Operating Systems
Unit - I

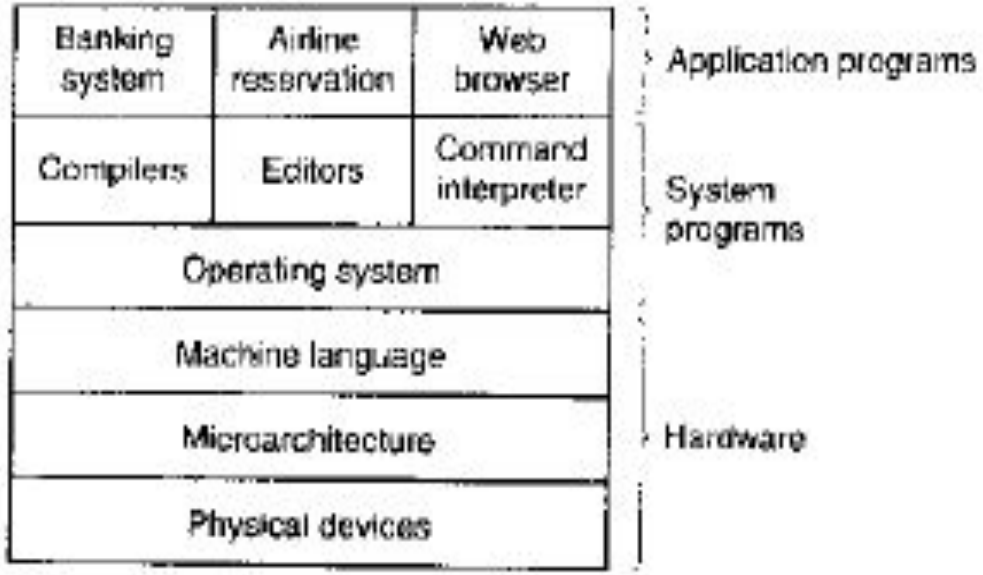
Dr.S.Nirmala Devi
AP/ CS
GASC, KPM

Computer

- Mordern Computer System Consists of
  - One or more processors
  - Main Memory
  - Disks
  - Printers
  - Keyboard
  - Display
  - Network Interfaces
  - I/O Devices

Operating System

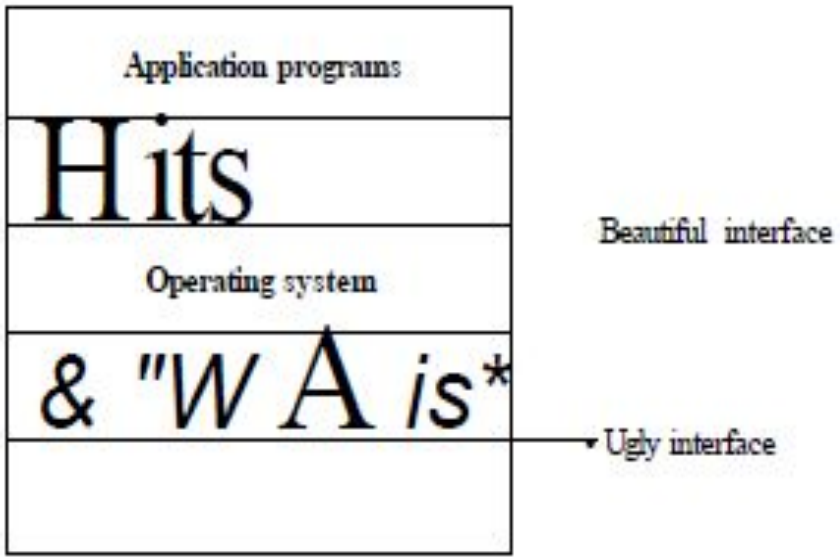
- Software
- Manage all the devices
- Acts as an interface between user and hardware



Operating System

- Operating System as an Extended Machine
- Operating System as an Resource Manager

Operating system as an Extended Machine



Abstraction required at the user level
Defining and implementing the abstractions
Use abstractions to solve problem

Operating System as a Resource Manager

- Allows multiple programs to run at the same time
- Manages and Protects the memory
- Manages I/O Devices

Operating System as a Resource Manager (contd)

Resource Management

Time

Different programs or users take turns and use it

Space

Each process gets part of the resoure

Eg:-

Main Memory divided among several programs

History of Operating Sytem

- First Digital Computer designed by

Generations:

- (1945-55) Vacuum Tubes
- (1955-65) Transistors and Batch Systems
- (1965-1980) ICs and Multiprogramming
- (1980-Present) Personal Computers

First Generation (1945 – 55) Vaccum tubes

First Generation (1945 – 55) Vacuum tubes

- Vacuum Tubes are used to build the computers in early days
- Group of people designed, built, programmed, operated and maintained each machine
- Programming was done in Machine Language
- Operating system not available

Second Generation (1955 – 65) Transistors and Batch systems

- Mainframe Computers
  - Developed using Transistors
  - Kept in a air-conditioned room
  - Only large corporations, universities could buy it.

