OPERATING SYSTEMS [20MCA15C]

UNIT – I
“Introduction, Operating System Structures”

FACULTY:

Dr. R. A. Roseline, M.Sc., M.Phil., Ph.D.,

Associate Professor and Head,
Post Graduate and Research Department of Computer Applications,
Government Arts College (Autonomous), Coimbatore – 641 018.

Operating Systems [20MCA15C] Syllabus

- **UNIT I: Introduction**: What is an OS - Mainframe systems - Desktop systems - Multiprocessor systems - Distributed systems - Clustered systems - Real-Time systems. **Operating system structures:** Systems components - OS services - System calls - System Programs - Systems structure - Virtual machines - System Design & Implementation - System Generation.
  (Chapter 1, 3)
- **UNIT II: Process Management:** Process concept - Process scheduling - Operations on process - Cooperating process - Inter-process communication. **CPU scheduling:** Scheduling criteria - Scheduling algorithms - Multiple-processor Scheduling - Real-Time Scheduling. **Deadlocks:** Deadlock characterization - Methods for handling Deadlocks - Deadlocks prevention - Deadlock avoidance - Deadlock detection - Recovery from Deadlock.
  (Chapter 4, ,6, 8)

- **UNIT III: Memory Management:** Background - Swapping - Contiguous memory allocation - Paging - Segmentation - Segmentation with paging. **Virtual memory:** Demand paging - Process creation - Page replacement - Allocation of frames - Thrashing.
(Chapter 9, 10)
- **UNIT IV: I/O Systems:** Disk structure - Disk scheduling - Disk management - Swap - Space management. **File systems:** File concept - Access methods Directory structure - File system structure - File system implementation - Directory implementation - Allocation methods - Free space management.
(Chapter 11, 12, 14)

- **UNIT V: CASE STUDY: Linux:** Design Principles - Kernel modules - Process management, scheduling - Memory management - File systems - Input & Output - Inter-process Communication - Network structure - Security. *(Chapter 20)*

- **TEXT BOOKS:**

1. Silberschatz, Galvin, Gagne, "*Operating Systems Concepts*", Sixth Edition, John Wiley & Sons, 2013.

- **REFERENCE BOOKS:**

1. Tanenbaum, "*Operating systems: Design & Implementation*", PHI, Second Edition, 1998.
2. Deital, "*Operating Systems*", Pearson Education Asia, Second Edition, 2001.
3. D. M. Dhamdhere, "*System Programming and Operating Systems*", TMH, 2000.

Introduction

- What is an Operating System?
- Mainframe Systems
- Desktop Systems
- Multiprocessor Systems
- Distributed Systems
- Clustered System
- Real-Time Systems
- Handheld Systems
- Computing Environments

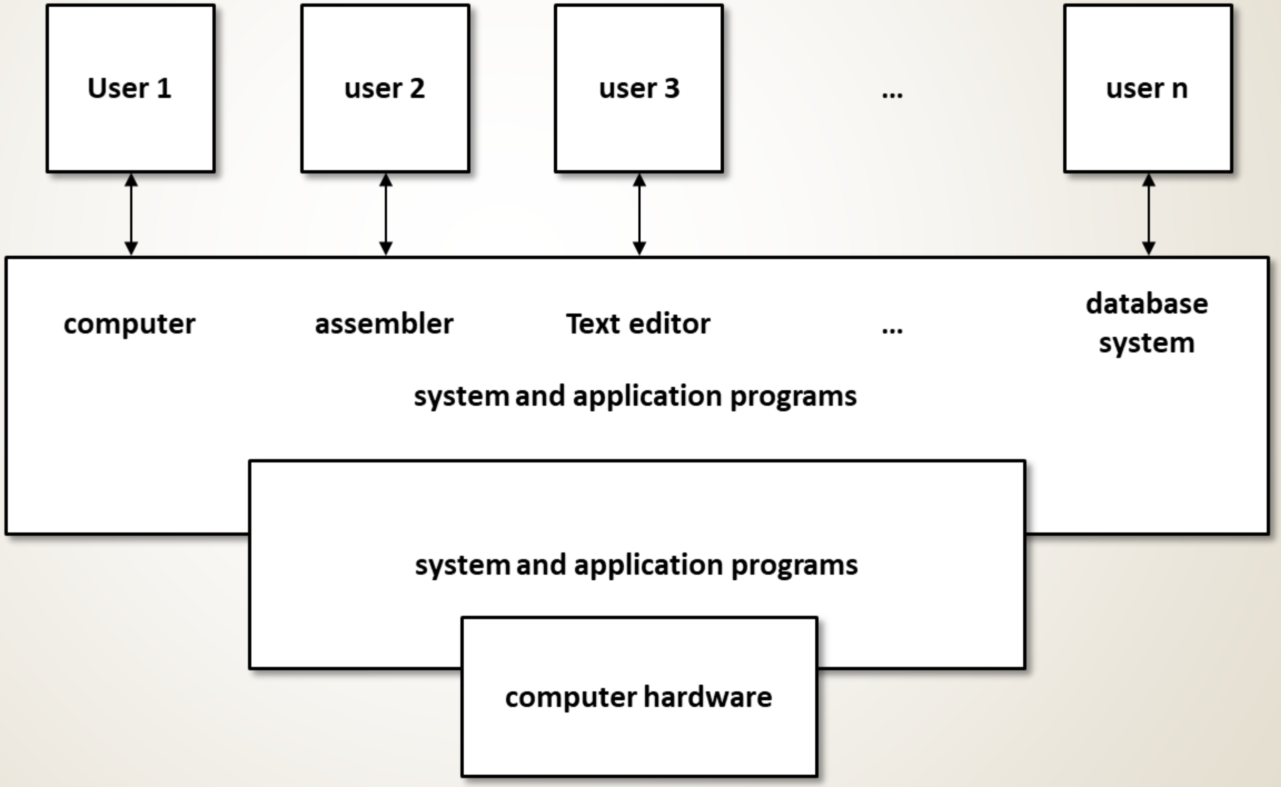
What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier.
  - Make the computer system convenient to use.
- Use the computer hardware in an efficient manner.

