

## 3장 시스템 소프트웨어의 구성

---

3.1 시스템 소프트웨어의 체계

3.2 운영체제(OS)

3.3 프로그래밍 언어 및 컴파일러

3.4 데이터베이스 관리시스템(DBMS)

3.5 유틸리티 소프트웨어

# 3.1 시스템 소프트웨어의 체계

- ◆ 소프트웨어의 개념
- ◆ 시스템 소프트웨어의 계층적 체계와 기능

# 소프트웨어의 개념

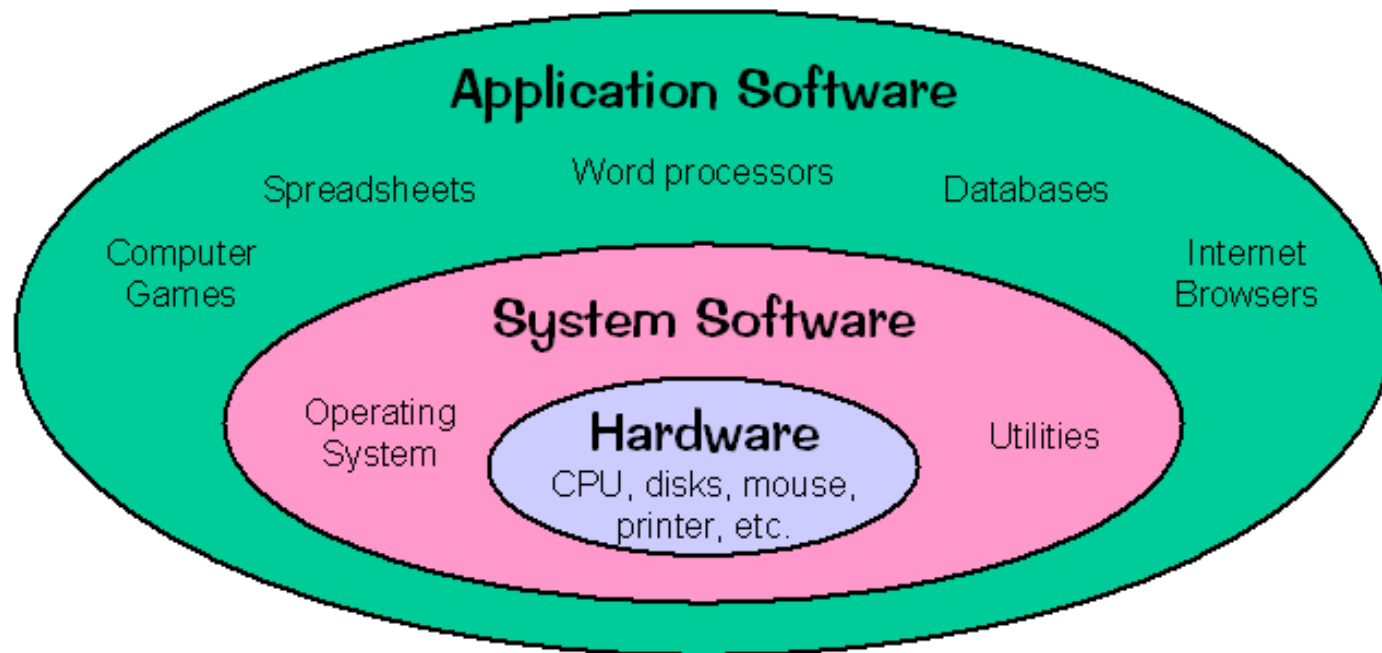
### ◆ 하드웨어와 소프트웨어

- 하드웨어: 컴퓨터를 구성하는 기기 장치
- 소프트웨어: 컴퓨터 시스템이나 주변기기 등의 하드웨어를 작동시켜 원하는 작업 결과를 얻기 위한 프로그램 또는 명령어
  - 제공 방법 : 하드디스크에 설치, CD/DVD로 제공, 인터넷 다운로드
  - 일단 설치하고(install) 원할 때 실행하여(run) 결과를 얻는다

### ◆ 시스템 소프트웨어 vs. 응용 소프트웨어

- 응용 소프트웨어
  - 문서작성기, 그림판, 게임 등 특정한 작업을 수행하기 위한 프로그램
- 시스템 소프트웨어
  - 컴퓨터나 주변기기들을 제어하는 프로그램

## ◆ 하드웨어와 소프트웨어



# 소프트웨어의 개념

### ◆ 시스템 소프트웨어의 역할

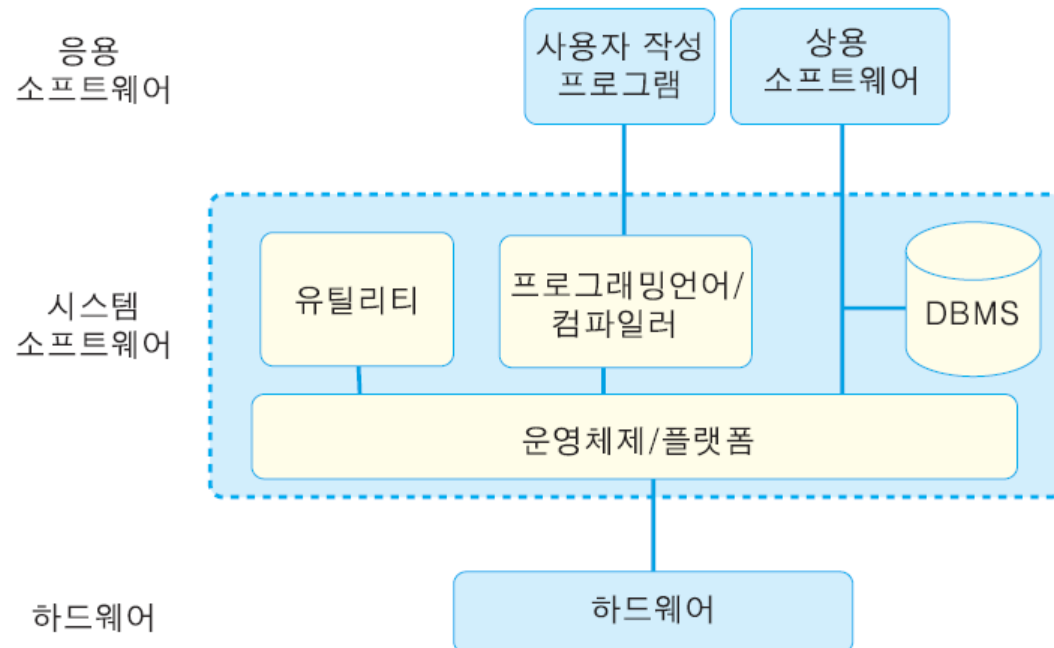
- 운영체제 : 하드웨어를 제어, MS Windows 나 Unix 등
- 드라이버 : 주변장치를 관리하는 프로그램
- 네트워크 프로그램 : 인터넷 접속 등의 네트워크 기능 수행



# 시스템 소프트웨어의 계층적 체계와 기능

## ◆ 소프트웨어의 계층적 체계

- 운영체제
- 컴파일러, 데이터베이스, 유틸리티



# 시스템 소프트웨어의 계층적 체계와 기능

### ◆ 운영체제

- 하드웨어를 직접 제어하고 자원을 관리해주는 계층
- 응용 프로그램들의 실행환경
  - 다른 응용프로그램들의 실행 기반이라는 의미로 ‘플랫폼’이라고 호칭

### ◆ 컴파일러/인터프리터

- 응용 소프트웨어를 개발하거나 사용자 자신의 프로그램을 개발하기 위하여 컴퓨터가 이해하는 언어로 번역해주는 소프트웨어
- 프로그래밍 언어
  - 컴퓨터가 읽고 사용하는 명령이나 코드의 집합
  - BASIC, C, Java 등의 언어
  - 원하는 프로그램을 개발하는데 사용

# 시스템 소프트웨어의 계층적 체계와 기능

### ◆ 데이터베이스 관리시스템

- 응용 소프트웨어와 운영체제 사이에서 대용량 데이터를 효율적으로 관리하기 위한 시스템 소프트웨어

### ◆ 유틸리티 소프트웨어

- 사용자가 시스템을 사용하기 편리하게 도움을 주는 프로그램
- 파일관리, 파일설치, 압축, 보안, 네트워크 등의 기능을 수행

### ◆ 시스템 소프트웨어와 응용 소프트웨어를 구별하기 어려운 경우도 있다.

- 예, MS Windows에서는 웹브라우저, 메모장, 그림판, 계산기 등 다수의 응용 프로그램을 운영체제에 기본적으로 포함



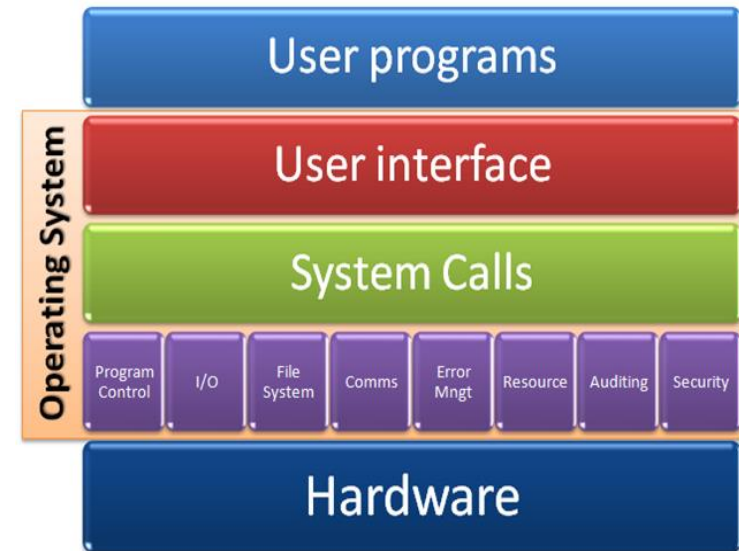
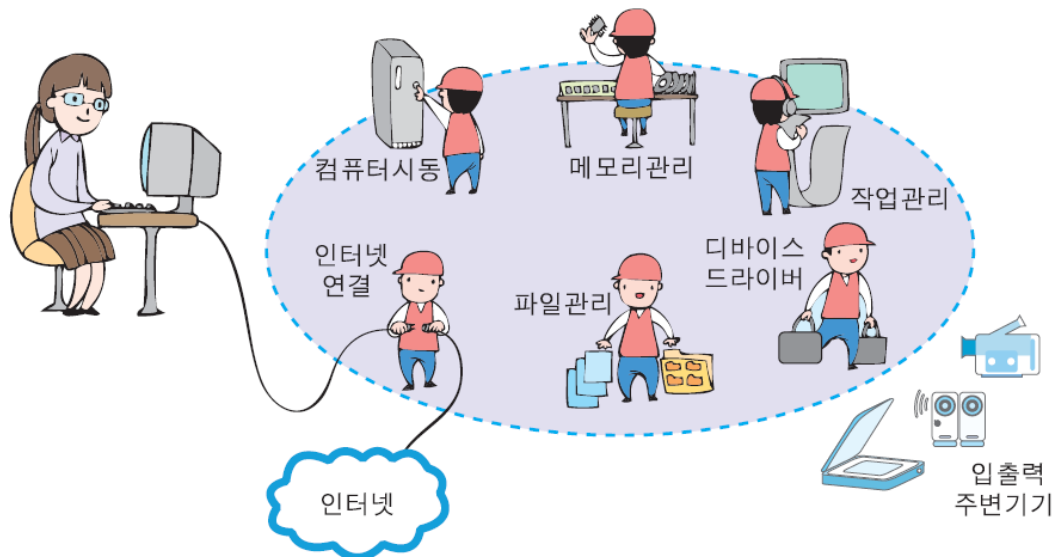
## 3.2 운영 체제(OS)

- ◆ 운영체제의 개념과 발전
- ◆ 운영체제의 기능
- ◆ 운영체제의 분류 및 사례

# 운영체제의 개념과 발전

## ◆ 운영체제의 역할

- 컴퓨터 시스템 내에서 플랫폼 역할 : 사용자의 작업명령이나 응용 소프트웨어의 실행을 위해 하드웨어 자원을 할당하고 제어
- 컴퓨터의 시동을 관장, 사용자 인터페이스를 제공, 프로그램을 실행, 메모리나 파일을 관리, 주변기기를 관리, 네트워크에 연결



# 운영체제의 개념과 발전

## ◆ 운영체제의 발전

- 초창기 컴퓨터 시스템 : 별도의 운영체제 개념이 없음
- 1960년대 메인프레임 - IBM System/360 시리즈
  - 유사한 기종에 같은 시스템 소프트웨어를 적용, 운영체제 개념이 태동
- 1970년대 미니컴퓨터 :
  - UNIX, VMS, 마이크로컴퓨터에서 CP/M, DOS, Mac OS 등 운영체제
- 1990년대 그래픽 인터페이스 방식
  - MS Windows와 같은 GUI 방식의 운영체제가 주류
- 2000년대 임베디드 운영체제
  - 모바일 기기 등 소형기기에 사용

