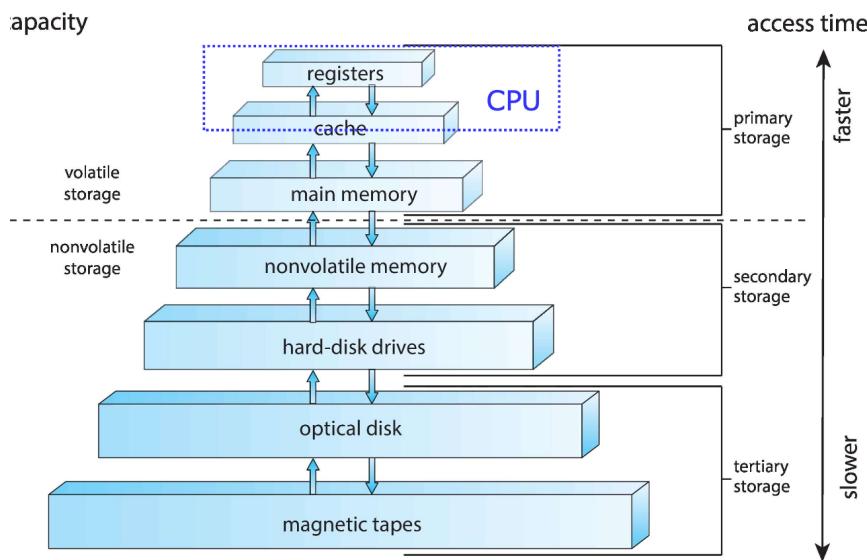


CH 01 - 2 [ch1 12(P.72~130)]

Storage Hierarchy



- 저장 시스템은 계층에 의해 조직화되어 있다.
 - 속도, 비용, 휘발성, 크기
- 캐싱 - 더 빠른 저장 시스템으로 정보를 복사; 메인 메모리는 세컨더리 저장 공간을 위한 캐시로 볼 수 있다.

Storage Structure

메인 메모리 - CPU가 직접적으로 접근할 수 있는 유일한 저장장치

- 랜덤 액세스, 일반적으로 휘발성
- 일반적으로 DRAM(Dynamic Random-access Memory) 형태의 랜덤 액세스 메모리
- 랜덤액세스 : 데이터를 저장하는 블록을 한번에 여러 개 액세스하는 것이 아니라 한 번에 하나의 블록만을 액세스하는 방식

보조기억장치 - 큰 비휘발성 저장 공간을 제공

하드디스크 드라이브(HDD) - 자기 기록 재료로 덮인 단단한 금속 또는 유리 플레이터

- 디스크는 보통 track 단위로 나뉘며, sector 형태로 한 번 더 나뉜다.

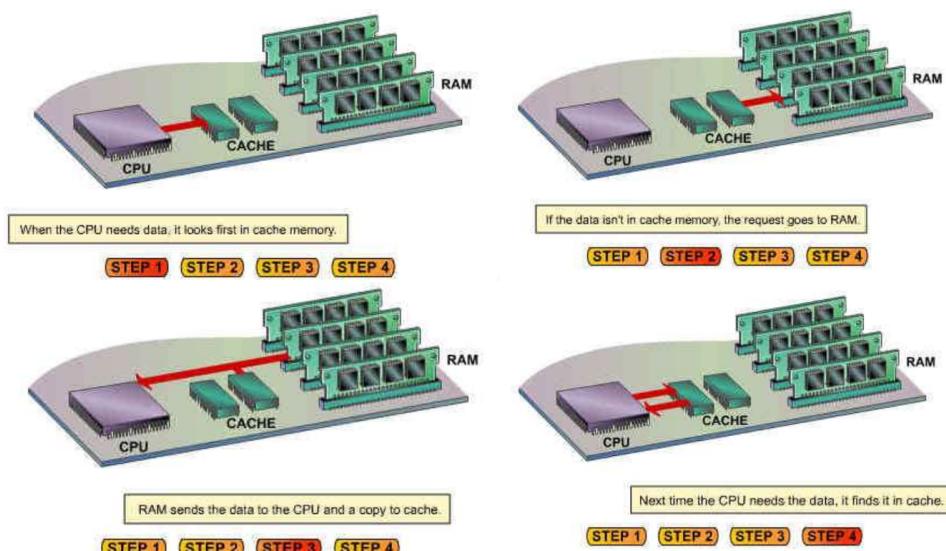
- 디스크 컨트롤러는 디바이스와 컴퓨터 간의 논리적인 교류점을 제공한다.