Computer System Components

- Hardware – provides basic computing resources (CPU, memory, I/O devices).
- Operating system – controls and coordinates the use of the hardware among the various application programs for the various users.
- Applications programs – define the ways in which the system resources are used to solve the computing problems of the users (compilers, database systems, video games, business programs).
- Users (people, machines, other computers).

Abstract View of System Components



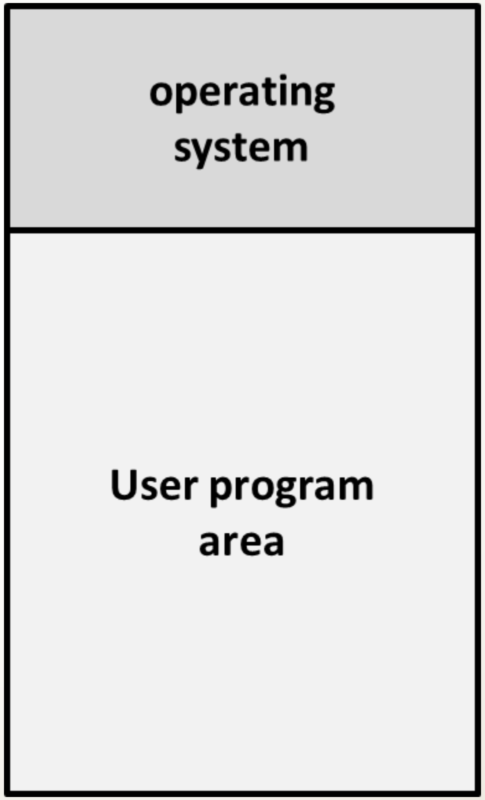
Operating System Definitions

- Resource allocator – manages and allocates resources.
- Control program – controls the execution of user programs and operations of I/O devices .
- Kernel – the one program running at all times (all else being application programs).

Mainframe Systems

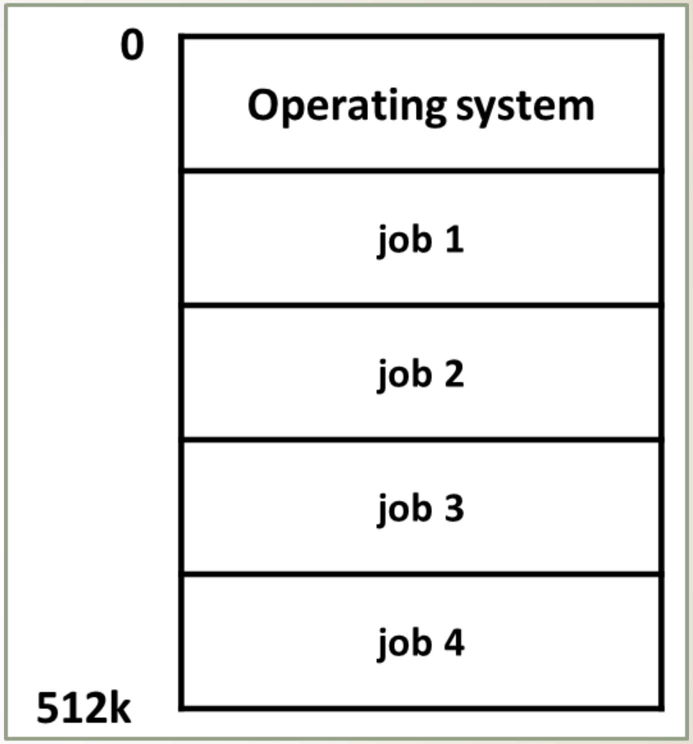
- Reduce setup time by batching similar jobs
- Automatic job sequencing – automatically transfers control from one job to another. First rudimentary operating system.
- Resident monitor
  - initial control in monitor
  - control transfers to job
  - when job completes control transfers pack to monitor

Memory Layout for a Simple Batch System



Multi-programmed Batch Systems

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



0

Operating system

job 1

job 2

job 3

job 4

512k

OS Features Needed for Multiprogramming

- I/O routine supplied by the system.
- Memory management – the system must allocate the memory to several jobs.
- CPU scheduling – the system must choose among several jobs ready to run.
- Allocation of devices.

Time-Sharing Systems–Interactive Computing

- The CPU is multiplexed among several jobs that are kept in memory and on disk (the CPU is allocated to a job only if the job is in memory).
- A job swapped in and out of memory to the disk.
- On-line communication between the user and the system is provided; when the operating system finishes the execution of one command, it seeks the next “control statement” from the user’s keyboard.
- On-line system must be available for users to access data and code.

Desktop Systems

- Personal computers – computer system dedicated to a single user.
- I/O devices – keyboards, mice, display screens, small printers.
- User convenience and responsiveness.
- Can adopt technology developed for larger operating system' often individuals have sole use of computer and do not need advanced CPU utilization of protection features.
- May run several different types of operating systems (Windows, MacOS, UNIX, Linux)

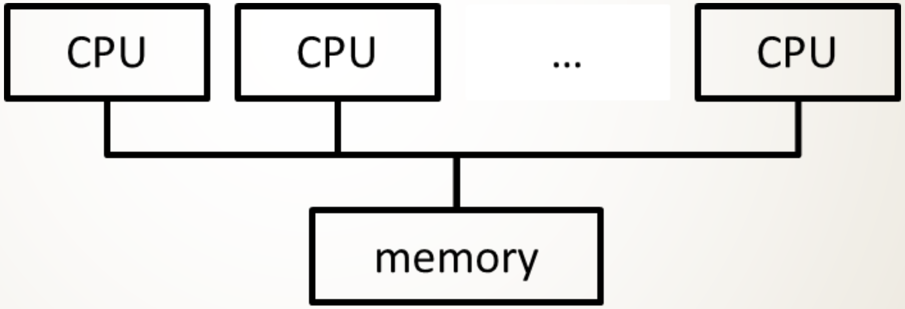
Parallel Systems

- Multiprocessor systems with more than on CPU in close communication.
- Tightly coupled system – processors share memory and a clock; communication usually takes place through the shared memory.
- Advantages of parallel system:
  - Increased throughput
  - Economical
  - Increased reliability
    - graceful degradation
    - fail-soft systems

Parallel Systems (Cont.)

