


Comp 310
Computer Systems and Organization

Lecture #23

Modern OS Overview

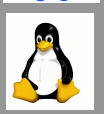
Prof. Joseph Vybihal




Announcements

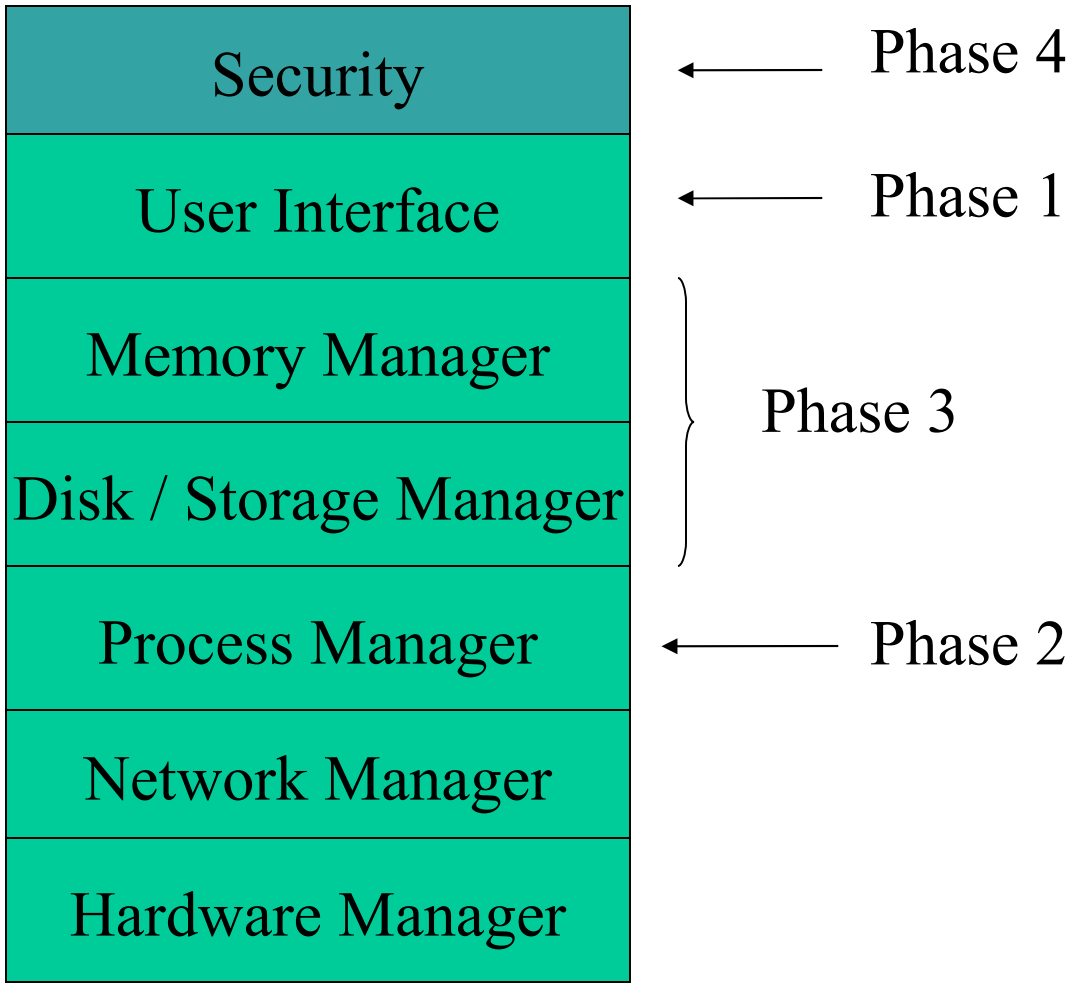
- Thursday last class!
  - Will talk about the final exam
  - Exam tutorials (next week)
  - Lecture on OS security
- Course evaluations...



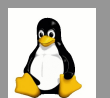



Basic OS Architecture

(Course Table of Contents)








Part 1

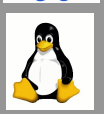
What are Operating Systems?




Tri-Nature

- Low-level interface with the hardware
  - Performance & compatibility
- High-level API to the software & user
  - Features
- Manager to the system
  - Security & Paradigms






Modes

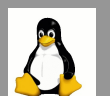
- Real:
  - Assembler commands have full access to all addresses and ports in the computer.
  - Protected and Virtual modes turned off or non-existent
- Protected:
  - CPU has boundary registers
  - Assembler commands can only reference addresses between the boundary registers




Paradigms

- Stand-alone
  - Real mode computer
  - One process executes
- Multi-processing
  - Protected mode computer
  - Many processes (and users) executing
- Virtual
  - Memory and addressing
  - Simulated “read mode”
- Distributed processing
  - Execution occurs across more than one computer






Part 2

MSDOS

Architecture of MSDOS

- Real mode only
- Single process
- TSR abilities

Kernel
API Table
Command.com

- Interrupts
- I/O function
- File system
- Jump addresses
- Expandable
- The shell
- Basic Lang?

RAM

OS
Video
Free
Zero

- Byte = colour
- Pixel = address
- Move code addr

- Single process fills free space
- R-T Stack, Heap, Code and Static data
- No PCB, simple task switch OS/Process

- RAM address directly mapped to card pins in motherboard's slot or port (COM, PRN)

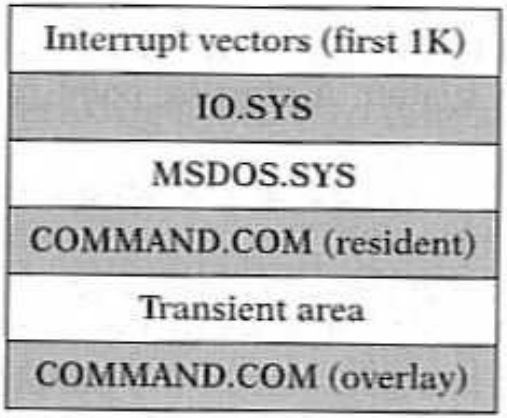
ProDOS

```
]CATALOG
DISK VOLUME 254

 A  006  HELLO
 T  018  ANIMALS
 I  004  APPLESOFT
 I  002  APPLEVISION
 B  010  BOOTI3
 A  006  BRIAN'S THEME
 B  003  CHAIN
 I  003  COLOR DEMO
 A  003  COLOR DEMOSOFT
 I  003  COPY
 B  003  COPY.OBJ0
 A  003  COPYA
 B  002  FID
 B  005  FPBASIC
 B  005  INTBASIC
 B  002  LITTLE BRICK OUT
 B  003  MASTER CREATE
 B  027  MUFFIN
]
```



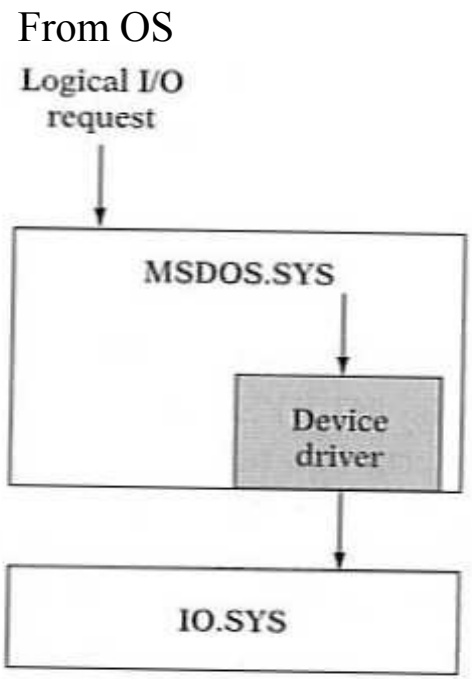

MSDOS Kernel



Heap space for Kernel




MSDOS.SYS & IO.SYS



Handshaking protocols, polling protocols

Byte streaming routines, hardware dependant

Processes can write directly to hardware devices without the requirement to use the .SYS library.