비휘발성 메모리(NVM) 디바이스

- 하드디스크보다 빠르고 비휘발성
 - 다양한 기술들, 수용력과 성능은 향상되고 가격은 떨어지면서 더 잘 쓰이게 됨

Caching

- 컴퓨터의 여러 수준에서 수행되는 중요한 원리(하드웨어, 운영 체제, 소프트웨어)
- 사용 중인 정보가 더 느린 스토리지에서 더 빠른 스토리지로 일시적으로 복사됨
 - 자주 사용되는 정보만 복사
- 정보가 있는지 확인하기 위해 먼저 빠른 스토리지(캐시) 확인, 만약 있으면 캐시에서 가져오고 없다면 데이터를 캐시로 복사해 온 후 사용



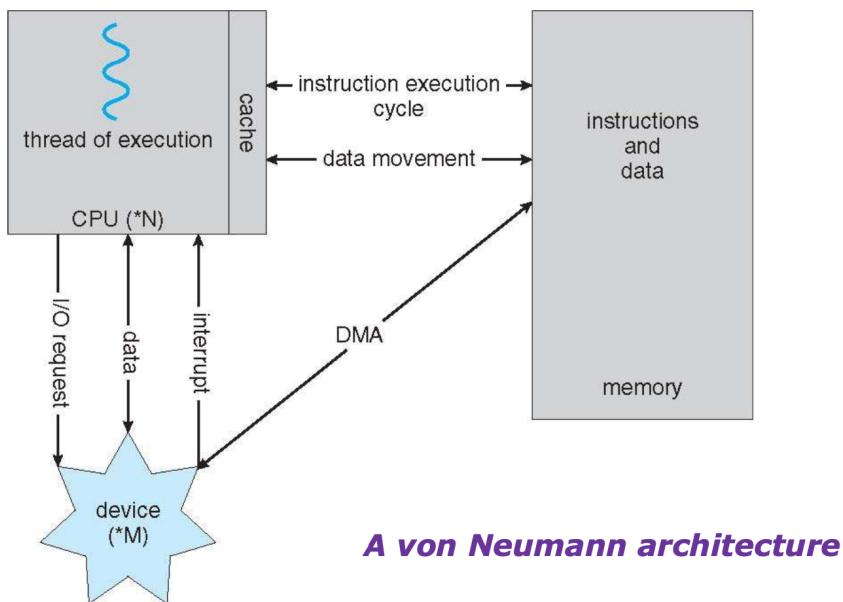
- 만 CPU가 데이터가 필요한 경우 먼저 캐시 메모리를 확인, 존재한다면 데이터를 사용
- 없다면 RAM에 요청을 보냄. RAM이 데이터를 CPU에게 보내고 캐시에 복사해옴
- 다음 CPU가 해당 데이터가 필요할 때, 캐시에서 찾아서 가져옴

Computer System Architecture

- 대부분의 시스템은 하나의 범용 프로세서를 사용, 추가적으로 특수 목적의 프로세서들을 가짐
- 멀티 프로세서(parallel system(병렬 시스템), tightly-coupled system)
 - 처리량 증가, 경제성, 신뢰성 향상 (적절한 결합 혹은 내결합성)

- Two Types
 - 비대칭형 멀티 프로세싱(Asymmetric Multiprocessing) : 각 프로세서가 특정 작업에 할당
 - 각각의 프로세서의 각각의 역할이 존재하며, 마스터(Master) 프로세서와 노예(Slave) 프로세서로 나뉜다. 마스터 프로세서인 CPU가 slave 프로세서에게 일을 할당하는 방식이다.
 - 대칭형 멀티 프로세싱(Symmetric Multiprocessing) : 각 프로세서가 모든 작업을 수행
 - 각각의 프로세서가 동일한 운영체제 사본을 가지고 수행한다. 또한 많은 프로세서가 퍼포먼스가 나빠지지 않은 상태로 동시에 수행된다. 많은 OS에서 이러한 방식을 채택한다.

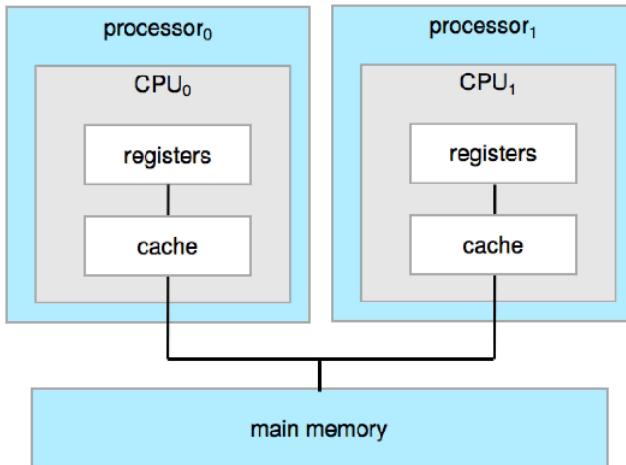
How a Modern Computer Works : 폰노이만 구조



- 프로그램을 메모리에 저장하는 구조

Symmetric Multiprocessing Architecture

UMA(Uniform Memory Access)