- Symmetric multiprocessing (SMP)
  - Each processor runs and identical copy of the operating system.
  - Many processes can run at once without performance deterioration.
  - Most modern operating systems support SMP
- Asymmetric multiprocessing
  - Each processor is assigned a specific task; master processor schedules and allocated work to slave processors.
  - More common in extremely large systems

Symmetric Multiprocessing Architecture



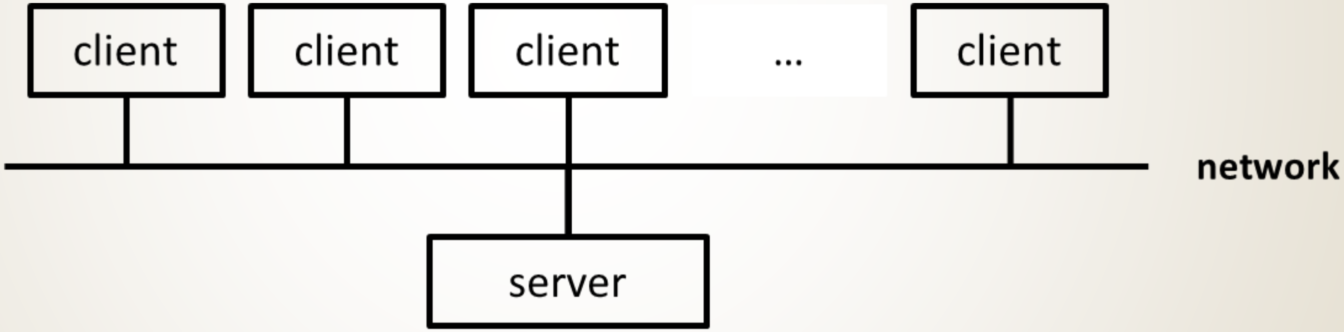
Distributed Systems

- Distribute the computation among several physical processors.
- Loosely coupled system – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or telephone lines.
- Advantages of distributed systems.
  - Resources Sharing
  - Computation speed up – load sharing
  - Reliability
  - Communications

Distributed Systems (cont)

- Requires networking infrastructure.
- Local area networks (LAN) or Wide area networks (WAN)
- May be either client-server or peer-to-peer systems.

General Structure of Client-Server



Clustered Systems

- Clustering allows two or more systems to share storage.
- Provides high reliability.
- Asymmetric clustering: one server runs the application while other servers standby.
- Symmetric clustering: all N hosts are running the application.

Real-Time Systems

- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.
- Well-defined fixed-time constraints.
- Real-Time systems may be either hard or soft real-time.

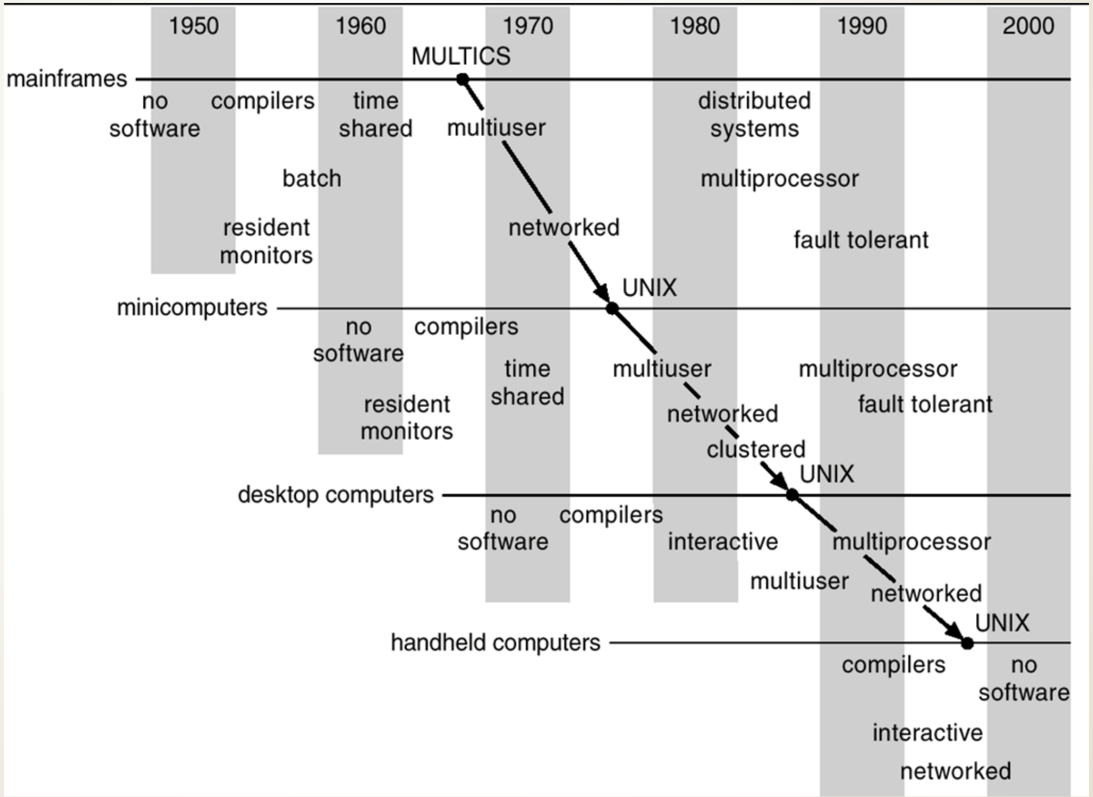
Real-Time Systems (Cont.)

- Hard real-time:
  - Secondary storage limited or absent, data stored in short term memory, or read-only memory (ROM)
  - Conflicts with time-sharing systems, not supported by general-purpose operating systems.
- Soft real-time
  - Limited utility in industrial control of robotics
  - Useful in applications (multimedia, virtual reality) requiring advanced operating-system features.

Handheld Systems

- Personal Digital Assistants (PDAs)
- Cellular telephones
- Issues:
  - Limited memory
  - Slow processors
  - Small display screens.

