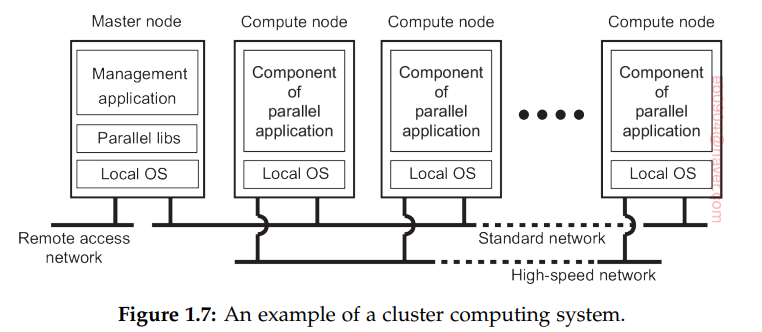
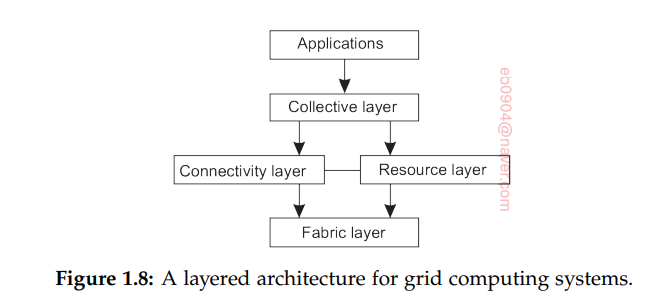
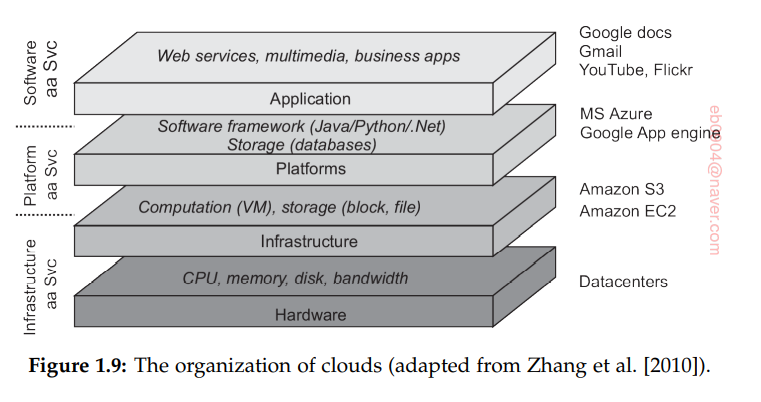
* 1. **Types of distributed systems (분산 시스템의 유형들)**
* distributed computing systems.
* distributed information systems.
* pervasive systems(naturally distributed).
* **High performance distributed computing**
* 분산 시스템은 고성능 컴퓨팅 작업에 사용되는 클래스가 중요하다.
* **cluster computing**에서 기본 하드웨어는 고속 근거리 통신망을 밀접하게 연결된 유사한 workstations 또는 PC의 모음으로 구성된다.
* 각 노드는 동일한 운영체제를 실행한다.
* **grid computing**은 상황이 다르다. 컴퓨터 시스템의 연합으로 구성되는 분산 시스템으로 구성된다.
* 전체 인프라를 아웃소싱한다. (?)
* **cloud computing :** 인프라를 동적으로 구성하고 사용 가능한 서비스에서 필요한 것을 구성할 수 있는 시설을 제공한다.
* 그리드 컴퓨팅과 달리 클라우드 컴퓨팅은 단순히 많은 리소스를 제공하는 것 이상이다.
* **Cluster computing**
* 개인용 컴퓨터와 workstations의 가격 대비 성능이 향상되면서 대중화 되었다.
* 클러스터 컴퓨팅은 단일 프로그램이 여러 시스템에서 병렬로 실행되는 병렬 프로그래밍에 사용된다.
* 클러스터 컴퓨터의 예 : Linux 기반 Beowulf 클러스터로 구성.



