Operating-System Structures

3.1 Common System Components

Systems as large and complex as operating systems can be created only by partitioning it into smaller pieces. Each piece should be well-delineated portion of the system, with carefully defined inputs, outputs, and functions.

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

3.1.1 Process Management

- ***Process*** is a program in execution (user processes such as compiler and word processing program, or operating system processes or task such as sending a file to a printer). A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task. These resources are given to a process while it is created or allocated while it is running .
- The execution of a process must be sequential. The CPU executes one instruction of the process after another until the process is complete.
- The operating system is responsible for the following activities in connection with process management.
    - Process creation and deletion both user and system processes
    - process suspension and resumption.
    - Provision of mechanisms for:
        - process synchronization
        - process communication

3.1.2 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 (such as reading / writing data from memory during instruction fetch) and I/O devices (such as reading and writes data to main memory via Direct Memory Access - DMA).
- **Main memory** is a volatile storage device. It loses its contents in 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.

3.1.3 File Management

- Computers can store information on several different types of physical media (such as magnetic tapes, magnetic disks .. etc). Each of types has its characteristics such as access speed, capacity, data-transfer rate, and access methods (sequential or random).
- A **file** is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data. It consist of sequence of bits, bytes, lines or records . Data files may be numeric, alphanumeric , alphabetic.
- 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

3.1.4 I/O System Management

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

3.1.5 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.
- Most programs including compilers, assemblers, editors .. etc are stored on disk until loaded into memory, and then use a disk as both the source and the destination of their processing.
- It must be used efficiently and the entire speed of the operation of the computer might hang on the speed of the disk subsystem and the algorithm that manipulate that subsystem.

- The operating system is responsible for the following activities in connection with disk management:
    - Free space management
    - Storage allocation
    - Disk scheduling

3.1.6 Networking (Distributed Systems)

- A ***distributed*** system is a collection processors that do not share memory or a peripheral devices. Each processor has its own local memory and peripheral devices.
- The processors communicate with each other through various communication lines such as high-speed bus or network.
- The processors in the system are connected through a communication network.
- Communication takes place using a ***protocol****s (such as File-Transfer Protocol (FTP), Network File-System (NFS) protocol, and Hypertext Transfer Protocol (http)).*
- A distributed system provides user access to various system resources.
- Access to a shared resource allows:
    - