2강. 클라우드 컴퓨팅의 기초

**<클라우드 등장 배경>**

1. 사업적 동인
   1. 용량계획
      1. 사용 부하의 변동을 예측하는 것이 어려웠음
      2. 그래서, 불필요한 IT인프라 지출을 하지 않으면서도 중단 없이 운영해야 했음
   2. 비용절감 & 운영 오버헤드
      1. 소유비용 (감가상각비)를 고려
      2. IT 인프라는 유지 비용이 들어가지만, 대부분 활용도가 낮았음
   3. 조직적 민첩성: 비즈니스에 대응하는 정도
      1. 이전보다 더욱 가용성과 신뢰성이 높은 IT자원을 필요로 함
2. 기술혁신
   1. 그리드 컴퓨팅: 전력망에서 전기를 쓰는 것처럼 컴퓨팅 자원을 사용
   2. 웹/클러스터링
      1. 웹 응용프로그램은, 수요자 중심적이라 사용부하가 일정치 않음
      2. 그래서 Load Balancer 같은 장비를 통해 부하를 분산함
   3. 가상화 기술
      1. 서버의 탄력성 (하나의 물리적 서버로 여러 논리적 서버 구성)을 구현
      2. 현대 클라우드 기술의 핵심 기술

**<클라우드 컴퓨팅의 정의>**

“클라우드 컴퓨팅은 **확장성** 있고 **사용량 측정**이 가능한 IT자원을 **원격으로 제공하여 사용**하도록 고안된 분산 컴퓨팅의 특수한 형태이다.”

**<개요>**

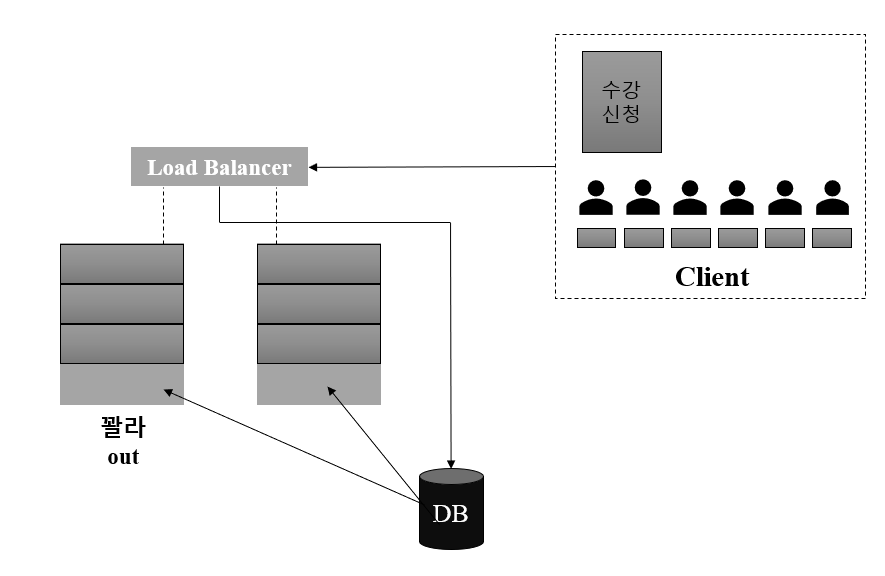
1. IT 자원(IT resources): 물리적 또는 가상의 IT 관련 산물 (서버, 프로그램, 서비스 등)
2. 가상화(virtualization): 물리적IT자원이 복수의 가상 이미지 제공 🡪 처리기능 개별적 공유  
   물리적 IT 자원의 소유자는 관리 제어 집중화 🡪 가상 IT 자원의 세부 구현 사항 소비자에게 의도적으로 숨김

스크린샷이(가) 표시된 사진

자동 생성된 설명

* 1. 가상화 환경 필요
  2. 가상 서버는 물리적 컴퓨터를 에뮬레이션 하는 가상화
  3. 독립적인 물리적 서버로 나타남

1. Scaling: 증가하거나 감소하는 IT자원 사용요구를 정상적으로 처리할 수 있는 능력
   1. 수평 확장: 동일한 유형의 자원을 할당하거나 해제.
      1. **스케일링아웃(scaling out)**: 자원의 수평적 할당
      2. **스케일링인(scaling in)**: 수평적 자원의 해제
         * 클라우드 환경 내에서 일반적인 스케일링 유형



상태 저장인 서버에서 스케일링아웃을 하면, 한쪽의 서버의 부하 증가  
🡪 차례로 죽음

이유

* 상태 저장이면, 기존의 부하가 서버에 저장이 되기 때문
* 모든 부하를 균등하게 분배하는 Load Balancer에 의해
* 기존에 있던 서버의 부하 + Load Balancer의 부하로 인해

해결책

* + - * + 부하를 임시로 저장하게 하는 저장소를 Load Balancer에서 연결
        + 요청이 오면 저장소에 부하 저장
        + 각 서버에 여유가 생기면, 저장소에 있는 부하를 각 서버에서 처리
  1. **수직 확장**: 기존 자원이 다른 자원으로 대체될 때 발생
     1. **스케일링업(scaling up)**: 용량이 더 큰 자원으로 대체
     2. **스케일링다운(scaling down)**: 용량이 더 적은 다른 것으로 대체
        + 교체 진행 중 중단 시간이 필요하기에 일반적인 유형이 아님

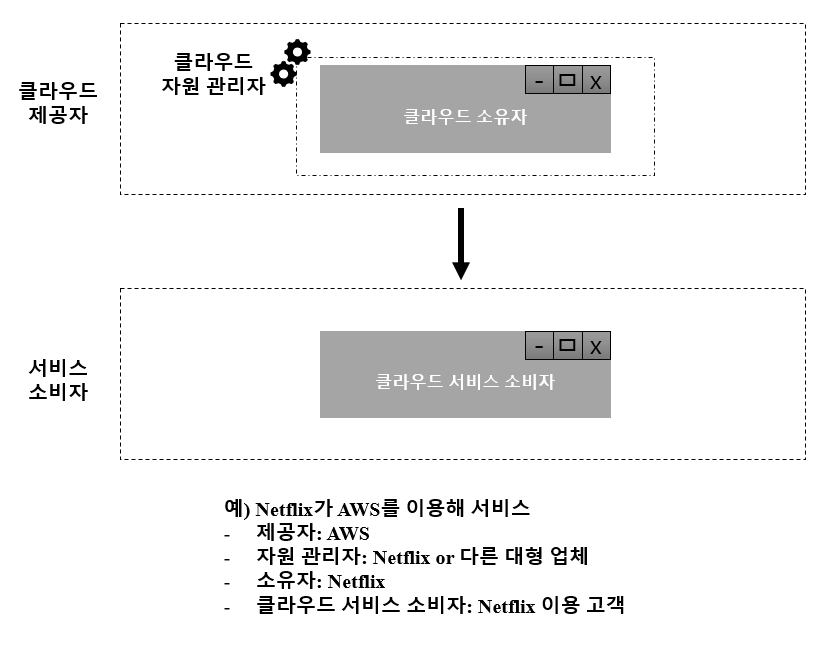
1. 클라우드
   1. 정의: **확장성** 있고 **사용량 측정**이 가능한 IT자원을 **원격으로 자공할 목적**으로 설계된 독립된 원격 IT 환경
   2. 정확한 물리적 위치를 알 수 있고 알지 못할 수 있음.
   3. 클라우드가 웹 기반일 필요 없음.
   4. 모든 원격 접속 프로토콜과 함께 사용
2. 온프레미스(On-Premise): 내부 IT엔터프라이즈 환경내에 있는 IT자원을 규정하는데 사용됨.  
   🡨🡪 클라우드와 상반된 개념으로 사용
3. 서비스: 서비스 사용 규약으로 통칭되는 기술 인터페이스 (or API)를 통해 원격 호출 가능한 소프트웨어 프로그램