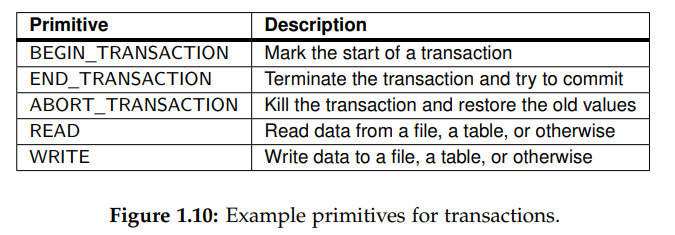
* 마스터는 특정 병렬 프로그램에 대한 노드 할당을 처리하고 제출된 작업의 일괄 처리 queue를 유지하며 시스템 사용자를 위한 인터페이스를 제공한다.
* 마스터는 실제로 프로그램 실행과 클러스터 관리에 필요한 미들웨어를 실행하고,
* 컴퓨팅 노드는 communication, storage, fault tolerance(내결함성) 등의 전형적인 미들웨어 기능을 확장한 표준 운영체제를 탑재한다.
* -따라서 마스터 노드를 제외하고 컴퓨팅 노드는 매우 동일하다.
* 클러스터 컴퓨터는 단일 컴퓨터처럼 나타나서 프로세스에 대한 궁극적인 distribution transparency(분산 투명성)을 제공한다.
* **Grid computing**
* 기존 cluster computing의 특징은 homogeneity(동종성)이다.
* 대부분의 cluster의 컴퓨터는 대체로 동일하고 운영체제가 동일하며, 모두 동일한 네트워크를 통해 연결된다.
* Grid computing system의 핵심 문제는 서로 다른 조직의 리소스가 함께 모여 서로 다른 조직의 사람들 그룹이 협업할 수 있도록 하여 실제로 시스템 연합을 형성한다는 것이다.
* 일반적으로 리소스는 compute servers, storage facilities, and databases.
* Given its nature, much of the software for realizing grid computing evolves around providing access to resources from different administrative domains, and to only those users and applications that belong to a specific virtual organization. (그 특성을 감안할 때 그리드 컴퓨팅을 구현하기 위해 많은 sw는 다양한 관리 도메인의 리소스에 대한 액세스를 제공하고, 특정 가상 조직에 속한 사용자와 애플리케이션에만 액세스할 수 있도록 진화한다?)
* 이러한 이유의 초점은 아키텍처 문제.