Example:

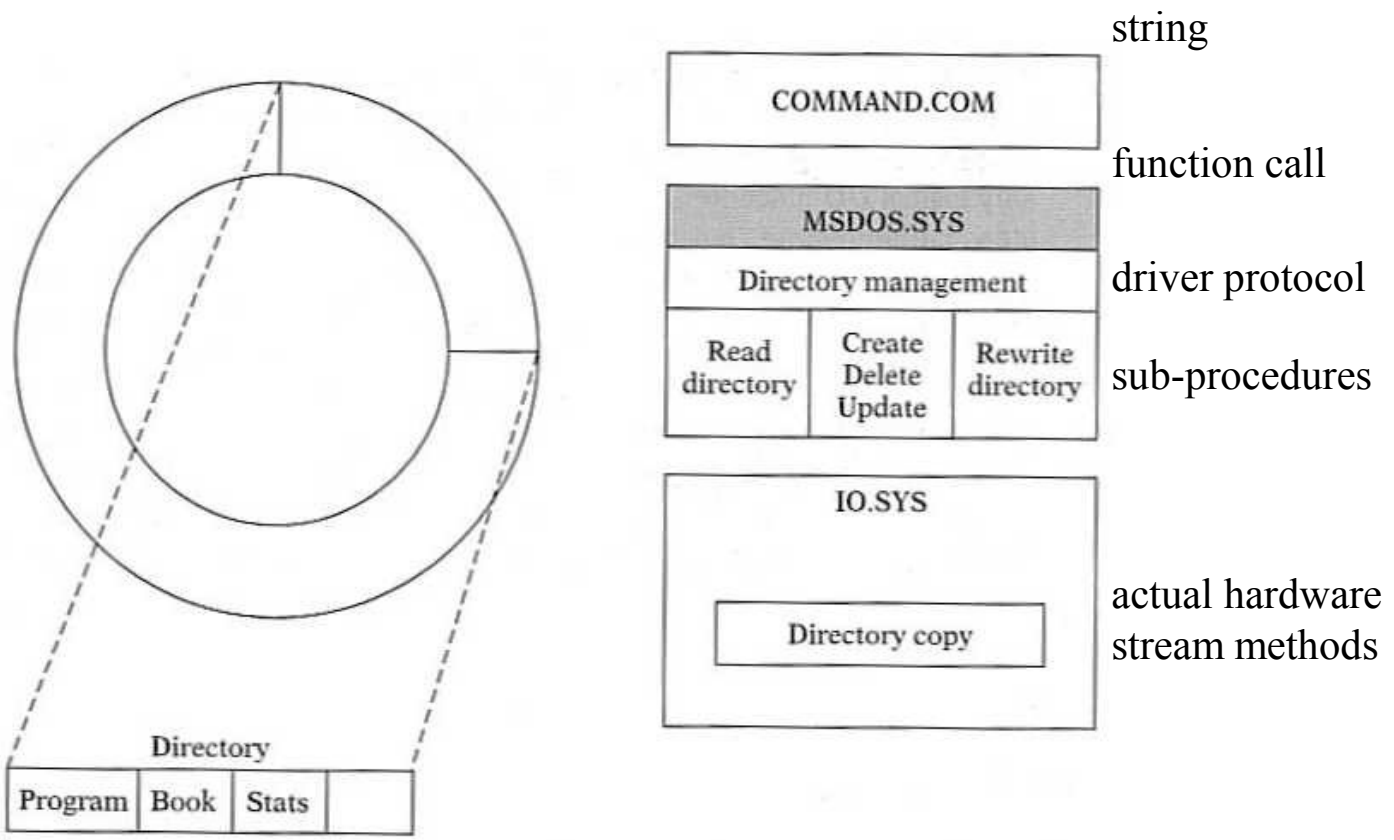
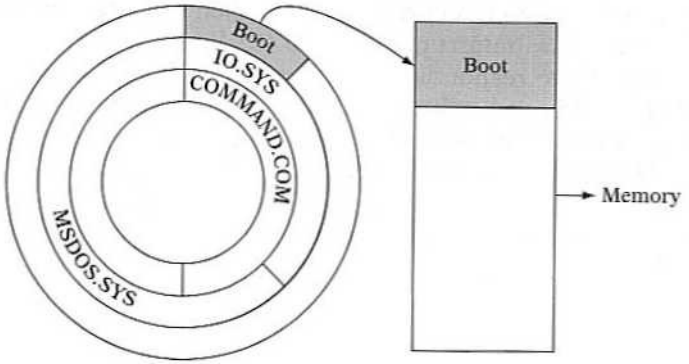


FIGURE 11.5

MSDOS.SYS is responsible for directory management.

a. Following a MKDIR command, COMMAND.COM calls MSDOS.SYS.

The File System



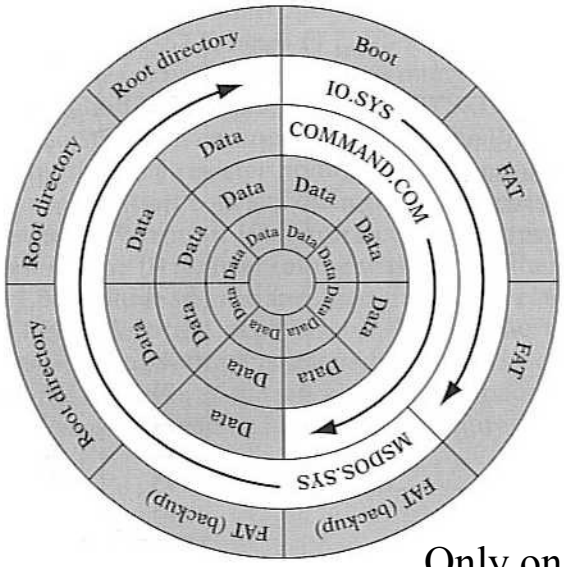
Directory

File name	First cluster
IO.SYS	8
MSDOS.SYS	11
COMMAND.COM	16
MYFILE	20
LETTERS	22

FAT

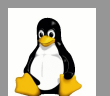
Cluster	Pointer
0–7	System
8	9
9	10
10	FF
11	12
•	•
•	•
•	•
20	21
21	24
22	23
23	FF
24	FF
25	