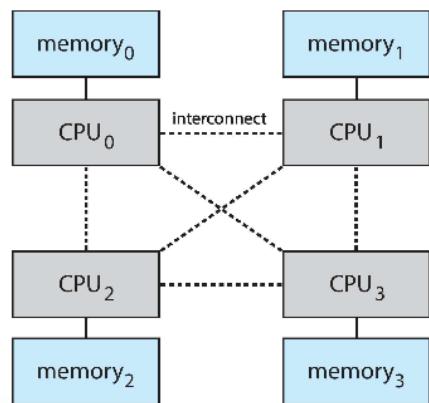
- 두 개 이상의 프로세서가 한 개의 공유 메모리를 사용하는 다중 프로세서 컴퓨터 구조이다, 각 프로세서에는 개별 레지스터 또는 캐시가 존재한다.
- 메인 메모리는 시스템 버스를 통해 공유하며 한 번에 한 개의 프로세서만이 메모리에 접근할 수 있다.

장점 : 많은 프로세스를 동시에 실행할 수 있다.

단점 : CPU가 독립적이기 때문에 비효율적일 수 있다. 프로세서가 특정 자료구조를 공유한다면 비효율성을 비할 수 있다.

→ CPU를 너무 많이 추가하면 시스템 버스에서 병목 현상이 일어나 성능이 저하된다. 이를 해결하기 위해 각각의 CPU에 자체 메모리를 존재하게 하는 NUMA가 고려되었다.

NUMA(Non-Uniform Memory Access)



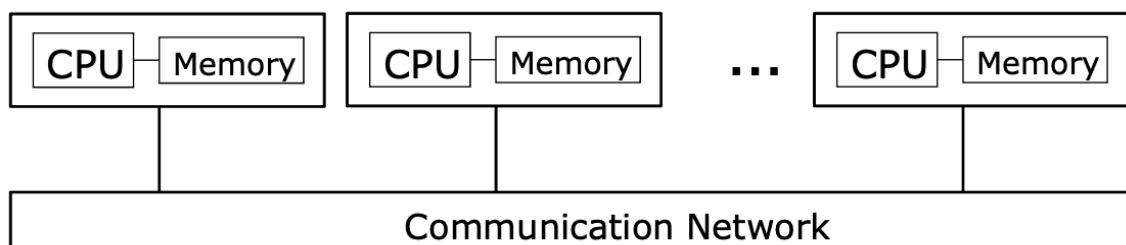
- NUMA는 모든 CPU가 공유 시스템으로 연결되어있으면서 각각의 CPU가 로컬 메모리를 가지는 컴퓨터 아키텍처이다.

장점 : 시스템 상호 연결에 대한 경합이 없다.

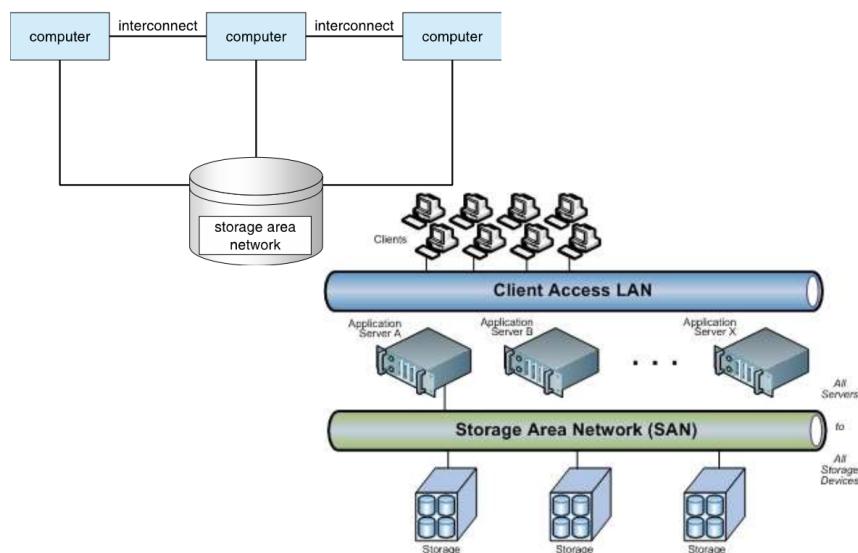
단점 : CPU가 원격 메모리에 액세스해야 할 때 지연 시간이 존재한다. 자체 로컬 메모리에는 빠르게 접근할 수 있으나 다른 CPU의 메모리에 접근할 때는 성능이 저하된다

Loosely Coupled Systems

- 분산 시스템
 - 공유 메모리가 없다.
 - 커뮤니케이션 네트워크를 통해 메세지를 주고받음으로써 의사소통
 - 각각의 프로세서는 각자의 메모리를 가지고 있다. (병렬 시스템은 메모리를 공유)
 - 속도가 빠르며 신뢰성이 높고 Communication을 하는 것이 특징이다.