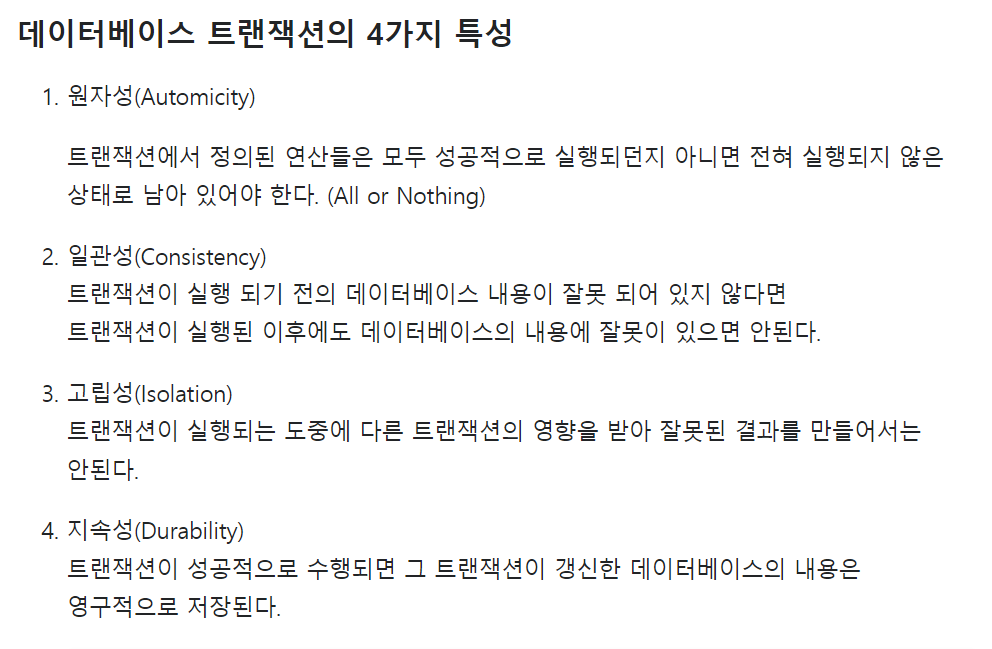
* connectivity layer : 사용자로부터 프로그램에 권함 위임, 인증.
* resource layer : connectivity layer에서 제공하는 기능을 사용하고, fabric layer에서 사용할 수 있는 인터페이스 직접 호출.
* collective layer : 여러 리소스에 대한 액세스 처리(resource discovery, allocation and scheduling, data replication,)
* application layer : virtual organization 내에서 작동하고, 그리드 컴퓨팅 환경을 활용.
* **Cloud computing**
* 클라우드 컴퓨팅은 쉽게 사용 가능하고 액세스 가능한 가상화된 리소스 풀이 특징이다.
* 어떤 리소스가 어떻게 사용되는지를 동적으로 구성하여 확장성을 위한 기반을 제 공할 수 있다.

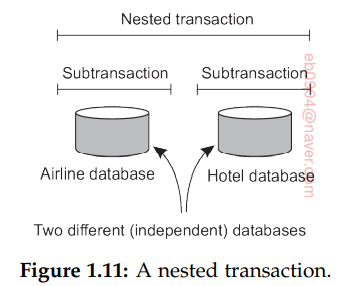


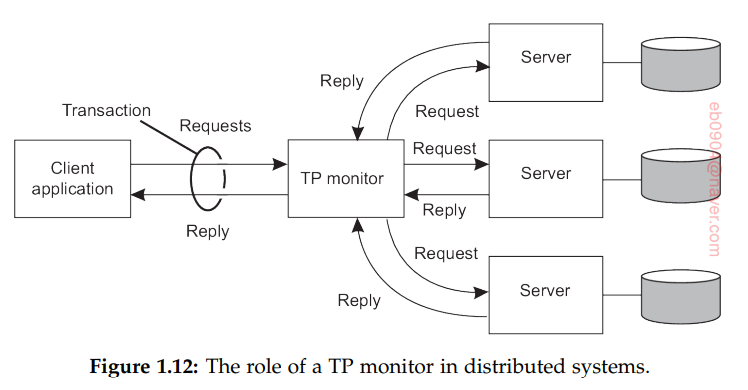
* Hardware : 가장 낮은 계층은 프로세서, 라우터, 전력 및 냉각 시스템과 같은 필요한 하드웨어를 관리하는 수단에 의해 형성. 일반적으로 데이터 센터에서 구현되며 일반적으로 고객이 직 접 볼 수 없는 리소스를 포함.
* Infrastructure : virtualization 스토리지 및 컴퓨팅 리소스로 구성된 인프라를 고객에게 제공하기 위해 가상화 기술을 배포한다. 클라우드 컴퓨팅은 가상 저장 장치와 가상 서버를 할당하고 관리하는 방향으로 발전.
* Platform : 플랫폼 계층은 운영 체제가 애플리케이션 개발자에게 제공하는 것.
* Application : 실제 응용 프로그램은 이 계층에서 실행되며 추가 사용자 지정을 위해 사용자에게 제공됨.
* three different types of services :
  + Infrastructure-as-a-Service (IaaS) : 하드웨어 및 인프라 계층 포함
  + Platform-as-a-Service (PaaS) : 플랫폼 계층 포함
  + Software-as-a-Service (SaaS) : 해당 애플리케이션이 포함
* **Distributed information systems**
* 풍부한 네트워크 애플리케이션을 직면했지만 상호 운용성(interoperability)은 문제.
* 기존 미들웨어 솔루션의 대부분은 애플리케이션을 정보 시스템에 더 쉽게 통합할 수 있는 인프라로 작업했다.
* 클라이언트는 특정 작업을 실행하기 위해 서버에 요청을 보낸 후 응답을 다시 보낸다.
* 가장 낮은 수준의 통합을 통해 클라이언트는 가능한 다른 서버에 대한 여러 요청을 하나의 더 큰 요청으로 래핑하고 분산 트랜잭션으로 실행할 수 있다.
* 여기서 핵심은 모든 요청이 실행되거나 실행되지 않는다는 것이다.
* 응용 프로그램이 독립적인 구성 요소로 분리됨에 따라 서로 직접 통신할 수 있도록 하여 통합도 이루어져야 한다. 🡪 EAI(Enterprise Application Integration).
* **Distributed transaction processing**
* 데이터베이스에 대한 작업은 트랜잭션의 형태로 수행된다.
* 트랜잭션을 사용하려면 기본 분산 시스템이나 언어 런타임 시스템에서 제공해야 하는 기본 요소가 필요하다.