Note: FF means end of chain



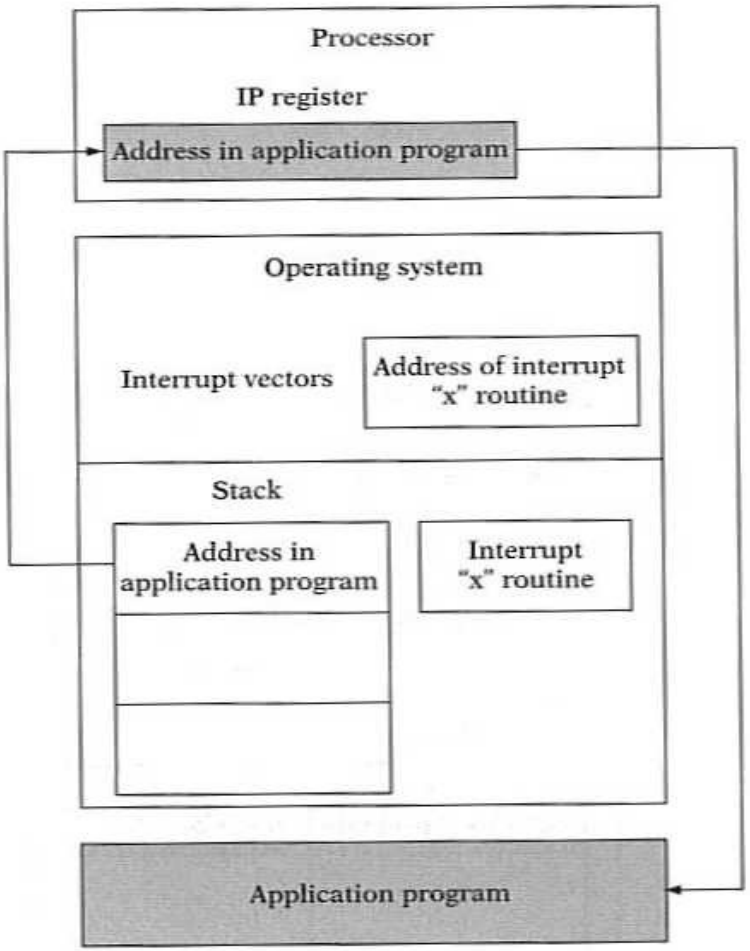
Only on disks formatted with an OS






Interrupt Vector

- Interrupt invoked
  - SYSINT #
  - Hardware signal
- Process IP stacked
- IP = vector(#)
- When done
  - IP = pop(stack)
- Process continues





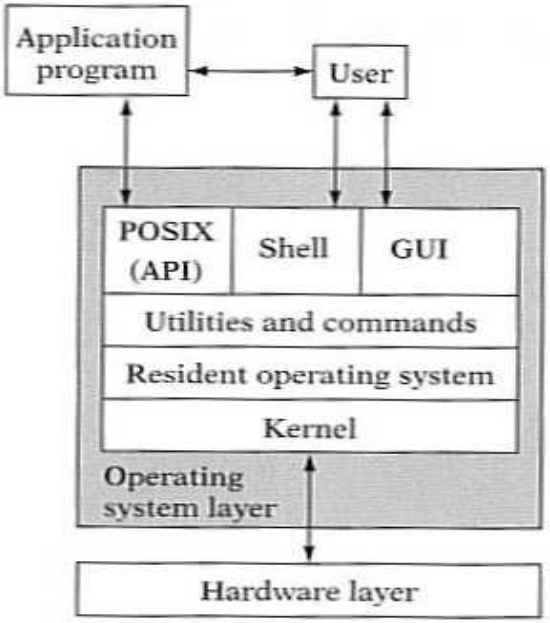

Part 3

Unix and Linux



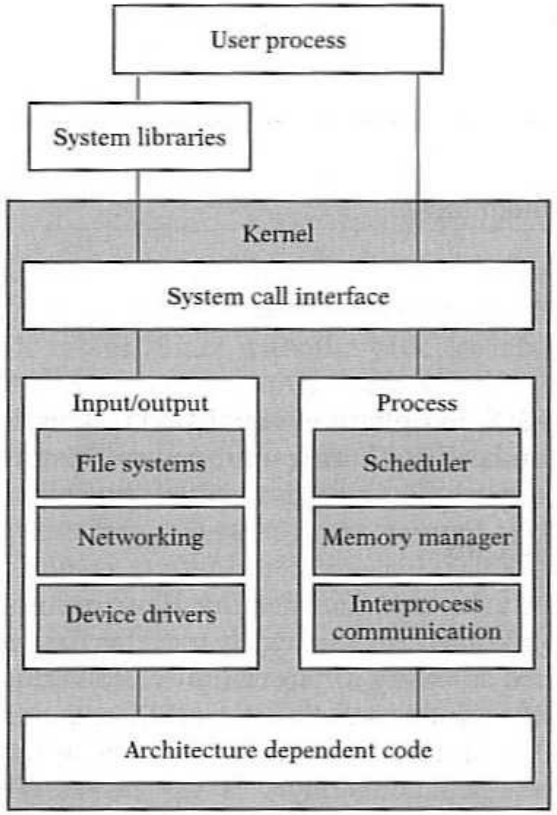

Unix & Linux Architectures

- Protected Mode
- Multi-process
- Virtual and Distributed



Unix

Simple Command Paradigm

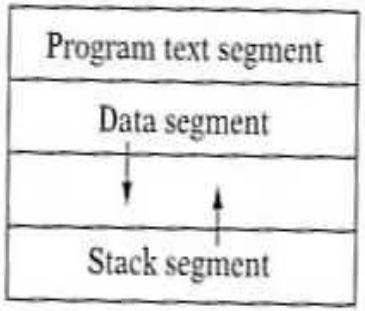


Linux

Layered OS Paradigm

The Process

A Process

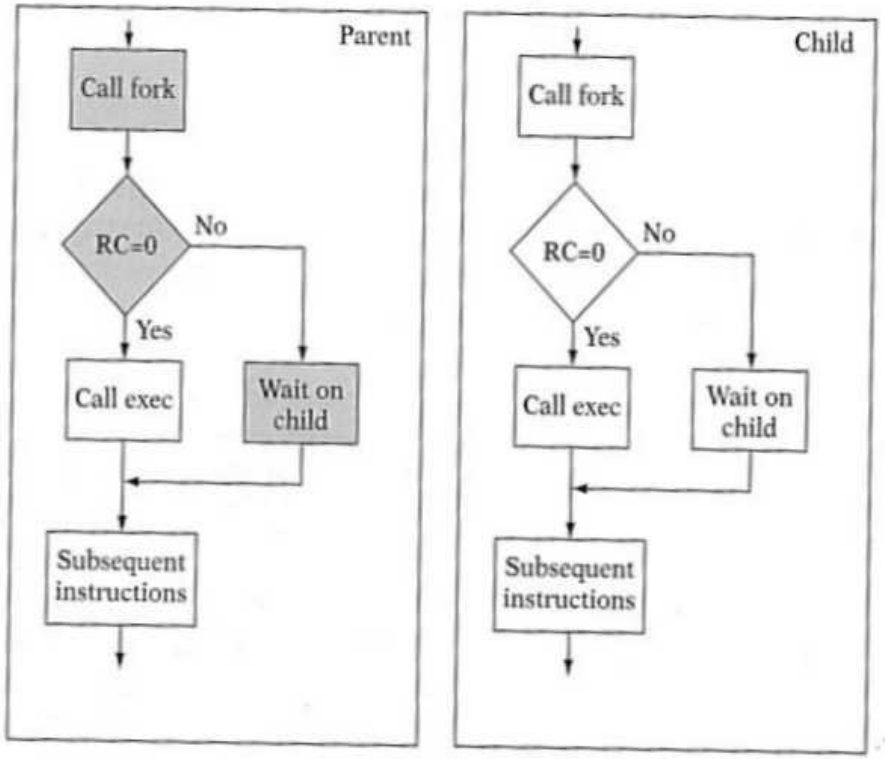


PCB

One entry per process.