# 운영체제의 개념과 발전

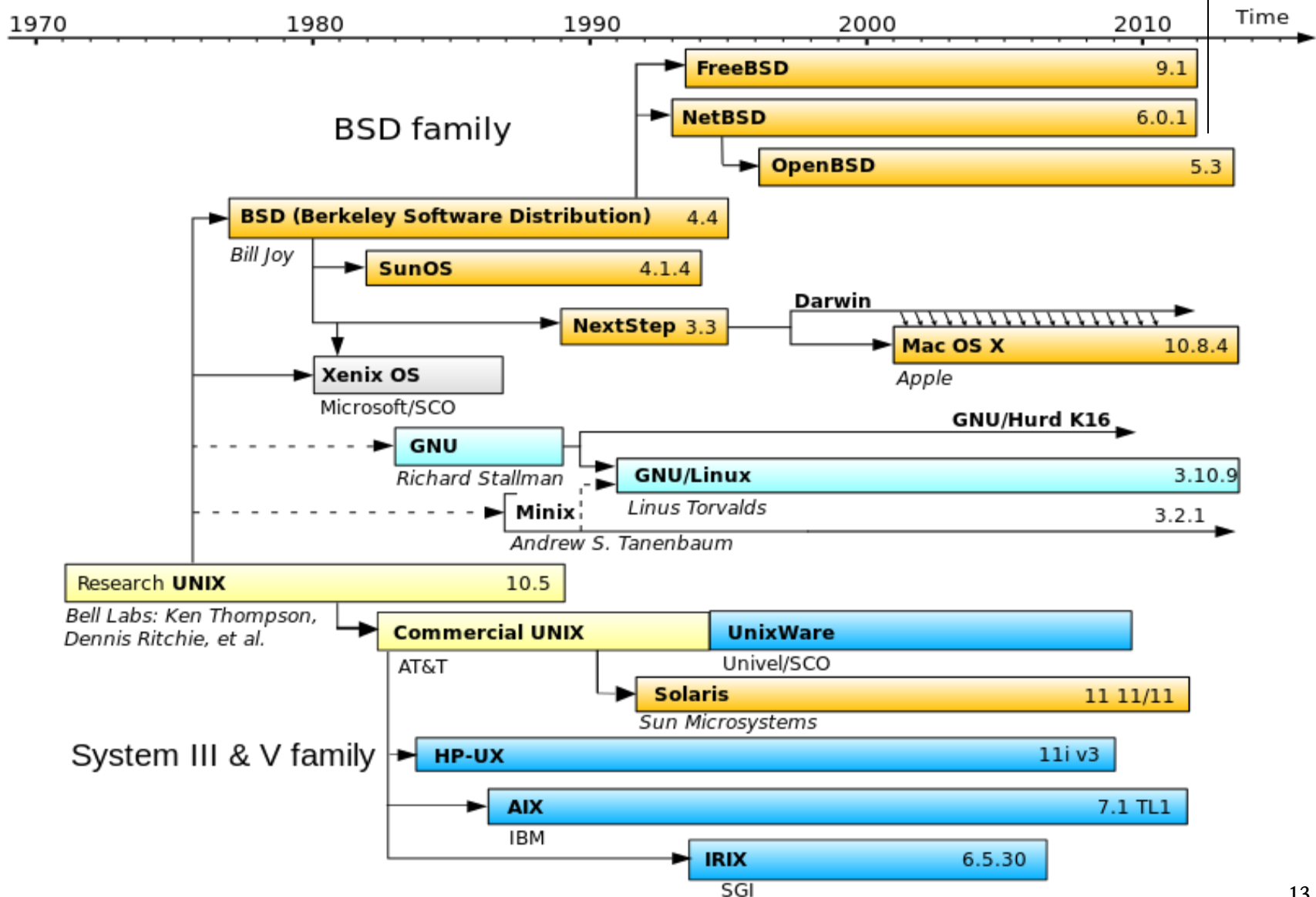
## ◆ 운영체제의 제공 형태

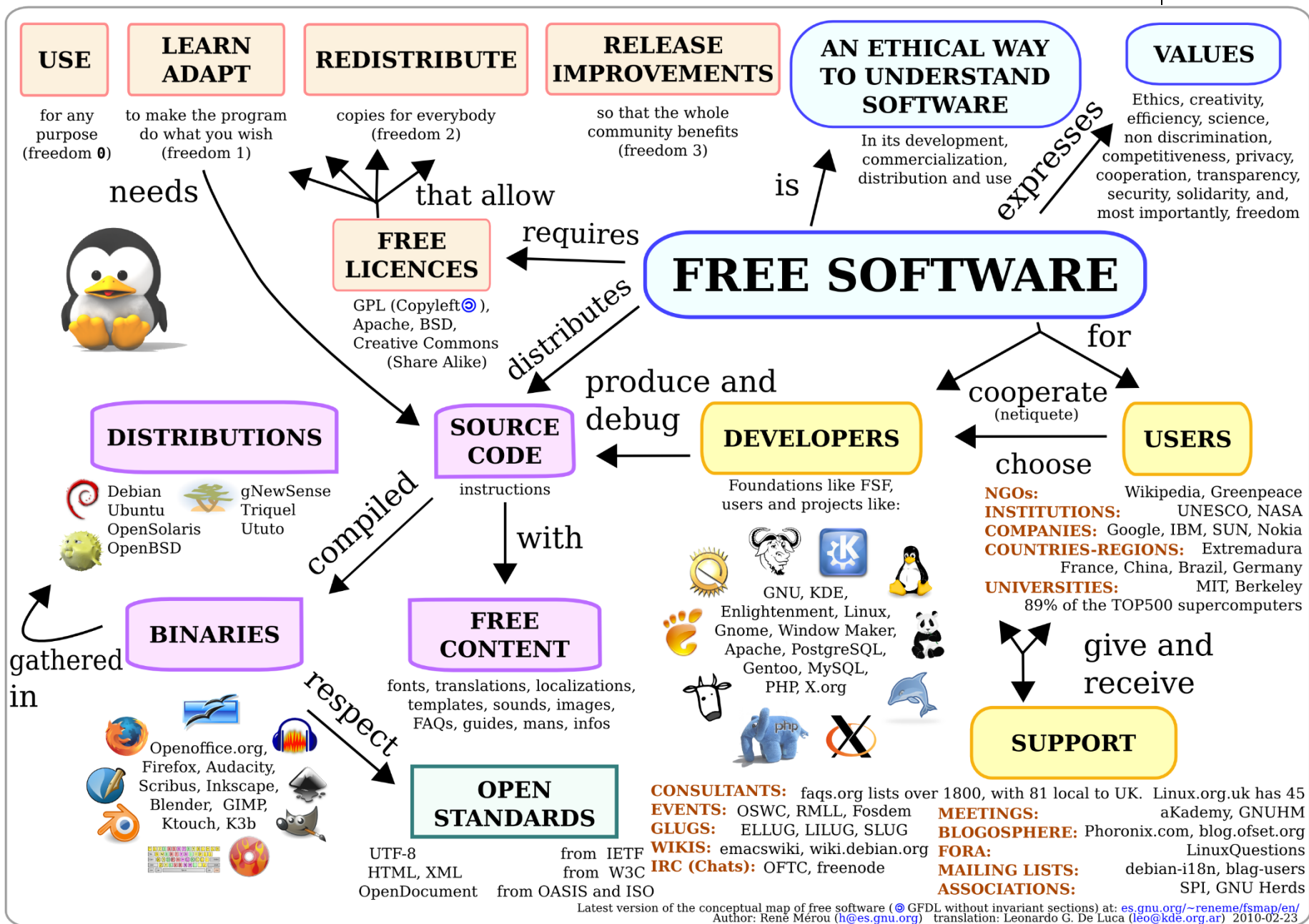
- 대부분의 경우 하드디스크에 미리 설치되어 제공
- 업데이트나 재설치를 위해 CD나 DVD에 별도로 제공
- PDA나 모바일 기기의 경우에는 ROM 칩에 저장하여 제공되기도

## ◆ 운영체제의 종류

- 컴퓨터의 크기별 구분
  - 메인프레임용 운영체제, 미니컴퓨터 및 워크스테이션용 운영체제, PC용 운영체제, 모바일 운영체제
- 대부분의 운영체제는 컴퓨터 하드웨어에 종속적
  - 경우에 따라서는 특정 시스템 전용으로 개발하기도
- 같은 유형이라도 컴퓨터 기종에 따라 혹은 사용자 선택에 따라 다른 운영체제를 사용
  - 예, 데스크탑 PC 경우 MS Windows, Linux, Mac OS 등 다른 OS 사용

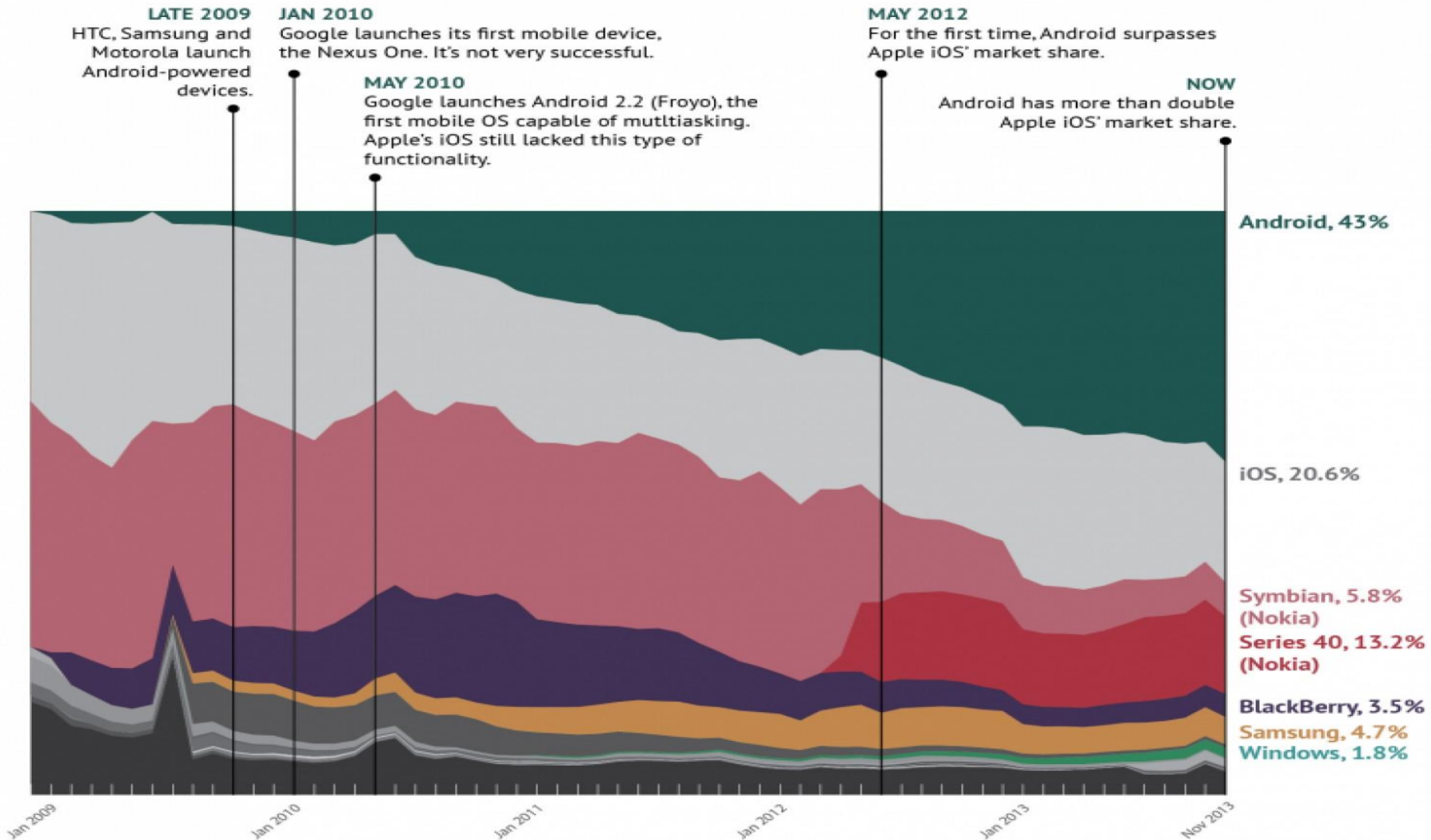
# UNIX vs LINUX





# A TIMELINE OF MOBILE OS MARKET SHARE

*How Google's Android Took Over The Market:*



By Lisa Mahapatra

INTERNATIONAL BUSINESS TIMES

Data Source: Global Stats StatCounter ([gs.statcounter.com](http://gs.statcounter.com))

# 운영체제의 기능

## ◆ 컴퓨터의 시동(Booting)

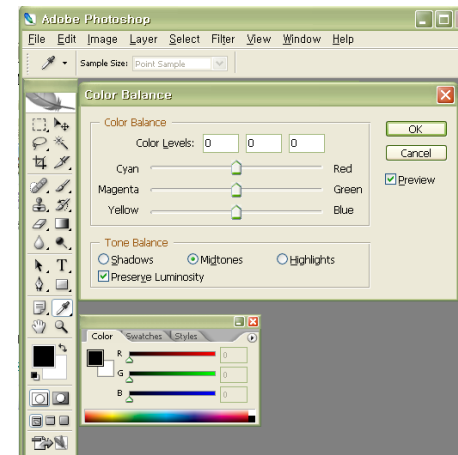
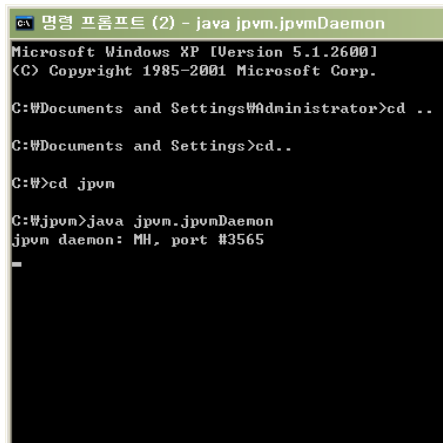
- 운영체제의 가장 기본적인 기능 중 하나
- 부팅 : 컴퓨터를 시작 또는 재시작 시키는 과정.
  - 콜드 부팅(cold-booting) : 컴퓨터에 전원을 연결하여 시동
  - 워мбу팅(Warm-booting) : 운영체제에서 컴퓨터를 재시동 시키는 것
- 펌웨어(Firmware) : 하드웨어를 직접 제어하는 부분
- 하드디스크에 있던 kernel 부분을 메모리에 로드
  - 메모리 상주 프로그램으로 대부분 눈에 보이지 않게 백그라운드에서 작업 수행. 나머지는 필요할 때 로드
- 시스템 구성 관리: 어느 하드웨어 가 연결되어 있는지 목록을 관리
  - 주변기기와 통신하려면 device driver 필요, 요즘은 주변기기를 연결하면 해당 드라이버를 찾아서 자동연결 (Plug & Play)



# 운영체제의 기능

## ◆ 사용자 인터페이스 제공

- 입력을 운영체제에서 컴퓨터가 이해할 수 있는 형태로 변환
  - 처리 결과 및 정보를 화면에 출력
- 종류
  - 명령어방식(command-line): 문자로 구성된 명령을 직접 타이핑
  - 메뉴방식(menu-driven): 기능이 할당된 키 또는 화면의 메뉴 선택
  - 그래픽방식(GUI): 요즘에는 GUI 방식이 많이 사용.