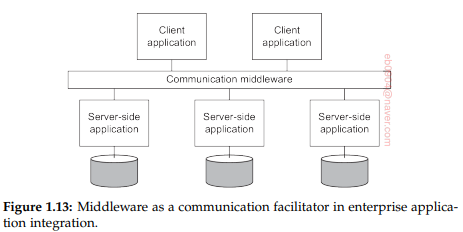


* 트랜잭션의 네가지 특성 :





* 중첩된 트랜잭션을 형성하는 여러 하위 트랜잭션으로 구성된다.
* 중첩된 트랜잭션은 여러 시스템에 트랜잭션을 분산하는 방법을 제공하기 때문에 분산 시스템에서 중요하다. Ex) 항공편 예약
* 
* **Enterprise application integration**
* 애플리케이션 구성 요소는 트랜잭션 처리 시스템에서 지원하는 요청/응답 동작을 통해서가 아니라 서로 직접 통신할 수 있어야 한다.



* 여러 유형의 통신 미들웨어가 존재한다.
* **RPC(remote procedure calls : 원격 프로시저 호출)** 를 사용하면 응용 프로그램 구성 요소가 로컬 프로시저 호출을 수행하여 다른 응용 프로그램 구성 요소에 요청을 보낼 수 있다.
* 결과가 다시 전송되어 프로시저 호출의 결과로 응용 프로그램에 반환된다.
* **RMI(remote method invocations : 원격 메서드 호출)**은 기능 대신 객체에서 작동한다는 점을 제외하고는 RPC와 동일하다.
* 원격 개체에 대한 호출을 허용하는 기술.
* RPC와 RMI는 호출자와 호출 수신자 모두 통신 시 작동하고 실행해야 하는 단점이 있다. 서로를 지칭하는 방법을 정확히 알아야 한다.
* 이러한 것은 문제점이 될 수 있고, message-oriented middleware(메시지 지향 미들웨어) 또는 단순히 MOM으로 알려진 문제로 이어졌다.
* . Likewise, applications can indicate their interest for a specific type of message, after which the communication middleware will take care that those messages are delivered to those applications. (특정 유형의 메시지에 대한 관심을 표시할 수 있고, 그 후에 통신 미들웨어는 해당 메시지가 해당 애플리케이션에 전달되도록 처리한다.) (?)
* 이러한 **publish-subscribe** 시스템은 중요하고 확장되는 분산 시스템 클래스를 형성한다.
* **Pervasive systems (?)**
* 지금까지 논의된 분산 시스템은 주로 stability(안정성)이 특징이다.
* 모바일 및 임베디드 컴퓨팅 장치의 도입 이후 문제가 변경되어 pervasive systems이 나왔다.(?)
* **Ubiquitous computing systems**
* 유비쿼터스 컴퓨팅 시스템에서 시스템이 널리 퍼져 있고 지속적으로 존재한다.
* 사용자가 시스템과 지속적으로 상호작용할 것임을 의미.

