Applied Operating System

Chapter 1: Operating Systems Types and Structure

Prepared By:
Amit K. Shrivastava
Asst. Professor
Nepal College Of Information Technology

Batch Systems

▪Jobs with similar needs are batched together and run through the computer as a group by an operator or automatic job sequencer. Performance is increased by attempting to keep CPU and I/O devices busy at all times through buffering, off-line operation, spooling, and multiprogramming. Batch is good for executing large jobs that need little interaction; it can be submitted and picked up later. The problems with Batch Systems are following.

➤Lack of interaction between the user and job.
➤Difficult to provide the desired priority.

Time-Sharing Systems

▪ This systems uses CPU scheduling and multiprogramming to provide economical interactive use of a system. The CPU switches rapidly from one user to another. Instead of having a job defined by spooled card images, each program reads its next control card from the terminal, and output is normally printed immediately to the screen. Advantages of Timesharing operating systems are - Provide advantage of quick response, Avoids duplication of software, Reduces CPU idle time.

Personal-Computer Systems

▪A Personal Computer(PC) is a small, relatively inexpensive computer designed for an individual user. All are based on the microprocessor technology that enables manufacturers to put an entire CPU on one chip. At home, the most popular use for personal computers is for playing games. Businesses use personal computers for word-processing, accounting, desktop publishing, and for running spreadsheet and database management applications. The goals of these operating systems is not only maximizing CPU and peripheral utilization, but also maximizing user convenience and responsiveness.

Parallel Systems

▪Parallel operating systems are used to interface multiple networked computers to complete tasks in parallel. The architecture of the software is often a UNIX-based platform, which allows it to coordinate distributed loads between multiple computers in a network. Parallel operating systems are able to use software to manage all of the different resources of the computers running in parallel, such as memory, caches, storage space, and processing power. Parallel operating systems also allow a user to directly interface with all of the computers in the network. Its one advantage is increased throughput.

Real-Time Systems
▪Often used in a dedicated application, this system reads information from sensors and must respond within a fixed amount of time to ensure correct performance. In this Response Time is already fixed. Means time to Display the Results after Possessing has fixed by the Processor or CPU. Real Time System is used at those Places in which we Requires higher and Timely Response.

Distributed Systems
▪This system distributes computation among several physical processors. The processors do not share memory or a clock. Instead, each processor has its own local memory. They communicate with each other through various communication lines, such as a high-speed bus or telephone line. The advantages of distributed systems are- With resource sharing facility user at one site may be able to use the resources available at another, Speedup the exchange of data with one another via electronic mail, If one site fails in a distributed system, the remaining sites can potentially continue operating, Better service to the customers, Reduction of the load on the host computer.

Operating-System Structures

System Components: Common System Components are
• Process Management
• Main Memory Management
• Secondary-Storage Management
• I/O System Management
• File Management
• Protection System
• Networking
• 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.

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

I/O System Management

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

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.

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.

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.
• A distributed system provides user access to various system resources.
• Access to a shared resource allows:
  – Computation speed-up
  – Increased data availability
  – Enhanced reliability

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

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.

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.
• 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.

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.

System Structure – Simple Approach

- 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.
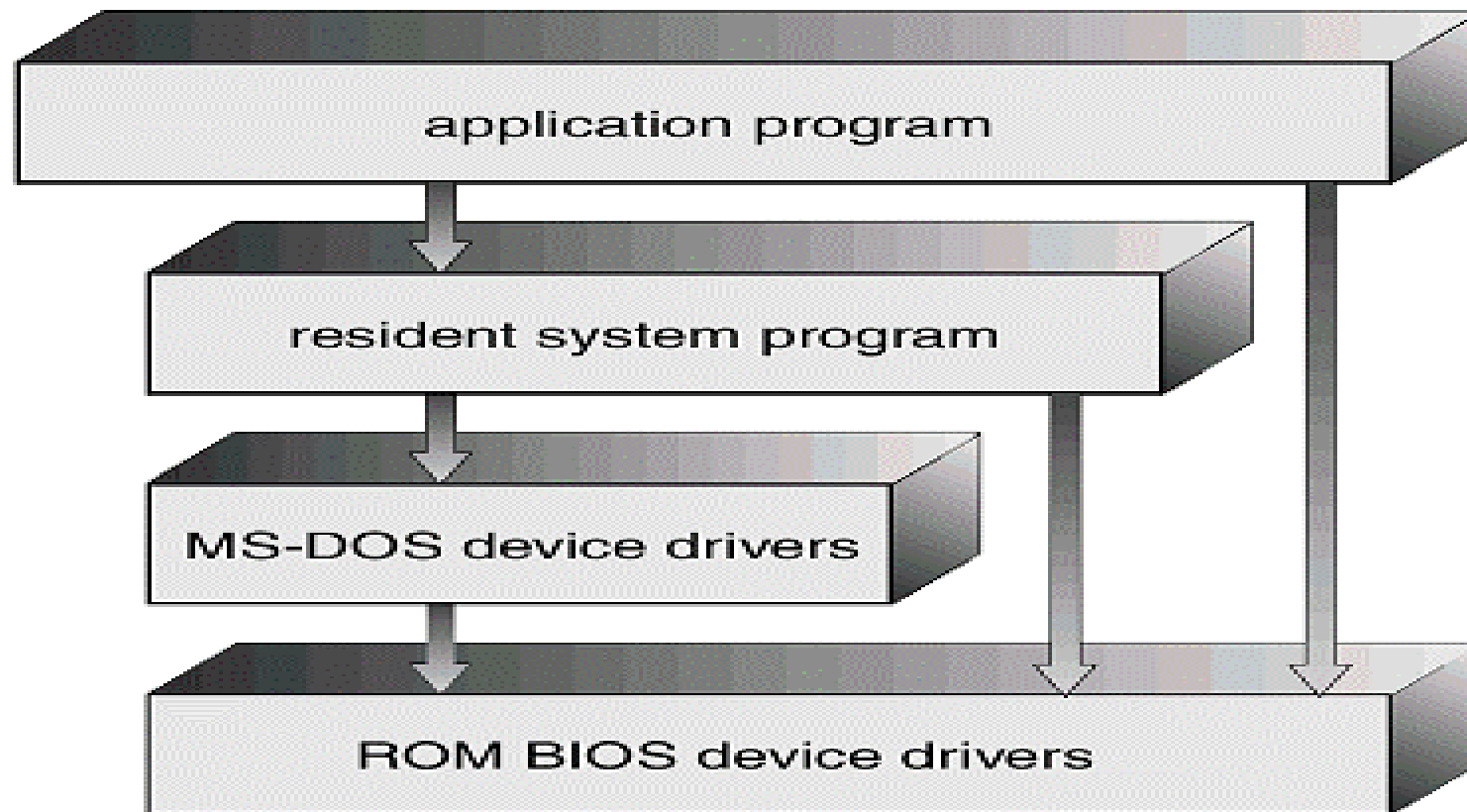

Fig: MS-DOS Layer Structure

System Structure – Simple Approach (Cont.)
• 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.

(the users)		
shells and commands compilers and interpreters system libraries		
system-call interface to the kernel		
signals terminal handling character I/O system terminal drivers	file system swapping block I/O system disk and tape drivers	CPU scheduling page replacement demand paging virtual memory
kernel interface to the hardware		
terminal controllers terminals	device controllers disks and tapes	memory controllers physical memory

kernel

System Structure – 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.

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

Structure of the THE operating system.

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.
• 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.

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.

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 concering 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.