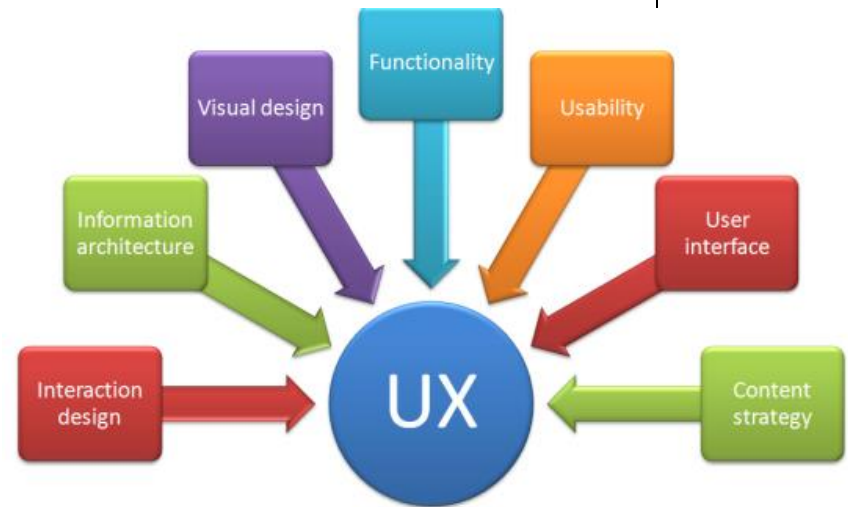
# UX IS NOT UI

## HOW UX WANTS TO BE SEEN

- Field research
- Face to face interviewing
- Creation of user tests
- Gathering and organizing statistics
- Creating personas
- Product design
- Feature writing
- Requirement writing
- Graphic arts
- Interaction design
- Information architecture
- Usability
- Prototyping
- Interface layout
- Interface design
- Visual design
- Taxonomy creation
- Terminology creation
- Copywriting
- Presenting and speaking
- Working tightly with programmers
- Brainstorm coordination
- Design culture evangelism

## HOW UX IS TYPICALLY SEEN

- Field research
- Face to face interviewing
- Creation of user tests
- Gathering and organizing statistics
- Creating personas
- Product design
- Feature writing
- Requirement writing
- Graphic arts
- Interaction design
- Information architecture
- Usability
- Prototyping
- Interface layout
- Interface design
- Visual design
- Taxonomy creation
- Terminology creation
- Copywriting
- Presenting and speaking
- Working tightly with programmers
- Brainstorm coordination
- Design culture evangelism



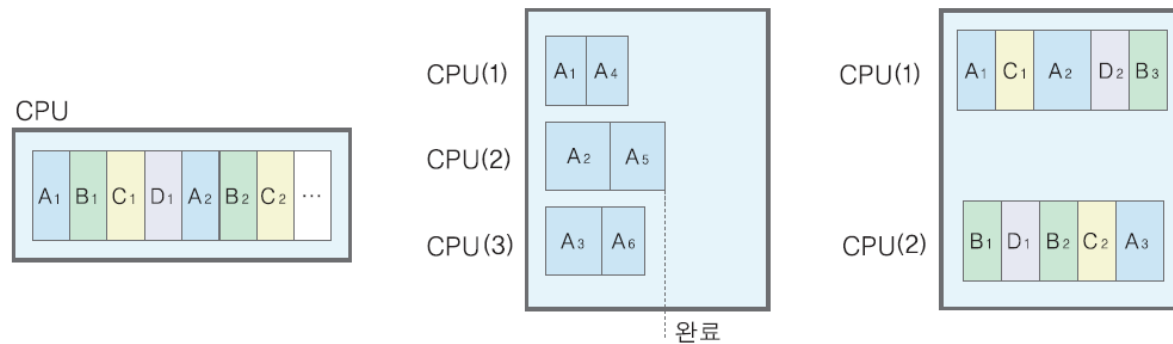
# 운영체제의 기능

## ◆ 프로그램 실행관리

- 시스템 자원을 프로그램이나 주변 기기에 할당하며 실행을 제어
  - 단일 사용자 환경의 단일 작업 경우는 문제 없음
  - 여러 개의 프로그램을 동시에, 또는 하나의 프로그램을 더 빠르게 실행하고자 할 때 다양한 기술이 필요
- 다중작업(Multitasking)
  - 여러 개의 프로그램 또는 작업을 동시에 실행
  - 예) PC에서 문서+그림+음악+이메일 동시 실행
  - 실제로 한 순간에는 하나의 작업만 가능하므로 시간을 쪼개서 순서대로 돌아가며 실행

# 운영체제의 기능

- 다중작업(Multitasking)의 개념
  - 시분할(Time Sharing) 기법 : 시간을 분할하여 순서대로 돌아가며 실행
  - 병렬처리(Parallel Processing) : 하나의 작업을 여러 개의 CPU에 할당하여 속도를 더 빠르게 하는 기술
  - 다중처리(Multiprocessing) : 여러 개의 프로그램을 여러 개의 CPU가 실행하여 전체적인 성능을 향상시키는 기술



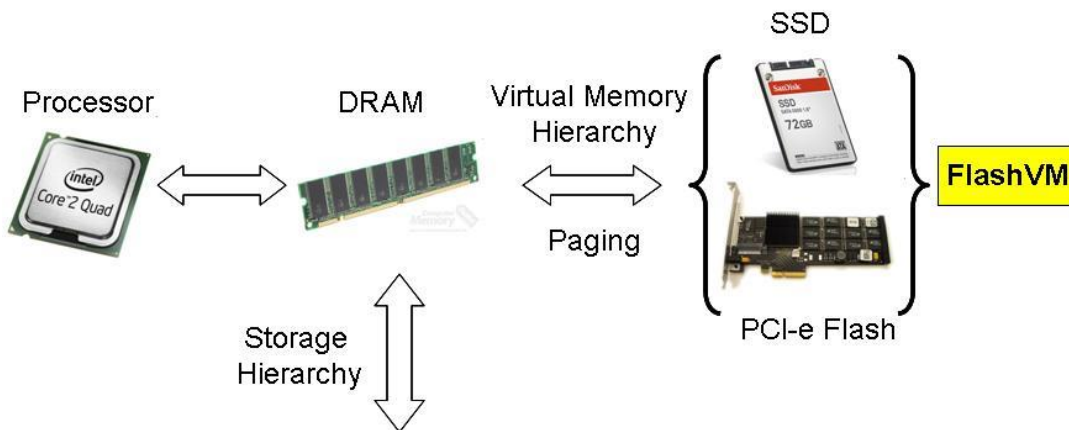
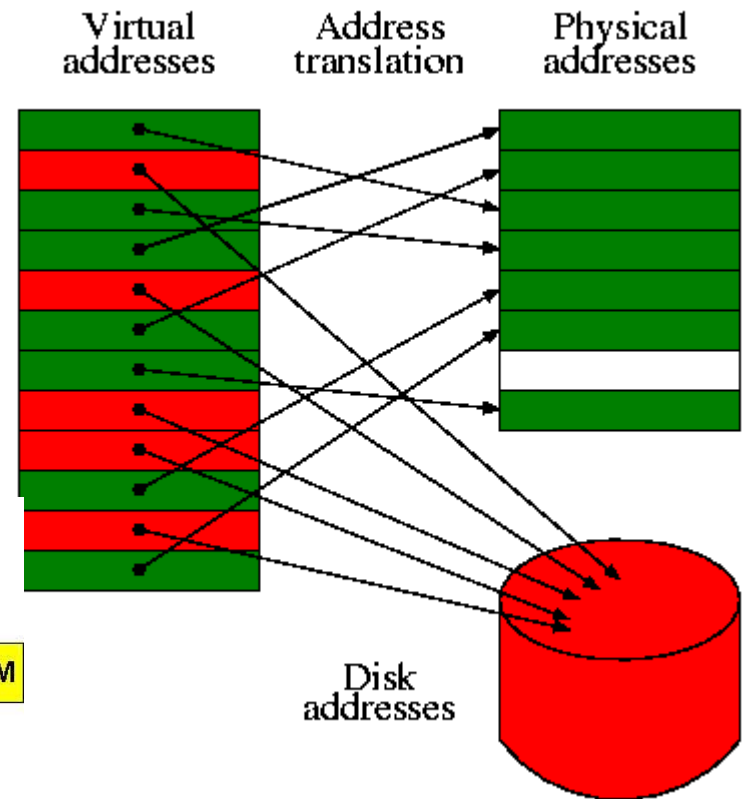
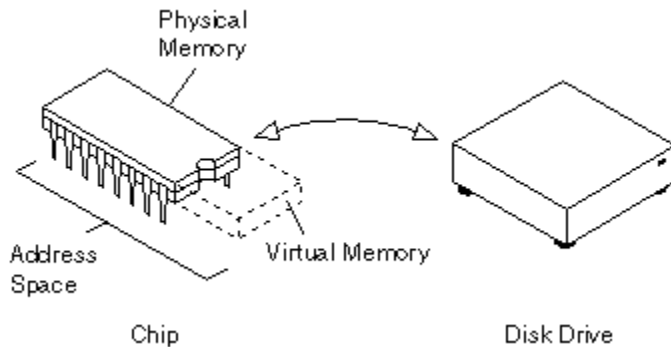
(a) 시분할(Time Sharing)      (b) 병렬처리(Parallel Processing)      (c) 다중처리(Multiprocessing)

# 운영체제의 기능

## ◆ 메모리 관리

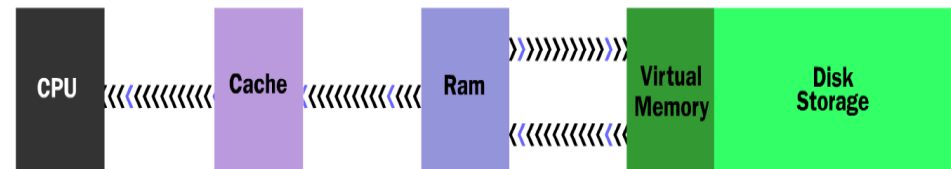
- 빠른 속도의 레지스터와 캐시 메모리, 그리고 RAM 및 디스크 등의 메모리 사용을 최적화 하는 것이 목적
- 현재 수행 중인 작업에서 프로그램과 데이터를 적절한 메모리 영역에 할당하고 프로그램의 실행이 완료된 후에는 이들을 제거
  - RAM 상주 부분 : 운영체제에서 커널 부분과 상황에 따라 필요한 기능, 실행 중인 프로그램, 작업 중인 데이터 등을 주메모리인 RAM에 상주
- 가상메모리(Virtual Memory) 기법
  - RAM 용량이 부족한 경우 하드디스크 일부를 RAM처럼 활용
  - 실행 중인 프로그램을 적당한 크기로 나누어 현재 실행에 꼭 필요한 부분은 RAM에 배치하고 당장 필요하지 않는 부분은 하드디스크에 배치
  - 필요할 때 바꾸어 로드하여 RAM 용량이 훨씬 큰 것처럼 느끼도록

# 가상메모리(Virtual Memory) 기법



Disks

## Memory Management



# 운영체제의 기능

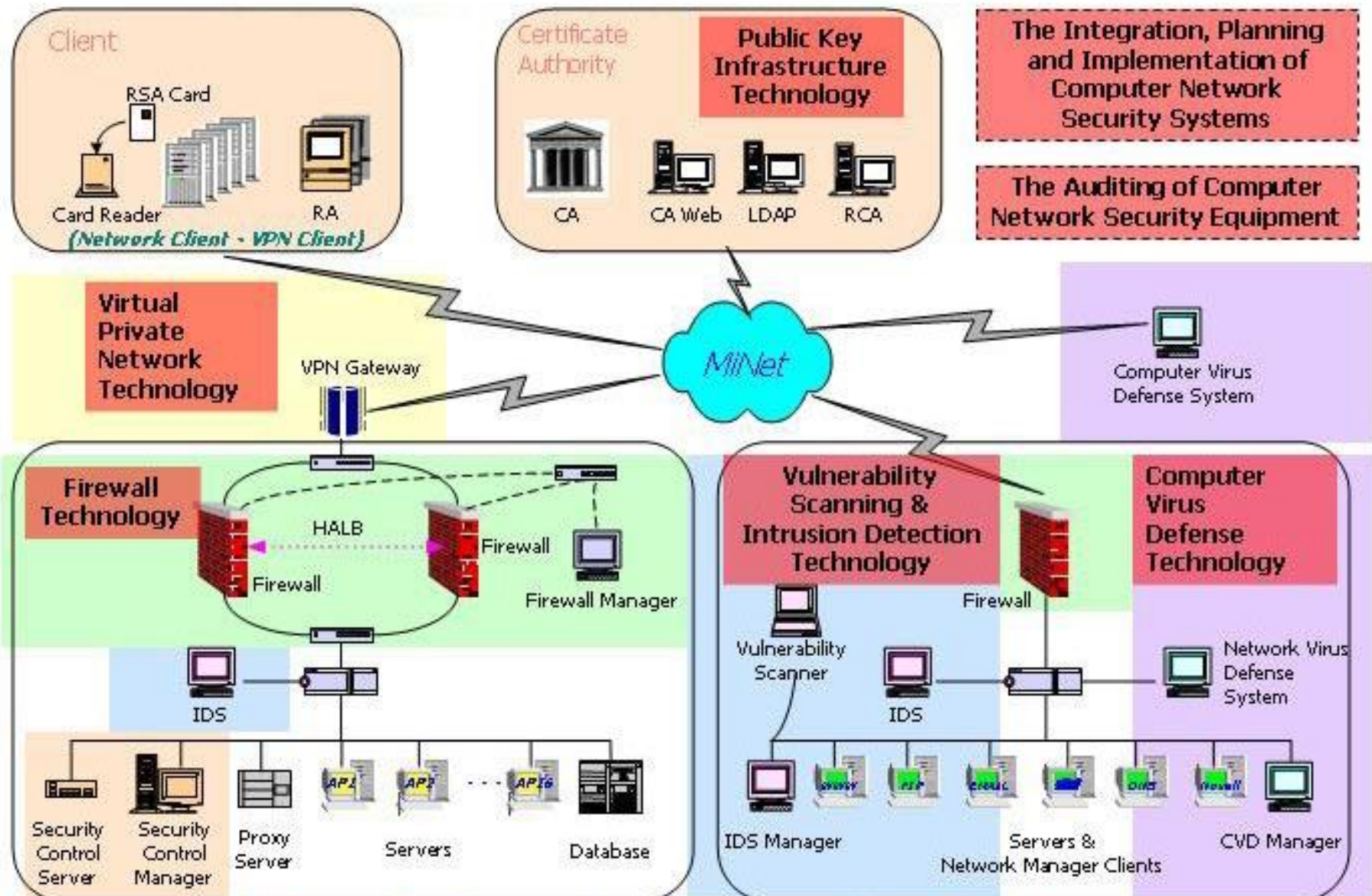
## ◆ 파일 관리

- 하드디스크에 파일을 계층적 구조의 폴더에 저장
- 파일경로
  - 하드디스크의 계층적 구조에서 찾고자 하는 파일의 위치를 표현한 것  
예) 웹의 경우 C://Introduction to Computers/text/ch03\_software.ppt  
예) MS DOS의 경우 C:\컴퓨터 개론\text\03장\_software.hwp
- 파일이름 : 문자나 숫자 등으로 구성되며
- 파일 확장자 : 이름 끝에 점(.) 다음에는 파일의 종류를 표시  
예) 실행프로그램 .exe , 한글 문서 .hwp , 웹문서의 경우 .html 등

## ◆ 기타 기능

- 보안(security) 기능, 암호화 및 압축 기능, 인터넷 연결 작업, 네트워크 제어 기능, 성능 모니터링 기능 등





**Network Security System Architecture**



# 운영체제의 분류 및 사례

## ◆ 데스크탑 운영체제

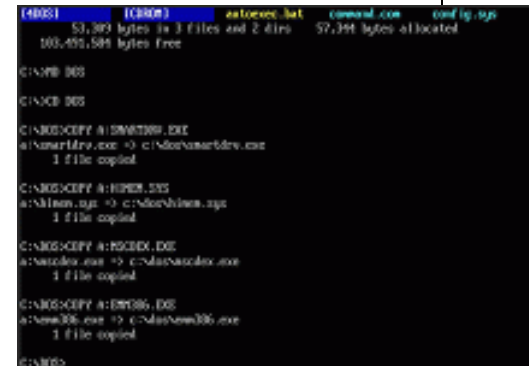
### ● DOS

- 1980년대 초반 IBM PC에서 등장,
- 1990년대 초반까지 널리 사용
- 명령어 방식의 인터페이스
  - 예) 파일의 목록을 보려면 'dir',  
파일을 삭제하려면 'del (파일이름)'