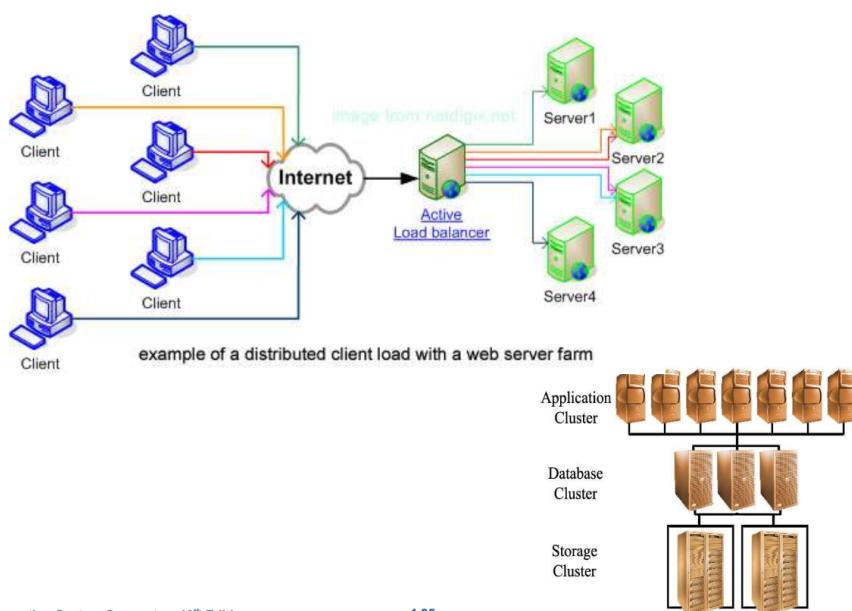


Clustered Systems



- 네트워크에 있는 여러 컴퓨터들을 병렬적으로 거대한 시스템으로 만드는 것.
 - 비동기식 클러스터링(Asymmetric clustering) : 다른 서버들이 대기하는 동안 하나의 서버가 응용 프로그램을 실행하는 구조

- 동기식 클러스터링(Symmetric clustering) : 모든 N개의 호스트 서버들이 응용 프로그램을 실행하는 구조
- 클러스터형은 storage-area network(SAN)을 통하여 스토리지를 공유한다.
- 클러스터는 만약 응용프로그램이 병렬처리가 가능하도록 설계 되었다면 다수의 컴퓨터 시스템을 네트워크로 연결하여 구성하기 때문에 고성능 컴퓨팅 환경(HPC, high-performance computing)을 제공할 수 있다.
- 클러스터는 여러 호스트가 공유 저장장치에 저장된 동일한 데이터를 접근할 수 있게되고 일부 클러스터는 접근 간의 충돌이 발생하지 않는 것을 보장하기 위한 접근 제어와 잠금 기법을 제공해야 하는데 이 기법을 분산 잠금 관리자라고 한다.



Multiprogramming과 Multitasking

운영체제는 컴퓨터 시스템의 자원을 효율적으로 사용할 수 있도록 한다. 여러 프로그램들을 동시에 실행했을 때, CPU 가 각 프로그램들을 끊겨다니며 효율적으로 실행하는 것도 이 중 하나다. 운영체제가 여러 프로그램을 실행시키는 경우 **멀티 프로그래밍**과 **시분할 시스템** 방식이 존재한다.

[멀티 프로그래밍]

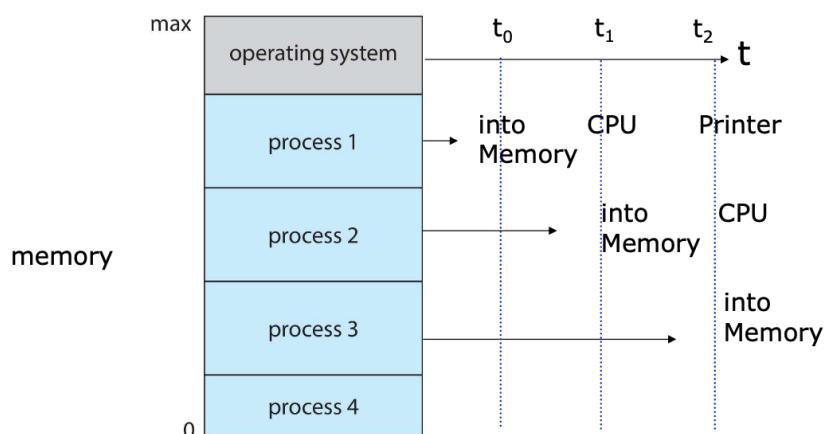
- **CPU의 활용도를 최대화해서 여러 프로그램이 동시에 실행되는 것**
- 멀티 프로그래밍은 하나의 프로그래밍이 다른 작업을 수행하는 동안에도 CPU는 다른 프로그램을 실행하여, CPU의 활용도를 최대화하고 프로그램들의 실행 시간을 겹쳐서 단축시킬 수 있다.