Batch System

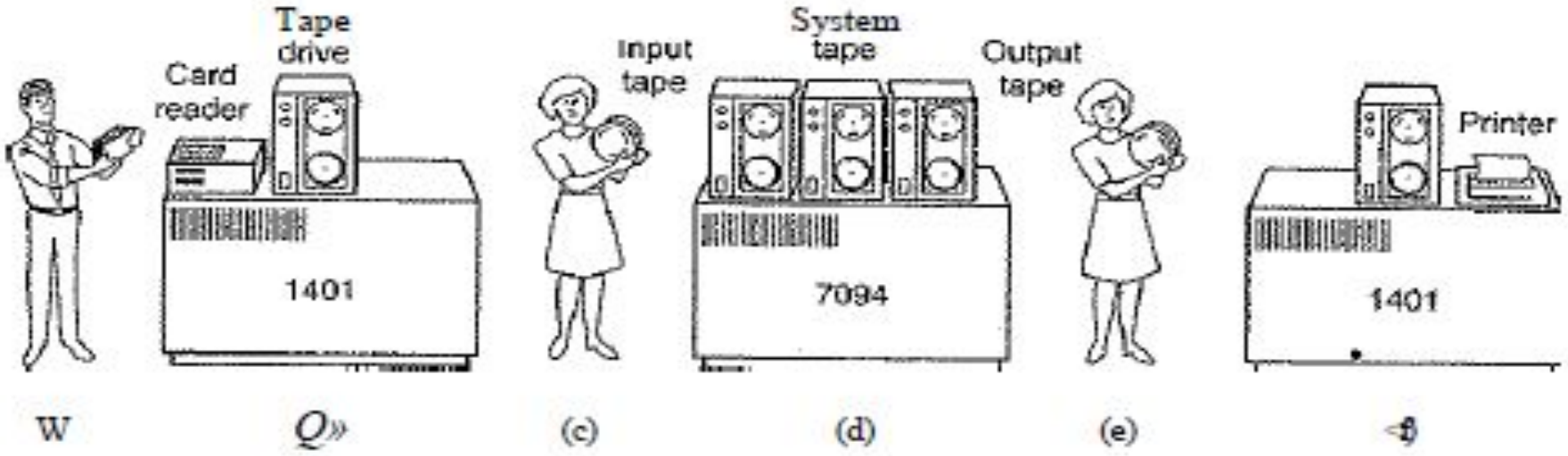
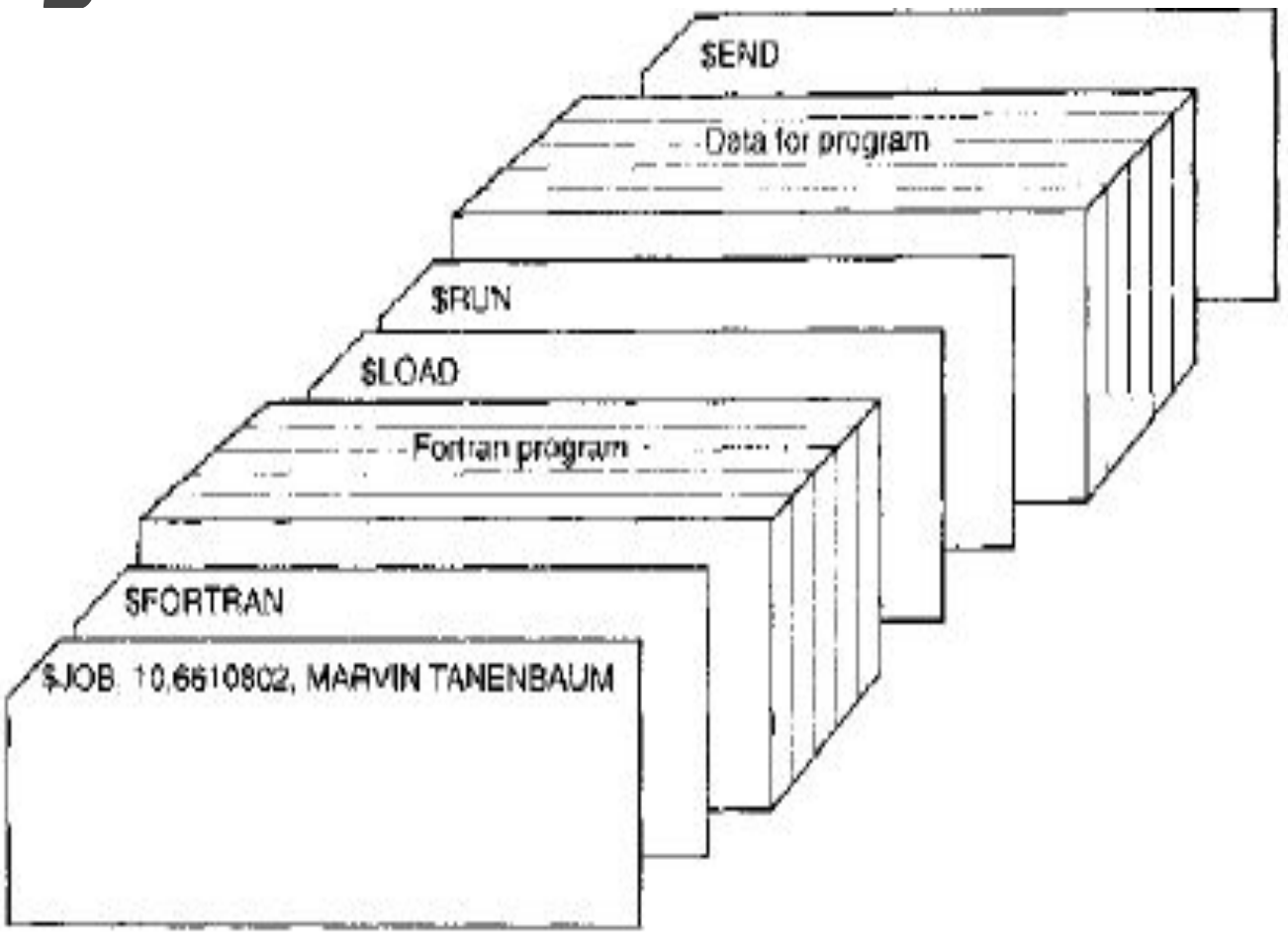


Figure 1-3. An early batch system, (a) Programmers bring cards to 1401. (b) 1401 reads batch of jobs onto tape, (c) Operator carries input tape to 7094. (d) 7094 does computing, (e) Operator carries output tape to 1401. (f) 1401 prints output