Migration of Operating-System Concepts and Features



Computing Environments

- Traditional computing
- Web-Based Computing
- Embedded Computing

Operating-System Structures

- System Components
- Operating System Services
- System Calls
- System Programs
- System Structure
- Virtual Machines
- System Design and Implementation
- System Generation

Common System Components

- Process Management
- Main Memory Management
- File Management
- I/O System Management
- Secondary Management
- Networking
- Protection System
- Command-Interpreter System

Process Management

- A process is a program in execution. A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task.
- The operating system is responsible for the following activities in connection with process management.
  - Process creation and deletion.
  - process suspension and resumption.
  - Provision of mechanisms for:
    - process synchronization
    - process communication

Main-Memory Management

- Memory is a large array of words or bytes, each with its own address. It is a repository of quickly accessible data shared by the CPU and I/O devices.
- Main memory is a volatile storage device. It loses its contents in the case of system failure.
- The operating system is responsible for the following activities in connections with memory management:
  - Keep track of which parts of memory are currently being used and by whom.
  - Decide which processes to load when memory space becomes available.
  - Allocate and deallocate memory space as needed.

File Management

- A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data.
- The operating system is responsible for the following activities in connections with file management:
  - File creation and deletion.
  - Directory creation and deletion.
  - Support of primitives for manipulating files and directories.
  - Mapping files onto secondary storage.
  - File backup on stable (nonvolatile) storage media.

I/O System Management

- The I/O system consists of:
  - A buffer-caching system
  - A general device-driver interface
  - Drivers for specific hardware devices

Secondary-Storage Management

- Since main memory (primary storage) is volatile and too small to accommodate all data and programs permanently, the computer system must provide secondary storage to back up main memory.
- Most modern computer systems use disks as the principle on-line storage medium, for both programs and data.
- The operating system is responsible for the following activities in connection with disk management:
  - Free space management
  - Storage allocation
  - Disk scheduling

Networking (Distributed Systems)

- A distributed system is a collection processors that do not share memory or a clock. Each processor has its own local memory.
- The processors in the system are connected through a communication network.
- Communication takes place using a protocol.
- A distributed system provides user access to various system resources.
- Access to a shared resource allows:
    - Computation speed-up
    - Increased data availability
    - Enhanced reliability

Protection System

- Protection refers to a mechanism for controlling access by programs, processes, or users to both system and user resources.
- The protection mechanism must:
  - distinguish between authorized and unauthorized usage.
  - specify the controls to be imposed.
  - provide a means of enforcement.

Command-Interpreter System

- Many commands are given to the operating system by control statements which deal with:
  - process creation and management
  - I/O handling
  - secondary-storage management
  - main-memory management
  - file-system access
  - protection
  - networking

Command-Interpreter System (Cont.)

- The program that reads and interprets control statements is called variously:

  - command-line interpreter
  - shell (in UNIX)

-

- Its function is to get and execute the next command statement.

Operating System Services

- Program execution – system capability to load a program into memory and to run it.
- I/O operations – since user programs cannot execute I/O operations directly, the operating system must provide some means to perform I/O.
- File-system manipulation – program capability to read, write, create, and delete files.
- Communications – exchange of information between processes executing either on the same computer or on different systems tied together by a network. Implemented via shared memory or message passing.
- Error detection – ensure correct computing by detecting errors in the CPU and memory hardware, in I/O devices, or in user programs.

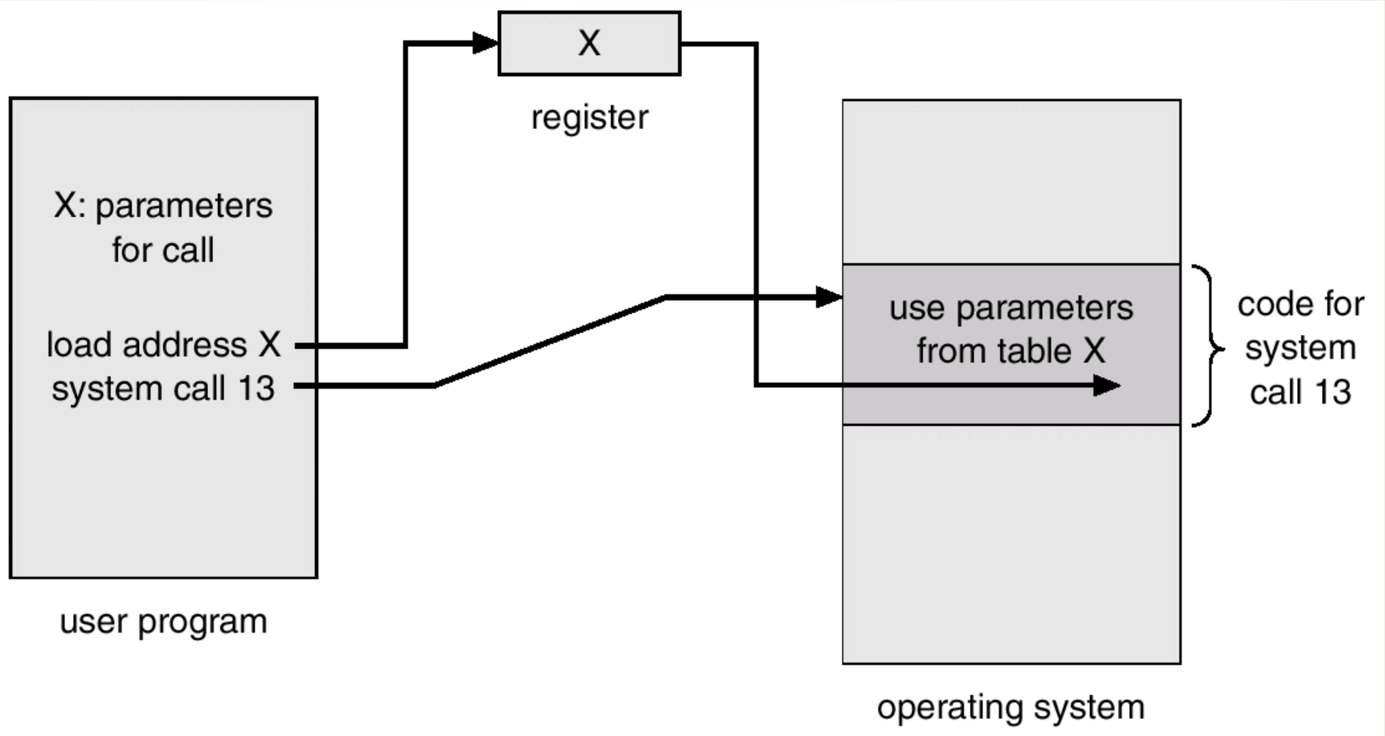
Additional Operating System Functions

- Additional functions exist not for helping the user, but rather for ensuring efficient system operations.
  - Resource allocation – allocating resources to multiple users or multiple jobs running at the same time.
  - Accounting – keep track of and record which users use how much and what kinds of computer resources for account billing or for accumulating usage statistics.
  - Protection – ensuring that all access to system resources is controlled.

