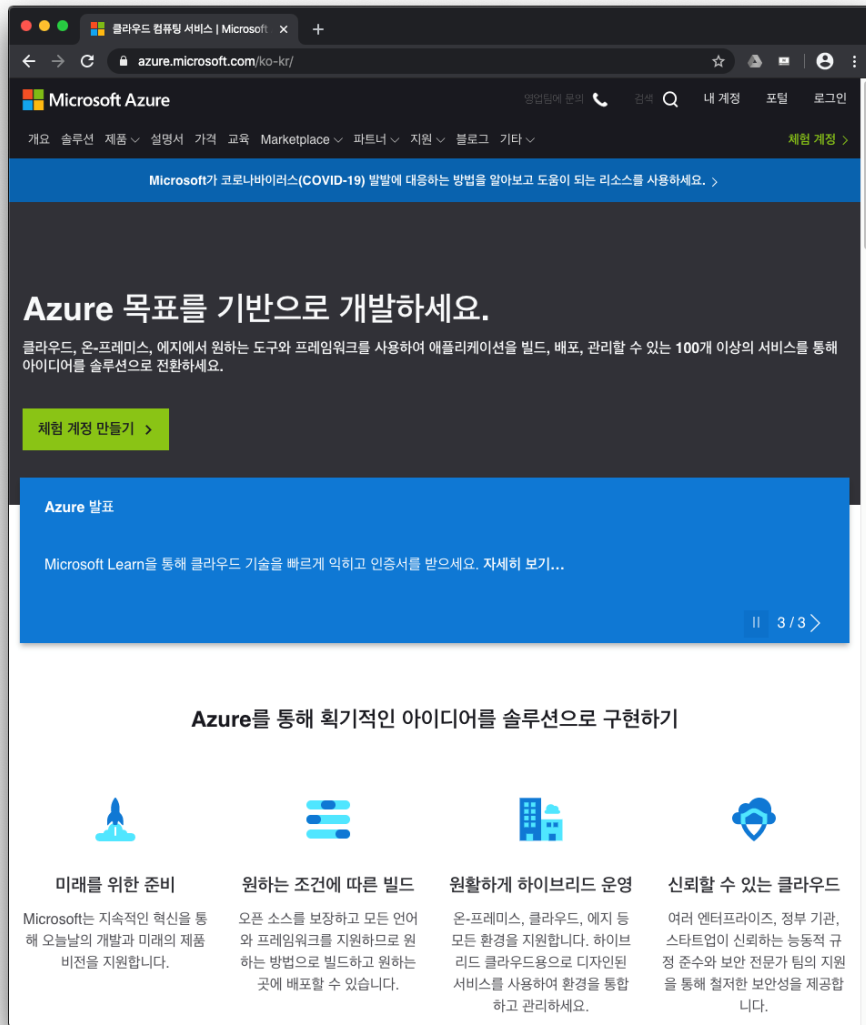


<https://aws.amazon.com/>

- IaaS 영역이 강조된 형태의 서비스를 운용하여 클라우드 컴퓨팅 서비스 시장을 선점
- 데이터베이스와 보안, 자동화 기술 등을 강조
- 큰 기업 고객을 많이 확보하고 있으며 기업, 학교 및 일반 대중을 상대로 교육, 홍보에 집중



Microsoft Azure



- PaaS 를 기초로 서비스를 시작하여 이후 IaaS 서비스로 확장
- 윈도우즈 플랫폼을 이용하는 소프트웨어 개발에 유리한 측면
- Red Hat Linux 에 통합 지원을 제공함으로써 자사의 플랫폼 종속성을 탈피하려는 노력

<https://azure.microsoft.com/ko-kr/>

요약

- 추상화, 가상화
 - 컴퓨터는 추상화의 집합체 (전기적 신호 \leftrightarrow 정보)
 - 가상화 기술을 통하여 시스템 이용률을 향상시키고 비용을 절감할 수 있음
- 클라우드 컴퓨팅의 확산
 - 소유에서 접근으로
 - 가상화 기술을 바탕으로 하고 있음
 - IaaS, PaaS, SaaS



Rifkin, Jeremy. "The Age of Access"