**<클라우드의 특징>**

1. 온디맨드식 사용
   1. 자체 공급(self-provisioning)  
      예) gmail 계정을 “내가” 만들고 “등록하기만 하면 된다.”
   2. 일방(unilaterally)
2. 언제 어디에서나 가능한 접근(Ubiquitous Access)
   1. 넓은 지역에서 접근 🡪 될 수 있으면 많은 기술 지원
   2. 일정 범위의 전송 프로토콜 인터페이스, 보안 기술의 지원 필요
3. 멀티테넌시와 자원 풀링(Multitenancy & pooling)
   1. 테넌시: 집터
   2. 멀티테넌시: 여러 소비자에게 프로그램 인스턴스 제공하여 각 소비자가 독립적으로 사용할 수 있게 하는 소프트웨어 프로그램의 특징 🡪 논리적으로 분리
   3. 풀링: 멀티테넌시 모델 이용하여 여러 클라우드 서비스 소비자가 사용할 수 있는 IT자원 🡪 물리적으로 분리
4. 탄력성(Elasticity)
   1. 예) 고무줄
   2. 수요↑ : 자원↑ / 수요↓ : 자원↓ // 스케일링과 연관
5. 사용량 측정
   1. 소비자가 사용한 IT 자원의 사용량을 기록  
      무엇보다도 우선되고 투명하게 측정!
   2. 청구하지 않는 클라우드에도 관련이 있음.
6. 복원력(Resiliency)
   1. 장애 복구(failover)의 한 형태: 복원력이 있는 컴퓨팅은 복수의 물리적 위치에 걸쳐 IT자원을 중복해서 구축
   2. 문제 발생 시, 다른 중복된 IT자원이 자동으로 처리(automatic hand-over)
   3. 동일한 클라우드(다른 물리적 위치) 또는 여러 클라우드 간에 IT 자원을 중복해서 배치할 수 있음을 의미함.

**<역할과 경계>**

1. 클라우드 서비스 소비자: 프로그램이 클라우드 서비스에 접근할 때 임시 런타임 부여
2. 클라우드 제공자: 클라우드를 소유(제공)하는 조직
3. 클라우드 소비자: 클라우드 서비스에 접근하여 클라우드 서비스 소비자를 사용하는 조직 or 개인
4. 클라우드 서비스 소유자
   1. 법적으로 클라우드 서비스를 소유하고 있는 개인 혹은 조직
   2. 클라우드 기반으로 또 다른 서비스를 제공하는 것  
      예) Netflix는 AWS를 이용해 서비스  
       (이때, Netflix가 클라우드 서비스 소유자가 된다.)
5. 클라우드 자원 관리자 (클라우드를 아웃소싱으로 관리)
   1. 클라우드 기반 IT자원 (클라우드 서비스도 포함하는) 관리의 책임을 맡고 있는 개인 혹은 조직
   2. 클라우드는 급격히 바뀌기 때문에, 일반적인 IT부서에서 일일이 관리하기 힘듦  
      이러한 클라우드를 관리하는 역할
6. 역할 요약



**<클라우드 전달 모델>**

1. Infrastructure-as-a-Service(IaaS)
   1. OS 관리자 권한 가짐.
   2. 서버 대여
   3. 예) 클라우드 웹 서버
2. Platform-as-a-Service(PaaS)
   1. 미들웨어 관리자 권한 까지만
   2. 미들웨어가 깔려 있는 서버대여
   3. 예) 클라우드 DB
3. Software-as-a-Service(SaaS)
   1. 관리자 계정 X
   2. SW 이용
   3. 예) gmail / dropbox
4. 위 1,2,3번에 관한 총괄적인 특징 및 전형적인 예
   1. 특징: 3 🡪 2 🡪 1 순으로 하위 단계 권한이 높아짐.
   2. 전형적인 예 (Pizza as a Service)