System Calls

- System calls provide the interface between a running program and the operating system.
  - Generally available as assembly-language instructions.
  - Languages defined to replace assembly language for systems programming allow system calls to be made directly (e.g., C, C++)
- Three general methods are used to pass parameters between a running program and the operating system.
  - Pass parameters in registers.
  - Store the parameters in a table in memory, and the table address is passed as a parameter in a register.
  - Push (store) the parameters onto the stack by the program, and pop off the stack by operating system.

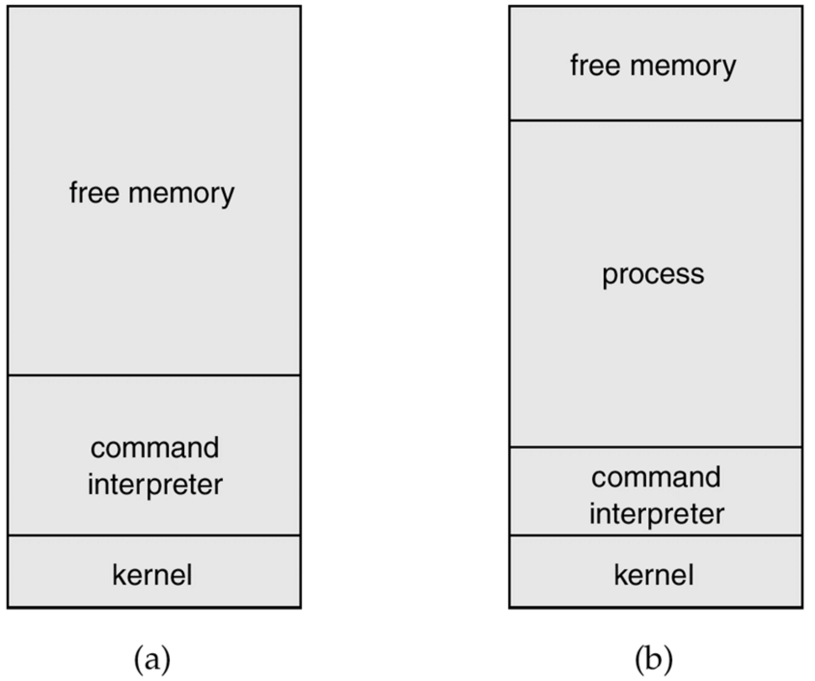
Passing of Parameters As A Table



Types of System Calls

- Process control
- File management
- Device management
- Information maintenance
- Communications