Each entry contains:

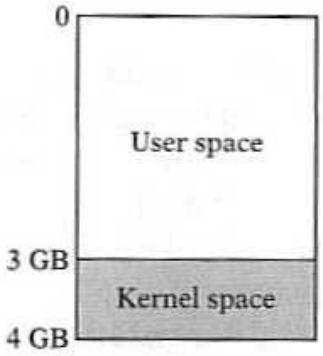
- Process number
- Process state (ready, waiting)
- Process priority
- Event number process is waiting on
- Text table address
- Data segment address
- Stack segment address
- System data segment address



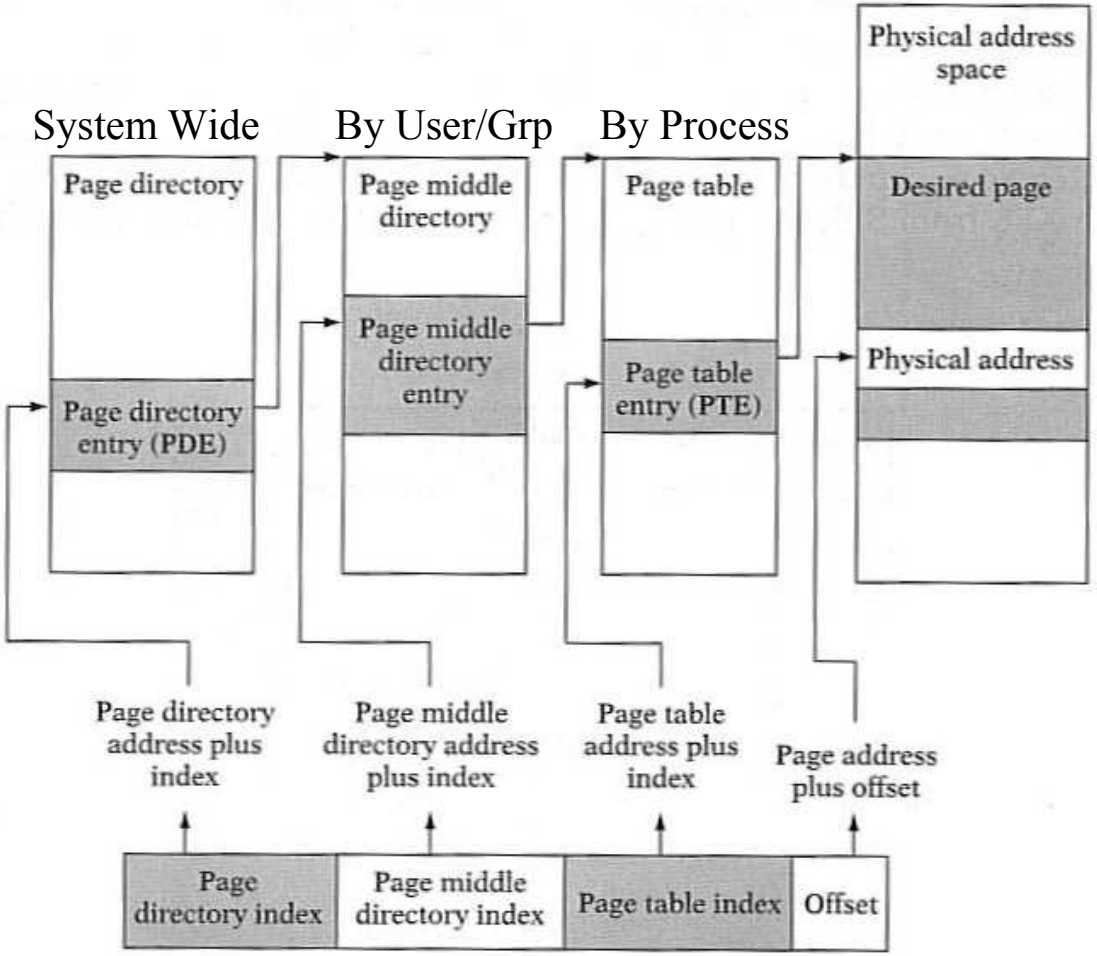
Fork() used to invoke a process

Special system process called Init() which is used to fork() and then exec() from (if needed).

Memory Management



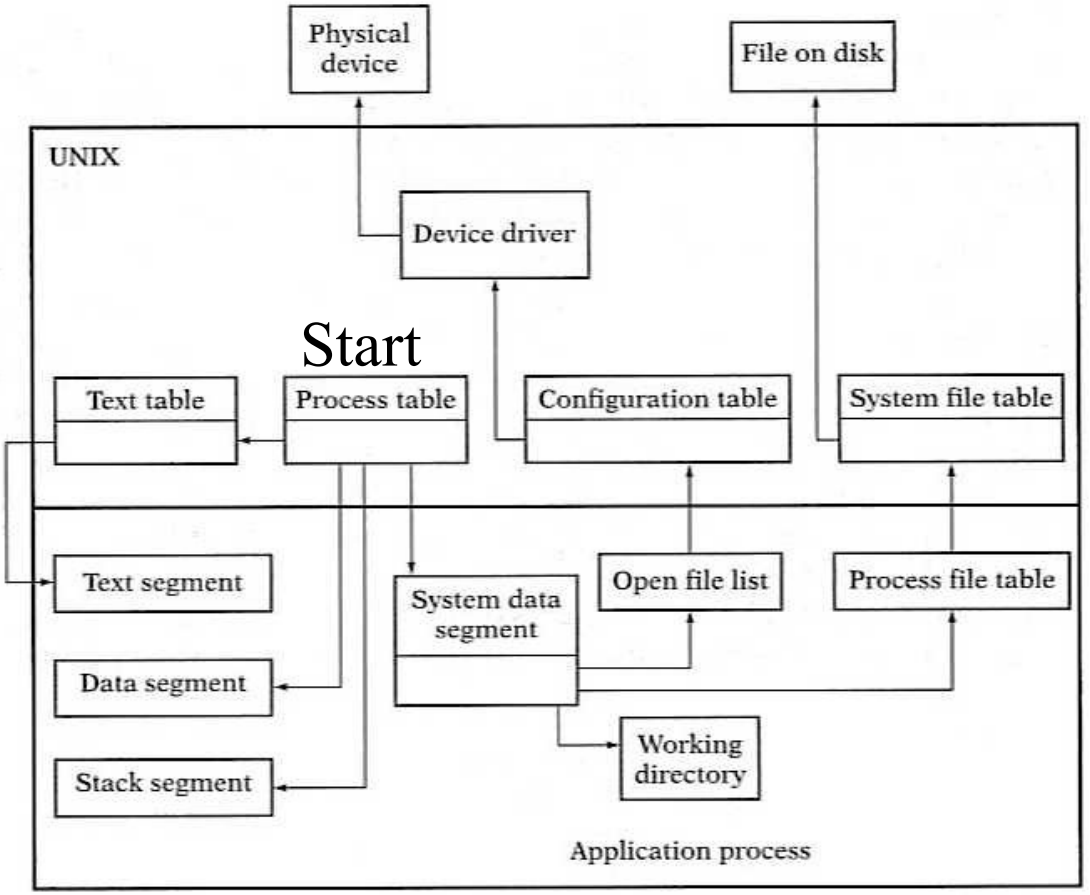
In the Virtual World



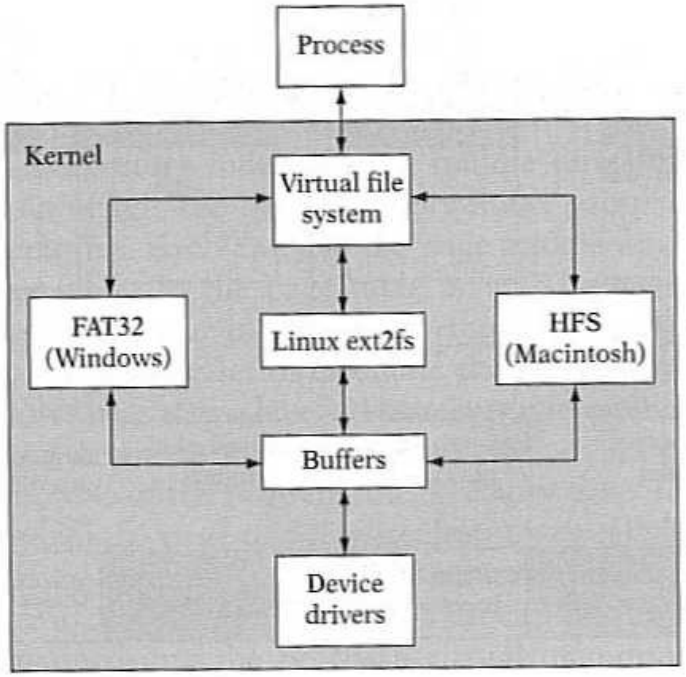
3 Levels of Paging




Run-time Tables



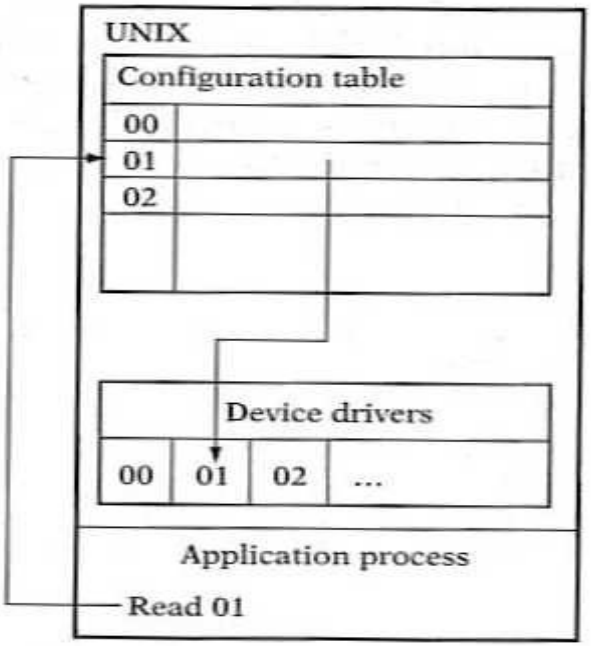
I/O System



Linux

Layered and Modularized

- VFS: Standardized function calls
  - Determines file system type
  - Calls proper functions using correct protocols
- Buffering system provides improved services
- Driver functions directly communicate with hardware



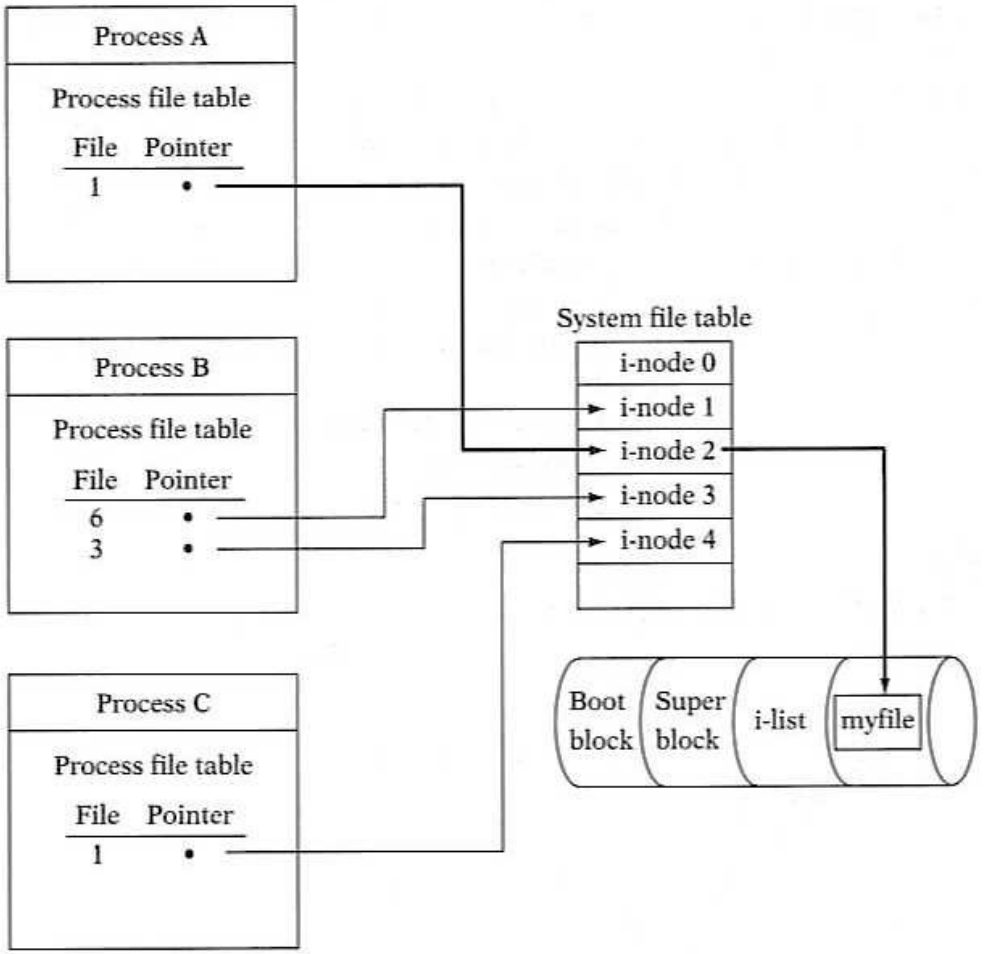
Unix

Remember MSDOS?

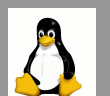
The Configuration table provides for a PCB based interrupt table for security reasons.




File System


Process A
Process file table
File Pointer
1
Process B
Process file table
File Pointer
6
3
Process C
Process file table
File Pointer
1
System file table
i-node 0
i-node 1
i-node 2
i-node 3
i-node 4
Boot block
Super block
i-list
myfile