Structure of a FORTRAN program



Third Generation (1965 – 1980) ICs and Multiprogramming

Multiprogramming

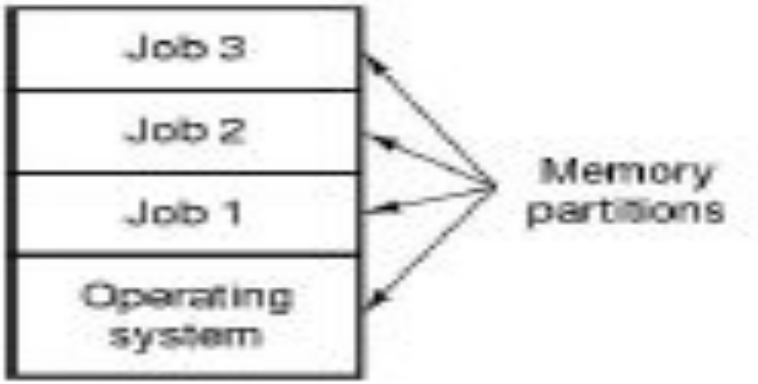


Figure 1-5. A multiprogramming system with three jobs in memory.

Third Generation (1965 – 1980) ICs and Multiprogramming

- **Spooling**
  Simultaneous Peripheral Operation Online
  More than one I/O operation can be done simultaneously

- **Timesharing**
  - **CTSS - Compatible Time Sharing System**
  - **Multics – (MULTiplexed Information and Computing Service)**

The Fourth Generation (1980 – Present)

- Personal Computers (Initially called Microcomputers)
- Control program for microcomputers (CP/M)

- IBM designed IBM – PC
- OS developed for it was by microsoft so called MS-DOS

- GUI – User Friendly
- Windows – windows NT Windows ME ……

Kinds of Operating Systems

Kinds of Operating System

- Mainframe Operating Systems
- Server Operating Systems
- Multiprocessor Operating Systems
- Personal Computer Operating Systems
- Real Time Operating Systems
- Embedded Operating Systems
- Smart Card Operating Systems

Mainframe Operating Systems

Mainframe OS handles processing many jobs at once.

Offers three kinds of services as

- Batch system - Processes routine jobs without any user intervention
- Transaction Processing - Handles large number of small requests
- Timesharing - Allows multiple remote users to run jobs on the computer at once

Example : z/ OS, OS/360, OS/390

https://www.youtube.com/watch?v=C1YGE7m33iQ

Server Operating Systems

- Run on Servers
- Serve multiple users at once over a network
- Internet providers run many server machines to support their customers and websites
- Server OS are solaris, FreeBSD, Linux and Windows Server 200x
- https://www.youtube.com/watch?v=avP5d16wEp0

Multiprocessor Operating System

- Connect multiple CPU's into a single system
- Based on connection these systems are called parallel systems, multicomputers or multiprocessors
- Special Operating Systems with features for communication and connectivity

Personal Computer Operating System

- Provides good interface to a single user
- Used for word processing, spreadsheets and internet access
- Examples
  - Windows 98, Windows 2000, Machintosh and linux

Real - Time Operating System

- Time is a key parameter
  Example
    Industrial Process control Systems

- Hard Real System
  Time are crucial factor. Example: Car Assembly

- Soft Real System
  Time not so important. Example: digital Audio, Multimedia systems

- Vxworks and QNX are well known operatng systems

Embedded Operating System

A Palmtop Computer or PDA

Example :- palmOS and Windows CE (Consumer Electronics)

Smart Card Operating System

- Smallest Operating system runs on smart cards
- Contains a CPU chip

- It has severe processing power and memory constraints

Some cards are Java oriented

ROM on smart card holds JVM.

Java applets are downloaded and interpreted by JVM

Operating System Concepts

Concepts

- **Process**
- **Deadlocks**
- **Memory Management**
- **Input / Output**
- **Files**
- **Security**
- **The Shell**

Process

- A Program in execution
- Each process is stored in an address space
- The address space consists of
  - Executable Program
  - Program Data
  - Stack
- The set of resources required for the process to execute are
  - Registers (including program counter and stack)
  - List of open files
  - Outstanding alarms
  - List of related processes

Interprocess Communication

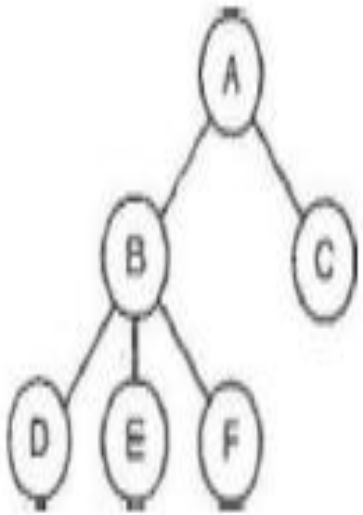
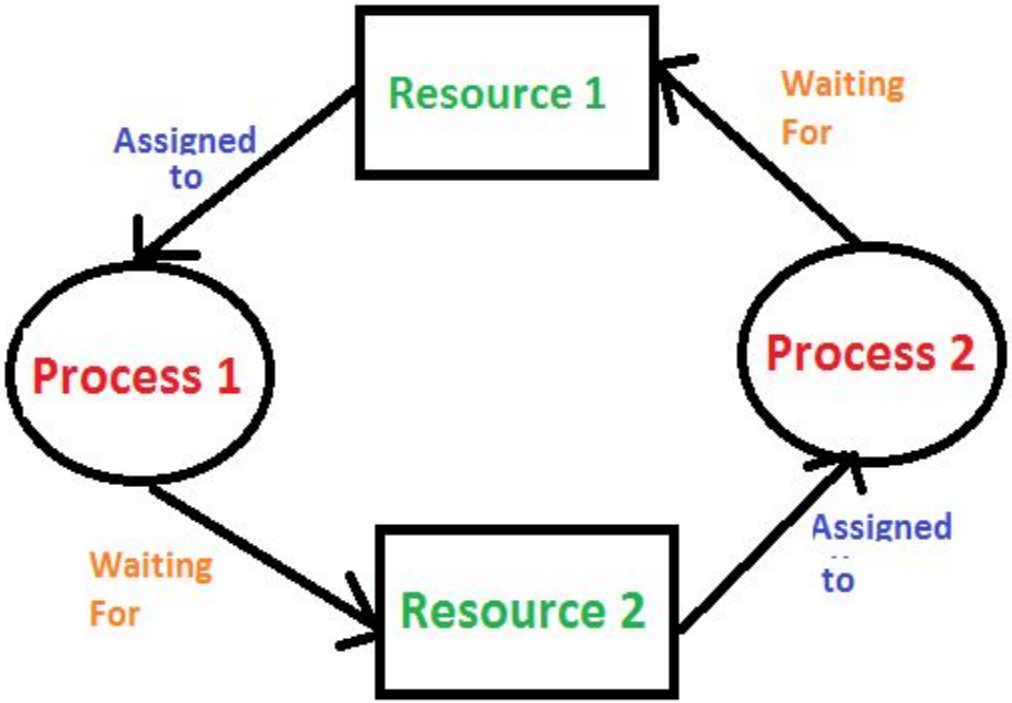