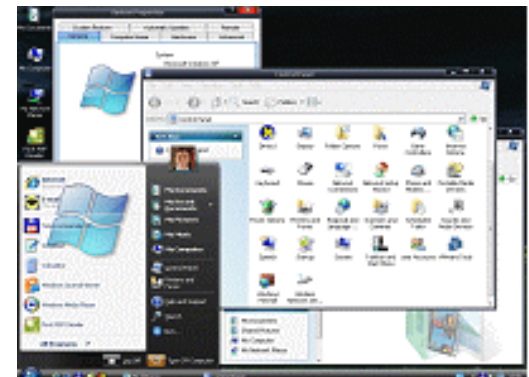
### ● MS Windows

- 1980년대 후반 Microsoft사가 개발
- 대표적인 GUI 방식
- 현재 대부분의 PC에서 널리 사용
- Windows 95버전부터 매킨토시의 인터페이스와 유사하게 개발
- Windows NT, 98, 2000, XP 등을 거쳐 2007년 Windows Vista 발표

DOS 실행화면



Windows Vista 실행화면



# 운영체제의 분류 및 사례

- Mac OS
  - Apple사 1984년 매킨토시 기종에 초기부터 탑재
  - 최근 2007년 Leopard 버전이 발표
  - GUI 방식 인터페이스를 최초로 탑재한 운영체제
- UNIX
  - 1970년대 초반 AT&T Bell 연구소에서 개발
    - 무상으로 제공되어 다양한 컴퓨터 기종에서 구현
  - 다중사용자 및 다중작업 기능을 지원하는 대표적인 운영체제
  - 명령어 방식의 인터페이스를 기본으로 한다.
    - 예) 파일의 목록을 보려면 'ls', 파일을 삭제하려면 'rm'
  - 1980년대부터는 다양한 버전으로 개조되어 판매
    - 대표적인 UNIX 계열의 운영체제 : HP-UX, Sun의 Solaris 등



Mac OS 실행화면

# 운영체제의 분류 및 사례

- Linux

- UNIX 계열의 운영체제 중 대표적인 공개 소프트웨어
  - 1991년 핀란드 헬싱키 대학의 리누스 토발즈(Linus Torvalds)가 개발
- 공개 소프트웨어 운동
  - MIT의 리처드 스톨만(Richard Stallman) 교수가 유닉스와 유사한 운영체제인 GNU를 개발하면서 시작
  - Linux가 최초의 공개소스 운영체제의 커널로 개발
- ‘오픈소스(Open Source)’
  - 소스코드를 공개하고 각자 수정한 결과를 또 다시 공유하자는 취지
  - 사용하는 것은 물론 수정, 재배포, 심지어 최종상품으로 판매까지 가능
- Linux의 배포
  - 대표적으로 RedHat사에서 배포판을 제작하여 오픈소스 비즈니스를 전개



# MEMORABLE LINUX MILESTONES

## CELEBRATING 20 YEARS OF LINUX

LINUS TORVALDS  
POSTS FAMOUS  
MESSAGE - "HELLO  
EVERYBODY OUT  
THERE..." - AND  
RELEASES FIRST  
LINUX CODE



1991

SLACKWARE  
BECOMES FIRST  
WIDELY ADOPTED  
DISTRIBUTION



1993

TECH GIANTS  
BEGIN ANNOUNCING  
PLATFORM SUPPORT  
FOR LINUX



1998

IBM RUNS  
FAMOUS LINUX  
AD DURING THE  
SUPERBOWL



2003

THE LINUX  
FOUNDATION IS  
FORMED TO PROMOTE  
PROTECT AND  
STANDARDIZE LINUX  
LINUX IS A FELLOW



2007

LINUX TURNS 20  
AND POWERS THE  
WORLD'S  
SUPERCOMPUTERS,  
STOCK EXCHANGES,  
PHONES, ATMS,  
HEALTHCARE  
RECORDS,  
SMART GRIDS, THE  
LIST GOES ON



2011



LINUS LICENSES  
LINUX UNDER  
THE GPL, AN  
IMPORTANT  
DECISION THAT  
WILL CONTRIBUTE  
TO ITS SUCCESS IN  
THE COMING YEARS

1992



LINUS VISITS  
AQUARIUM, GETS  
BIT BY A PENGUIN  
AND CHOOSES  
IT AS LINUX MASCOT

1996



RED HAT  
GOES PUBLIC

1999



LINUS APPEARS ON  
THE COVER OF  
BUSINESSWEEK WITH  
A STORY THAT HAILS  
LINUX AS A  
BUSINESS SUCCESS

2005



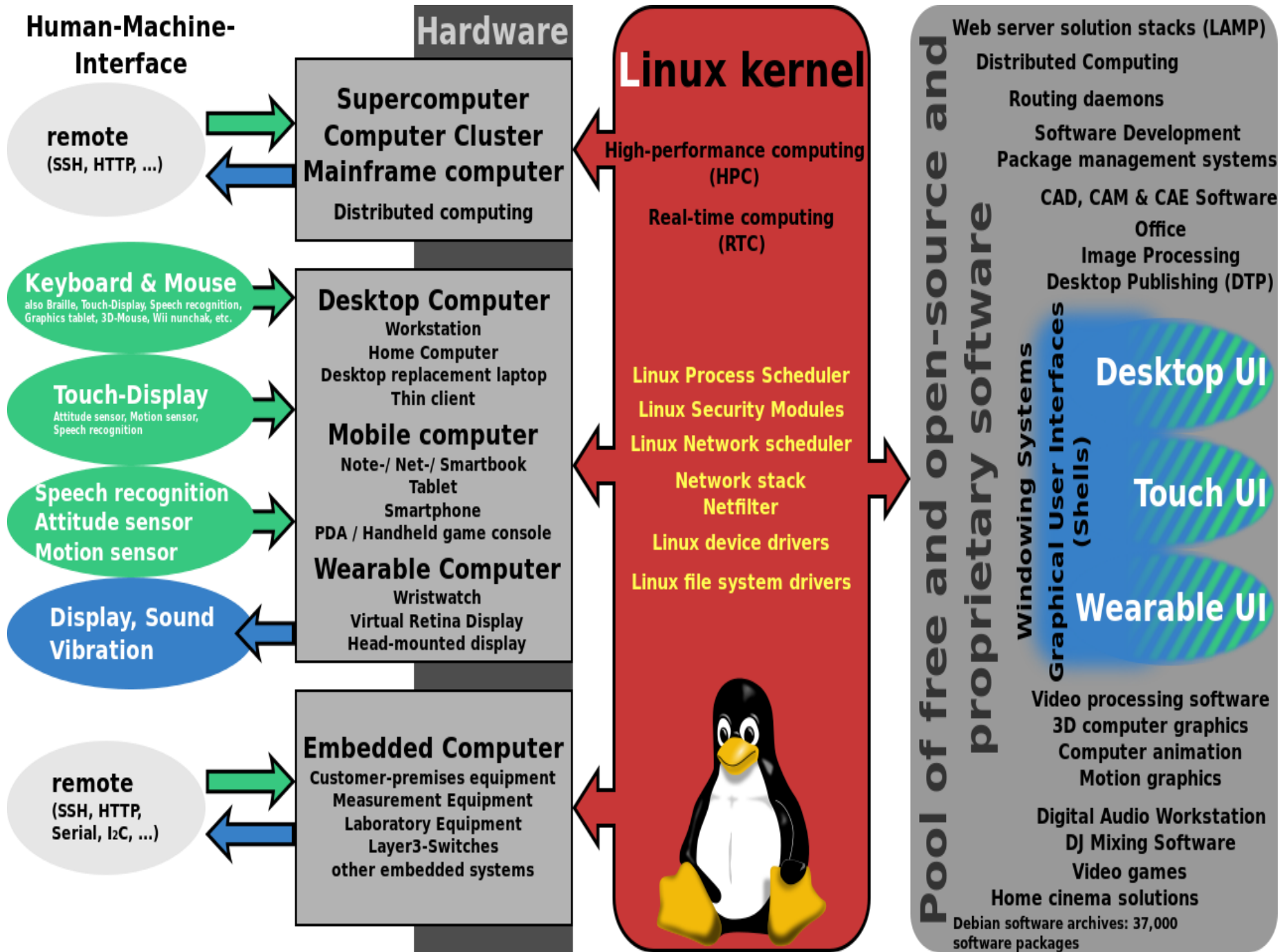
THE LINUX-BASED  
ANDROID OS  
OUTSHIPS ALL OTHER  
SMARTPHONE OSes  
IN THE U.S. AND  
CLIMBS TO  
DOMINANCE

2010



THE  
LINUX  
FOUNDATION  
<http://www.linuxfoundation.org/>





## 운영체제의 분류 및 사례

### ◆ 네트워크 기반 운영체제

- 여러 대의 컴퓨터가 연결되어 프로그램이나 데이터가 서로 연동
  - 1980년대 중반 Novell사 Netware가 대표적인 네트워크 운영체제
  - 이 외에도 Windows NT, SUN Solaris, IBM OS/2 등
  - UNIX나 Linux도 네트워크 서버용 운영체제로 사용

### ◆ 메인프레임 운영체제

- 시스템의 규모가 대형이고 구성 기술이 기종마다 많은 차이
  - 각 기종의 종류에 따라 독자적인 운영체제를 사용
  - IBM 컴퓨터의 경우 1960년대 IBM System/360 기종부터 시작하여 현재 IBM System Z까지 기종에 따라 다양한 운영체제를 사용
- 메인프레임 운영체제에서는 거대 용량의 디스크 스토리지 기기의 관리나 다중사용자 및 다중작업 처리 기능이 매우 중요









## 운영체제의 분류 및 사례

### ◆ 모바일 기기용 임베디드(Embedded) 운영체제

- 휴대폰, TV 셋탑박스, 산업용 전자기기, 로봇과 같은 장치
  - 컴퓨터 프로세서와 같은 칩이 장착(Embedded)
  - 이들 장치의 작동을 위하여 컴퓨터와 마찬가지로 운영체제가 필요
- 임베디드 운영체제
  - 임베디드 장치에 필요한 핵심적인 시스템 소프트웨어 부분
  - 고사양의 운영체제로는 Windows CE와 Embedded Linux가 많이 사용
  - 저사양 소형 기기에는 RTOS(Real Time Operating System) 계열 사용
- Windows CE
  - MS Windows 운영체제 기능을 축소하여 작은 기기에 사용하도록 개발
  - PDA 등의 pocket PC에 많이 사용
- Embedded Linux
  - Linux를 축소하여 소형기기에 맞도록 개발한 버전
  - PDA, 스마트폰, 셋탑박스, 제어기기 등에 많이 사용

# Windows Extends Beyond the PC





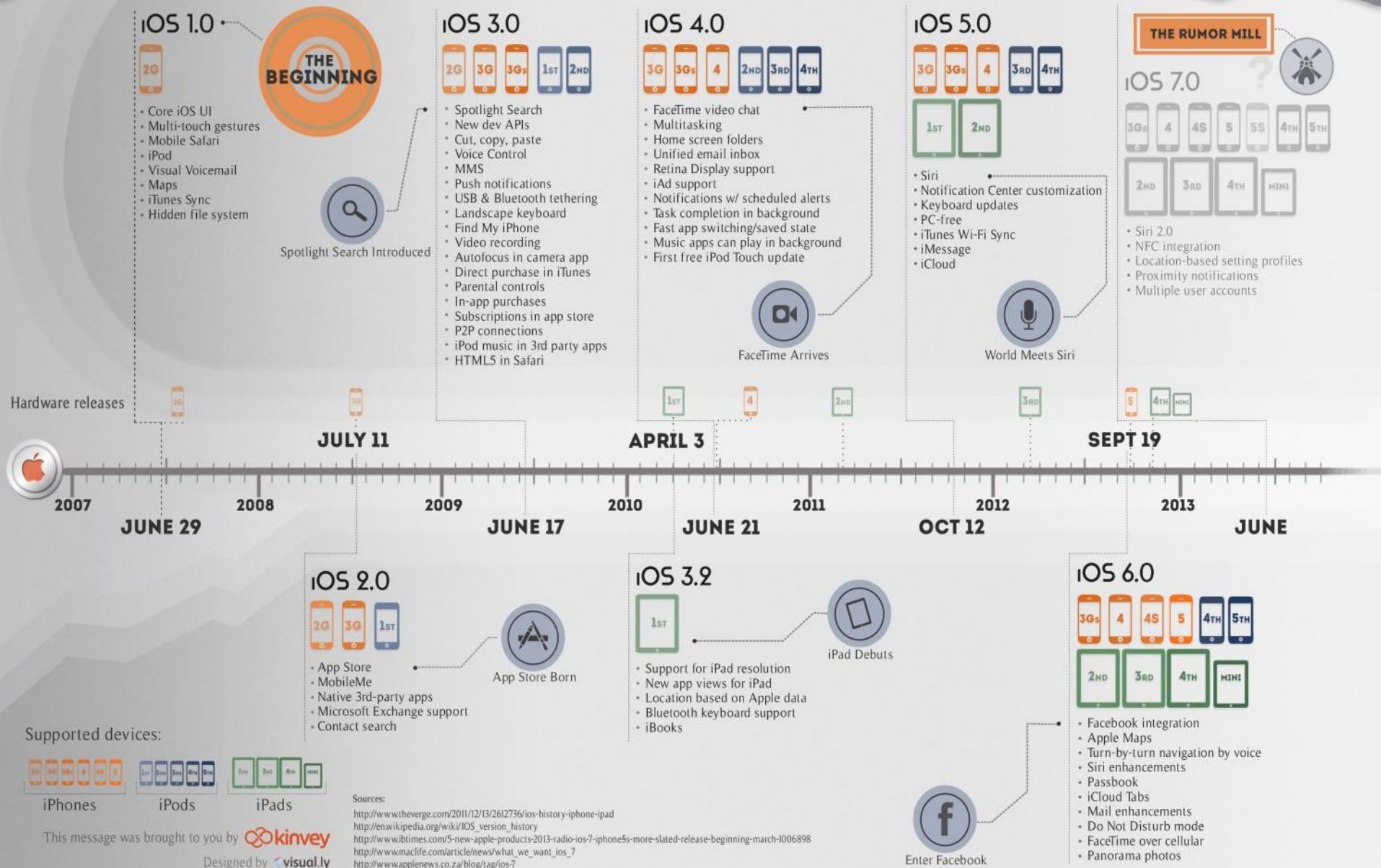
## 운영체제의 분류 및 사례

- Windows mobile
  - Windows CE 기반으로 모바일 기기에 적합하도록 개발
  - 스마트폰에서 많이 사용, Windows XP와 인터페이스 모습이 유사
- RTOS 계열
  - Palm OS : palm handheld 기기용 OS로 PDA, 스마트폰에 사용
  - Symbian OS : 일반 휴대폰이나 스마트폰에서 사용
    - 유명 모바일폰 회사들이 Symbian 사 설립 .
- 최근 모바일 기기 운영체제
  - Apple iOS : Apple iPhone 및 iPad에서 사용
  - Google Android : 개방형 모바일 운영체제, 다수의 스마트폰에서 사용



# iOS Version History: A Visual Timeline

In 5 1/2 years, Apple has released 9 hardware devices and 51 total iOS version updates. Here we've captured the major milestones.