- 한 유저는 CPU와 입출력장치를 쉬지 않고 동작시킬 수 없다. 멀티 프로그래밍은 작업(코드와 데이터)를 분배해서 CPU가 항상 일을 하도록 만든다. 시스템의 전체 작업은 메모리에 저장된다. 하나의 일은 선택되고 job scheduling 동안 실행된다. 입출력 작업 등으로 인해 대기해야 할 경우 운영체제는 다른 작업으로 전환한다.

[멀티태스킹]

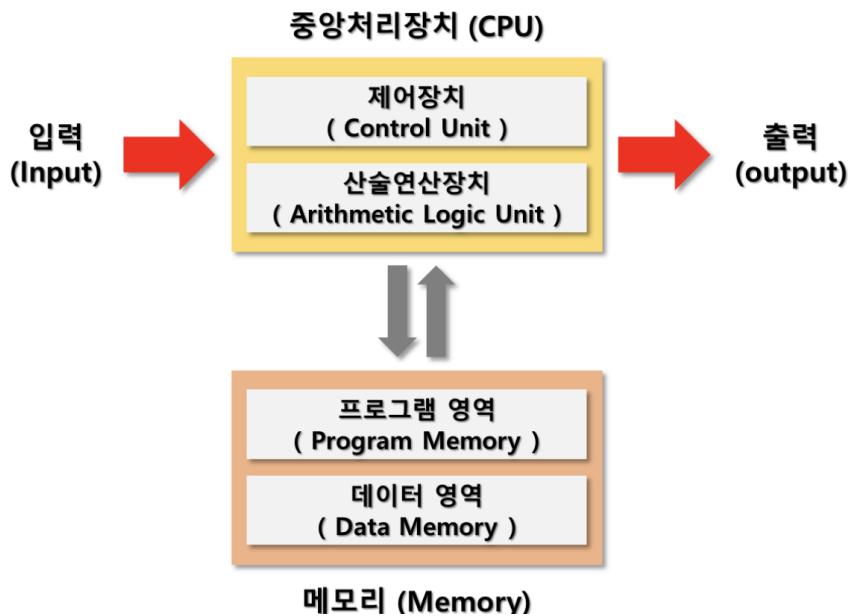
- 멀티 태스킹은 다수의 작업(Task)을 운영체제의 스케줄링에 의해 번갈아 가며 수행되도록 해주는 것을 의미
- CPU가 작동 중에 사용자가 각 작업과 상호 작용할 수 있을 정도로 자주 작업을 전환하여 대화형 컴퓨팅을 만드는 논리적 확장
- 응답 시간은 1초 미만이어야 하며, 각 사용자는 적어도 메모리에서 실행되는 하나의 프로그램을 가져야 한다. → 프로세스
- 만약 여러 개의 작업이 실행될 준비가 되었다면 CPU Scheduling을 하고, 프로세스가 메모리에 맞지 않으면 swapping을 통해 프로세스를 안팎으로 이동하여 실행
- 가상 메모리를 사용하면 메모리에 완전히 저장되지 않은 프로세스를 실행할 수 있다.

- Several jobs are kept in main memory at the same time, and the CPU is multiplexed among them.



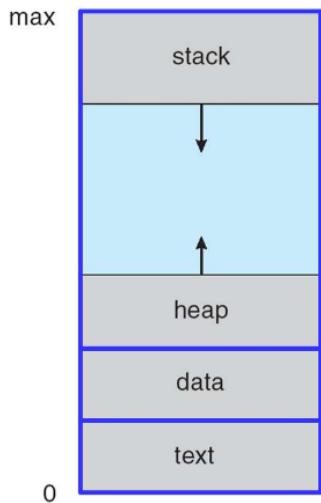
Stored Program Concept - Von Neumann 구조

폰 노이만 구조



- 폰 노이만 구조는 중앙 처리장치, 메모리, 프로그램 세 가지 요소로 구성된다. 이 구조에서 CPU와 메모리는 분리되어있으며 버스를 통해 데이터와 명령어를 주고받는다. 이러한 분리된 구조는 명령어와 데이터를 동시에 접근할 수 없게 한다.
- processing unit에선 ALU와 프로세서 레지스터들을 가진다. 제어 유닛은 명령어 레지스터와 프로그램 카운터(pc, 명령어 포인터)를 가진다.
- 폰 노이만 구조의 가장 큰 장점은 내장 프로그램이라는 것인데, 컴퓨터에 덧셈 뺄셈과 같은 산술 연산을 소프트웨어에 내장 시켜두었기 때문에 사용자가 하드웨어에 일일이 접근하지 않아도 된다.
- 본 구조에서 CPU는 메모리에 저장된 명령어를 가져와(fetch), 명령어를 해석하고(decode), 실행(execute)하고 결과를 저장(store) 한다.