Figure 1-13. A process free. Process *A* created two child processes, *B* and C. Process *B* created three child processes, £>, £, and *F*.

Deadlock

Deadlock is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.

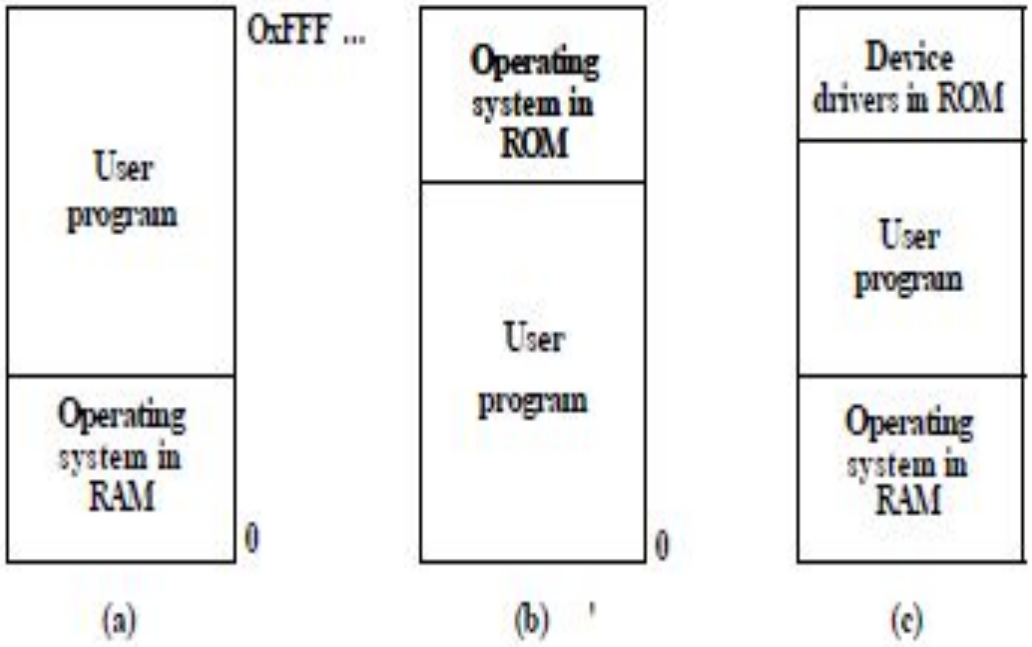


Memory

1 nsec	Registers	<1 KB
2 nsec	Cache	4MB
10 nsec	Main memory	512-2048 MB
10 msec	Magnetic disk	200-1000 GB
100 sec	Magnetic tape	400-800 GB

Figure 1-9. A typical memory hierarchy. The numbers are very rough approximations.

Memory Management



Files

- Provides the abstract model of device independent files
- System calls to create, remove, read and write file operations
- Directory - a way to group files
  - System calls to create and remove directory
  - System call to place a file in a directory, remove a file from a directory and to move from one directory to another
  - Directories are accessed by path

Files

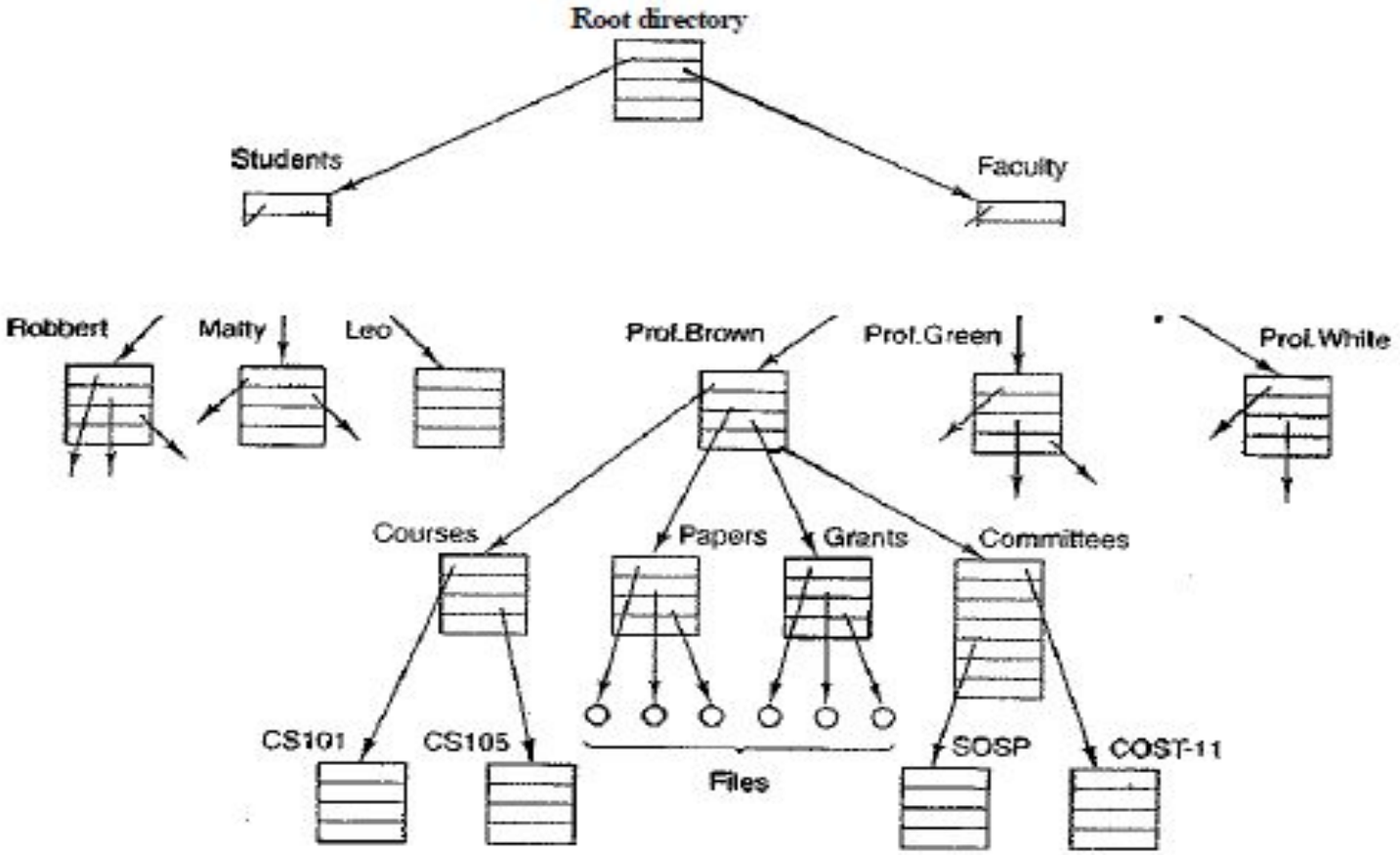


Figure 1-14. A file system for a university department.

I/O

Keyboards

Mouse

Monitor

OS handles all these

Security

File Security

Information stored securely

Shell

In unix OS

Shell commands

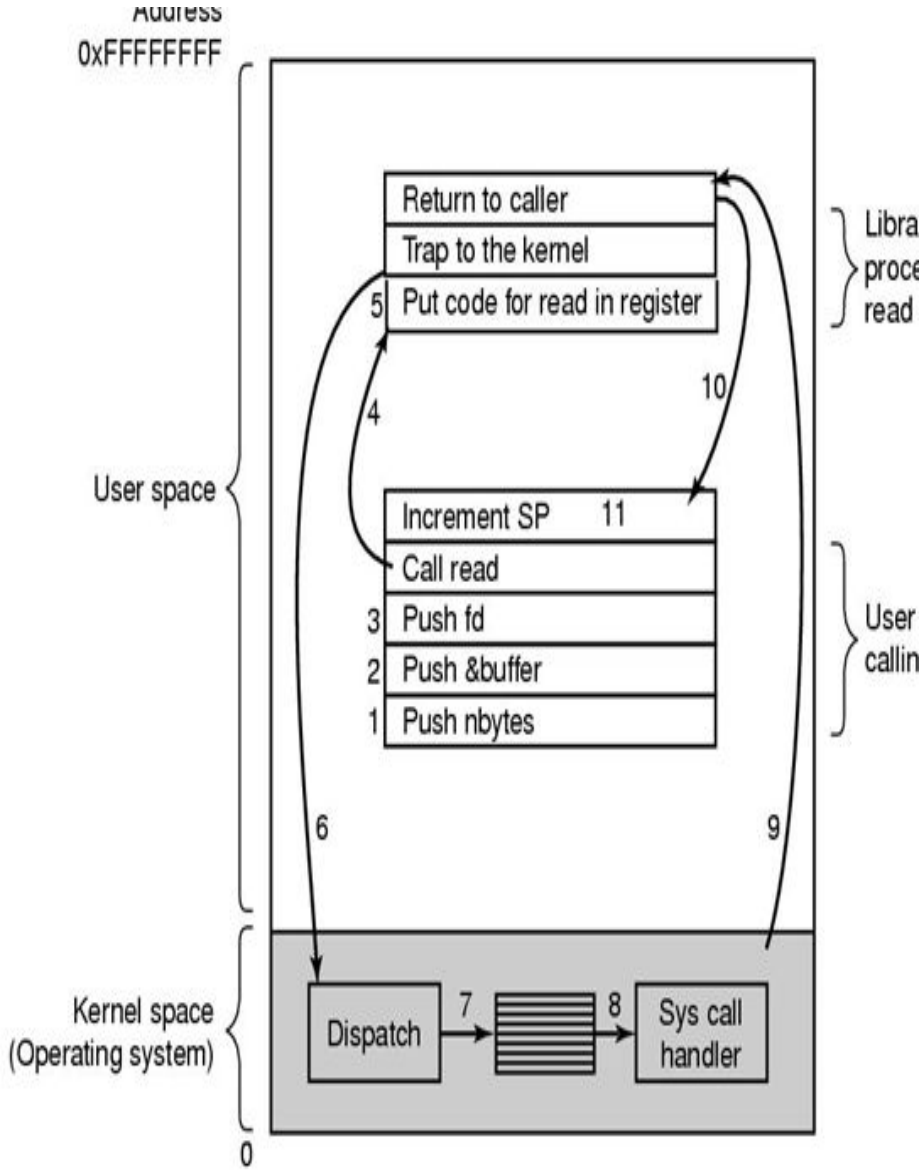
System Calls

System Calls

System calls is the interface users contact with OS and hardware

System calls vary from system to system, but the underlying concepts are similar

System Calls



System Calls for Process Management

Process management

Call	Description
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

System Calls for File Management

File management	
Call	Description
fd = open(file, how,...)	Open a file for reading, writing, or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information

System Calls for Directory Management

Directory and file system management

Call	Description
s = mkdir(name, mode)	Create a new directory
s = rmdir(name)	Remove an empty directory
s = link(name1, name2)	Create a new entry, name2, pointing to name1
s = unlink(name)	Remove a directory entry
s = mount(special, name, flag)	Mount a file system
s = umount(special)	Unmount a file system

System Calls for Miscellaneous

Miscellaneous	
Call	Description
s = chdir(dirname)	Change the working directory
s = chmod(name, mode)	Change a file's protection bits
s = kill(pid, signal)	Send a signal to a process
seconds = time(&seconds)	Get the elapsed time since Jan. 1, 1970

Windows API

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
Gpen	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
Iseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
crtdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

Operating System Structure

Monolithic System

Monolithic systems – basic structure:

- Procedures are written for each functionality, then combined.

- A main program that invokes the requested service procedure.
- A set of service procedures that carry out the system calls.
- A set of utility procedures that help the service procedures.

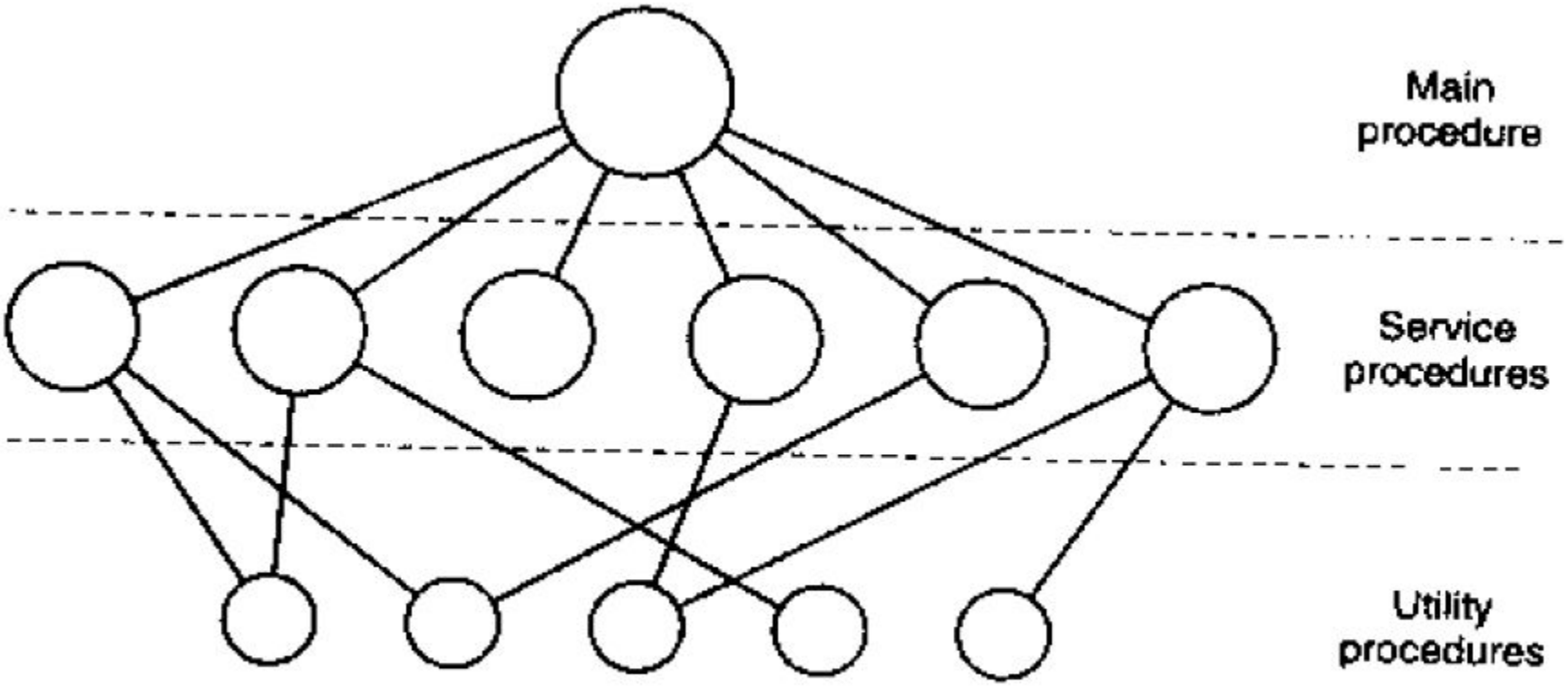


Figure 1-24. A simple structuring model for a monolithic system.

Layered Systems

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

Figure 1-25. Structure of the THE operating system.