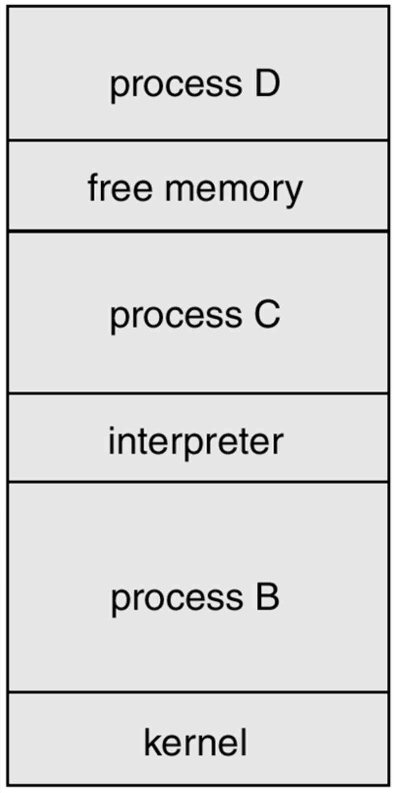
MS-DOS Execution



At System Start-up

Running a Program

UNIX Running Multiple Programs



process D

free memory

process C

interpreter

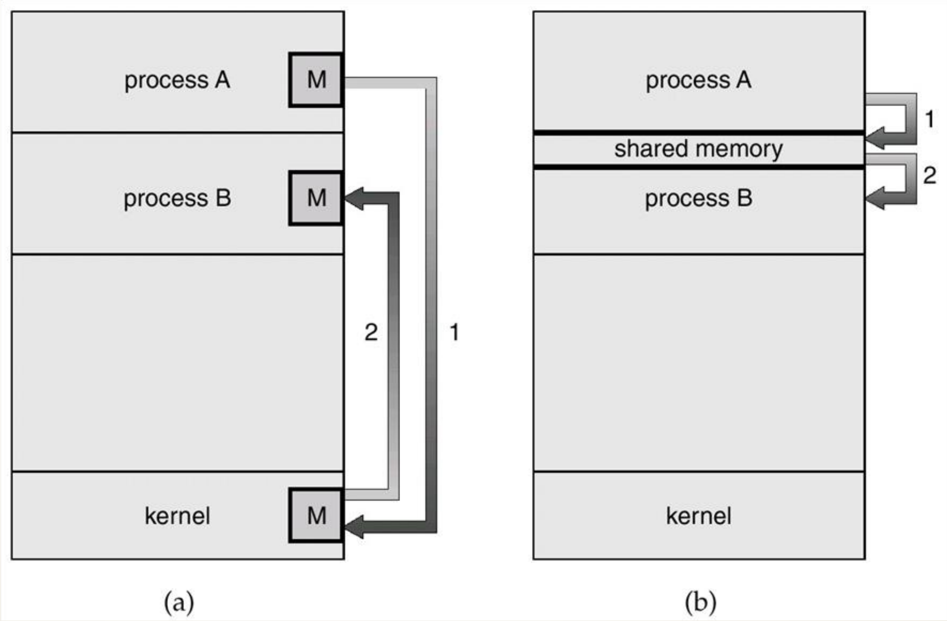
process B

kernel

Communication Models

Communication may take place using either

(a) message passing or (b) shared memory



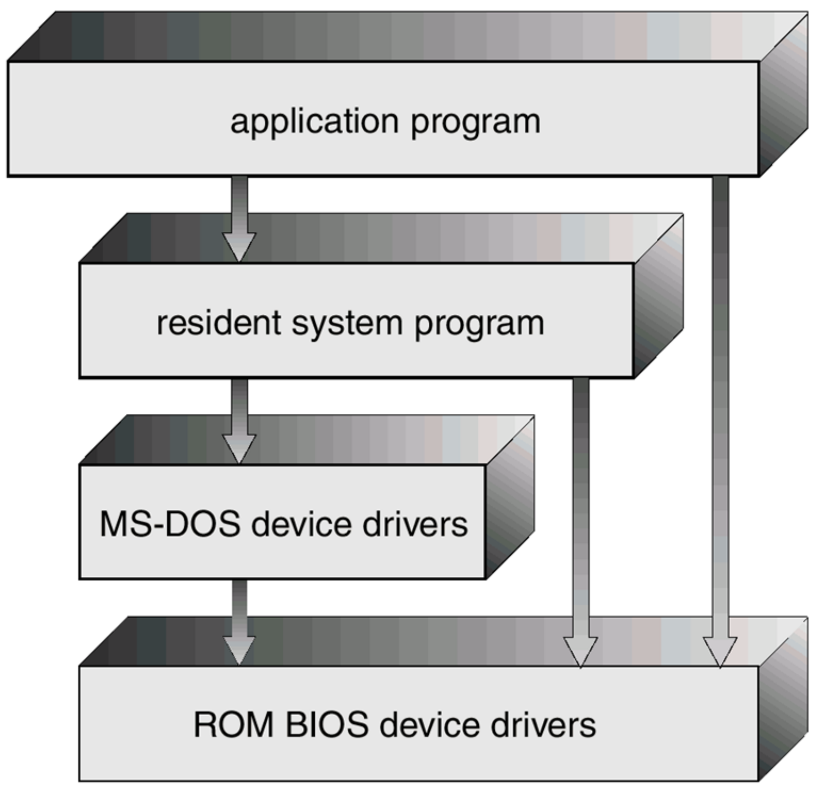
System Programs

- System programs provide a convenient environment for program development and execution. The can be divided into:
  - File manipulation
  - Status information
  - File modification
  - Programming language support
  - Program loading and execution
  - Communications
  - Application programs
- Most users' view of the operation system is defined by system programs, not the actual system calls.

MS-DOS System Structure

- MS-DOS – written to provide the most functionality in the least space
  - not divided into modules
  - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated

MS-DOS Layer Structure



UNIX System Structure

- UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring. The UNIX OS consists of two separable parts.
  - Systems programs
  - The kernel
    - Consists of everything below the system-call interface and above the physical hardware
    - Provides the file system, CPU scheduling, memory management, and other operating-system functions; a large number of functions for one level.