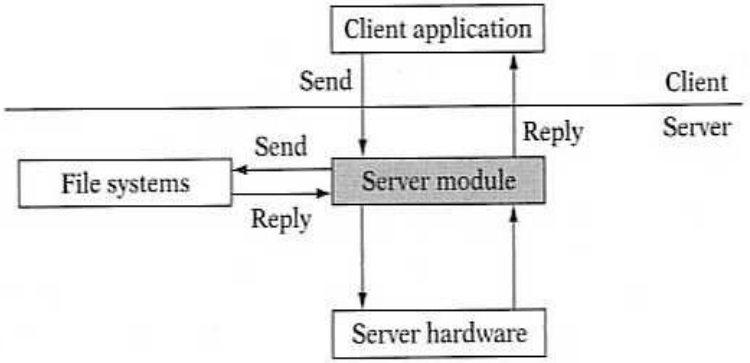





Part 4

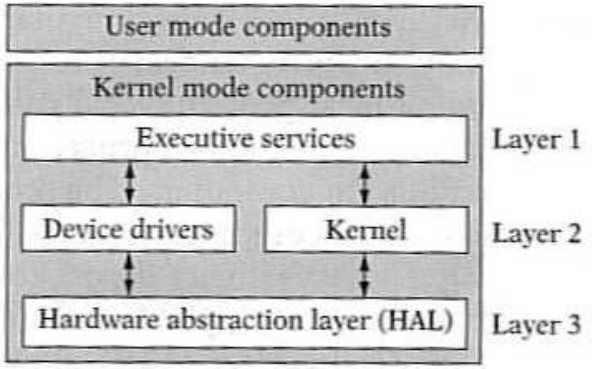
Windows XP

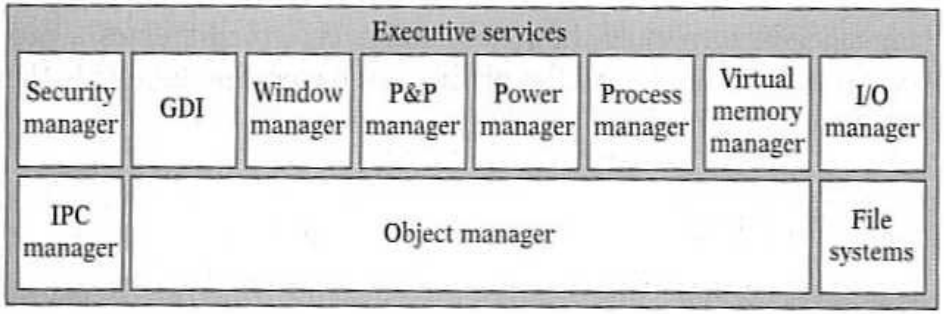
Architecture



A Client Server System

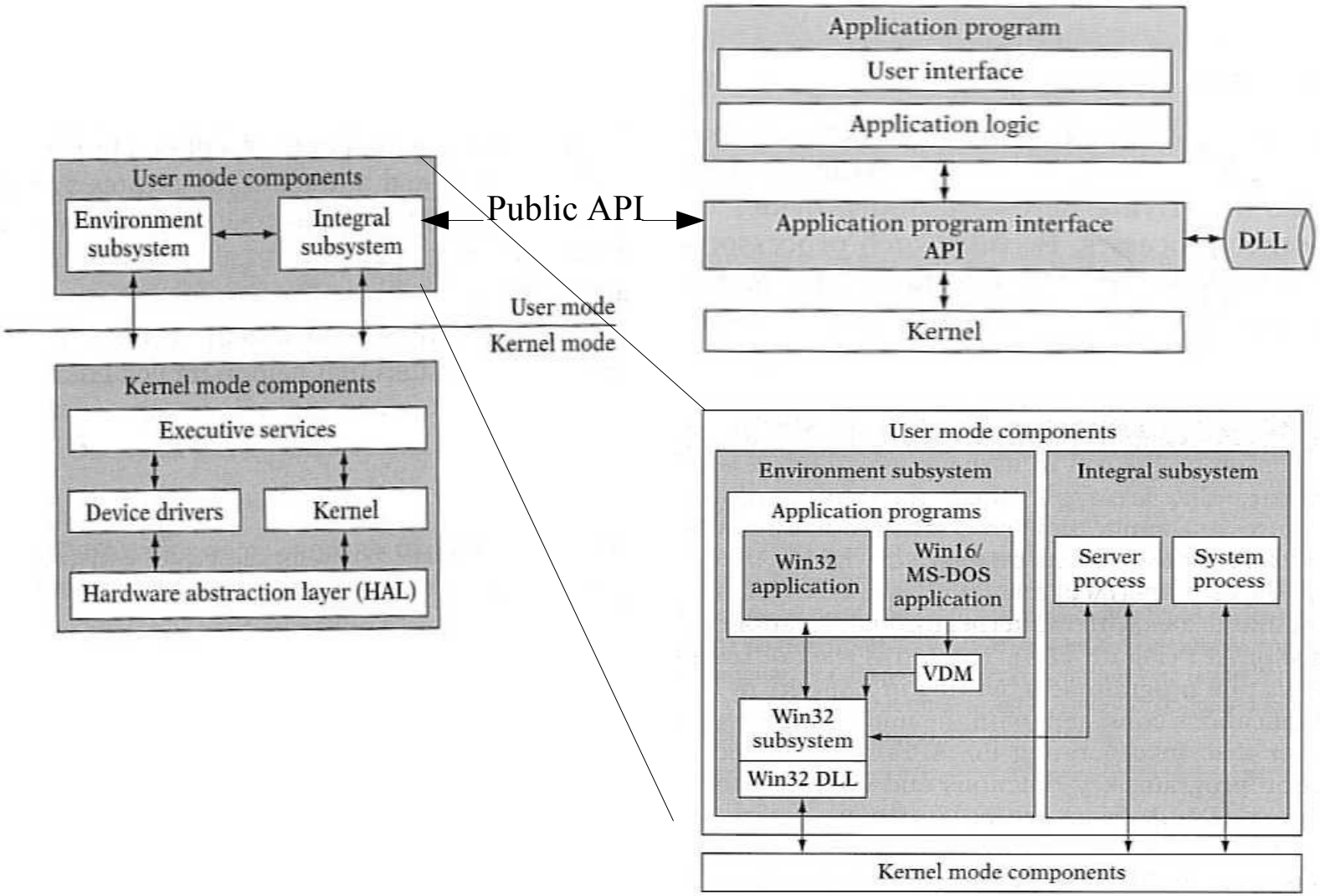
Kernel





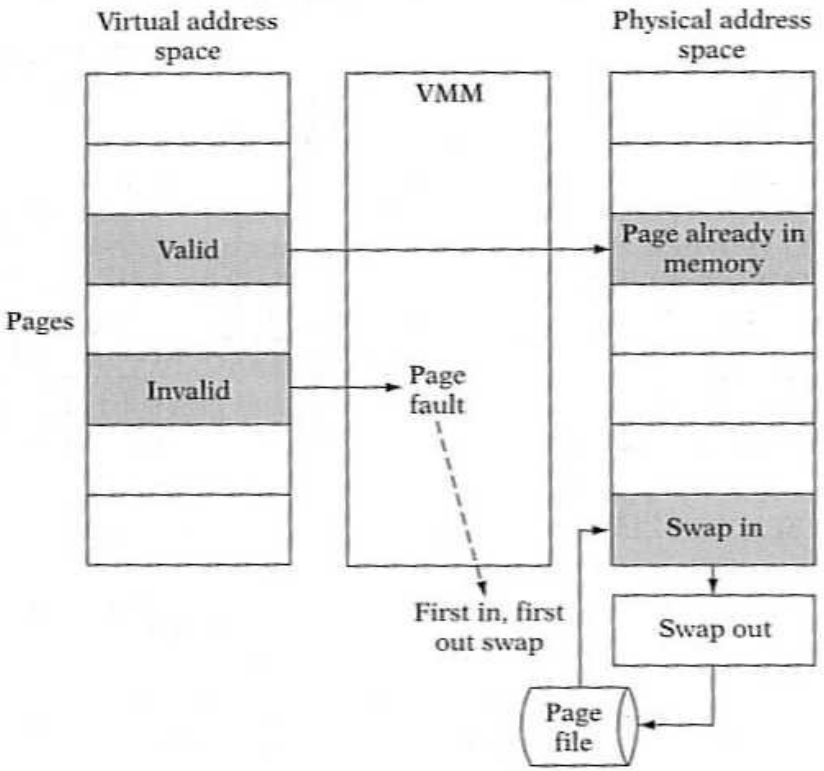
An API layer for kernel

The Process

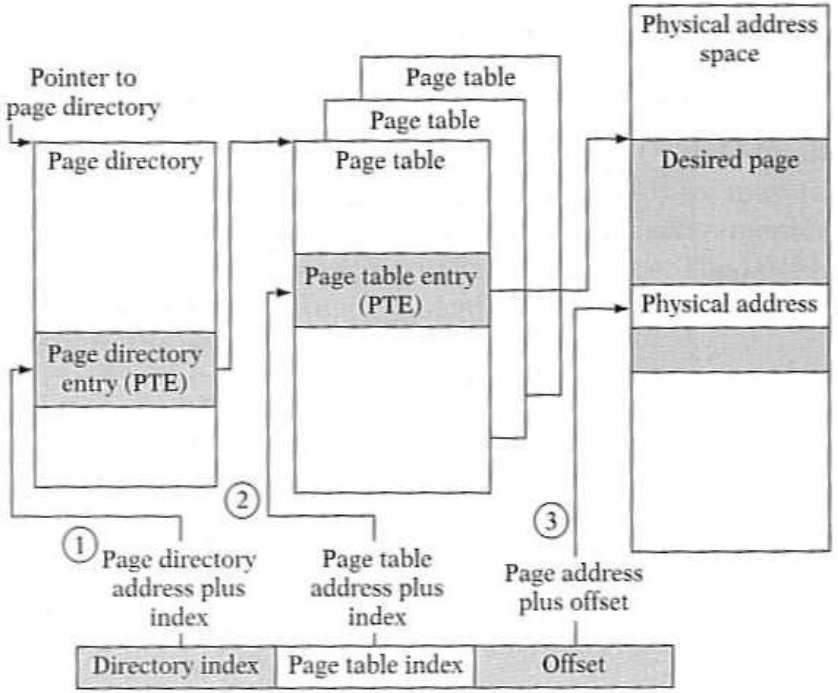

Application program
User interface
Application logic
Application program interface
API
DLL
Kernel
Public API
User mode components
Environment subsystem
Integral subsystem
User mode
Kernel mode
Kernel mode components
Executive services
Device drivers
Kernel
Hardware abstraction layer (HAL)
User mode components
Environment subsystem
Application programs
Win32 application
Win16/ MS-DOS application
VDM
Win32 subsystem
Win32 DLL
Integral subsystem
Server process
System process
Kernel mode components