Microkernels

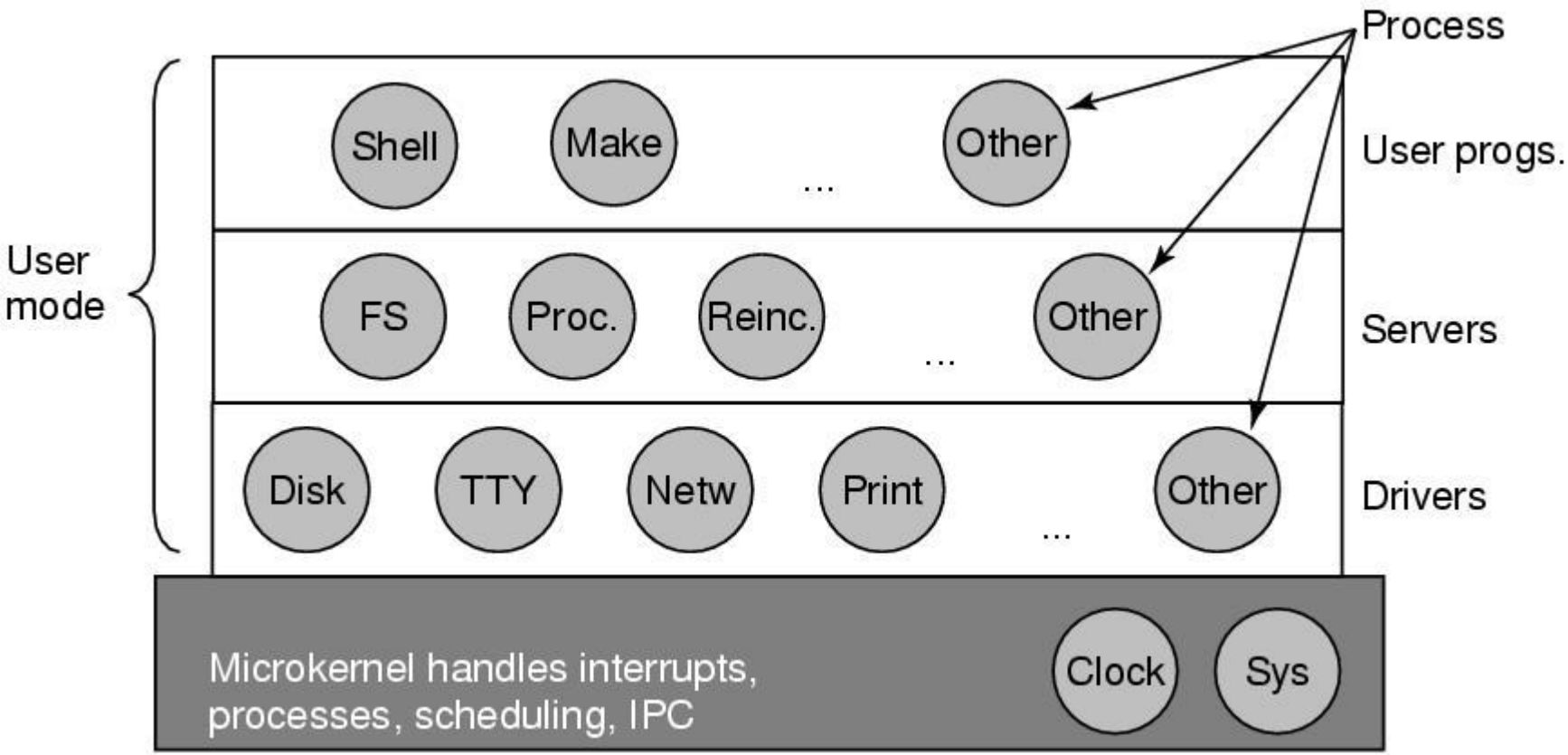


Figure 1-26. Structure of the MINIX 3 system.

Client-Server Model

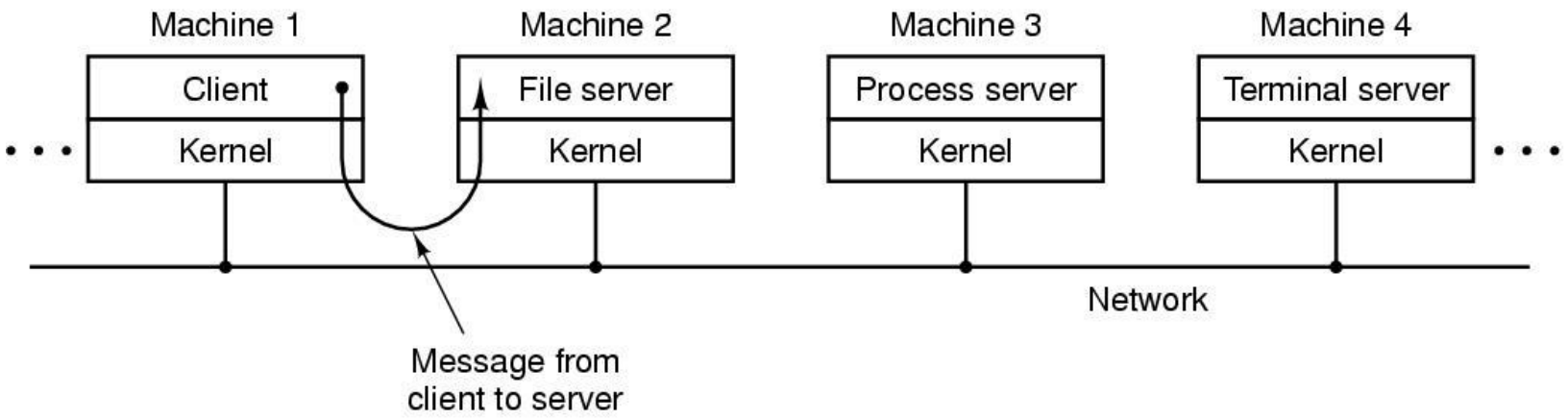


Figure 1-27. The client-server model over a network.