저장 프로그램 개념



- 프로그램(Text OR Code section)을 메모리에 저장한다는 개념.
- 데이터 섹션은 전역 변수와 static 변수를 포함한다.
- 스택은 일시적인 데이터를 저장한다. (함수 매개변수, 반환 주소, 지역 변수)
- 힙은 실행시간 동안 동적으로 할당된 메모리 영역이다.

Major Components of Operating Systems

Process 관리

- 프로세스는 실행 중인 프로그램이다. 프로그램은 수동적인, 프로세스는 능동적인 객체이다.
- 프로세스는 작업을 수행하기 위해 자원(CPU 시간, 메모리, 파일들, 입출력 장치들과 초기 데이터)이 필요하다.
- 프로세스를 종료하기 위해선 재사용 가능한 자원을 회수해야 한다.
- 단일 스레드 프로세스는 다음 실행할 명령어의 위치를 특정하는 program counter를 하나만 가지고 있다. 프로세스는 한 번에 한 개의 명령어를 순차적으로 실행한다.
- 멀티 스레드 프로세스는 스레드 당 하나의 프로그램 카운터가 있다.
- 일반적으로 시스템은 많은 프로세스, 몇몇의 사용자, 몇 개의 OS를 하나 혹은 그 이상의 CPU에서 동시에 실행한다.

Process Management Activities

- OS는 프로세스 관리를 위해 다음과 같은 활동을 해야 할 책임이 있다.

- 사용자와 시스템 프로세스들을 생성하거나 삭제
 - 프로세스 일시 중단 및 재개
 - 프로세스 동기화를 위한 메커니즘 제공
 - 프로세스 의사소통을 위한 메커니즘 제공
 - 데드록 핸들링을 위한 메커니즘 제공
-

Memory Management

- 프로그램을 실행시키기 위해서는 메모리에 모든 명령어/데이터가 있어야 한다.
 - 메모리 관리는 무엇을, 언제 메모리에 넣을지 결정한다.
 - cpu 활용성을 최적화하고 사용자에게 컴퓨터 응답을 제공한다.
 - 메모리 관리 활동
 - 현재 메모리의 어떤 부분이 누구에 의해 사용되고 있는지 추적하는 것
 - 메모리 내 및 메모리 외로 이동할 프로세스(또는 그 일부) 및 데이터 결정
 - 필요에 따라 메모리 공간 할당 및 할당 해제
-

File-System Management

- OS는 information storage에 대한 균일하고 논리적인 뷰를 제공
 - 물리적인 개체를 논리적인 저장 유닛으로 추상화 한 것 → 파일
 - 각 매체는 장치(즉, 디스크 드라이브, 테이프 드라이브)에 의해 제어
 - 다양한 속성에는 액세스 속도, 용량, 데이터 전송 속도, 액세스 방법(순차 또는 랜덤)이 포함
- 파일 시스템 관리
 - 파일은 directory에 의해 조직된다.
 - 대부분의 시스템에서의 권한은 누가 무엇을 볼 수 있는가를 결정
 - OS 활동은 다음을 포함한다.
 - 파일과 디렉토리를 생성하고 삭제
 - 파일 및 디렉토리를 조작
 - 파일을 보조기억장치에 맵핑
 - 파일을 비휘발성 저장 매체에 안전하게 백업

Mass-Storage Management(대용량 저장장치)

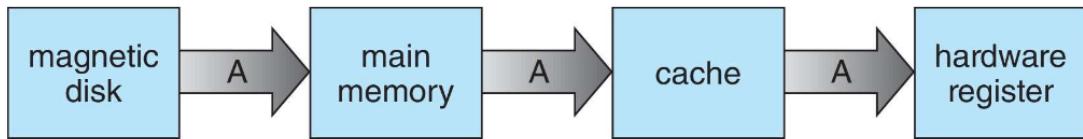
- 대부분 디스크는 메인 메모리에 맞지 않거나 데이터가 긴 기간동안 저장되어야 할 때 사용된다.
- 전체 컴퓨터 작동 속도는 디스크 서브시스템과 그 알고리즘에 달려있다.
- 운영체제 활동
 - mounting과 unmounting
 - mounting : 특정 위치로 이들 장치들이 연결되도록 하는 과정
 - free-space management
 - 저장공간 할당
 - 디스크 스케줄링
 - partitioning
 - 보호
- 이러한 저장공간은 빠를 필요가 없다.

다양한 타입의 저장공간들의 특성

Level	1	2	3	4	5
Name	registers	cache	main memory	solid-state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25-0.5	0.5-25	80-250	25,000-50,000	5,000,000
Bandwidth (MB/sec)	20,000-100,000	5,000-10,000	1,000-5,000	500	20-150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

데이터 'A'의 디스크에서 레지스터로의 이동

- 멀티테스킹 환경은 반드시 최근 데이터를 조심히 다뤄야 한다. (해당 데이터가 어느 storage 계층에 저장되어있든 상관 X)



- 멀티프로세서 환경은 반드시 CPU가 캐시의 최근 값을 갖도록 하드웨어에서 캐시 일관성을 제공해야 한다.
- 분산된 환경은 더 복잡하게 작동한다. (다양한 데이터의 복사본이 존재할 수 있다.)