**Alpha**  
Nov'07



**Beta**  
5 Nov'07



**Cupcake**  
30 Apr'09



**Donut**  
15 Sep'09



**Eclair**  
26 Oct'09



**Froyo**  
20 May'10



**Gingerbread**  
6 Dec'10



**Honeycomb**  
22 Feb'11



**Ice Cream Sandwich**  
19 Oct'11



**Jelly Bean**  
9 Jul'12



**KitKat**  
31 Oct'13

## 3.3 프로그래밍 언어 및 컴파일러

- ◆ 프로그래밍 언어와 컴파일러의 기능
- ◆ 초기 프로그래밍 언어
- ◆ 절차적 프로그래밍 언어
- ◆ 객체지향 언어
- ◆ 제4세대 언어 및 기타 언어

# 프로그래밍 언어와 컴파일러의 기능

## ◆ 저수준(Low-level) 언어

- 기계어(Machine Language)
  - 자료 표현의 기본 단위인 비트의 값 0과 1로 그대로 표기하는 언어
  - 컴퓨터 CPU에서 명령을 수행하기 위해 반드시 필요
  - CPU의 종류에 따라 각기 고유한 명령어로 구성
  - 컴퓨터 실행에는 효율적이지만, 사람이 작성하기에는 매우 불편

기계어 프로그램의 예

Addr	Object code
2048	
2064	
2068	00000010 10000000 00000000 00000110
2072	10000010 10000000 01111111 11111100
2076	10001000 10000000 01000000 00000010
2080	11001010 00000001 00000000 00000000
2084	00010000 10111111 11111111 11111011
2088	10000110 10000000 11000000 00000101
2092	10000001 11000011 11100000 00000100
2096	00000000 00000000 00000000 00010100
3000	00000000 00000000 00001011 10111000



# 프로그래밍 언어와 컴파일러의 기능

- 어셈블리 언어(Assembly Language)가
  - 컴퓨터 고유의 기계어 명령을 사람이 어느 정도 해독할 수 있도록 문자화하거나 기호화한 형태
  - 기계어와 마찬가지로 CPU 종류에 따라 명령어가 다르다
  - 일단 기계어로 번역되는 과정을 거쳐야 컴퓨터에서 실행이 가능

어셈블리 프로그램의 예

Label	Instruction
	.begin
a_start	.equ 3000
	ld length,%
	be done
	addcc %r1,-4,%r1
	addcc %r1,%r2,%r4
	ld %r4,%r5
	ba loop
	addcc %r3,%r5,%r3
done:	jmp1 %r15+4,%r0
length:	20
address:	a_start
a:	

# TExaS

## Source Code

```
;Example
PORTA equ    $0000
          org    $0800
DATA      rmb    100
CNT       rmb    1
          org    $F000
Start     lds    # $0C00
Rep       ldaa   SIZE
          staa   CNT
          ldx    #DATA
Inp       ldaa   PORTA
          staa   1,x+
          dec    CNT
          bne    Inp
          bra    Rep
NAME      fcc    "Jon"
SIZE      fcb    100
          org    $FFFE
          fdb    Start
```

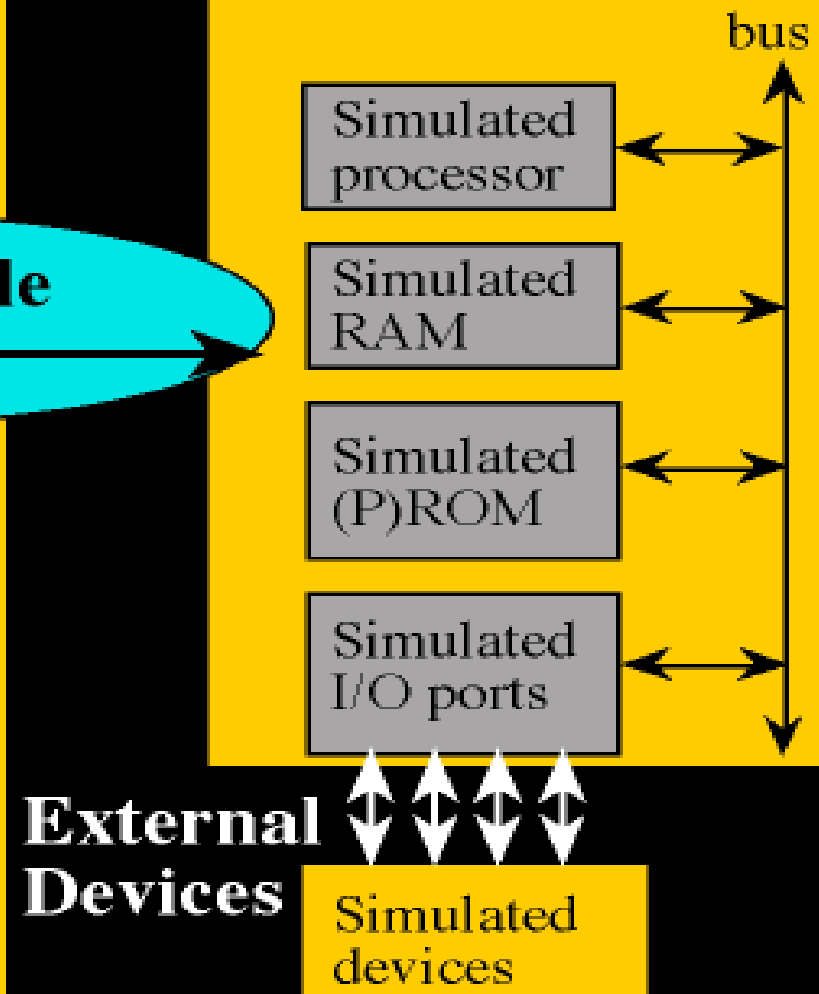
## Object Code

```
$F000 CF
$F001 0C
$F002 00
$F003 B6
$F004 F0
$F005 1A
$F006 7A
$F007 08
$F008 64
$F009 CE
```

**Assemble**

```
$F00A 20
$F00B 30
$F00C 73
$F00D 08
$F00E 64
$F00F 26
$F010 F7
$F011 20
$F012 EC
$F013 4A
$F014 6F
$F015 6E
$F016 64
$F017 F0
$F018 F0
$F019 F0
$F01A F0
$FFFE F0
$FFFF 00
```

## Simulated Microcomputer



# 프로그래밍 언어와 컴파일러의 기능

## ◆ 고수준(High-level) 언어

- 명령어가 일상적인 문장에 가까워 코딩 부분이 훨씬 간단
  - 기계어로 번역하는 복잡한 과정을 거쳐야 하지만 컴퓨터 하드웨어에 독립적이므로 프로그래밍 과정은 더욱 간단

- 예) 데이터 20개의 합과 평균

```
sum = 0.0;
for (i=0; i<20; i++)
    sum = sum + data[i];
if (sum > 0.0)
    average = sum/20;
```

- 절차적(Procedural) 언어
  - FORTRAN, COBOL, C, Pascal, BASIC 등이 대표적
- 최근에는 C++, Java 등의 객체지향 언어가 많이 사용
- 4세대 언어로는 SQL, RPG 등
- 인공지능 등 특수 목적의 언어로는 LISP, Prolog, SmallTalk 등

## 프로그래밍 언어와 컴파일러의 기능

### ◆ 컴파일러(Compiler)

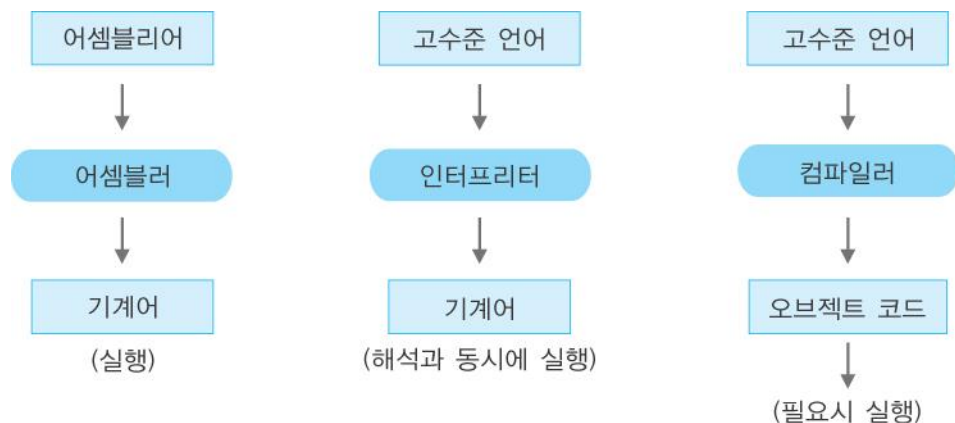
- 모든 프로그램은 기계어로 번역되어야 프로그램의 실행이 가능
  - 소스 프로그램(Source Program)을 기계어로 번역하여 오브젝트 코드(Object Code)라 불리는 실행 가능한 프로그램으로 저장
- 오브젝트 코드 형태의 프로그램은 항상 컴퓨터에서 실행이 가능
  - \*.exe 또는 \*.com 파일이 이에 해당
  - 대부분의 응용 프로그램들이 컴파일되어 실행파일 형태로 제공
- 프로그램 실행 시에는 번역이 필요 없으므로 실행 속도 빠르다
- 번역된 기계어는 사람의 해독이 어려워 소스코드의 보안에 좋다
- 일단 컴파일하여 제공되므로 오류가 발생하거나 새로운 내용으로 변경하고자 할 때 즉시 적용하는 것은 곤란

# 프로그래밍 언어와 컴파일러의 기능

## ◆ 인터프리터(Interpreter)

- 소스 프로그램 한 줄씩 기계어로 해석하여 바로 실행을 하는 방식
  - BASIC이 대표적, 웹페이지에 많이 사용되는 스크립트 언어(Script Language)들도 인터프리터 방식
- 실행 속도는 약간 느리더라도 문제가 안 되는 분야에서 많이 사용
- 장점으로는 개발이나 업데이트가 쉽다

어셈블러,  
인터프리터,  
컴파일러에서  
프로그램 처리과정



# 초기 프로그래밍 언어

## ◆ 어셈블리 언어

- 기계어를 0과 1의 숫자 대신 그대로 기호화
  - 더하기의 경우 A(Add), 곱하기는 M(Multiply), 문자열 비교의 경우 C(Compare), 데이터 로드할 때 L(Load) 등의 명령을 사용
  - 메모리 주소도 번지수 대신에 기호로 된 이름을 사용
- 기계어에 비해 사용하기 쉬우나 역시 배우기는 어렵다.
- CPU 종류에 따라 해당하는 어셈블리 언어가 다르다.
- 산업기계 제어용 프로그램에서 사용
  - 속도를 증가시키기 위하여 프로그램을 주로 어셈블리 언어로 작성
  - 최근에는 프로세서의 성능이 향상됨에 따라 C 또는 Java 등의 고수준 언어를 사용하는 추세



# 초기 프로그래밍 언어

## ◆ FORTRAN

(FORmular TRANslator)

- 최초의 고수준 프로그래밍 언어
  - 1954년 과학이나 공학 계산을 위한 프로그램용으로 개발
- 과학 공식 표현이 매우 쉽다.
  - 예)  $Y=X^2+5X+9$
- 우주선 궤도 계산, 자동차 설계, 선박 설계 등에 사용
  - 아직도 선박 설계용으로 사용하는 경우가 있다.

## FORTRAN 프로그램의 예

```

C  FORTRAN PROGRAM
C  AVERAGE OF INPUT NUMBERS
SUM = 0
READ (5,50) NUMBER
2  IF (NUMBER .EQ. 999) GO TO 3
SUM = SUM + NUMBER
COUNT = COUNT + 1
WRITE (6,60)
READ (5,50) NUMBER
GO TO 2
3  AVER = SUM / COUNT
WRITE(6,70) AVER
50  FORMAT (I3)
60  FORMAT (1X, 'ENTER : ')
70  FORMAT (1X, 'AVERAGE = ', F6.2)
STOP
END

```

# 초기 프로그래밍 언어

## ◆ COBOL

- 1960년대 초반  
데이터 처리를 위해  
그레이스 호퍼(Grace Hopper)  
가 개발
- 영어 표기와 유사, 읽기 쉽고  
작성 용이, 유지하기 용이
- 회계처리나 상거래와 같은  
비즈니스 응용에서 많이 사용