Virtual Machines (2)

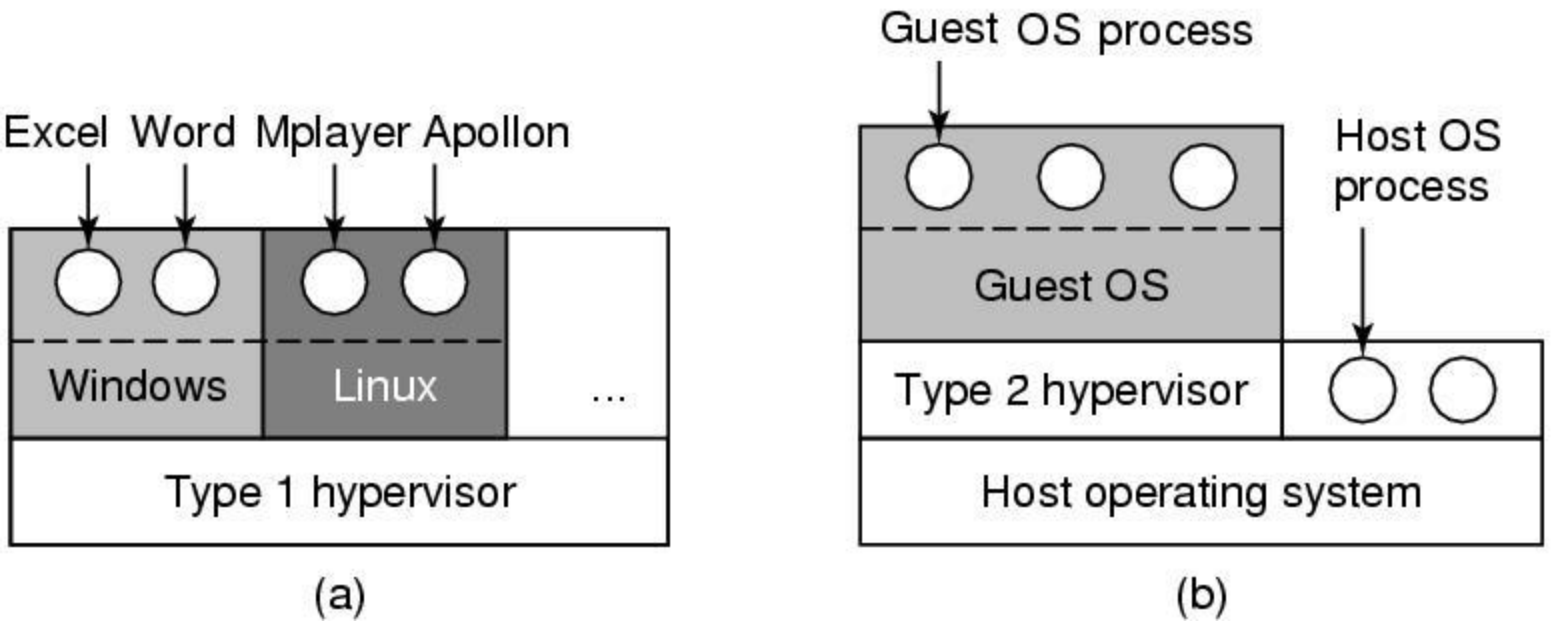


Figure 1-29. (a) A type 1 hypervisor. (b) A type 2 hypervisor.

The World According to C

- The C language
- Header files
- Large programming projects
- The model of run time

World according to C

- Operating systems are large C programs consisting of many pieces written by many programmers
- C language
  - Data types, variables, control statements…
  - Header files: declaration, definition, macros…
  - For a large programming project

The Model of Run Time

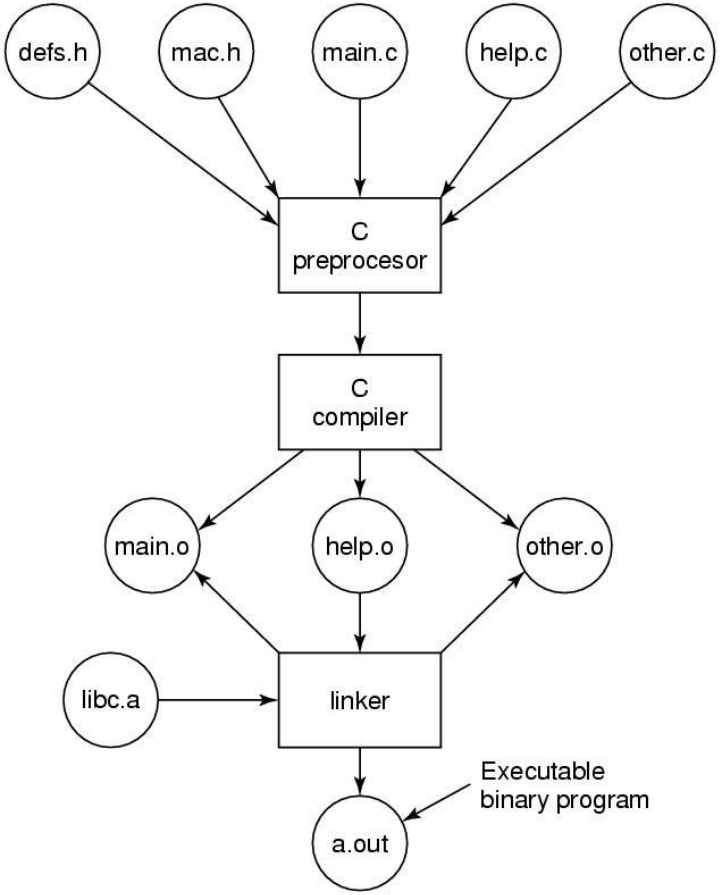


Figure 1-30. The process of compiling C and header files to make an executable.

Large programming projects

- C preprocessor:
  - Gets the header, expand macros, handling conditional compilation
- Compiler
  - .c -.o
- Linker
  - Combine all .o to an executable program; traditionally a.out

Gcc:

preprocess-assemble-compile-link

- gcc -E hello.c –o hello.i
- gcc -S hello.i -o hello.s
- gcc -c hello.s –o hello.o
- gcc hello.o –o hello
- ldd hello

Thank You