I/O SubSystem

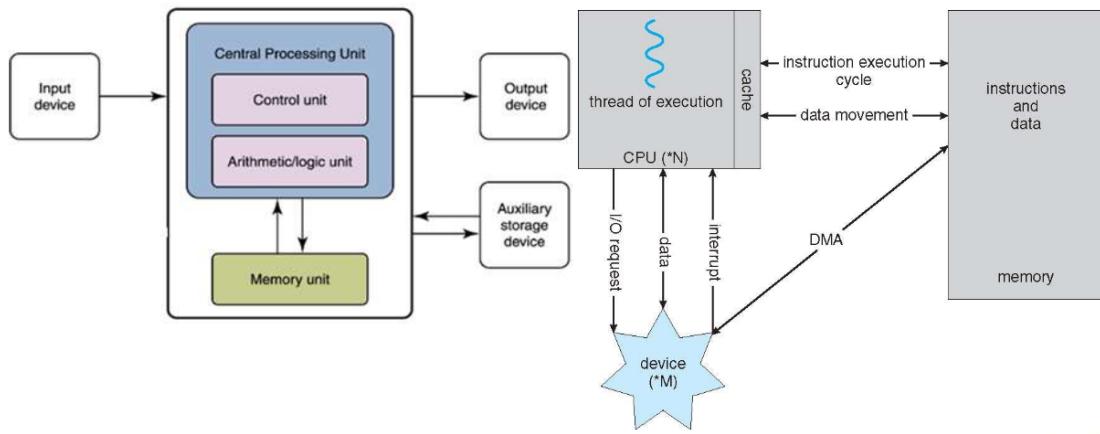
- OS의 한 가지 목적은 사용자로부터 하드웨어 장치의 특성을 숨기는 것이다.
- I/O Subsystem은 다음과 같은 책임이 있다.
 - buffering, caching, spooling을 포함하여 입출력의 메모리 관리를 할 책임
 - 일반적인 디바이스 드라이버 인터페이스를 제공
 - 특정 하드웨어 디바이스를 위한 드라이버를 제공

Protection and Security

- **Protection** : OS에 의해 정의된 자원에 대한 프로세스 또는 사용자의 접근을 제어하기 위한 임의의 메커니즘
- **Security** : 내부 및 외부 공격에 대한 시스템 방어
 - 시스템 사용자의 인증을 보장하여 시스템에 저장된 정보(데이터 및 코드 모두)와 컴퓨터 시스템의 물리적 리소스의 무결성을 보호
 - 무단 액세스, 악의적인 데이터 파괴 또는 변경, 불일치의 우발적 도입 방지
 - 서비스 거부, 웹, 바이러스, 신원 도용, 서비스 도용을 포함한 광범위한 범위 보호
- 시스템은 일반적으로 사용자를 잘 구별해내야 한다.
 - 사용자 인증 (user id, security id) / 그룹 인증 등

Emerging Trends in Computer-System Architecture

Von Neumann Architecture



- 대부분의 시스템은 하나의 범용 프로세서를 사용한다. 그리고 여러 개의 특수 목적 프로세서를 가지고 있다.

최신 동향은 그냥 확인하면 될 듯 (굳이 정리..? 필요없을듯..)

Types of Computing Environments

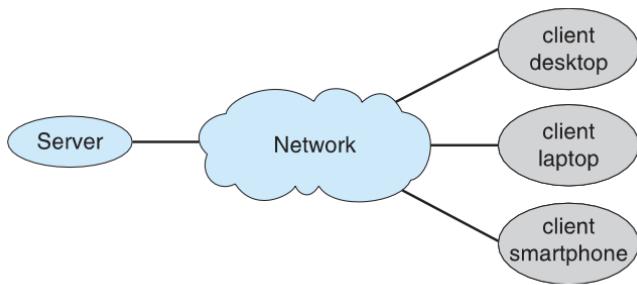
전통적인 컴퓨팅 환경

- 독립형 범용 기계
- '네트워크'의 등장 기점으로 변화했다고 보면 됨.
- 포털은 내부 시스템에 대한 웹 액세스를 제공하고, 네트워크 컴퓨터는 웹 터미널과 같다. 모바일 컴퓨터는 무선 네트워크를 통해 상호 연결한다.
- 네트워킹의 보편화 - 심지어 가정용 시스템도 방화벽을 사용하여 가정용 컴퓨터를 인터넷 공격으로부터 보호