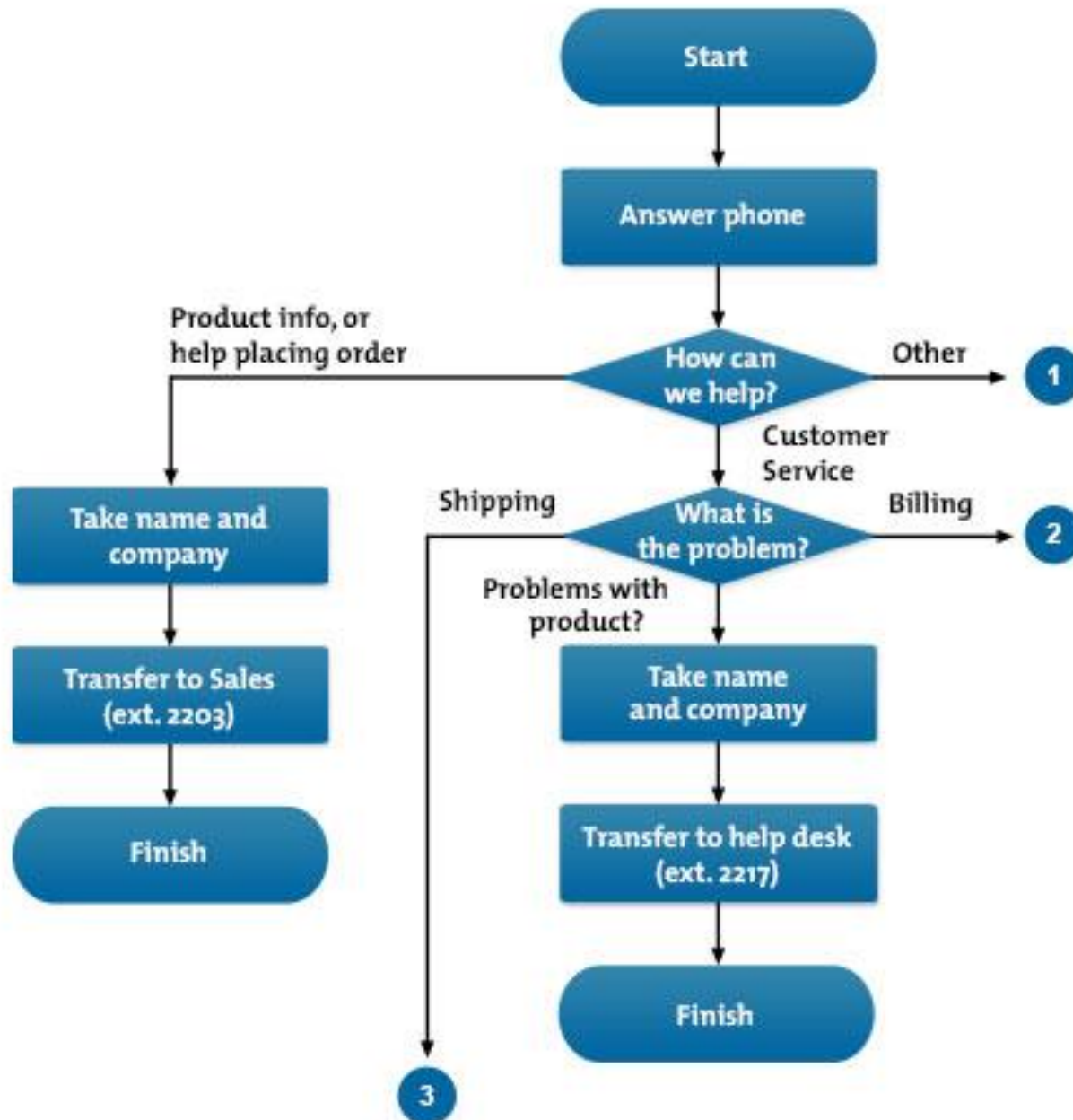
## COBOL 프로그램의 예

```
IDENTIFICATION DIVISION.  
PROGRAM-ID. SAMPLE  
* Example COBOL program using MULTIPLY  
  
DATA DIVISION.  
  
WORKING-STORAGE SECTION.  
01 Num1          PIC 9 VALUE ZEROS.  
01 Num2          PIC 9 VALUE ZEROS.  
01 Result        PIC 99 VALUE ZEROS.  
  
PROCEDURE DIVISION.  
    DISPLAY "Enter 1st number (1 digit): ".  
    ACCEPT Num1.  
    DISPLAY "Enter 2nd number (1 digit): ".  
    ACCEPT Num2.  
    MULTIPLY Num1 BY Num2 GIVING Result.  
    DISPLAY "Result is = ", Result.  
    STOP RUN.
```

# 절차적 프로그래밍 언어

### ◆ 절차적 언어의 필요성 대두

- 컴퓨터가 수행할 작업의 절차를 표현
  - 예) 입력, 계산 작업, 데이터의 처리 lac 저장, 작업의 반복, 출력
- 구조적 프로그래밍
  - 70년대부터 프로그래밍 기본 개념으로 적용
  - 반복 및 선택의 제어구조가 특징 : 언어에 따라 문법만 다를 뿐
  - 반복구조 for, while, do\_while
  - 선택구조 if, switch\_case
- 대표적 언어
  - C, Pascal, BASIC, PL/I, Ada 등



# 절차적 프로그래밍 언어

## ◆ C 언어의 소개

- 70년대 초반 AT&T Bell 연구소 UNIX 운영체제 작성을 위해 개발
  - 현재 OS 뿐만 아니라 각종 응용 프로그램 개발에 널리 사용
- 고수준 프로그램 뿐 아니라 저수준의 시스템 프로그램도 편리
  - 응용 프로그램, 시스템 프로그램, 산업기계제어 프로그램 등 널리 사용

### C 언어 프로그램의 예

```
/* C language program      */
main () {
    int number, count=0, sum=0;
    float aver;
    scanf("Enter : %d", &number);
    while (number < 999) {
        sum += number;
        count++;
        scanf("Enter : %d", &number);
    }
    aver = (float)sum / count;
    printf("\n Average = %.2f ", aver);
}
```

# 절차적 프로그래밍 언어

## ◆ BASIC 언어의 소개

- Beginner's All purpose Symbolic Instruction Code 의 약자
- 간단한 계산을 위한 초심자용, 배우기 쉽고 작성이 용이
- 80년대 및 90년대 PC에서 매우 널리 활용

### BASIC 프로그램의 예

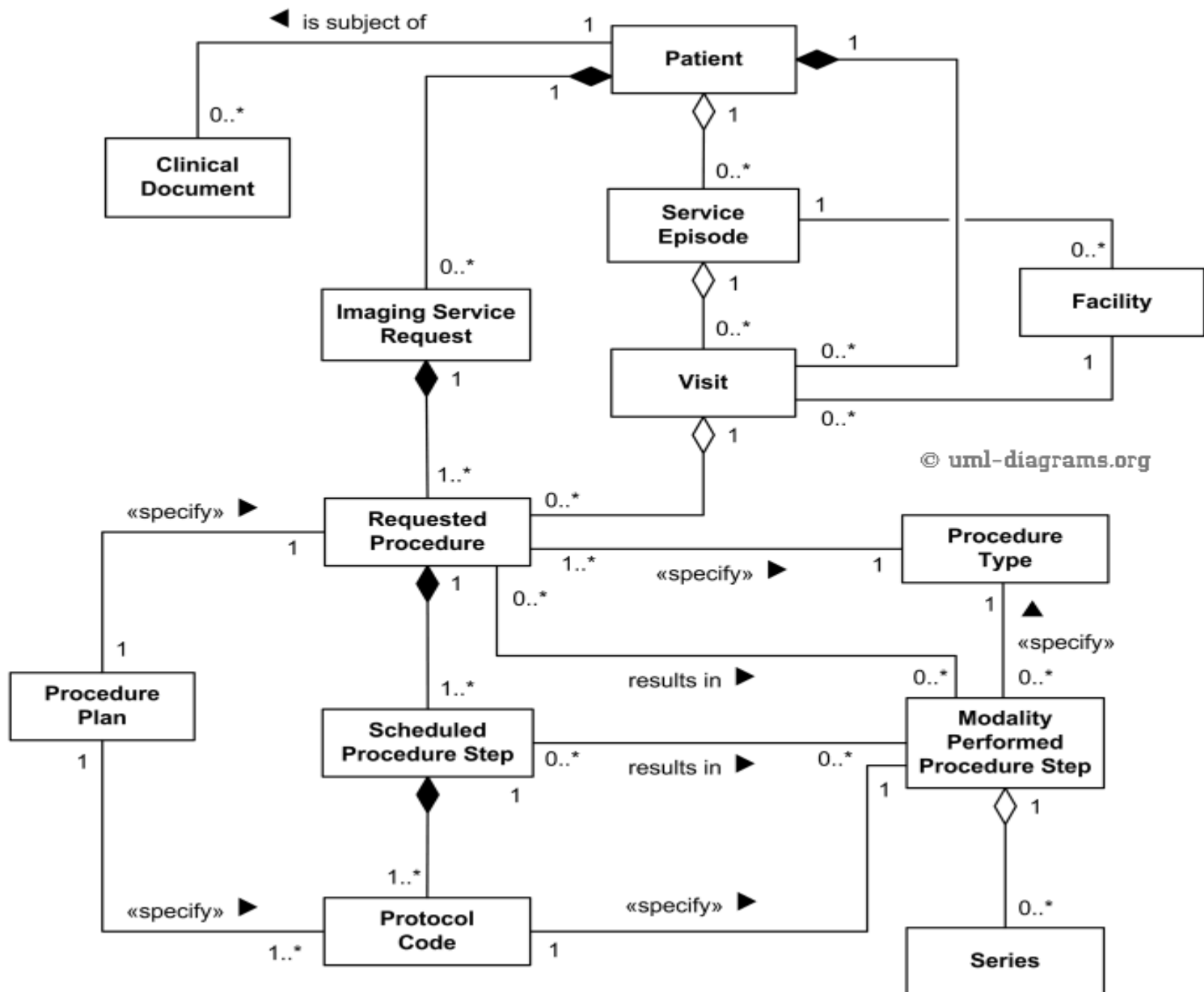
```
'EXAMPLE BASIC PROGRAM
SUM = 0
COUNT = 0
PRINT "ENTER A NUMBER"
INPUT NUMBER
DO WHILE NUMBER < 999
    SUM = SUM + NUMBER
    COUNT = COUNT + 1
    PRINT "ENTER A NUMBER"
LOOP
AVER = SUM / COUNT
PRINT "AVERAGE = "; AVER
```

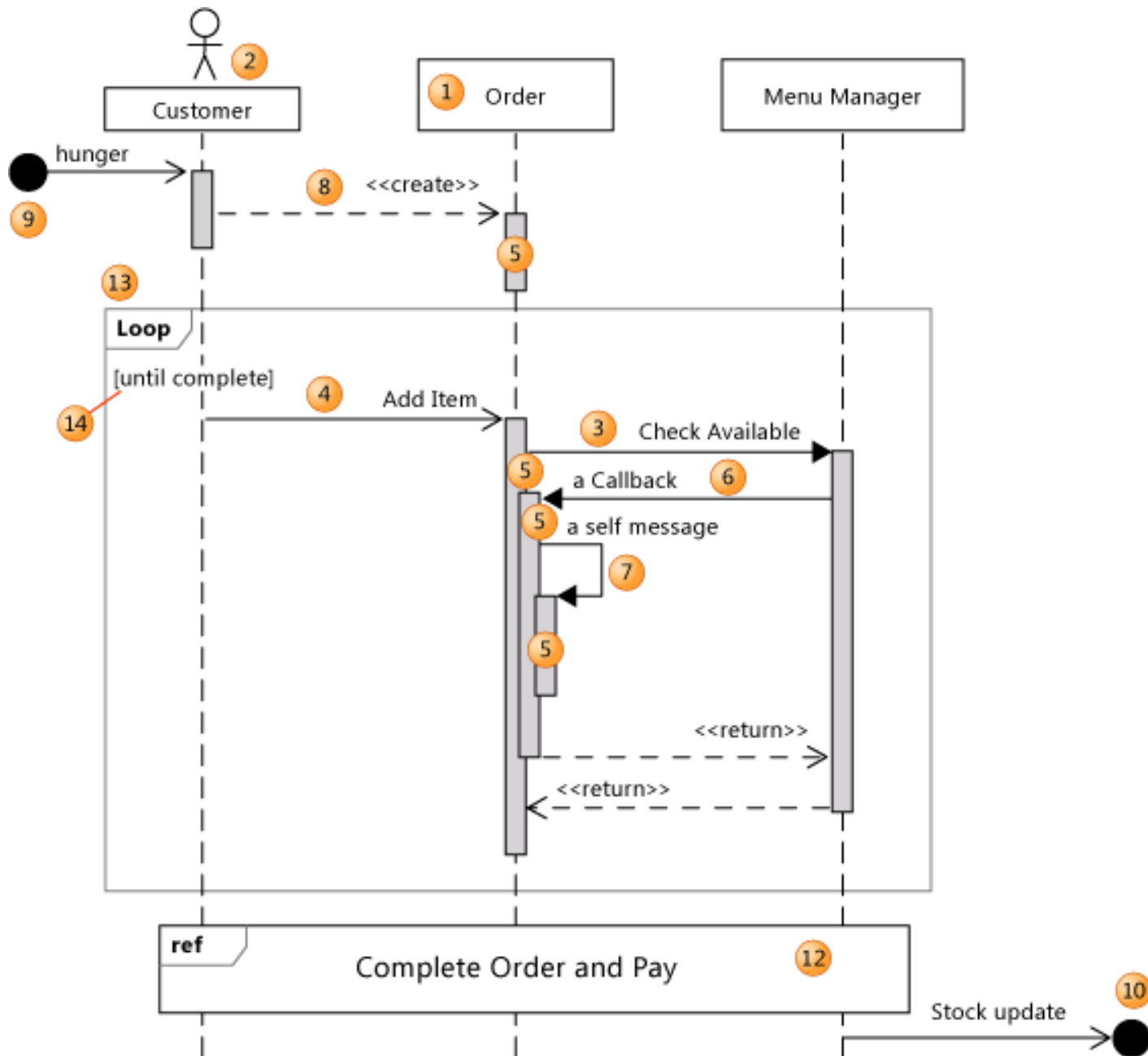


# 객체지향 언어

## ◆ 객체지향의 개념

- 인간이 이해하기 쉬운 사물이나 개념을 객체(Object)로 표현
- 객체의 형식과 처리방법을 클래스(Class)에 한 묶음으로 표현
  - 객체의 처리방법을 클래스에 정의하고 필요시(이벤트 발생시) 호출 해당 작업만을 수행 → 이벤트기반(Event Driven) 프로그래밍 기법
- 프로그램을 객체별로 분리하여 취급하므로 재사용과 수정이 용이
- 대표적 언어
  - SmallTalk, C++, Java, C# 등





# 객체지향 언어

## ◆ C++

- 80년대 Bell 연구소  
Bjarne Soustrup 개발
  - C 언어를 확장하여 object, class, event 등 객체지향 개념을 적용
- DB 활용, 웹 기반, 각종 응용 프로그램 개발에 많이 사용
- C# 은 마이크로 소프트에서 웹 서비스 응용에 적합하도록 개발
  - C++을 확장한 객체지향 언어

## C++ 프로그램의 예

```
/* C++ language program      */
main () {
    int  number, count=0, sum=0;
    float aver;
    cout << "Enter a number : "
    cin >> number;
    while (number < 999) {
        sum += number;
        count++;
        cout << "Enter a number : "
        cin >> number;
    }
    aver = (float)sum / count;
    cout << "\n Average = " << aver;
}
```

# 객체지향 언어

## ◆ Java

- 1980년대 후반 Sun Microsystems사가 차세대 언어로 개발
  - 웹 어플리케이션, 일반 어플리케이션, 소형 가전기기에도 사용
- Java로 작성된 응용 프로그램 명칭
  - Application : 일반 데스크탑 응용 프로그램
  - Applet : 웹 응용 프로그램
  - Xlet 또는 Jlet : 셋탑박스 나 모바일 기기 등에서의 Java 프로그램