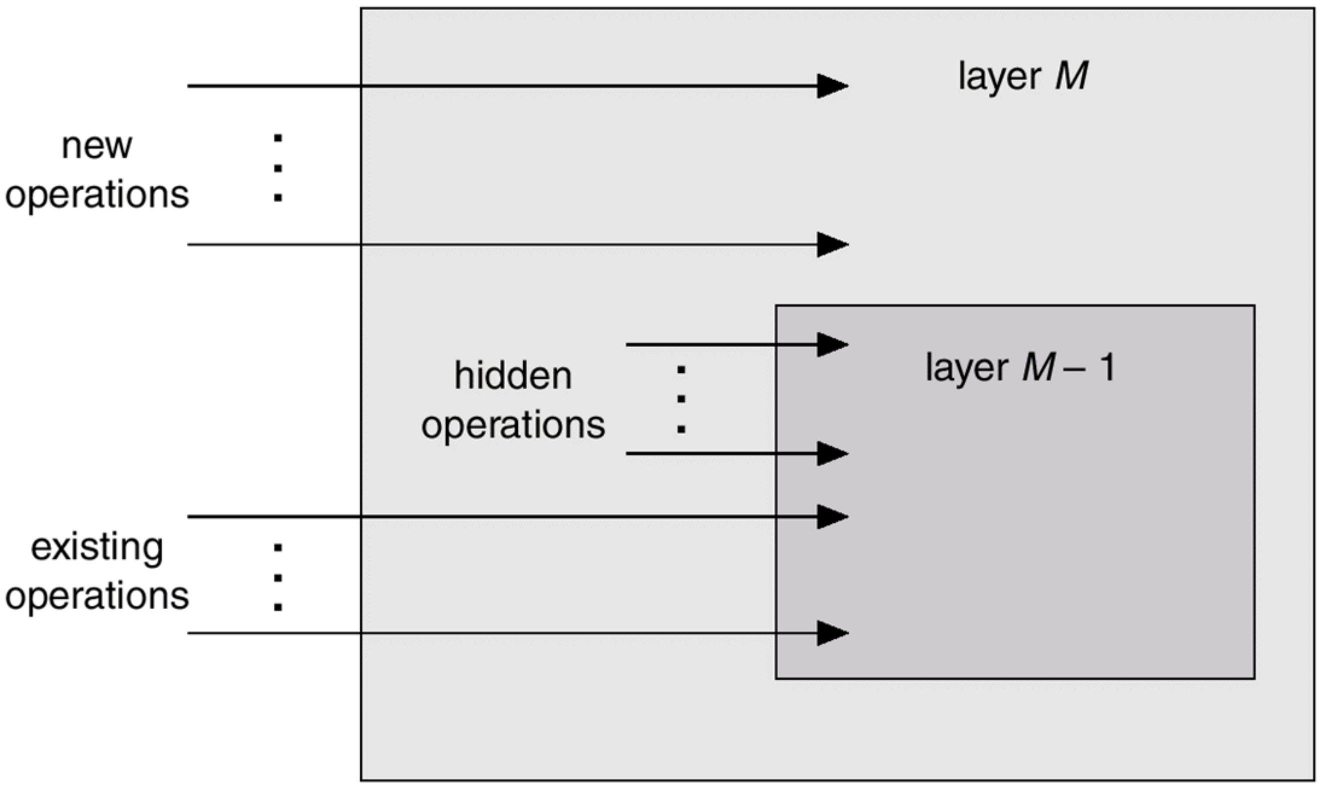
UNIX System Structure

<table>
<tr><td colspan="3">(the users)</td></tr>
<tr><td colspan="3">shells and commands
compilers and interpreters
system libraries</td></tr>
<tr><td colspan="3">system-call interface to the kernel</td></tr>
<tr><td>signal terminal handling
character I/O system
terminal drivers</td><td>file system
swapping block I/O System
disk and space tape drivers</td><td>CPU scheduling
page replacement
demand paging
virtual memory</td></tr>
<tr><td colspan="3">Kernel interface to the future</td></tr>
<tr><td>terminal controllers
terminals</td><td>device controllers
disk and tapes</td><td>memory controllers
physical memory</td></tr>
</table>

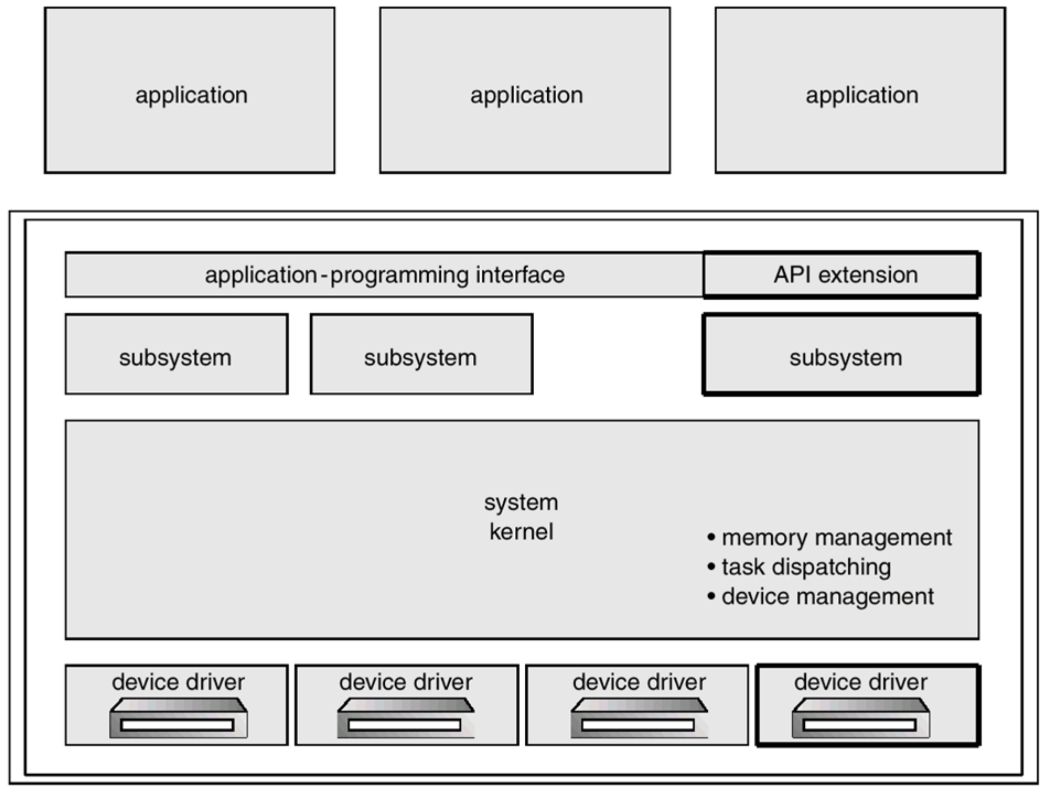
Layered Approach

- The operating system is divided into a number of layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers.

An Operating System Layer



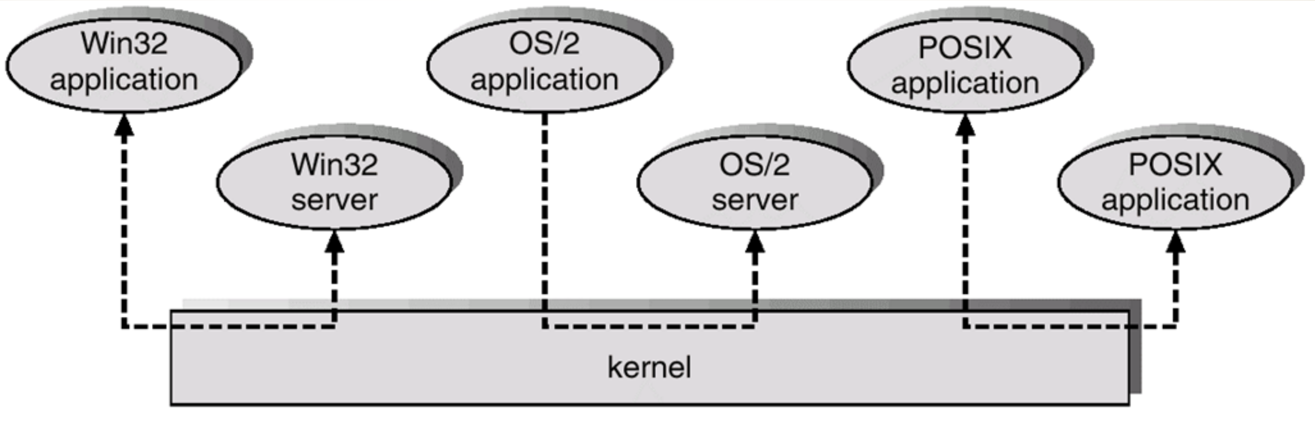
OS/2 Layer Structure



Microkernel System Structure

- Moves as much from the kernel into “user” space.
- Communication takes place between user modules using message passing.
- Benefits:
- - easier to extend a microkernel
- - easier to port the operating system to new architectures
- - more reliable (less code is running in kernel mode)
- - more secure