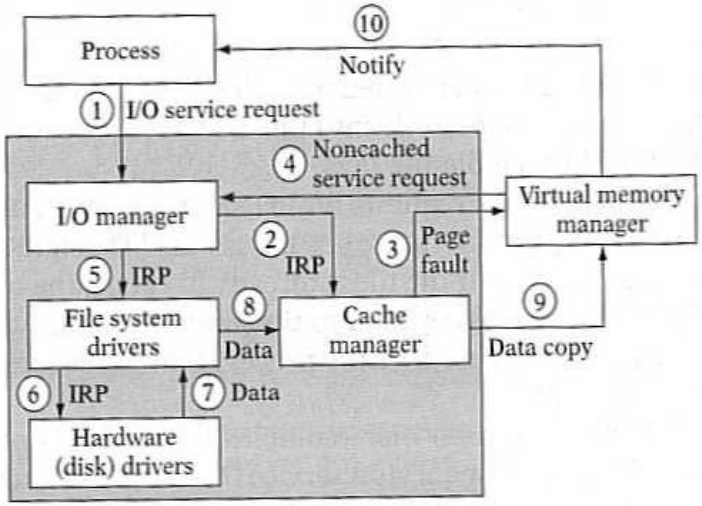

Virtual Memory



Memory Management



Addressing

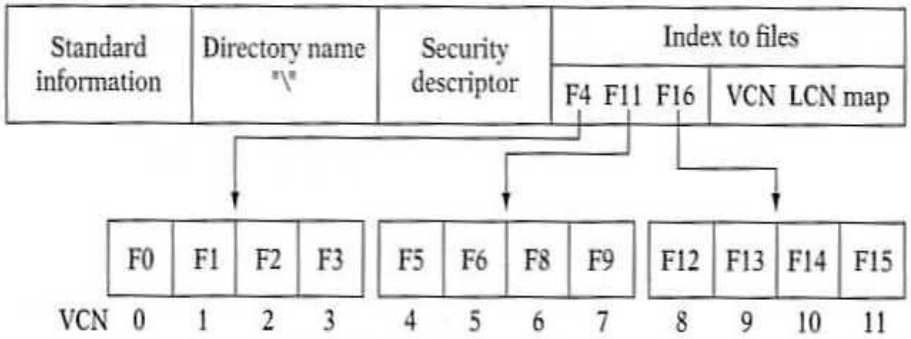


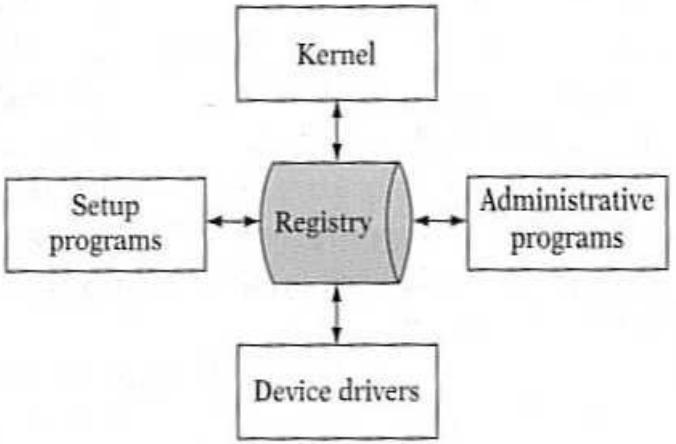
Cache




I/O and File System

FAT

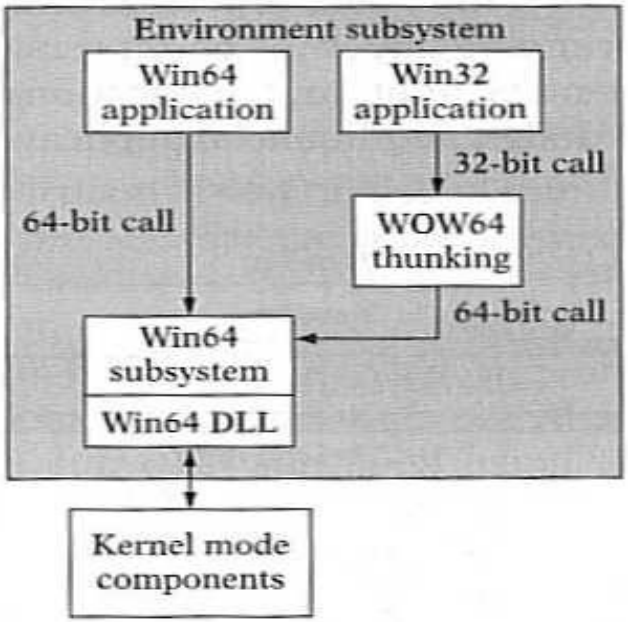




Registry




64 bit Thunking



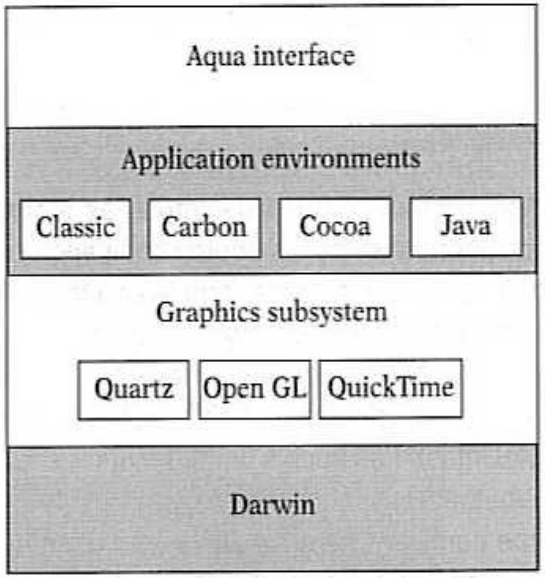


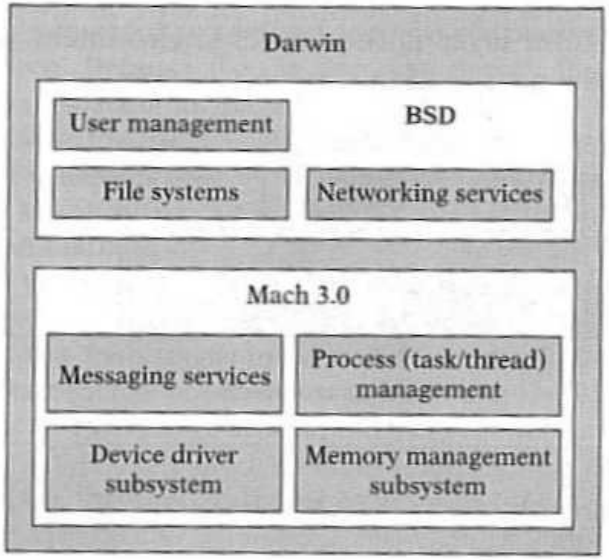

Part 5

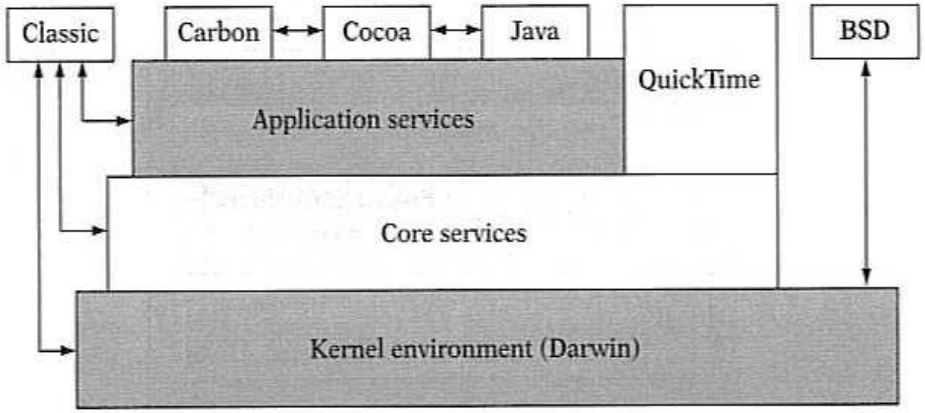