1. **Distribution** : 장치가 네트워크로 연결되고 분산되며 투명한 방식으로 액세스할 수 있다.
   1. 유비쿼터스 컴퓨팅 시스템은 분산 시스템의 한 예이다.
   2. 시스템의 노드를 구성하는 장치와 기타 컴퓨터는 단순히 네트워크로 연결되고 함께 작동하여 단일한 일관성 있는 시스템을 형성한다.
   3. 사용자 가까이에 있는 장치가 컴퓨터에 연결되어 보이지 않고, 원격으로 클라우드에서 작동할 수도 있다.
2. **Interaction** : 사용자와 장치 간의 상호작용은 거슬리지 않는다.(?)
   1. 사용자와의 상호 작용과 관련하여 유비쿼터스 컴퓨팅 시스템은 지금까지 논의한 시스템과 많이 다르다.
   2. 최종 사용자는 유비쿼터스 시스템 설계에서 중요한 역할을 한다.
   3. 사용자와 핵심 시스템 간의 상호 작용이 발생하는 방식에 특별한 주의를 기울여야 한다.
   4. 사용자는 입력이 컴퓨터 시스템에 제공되고 있다는 사실을 대부분 모를 수 있다.
3. **Context awareness** : 시스템은 상호 작용을 최적화하기 위해 사용자의 컨텍스트를 인식한다.
   1. 유비쿼터스 컴퓨팅 시스템이 수행해야 하는 작업은 상호 작용이 발생하는 컨텍스트를 고려하는 것이다.
   2. 분산 시스템 관점에서 다양한 센서에 의해 수집된 데이터가 애플리케이션에서 사용할 수 있는 추상화 수준으로 보여지는 것이다. Ex) GPS 좌표.
4. **Autonomy(자율성)** : 장치는 사람의 개입 없이 자율적으로 작동하므로 높게 자체 관리된다.
   1. 대부분의 유비쿼터스 컴퓨팅 시스템의 중요한 측면은 명시적인 시스템 관리가 최소한으로 축소되었다는 것이다.
   2. 시스템 전체가 자율적으로 작동하고 변경 사항에 자동으로 대응할 수 있어야 한다.
      1. **Address allocation:** 네트워크로 연결된 장치가 통신하려면 IP 주소가 필요함.
      2. **Adding devices :** 기존 시스템에 장치를 쉽게 추가할 수 있어야 함.
      3. **Automatic updates :** 유비쿼터스 컴퓨팅 시스템의 많은 장치는 sw 업데이트가 필요한지 여부를 인터넷을 통해 정기적으로 확인할 수 있어야 함.
5. **Intelligence :** 시스템은 전체적으로 광범위한 동적 동작과 상호 작용을 처리할 수 있다.
   1. 유비쿼터스 컴퓨팅 시스템이 종종 인공 지능 분야의 방법과 기술을 사용한다고 언급한다. 🡪 불완전한 입력을 처리하고 변화하는 환경에 신속하게 대응하며 예기치 않은 이벤트를 처리하는 등 다양한 고급 알고리즘과 모델을 배포해야 한다는 뜻.
   2. 인공 지능 분야의 많은 문제에 대한 분산 솔루션은 아직 발견되지 않았다. 🡪 네트워크 및 분산 장치의 첫번째 요구 사항과 고급 분산 정보 처리 사이에 긴장이 있을 수 있다.(-)

* **Mobile computing systems**
* Mobility은 pervasive systems의 중요한 구성 요소를 형성하는 경우가 많다.