## Java 프로그램의 예

```

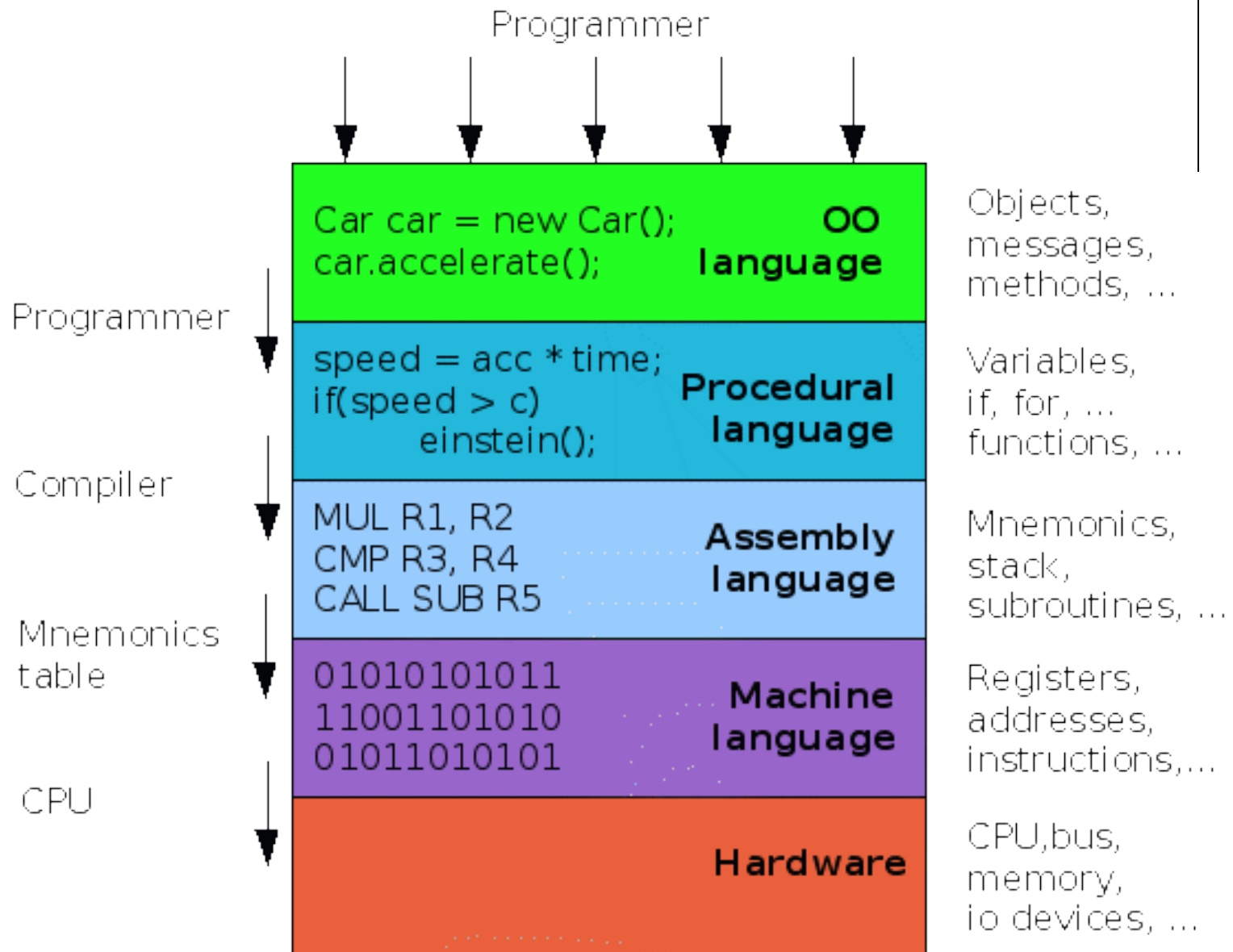
/* Java language program      */
public static void main(String args[]) {
    int  number, count=0, sum=0;
    float aver;
    DataInputStream dis =
        new DataInputStream();
    System.out.println("Enter : ");
    while ((number=dis.readInt())< 999) {
        sum += number;
        count++;
        System.out.println("Enter : ");
    }
    aver = (float)sum / count;
    System.out.println("Average = "+aver);
    dis.close;
}

```

# 객체지향 언어

- Java 컴파일러
  - 일종의 중간형태인 바이트코드(Byte Code) 형태로 번역
  - 실행 시에는 Java VM(Virtual Machine)에서 바이트코드를 해석하여 작업을 수행
  - Java 응용 프로그램이 수행되려면 사용자의 시스템에 Java VM이 설치되어 있어야 한다





## 제4세대 언어 및 기타 언어

### ◆ 제4세대 언어

- 초고수준 언어(very high level language)로 선언적 언어
  - 절차적 언어에서는 작업의 방법(how to do)을 기술
  - 선언적 언어에서는 작업의 대상(what to do)을 기술
  - SQL, RPG 등이 대표적
- 일반적으로 그래픽 프로그래밍 환경을 제공
  - 비주얼 프로그래밍 환경 : Visual Studio, Delphi, PowerBuilder 등

### ◆ 특수 목적 언어

- 인공지능 처리용 언어 : LISP, Prolog, SmallTalk 등
- 웹 환경 개발 언어 (6장에서 자세히 설명)
  - HTML, XHTML, XML 등 페이지 표현
  - Javascript, Perl, Tcl, Ajax 등 스크립트 언어

## 3.4 데이터베이스 관리시스템(DBMS)

- ◆ 데이터베이스의 개념
- ◆ DBMS의 원리
- ◆ DBMS의 종류 및 활용

## 데이터베이스의 개념

### ◆ 지식정보사회에서의 정보의 양이 매우 방대

- 행정전산망의 경우
  - 전 국민의 주민 정보가 데이터베이스에 수록 : 거주지 주소, 가족관계, 병역관계, 심지어 부동산 재산까지도 수록
  - 전국의 동사무소 어디에서나 민원서류 발급이 가능
  - 세금이나 교육기관 등 다른 국가 전산망과도 연동하여 막대한 분량의 정보가 데이터베이스에 저장되어 필요할 때마다 수정 및 검색 서비스
- 포털사이트의 뉴스 검색 및 지식 검색
  - 방대한 분량의 정보가 해당하는 웹 데이터베이스에 저장
  - 텍스트 이외에도 이미지나 동영상 등 멀티미디어 정보를 저장
  - 뉴스의 경우만 해도 매일 사진을 포함한 수 백 개의 데이터가 생산

# 데이터베이스의 개념

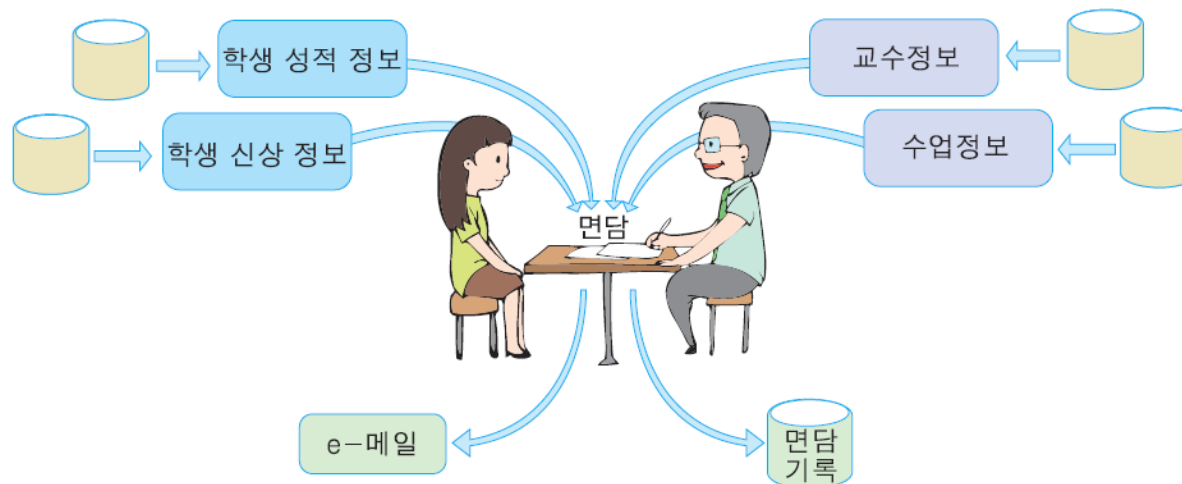
## ◆ 데이터와 정보

- 데이터 : 텍스트, 숫자, 이미지, 비디오 등으로 입력 받은 자료를 디스크와 같은 저장매체에 기록해둔 것
- 정보 : 데이터를 처리해서 사람이 이해하기 적합한 형태로 만든 것
  - 정보는 데이터가 구조화되어 의미 있고 유용성 있게 처리된 결과
- 방대한 분량의 데이터를 저장하거나 서로 연관된 정보를 검색하는 데에 데이터베이스가 매우 필요
  - 많은 분량의 데이터를 수집하여 데이터베이스에 체계적으로 정리해두면 데이터의 보관이나 원하는 정보의 검색을 쉽게 수행 가능

# 데이터베이스의 개념

## ◆ 여러 가지 유형의 정보가 연동되어 처리

- 예) 학교에서는 학생 신상 정보, 수업 및 성적 정보, 교수 정보, 행정 정보 등 여러 가지 데이터베이스가 운영
- 예) 백화점에서 상품 정보와 고객정보, 판매정보 등을 연동
- 예) 기업이나 은행에서 고객 및 거래 정보를 데이터베이스에 보관



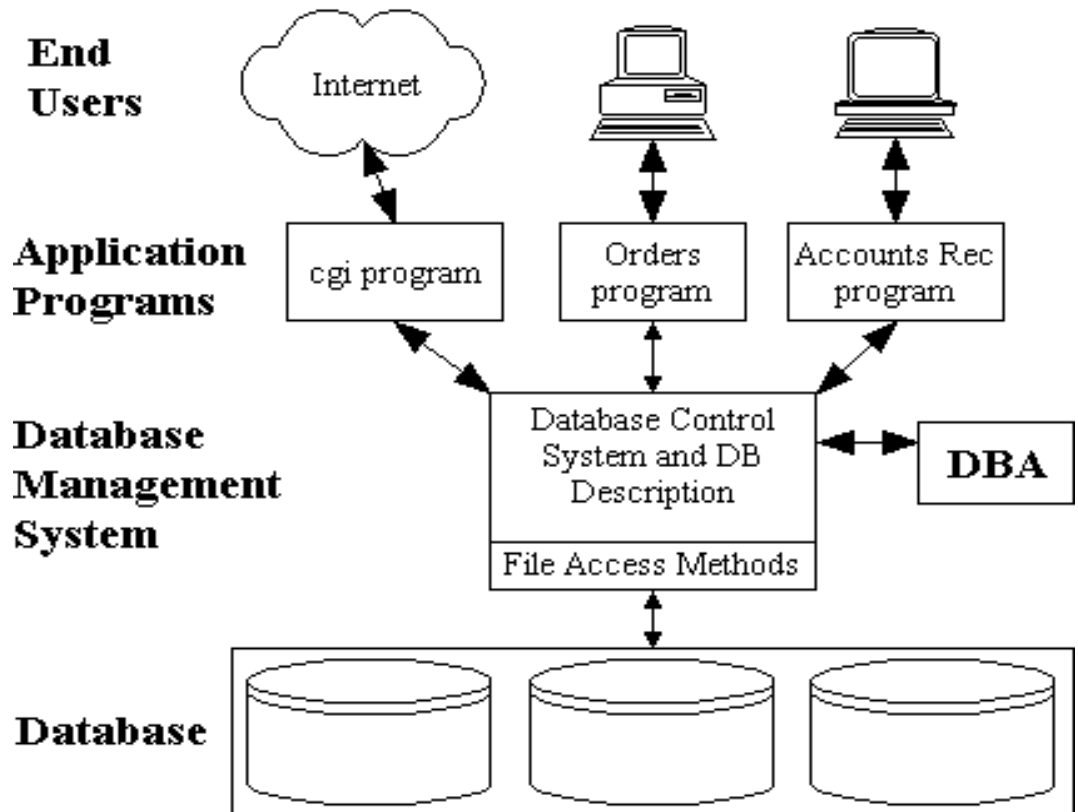
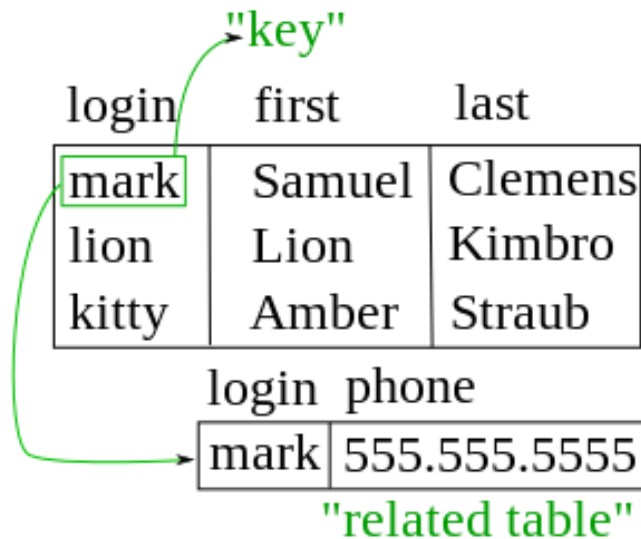


# 데이터베이스의 개념

### ◆ 데이터베이스의 개념

- 필요한 정보를 쉽게 검색하거나 사용할 수 있도록 데이터를 구조화하여 모아놓은 것
  - 일관된 형식으로 저장되어 다수의 사용자/프로그램이 공유 가능
- 데이터나 정보의 가치는 점차 중요해짐에 따라 데이터베이스의 필요성은 더욱 증가
  - 예) Walmart사나 Fedex사의 경우 상품이나 매장관리에 수억 개 이상의 레코드가 존재하고 수십 테라바이트 크기의 데이터베이스 사용
- 데이터의 공유를 위하여 데이터 중복을 최소화하며, 데이터 접근이 용이할 뿐 아니라 응용 소프트웨어의 개발시간도 단축

## 데이터베이스의 개념



# DBMS의 원리

## ◆ 관계형 데이터베이스의 계층적 구성

- 데이터의 가장 기본 단위는 문자 또는 숫자 => 필드(Field)를 구성
  - 각 필드는 필드명, 데이터형, 크기 등을 지정
- 각각의 가로 행에 해당하는 한 줄의 필드는 레코드(Record)
  - 이들 필드들은 서로 연관된 데이터를 가지고 있다
- 예) 그림 3-18 ‘학생정보’ 테이블
  - 첫 번째 세로 열 필드명이 ‘학생ID’이며, 데이터형은 문자, 크기는 7개
  - 각 행은 학생에 대한 정보를 가지고 있는 각 학생정보의 레코드

학생ID	이름	전화	이메일
0811021	김민지	010-6134-3568	kimmj@sm.ac.kr
0811041	나세리	010-9865-7622	naseri@sm.ac.kr
0811055	서보은	010-1373-2489	suhbe@sm.ac.kr
0811083	이은지	010-2832-7890	leeji@sm.ac.kr

# DBMS의 원리

- 인덱스 키(Index Key)
  - 테이블에서 특정한 필드 값을 가지는 레코드를 검색하기 위해 설정
  - 중복된 값을 가지지 않는 필드를 인덱스 키로 지정
  - 필요시 중복이 발생하지 않도록 복수 개의 필드를 인덱스로 지정
    - 예) '학생정보'의 경우 '학생ID' 를 인덱스로 사용
    - 예) '과목별\_성적부'의 경우 '과목코드'와 '학생ID', '이수학기'를 동시에 사용