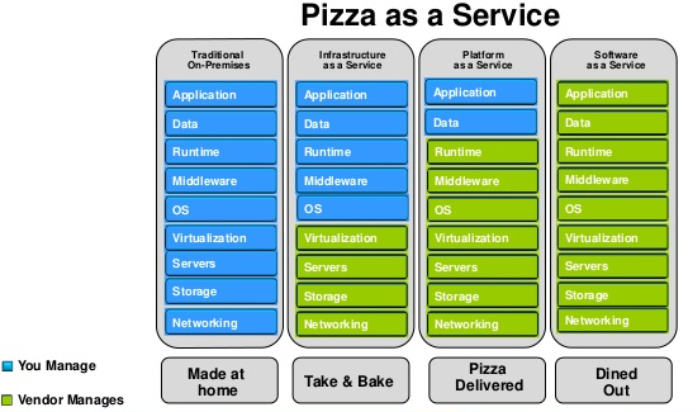


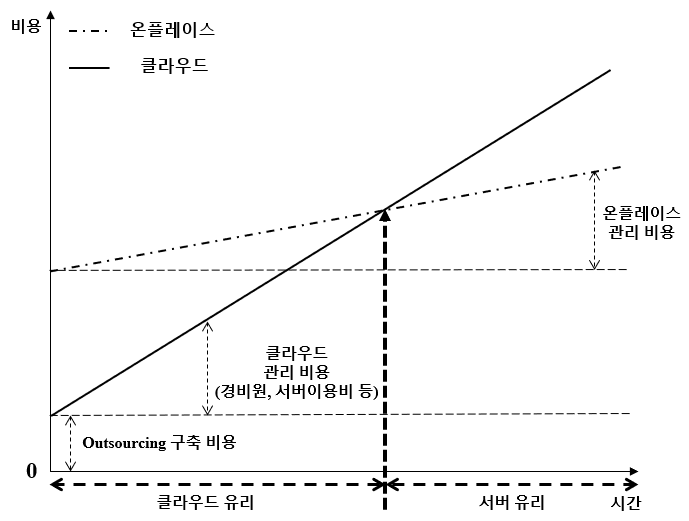
사진 출처URL: <https://cristinaoncloud.com/pizza-as-a-service/>

**<클라우드 배포 모델>**

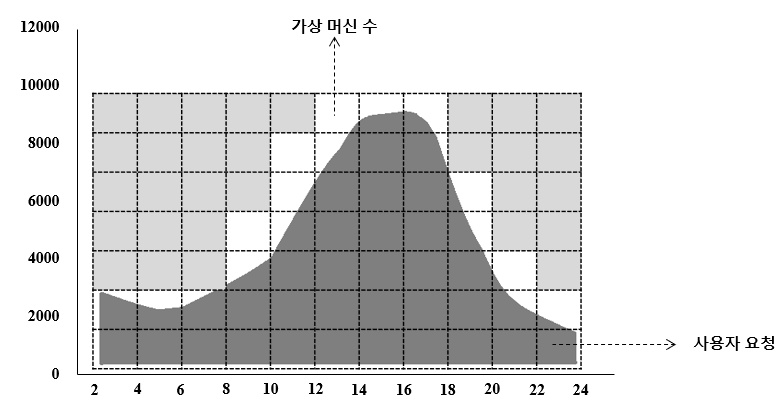
1. 클라우드 배포 모델이란?
   1. 클라우드 환경의 특정한 형태
   2. 소유권, 규모, 접근 방법에 의해 분류됨.
2. 종류
   1. 퍼블릭 클라우드
      1. 남과 같이 씀
      2. 특정회사가 가짐
   2. 커뮤니티 클라우드
      1. 공동 소유
      2. 특정 사람만 접근
   3. 프라이빗 클라우드
      1. 내 PC (내가 소유, 내부적 사용 클라우드)
      2. 장점: 더 많은 관리 권한을 가짐
      3. 단점: 돈이 많이 들어 감 / 관리 복잡
   4. 하이브리드 클라우드
      1. 퍼블릭 + On-premise
      2. 예) 배틀 그라운드 🡪 평소에 자사 서버의 용량 늘면 AWS

**<클라우드 컴퓨팅의 이점>**

1. 투자 절감과 비례 비용
   1. pay-as-you-go 모델 🡪 종량제로 쓴 만큼 지불
   2. 직접적인 비례 비용 🡪 실제 비용이 계산되기에 비용 계산 쉬움
   3. IT 내용 연수: 장비의 기한



1. 확장성(Scalability) 증대
   1. 필요한 대로 사용량 조정
   2. 급격하게 증가되는 사용량을 대비해 On-premise 구축 보다 필요한대로 사용량을 조정하여 사용하는 클라우드가 저 저렴할 수 있음.



**<클라우드 컴퓨팅의 위험>**

1. 들어가기 전
   1. 인터넷을 통해 연결 🡪 VPN 🡪 (종단 암호화)
   2. 다른 사용자의 데이터가 같은 서비스에 있음
2. 운영 관리 제어의 축소
   1. 낮은 관리 제어 수준 할당 받음
   2. 서비스 품질 계약(SLA)와 클라우드 제공자가 만든 다른 보장에 의존할 수밖에 없음.
3. 클라우드 제공자 사이의 제한된 이식 성
   1. 다양한 규모로 특허 받음
   2. 한 클라우드 제공자에서 다른 제공자로 이동하는 것은 어려운 일  
      🡪 SaaS로 올라갈수록 힘듦
4. 다중 영역 규제와 법적 이슈
   1. 물리적 위치에 관한 정보를 가지지 못하는 경우가 많음.
   2. 산업적 혹은 정부 규제와 정책 존재
   3. 규제와 정책을 준수할 수 없을 수도 있음.