모바일 컴퓨팅 환경

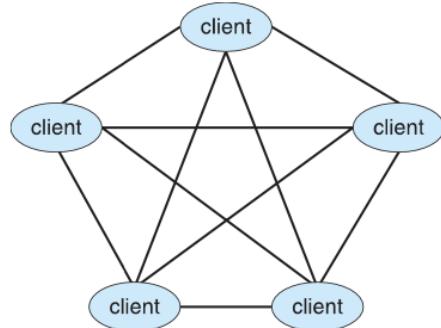
- 손으로 잡을 수 있는 스마트폰, 태블릿 등

Client-Server 컴퓨팅 환경



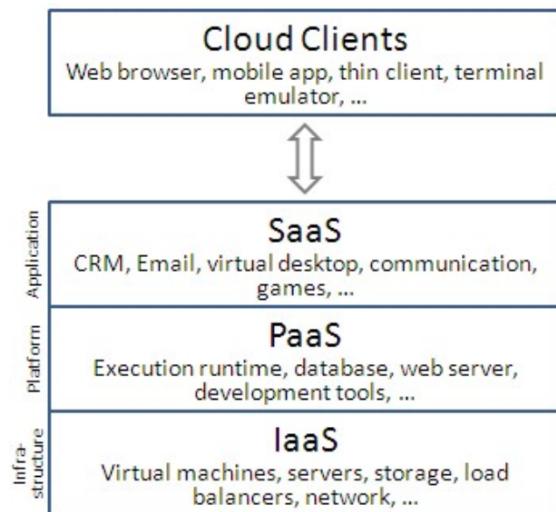
- 스마트 PC로 대체되는 Dumb 단말기
- 현재 많은 시스템이 서버를 설치하여 클라이언트가 생성한 요청에 응답
 - Compute-server system은 클라이언트에게 서비스(즉, 데이터베이스)를 요청할 수 있는 인터페이스를 제공
 - File-server system은 클라이언트가 파일을 저장하고 검색할 수 있는 인터페이스를 제공

Peer-to-Peer(P2P) 컴퓨팅 환경



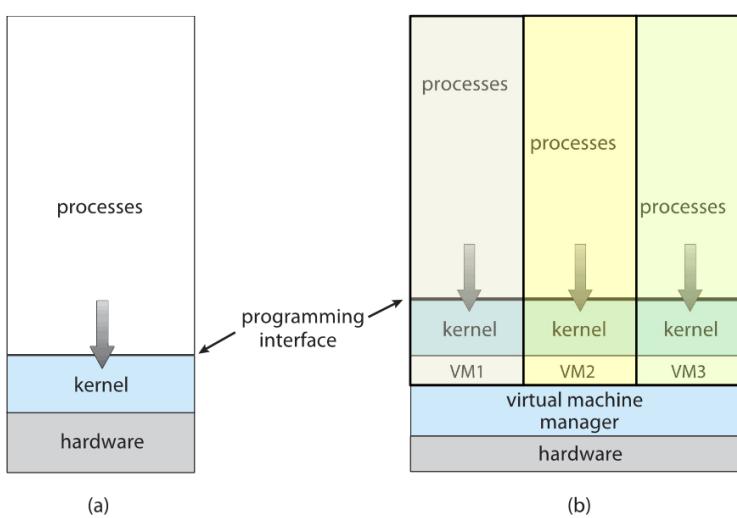
- 분산 시스템의 다른 모델
- P2P는 클라이언트와 서버를 구분하지 않고, 모든 노드들을 peer로 판단한다.
- 각 노드는 반드시 P2P 네트워크에 속해있어야 한다.

클라우드 컴퓨팅 환경



- 네트워크를 통해 컴퓨터, 저장공간 심지어 어플리케이션까지 서비스로서 제공한다.
- 가상화의 논리적인 확장이다.
- Public Cloud/Private Cloud/Hybrid Cloud로 나뉜다.
- SaaS(Software as a Service) : 하나 혹은 그 이상의 어플리케이션들이 인터넷을 통해 제공(ex: 워드 프로세스)
- PaaS(Platform as a Service) : 소프트웨어 스택이 인터넷을 통해 제공 (ex: 데이터베이스 서버)
- IaaS(Infrastructure as a Service) : 서버나 저장공간이 인터넷을 토에 제공 (ex: 백업을 위해 사용가능한 저장공간)

가상화



- 운영체제가 다른 운영체제들과 함께 동작될 수 있도록 함.
- 소스 CPU 유형이 대상 유형과 다른 경우 사용되는 에뮬레이션
- 가상화 - CPU용으로 기본 컴파일된 OS, 게스트 OS도 기본 컴파일됨
- 가상 머신은 논리적 결론에 대해 계층화된 접근 방식을 취한다. 이는 하드웨어와 운영 체제 커널이 모두 하드웨어인 것처럼 취급한다.

Real-Time 임베디드 시스템 컴퓨팅 환경

- 실시간성을 가진 운영체제를 뜻하며, 주어진 문제를 해결하기 위해 정해진 시간동안 처리하는 컴퓨팅 환경이다. 즉, 정해진 시간 내 행동할 수 없을 때 문제가 발생하게 된다.