Windows NT Client-Server Structure



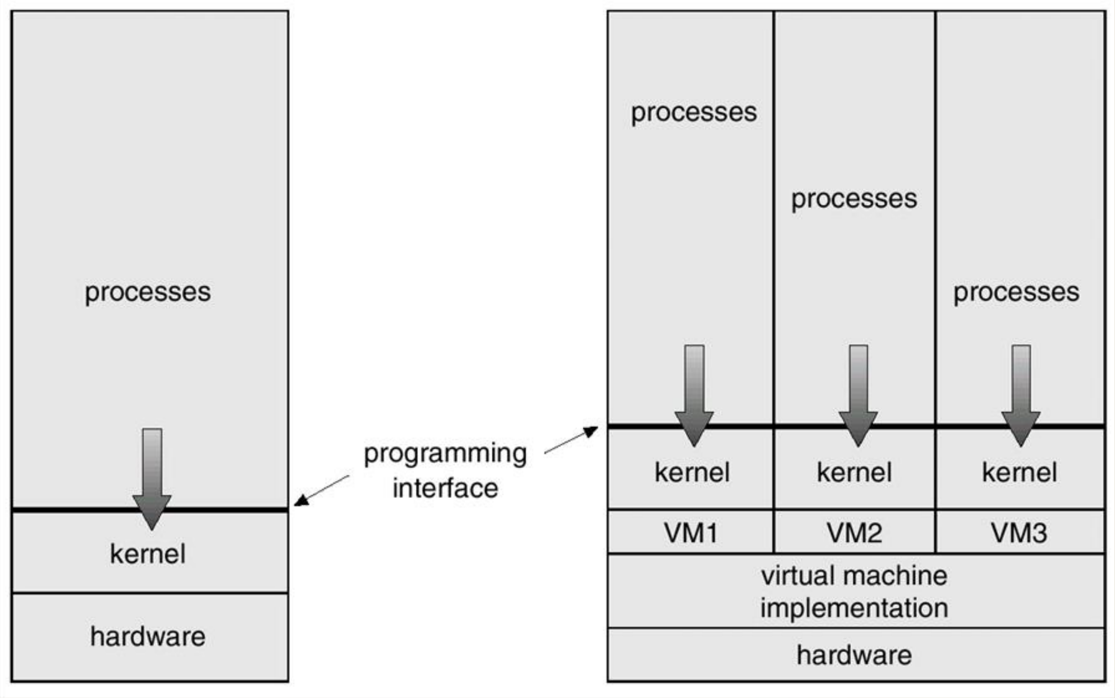
Virtual Machines

- A virtual machine takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware.
- A virtual machine provides an interface identical to the underlying bare hardware.
- The operating system creates the illusion of multiple processes, each executing on its own processor with its own (virtual) memory.

Virtual Machines (Cont.)

- The resources of the physical computer are shared to create the virtual machines.
  - CPU scheduling can create the appearance that users have their own processor.
  - Spooling and a file system can provide virtual card readers and virtual line printers.
  - A normal user time-sharing terminal serves as the virtual machine operator's console.

System Models



Non-virtual Machine

Virtual Machine

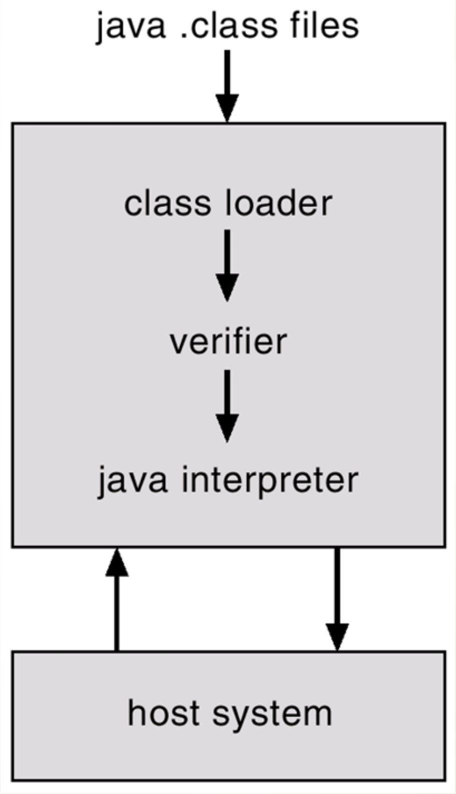
Advantages/Disadvantages of Virtual Machines

- The virtual-machine concept provides complete protection of system resources since each virtual machine is isolated from all other virtual machines. This isolation, however, permits no direct sharing of resources.
- A virtual-machine system is a perfect vehicle for operating-systems research and development. System development is done on the virtual machine, instead of on a physical machine and so does not disrupt normal system operation.
- The virtual machine concept is difficult to implement due to the effort required to provide an exact duplicate to the underlying machine.

Java Virtual Machine

- Compiled Java programs are platform-neutral bytecodes executed by a Java Virtual Machine (JVM).
- JVM consists of
  - class loader
  - class verifier
  - runtime interpreter
- Just-In-Time (JIT) compilers increase performance

Java Virtual Machine



System Design Goals

- User goals – operating system should be convenient to use, easy to learn, reliable, safe, and fast.
- System goals – operating system should be easy to design, implement, and maintain, as well as flexible, reliable, error-free, and efficient.

Mechanisms and Policies

- Mechanisms determine how to do something, policies decide what will be done.
- The separation of policy from mechanism is a very important principle, it allows maximum flexibility if policy decisions are to be changed later.

System Implementation

- Traditionally written in assembly language, operating systems can now be written in higher-level languages.
- Code written in a high-level language:
  - can be written faster.
  - is more compact.
  - is easier to understand and debug.
- An operating system is far easier to port (move to some other hardware) if it is written in a high-level language.

System Generation (SYSGEN)

- Operating systems are designed to run on any of a class of machines; the system must be configured for each specific computer site.
- SYSGEN program obtains information concerning the specific configuration of the hardware system.
- Booting – starting a computer by loading the kernel.
- Bootstrap program – code stored in ROM that is able to locate the kernel, load it into memory, and start its execution.

Thank you

The content in this Material are from the Textbooks and Reference books given in the Syllabus