**<비즈니스 비용 지표>**

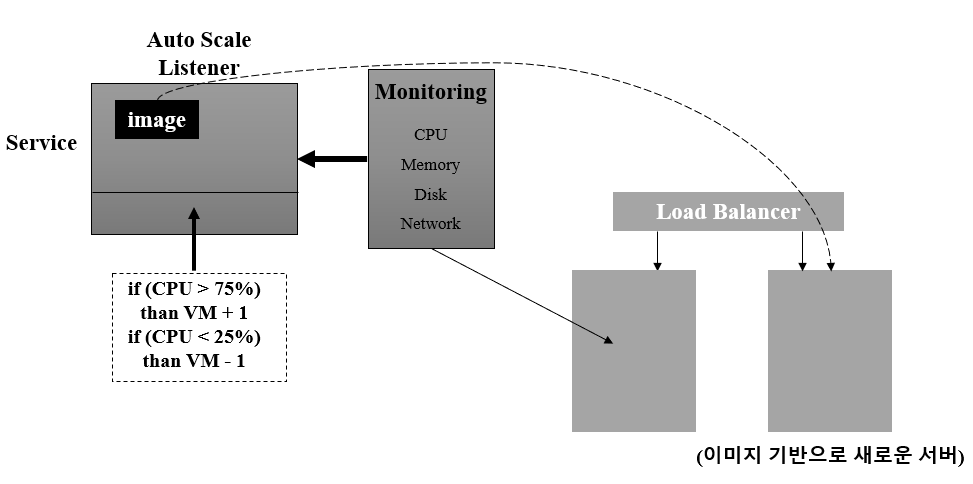
1. 선행 비용: 초기 투자 비용
2. 계속 비용: 유지 관리비
3. 자본 비용: 자본 구매 비용
4. 매몰 비용: 기존에 투자된 비용
5. 통합 비용: 시스템 통합 비용
6. 락인 (Lock-in) 비용: 시스템을 바꾸는 비용

**<서비스 품질 계약(SLA)>**

1. 가용성(Availability)
   1. 정의: IT자원에 접속될 수 있는 확률
   2. 해당 기간 동안 IT자원에 접근할 수 있는 시간의 양을 퍼센트(%)로 표시됨.
   3. 계산법
      1. IT자원이 가용하지 않는 시간(해당 기간)을 해당 기간의 총 시간으로 나눔
      2. i번 산출값에 100을 곱함.
      3. 100에 ii번 산출값을 뺌
   4. 특징: SLA에서 보장하는 가용성 값은 미래 정지 시간 추정에 기반해 있음.
   5. 미래 정지 시간: 정지가 발생할 확률과 IT자원을 재가동시키는데 소요되는 시간을 기준으로 계산
      1. 계산법
         1. 정지시간의 확률과 실제 예상 정지시간을 곱함
         2. i번 산출값에서 해당 가긴 총 시간을 나눔
         3. ii번 산출값에서 100을 곱함
         4. iii번 산출값에서 100을 뺌
2. 신뢰성(Reliability)
   1. 정의: IT자원이 미리 정해진 조건 아래에서 장애없이 목적한 기능을 수행할 수 있는 확률
   2. 일반적인 공식은 성공적으로 목적한 기능이 수행된 횟수를 그 기능을 수행한 총 횟수로 나누어서 얻음.
   3. 결과값을 100으로 곱한 값 🡪 신뢰성 퍼센트(%)

**<클라우드 컴퓨팅 메커니즘>**

1. 자동 확장 리스너(Automated Scaling Listener) ≒ Load Balancer



1. 대체 작동 시스템(Failover System): 죽었는지 체크해서 죽은 것의 복제품 투입
2. 다중 장치 중개자(Multi-Device Broker): 다양한 기술을 사용 가능하도록 자동화된 폼 적용
3. 사용량당 과금 모니터(Pay-Per-Use Monitor): 청구 목적으로 클라우드 소비자에 의해 클라우드 기반 IT자원 사용을 측정하는 서비스 에이전트
4. 상태 관리 데이터베이스(State Management Database): 요청이 오면 여기에 저장하고 여유여 생길 때 다시 꺼내 와 작업 수행