과목코드	과목명	학점	강사	학생ID	이름	점수	성적	이수학기
06104	컴퓨터의 이해	3	10150	0811021	김민지	88	B+	200803
06104	컴퓨터의 이해	3	10150	0811041	나세리	95	A	200803
06104	컴퓨터의 이해	3	10150	0811055	서보은	97	A+	200803
06104	컴퓨터의 이해	3	10150	0811083	이은지	85	B	200803

개인별 성적표 : 김민지

- 컴퓨터의 이해 (2008.03) B+
- 멀티미디어 (2008.03) A
- 데이터베이스 (2008.09) B

# DBMS의 원리

## ◆ 데이터베이스 스키마(Database Schema)

- 데이터베이스의 전체적인 구조와 가능한 상태를 정의
- 그림 3-18의 예
  - 데이터베이스 파일은 3개 테이블로 구성
  - 테이블과 필드의 관계는 다음의 스키마로 정의 (인덱스 키는 밑줄)
    - 학생정보 (학생ID, 이름, 전화, 이메일)
    - 교수정보 (교수ID, 이름, 학과, 이메일)
    - 과목별\_성적부 (과목코드, 과목명, 학점, 강사, 학생ID, 이름, 점수, 성적, ...)
  - 각 필드에 대한 데이터의 속성 예
    - 학생ID (문자, 8개)
    - 과목명 (문자, 제한없음)
    - 이메일 (문자, 제한없음)
    - 학점 (숫자, 0..30),
    - 성적 (숫자, 0..100)

# DBMS의 원리

### ◆ 데이터베이스 관리시스템(DBMS)

- 데이터베이스의 추가, 수정, 검색 등의 작업을 하기 위한 시스템 인터페이스를 제공하며 관리해주는 소프트웨어
- DBMS(Database Management System)의 필요성
  - 지속적으로 레코드를 추가, 수정 또는 삭제 하는 일이 빈번하게 발생
  - 경우에 따라서는 새로운 스키마의 테이블을 추가하거나 스키마 정보를 변경해야 하는 일도 발생
  - 원하는 정보를 검색하거나 추출하기 위한 질의어도 필요.
- DBMS의 시스템 인터페이스는 응용 소프트웨어가 데이터베이스에 접속하여 사용하는 경우가 많이 있으며, 오퍼레이터나 일반 사용자가 데이터베이스를 관리하는데에도 필요.



# DBMS의 원리

### ◆ DBMS의 특징

- 데이터를 효율적으로 관리
  - 응용 프로그램과 데이터베이스의 관리는 독립적으로 이루어지며, 데이터를 통합하여 관리함으로써 데이터의 일관성 유지가 용이
- 데이터 접근을 효율적으로 관리
  - 여러 사람이 동시에 한 데이터에 접근하더라도 혼자 사용하는 것처럼 동시접근을 제어하며, 권한 없는 사용자로부터 접근을 차단하여 데이터 보안을 지킬 수 있다.
- 질의어를 통해 효율적인 데이터 검색이 가능
- 보고서 작성기를 통하여 원하는 형태의 보고서를 바로 작성가능
- 시스템 오류 등의 경우를 대비하여 백업 및 복구 기능 보유
- 다양한 사용자 인터페이스를 제공하고 있다.

# DBMS의 원리

## ◆ 질의어(Query Language)

- 특정한 정보를 요청하는 질문을 수행하기에 최적화된 언어
- SQL(Structured Query Language)이 널리 사용
- 테이블이나 레코드의 내용을 수정하고, 원하는 레코드나 필드에 대해 검색하며, 결과를 원하는 리포트 형태로도 작성
- SQL의 사용 예

CREATE TABLE

SELECT 이름 FROM 학생정보      // 학번이 0811021인 학생의 이름  
WHERE 학생ID = '0811021'

SELECT 학생ID FROM 과목별\_성적부 // 멀티미디어 성적이 90점 이상인 학생ID  
WHERE 과목명 = '멀티미디어' AND 점수 >= '90'

SELECT 이메일      // 멀티미디어 수업듣는 학생의 이메일 주소  
FROM 학생정보 a, 과목별성적부 b  
WHERE b.과목명 = '컴퓨터의 이해' AND a.학생ID = b.학생ID

# DBMS의 종류 및 활용

## ◆ DBMS의 종류

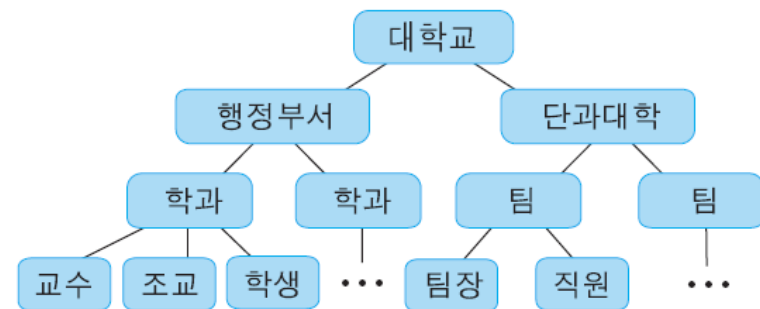
- 관계형 데이터베이스(Relational Database)
  - 지금까지 가장 널리 사용되는 방식
  - 테이블을 기반으로 하는 데이터 모델로서 표현이 간단하며 구조를 이해하거나 사용하기 쉽다는 장점
  - 1970년 IBM 연구소의 에드가 카드(E.F.Codd)가 제안
    - 초창기에는 Ingres와 System R이 대표적인 시스템, 질의어로 SQL을 사용
  - 현재 판매되는 대부분의 DBMS에서 사용하고 있는 모델
    - 대표적인 DBMS로는 MS Access, MS MySQL, Oracle, Informix 등

# DBMS의 종류 및 활용

- 객체지향 데이터베이스(Object-oriented Database)
  - 1980년대에는 객체지향 프로그래밍 패러다임을 기반으로 하는 모델
  - 관계형 데이터베이스의 단점
    - 텍스트와 이미지 동영상 등이 섞여 있거나 용량이 매우 큰 데이터
    - 트리 구조처럼 구조가 동일하지 않는 비정형적인 구조의 데이터
  - 멀티미디어 또는 복잡한 구조의 데이터를 위하여 사용
  - 대표적인 객체지향 DBMS : Ontos, GemStone, FastObjects, Versant
  - 객체관계형 DBMS : Informix, Oracle, UniSQL, JDataStore 등

이름	전화	이메일
김민지	010-6134-3568	kimmj@sm.ac.kr
나세리	010-9865-7622	naseri@sm.ac.kr
서보은	010-1373-2489	suhbe@sm.ac.kr
이은지	010-2832-7890	leeji@sm.ac.kr

(a) 테이블 구조



(b) 객체지향 트리구조

# DBMS의 종류 및 활용

## ◆ DBMS의 활용

- 전통적인 정보처리 분야
  - 개인이나 SOHO(Small Office Home Office) : 재고관리, 고객관리, 회계/재무 관리, 물류관리 등을 위한 데이터베이스
  - 일반 기업 : ERP 시스템이나 상거래 시스템을 위한 데이터베이스
  - 국가기관이나 공공기관 등의 대형 SI(시스템 통합) 사업
    - 대규모 데이터베이스 구축이 필수, 대표적인 사례로서 행정전산망, 교육전산망, 금융전산망 등의 대규모 국가정책사업
- 기술발전에 따라 새롭게 등장하는 응용분야
  - 이미지, 그래픽, 오디오, 비디오 등의 데이터를 요구하는 CAD 및 멀티미디어 데이터베이스, 웹 검색 데이터베이스가 대표적인 사례
  - GIS 분야에서는 공간정보와 관리정보를 연동하는 데이터베이스 사용
  - 소프트웨어 공학 라이브러리의 경우에는 프로그램 데이터베이스
  - 비정형화된 구조의 데이터인 경우에는 XML 형식의 데이터베이스

## DBMS의 종류 및 활용

### ◆ 데이터베이스에 관련된 인력

- 데이터베이스 관리자(Database Administrator) : 여러 부서의 다양한 요구를 만족시키기 위해 일관성 있는 데이터베이스를 생성하고 관리하는 일을 결정하고 총괄하는 사람
- 응용 프로그래머 : 요구되는 작업을 수행하기 위하여 데이터베이스를 사용하는 특정 프로그램이나 인터페이스를 구현하는 사람
- 데이터베이스 설계자 : 데이터 모델을 분석하여 데이터베이스 모델을 설계하고 수정이나 보완 작업을 직접 수행하는 사람
- 오퍼레이터 : DBMS가 실행되는 컴퓨터를 관리하는 사람
- 일반 사용자 : 데이터를 질의하거나 보고서를 생성하기 위하여 데이터베이스를 사용하는 사람

## 3.5 유틸리티 소프트웨어

- ◆ 범용 유틸리티 소프트웨어
- ◆ 네트워크 유틸리티



## 범용 유틸리티 소프트웨어

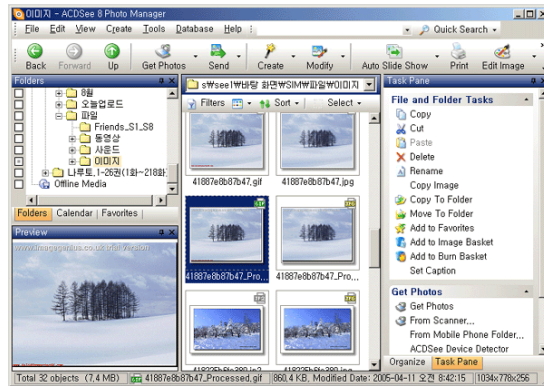
### ◆ 범용 유틸리티(Utility) 소프트웨어

- 사용자가 컴퓨터를 사용하거나 관리하는데 필요한 다양한 기능을 단독으로 수행하는 프로그램
- 대표적인 기능으로는 파일관리, 디스크관리, 시스템 상태 관리, 그리고 네트워크 관리 및 이에 따른 보안관리 기능 등

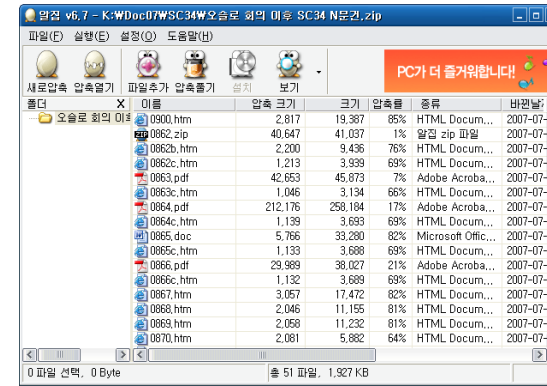
### ◆ 파일관리를 위한 기능

- 파일의 목록을 보여주고 파일을 복사하거나 이름을 바꾸고 삭제하며 DVD에 저장하는 등의 기능
- 별도로 파일관리 유틸리티로 제공되는 대표적인 제품
  - ACDSsystem의 ACDSsee, 국내에서는 이스트소프트의 알씨
- 파일의 압축 및 복원 기능은 필수적
  - Winzip이 가장 대표적, 국내에서는 이스트소프트의 알집도 많이 사용

# 범용 유틸리티 소프트웨어



ACDSee 프로그램



알집 프로그램

## ◆ 디스크 관리 기능

- 디스크 상태 보기 : 디스크의 사용 현황을 분석하여 필요 없는 파일을 삭제하거나 문제점을 해결하는 기능, 포매팅 기능도 포함
- 디스크 조각 모으기 : 디스크에서 사용하지 않는 작은 조각들을 모아 서 파일공간으로 사용하기 쉽도록 만들어 주는 기능

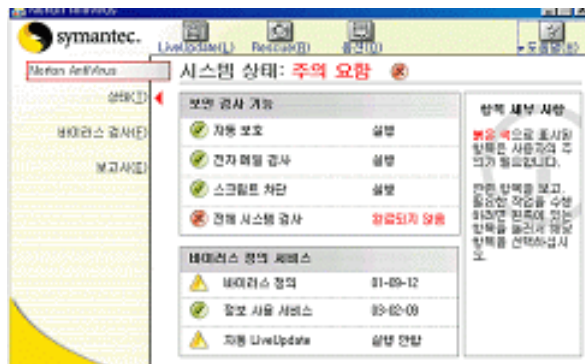
# 범용 유틸리티 소프트웨어

## ◆ 시스템 상태 보기

- 컴퓨터 하드웨어, 주변기기나 시스템 소프트웨어의 상태 보기
- 시스템의 문제를 진단하여 오류 수정까지 하는 기능이 포함
  - 디스크의 파일을 백업을 해주고 디스크 오류 발생시에 복구까지

## ◆ 종합 유틸리티 프로그램

- Norton Utility가 대표적, 그 외에 Symantec사의 PCAnywhere, 국내 회사인 이스트소프트사의 알툴즈



(a) Norton 유틸리티



(b) 알툴즈 프로그램

# 네트워크 유틸리티

## ◆ 네트워크 연결 도구 및 통신용 프로그램

- 전화접속 또는 유무선 네트워크에 접속
- KT의 네트워크 접속 프로그램이 대표적인 예

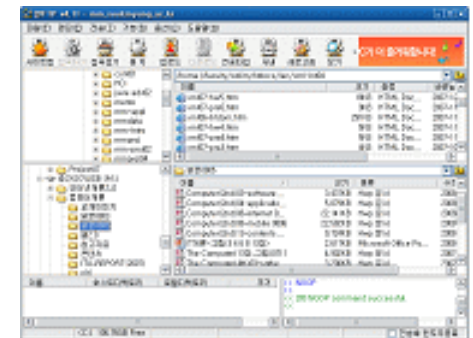


## ◆ 호스트 접속 프로그램

- 네트워크 호스트에 접속하여 텔넷(Telnet), 파일전송(FTP), 원격제어, 네트워크 프린터관리 등 다양한 서비스를 수행
- 이스트소프트사의 알 FTP, IPSwitch사의 WS FTP, InterSoft 사의 Secure Netterm이 많이 사용

## ◆ 네트워크 관리 및 분석도구

- 네트워크의 상태나 성능을 검사하고 관리



알 FTP

## 네트워크 유틸리티

### ◆ 인터넷 관련 소프트웨어

- 인터넷 접속 프로그램, 웹 브라우저, 메신저 및 통신 프로그램 등

### ◆ 보안관련 유틸리티 소프트웨어

- 바이러스를 방지하고 제거하기 위한 백신 프로그램
- 스파이웨어, 애드웨어 등의 악성코드를 제거하는 보안 유틸리티
- 안철수연구소, 하우리, McAfee, Symantec 등에서 제공

### ◆ 웹 접근 제한 프로그램

- 인터넷에서 특정사이트를 제한, 특정내용 차단, 팝업 윈도우 제한
- 스팸메일을 차단하는 이메일 필터링

# Q & A