MAC OS X



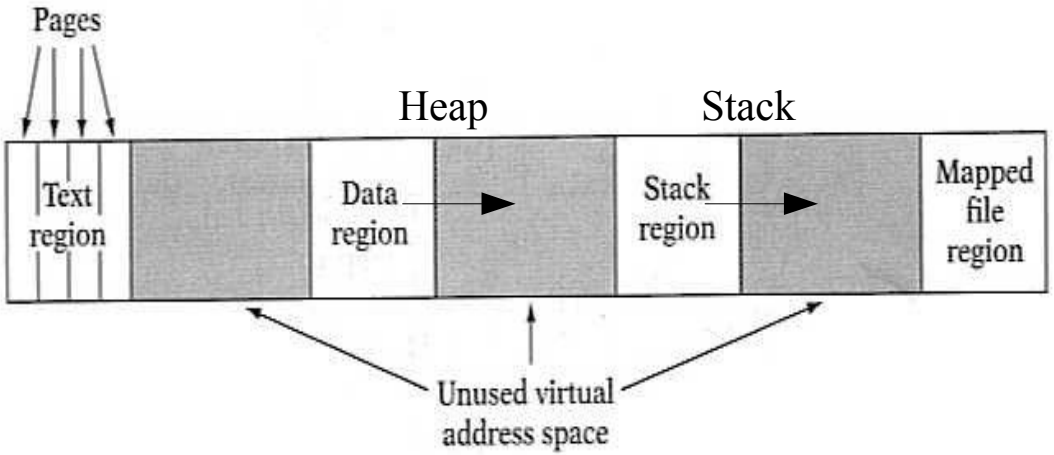

Architecture



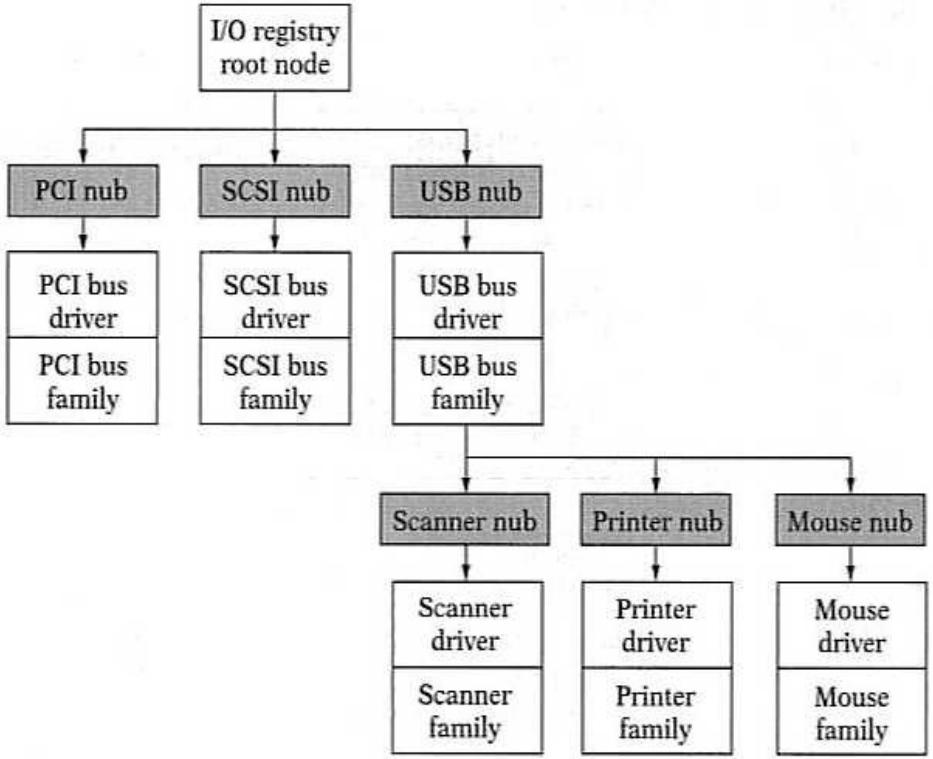


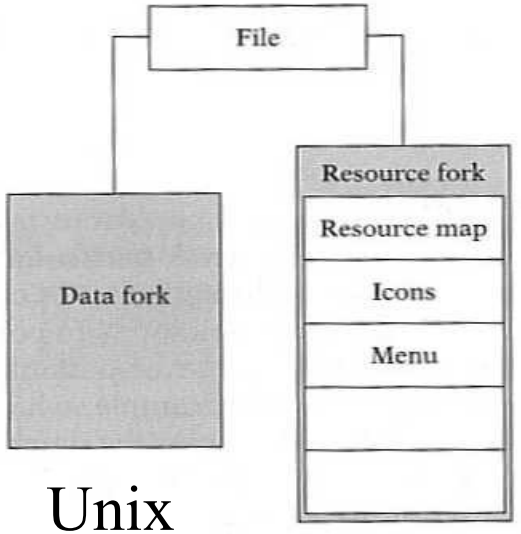


The Process


Pages
Heap
Stack
Text region
Data region
Stack region
Mapped file region
Unused virtual address space

File and I/O System





Unix

MAC




Part 3

At Home




Things to try out

1. Get your hands on lots of OS systems and play around.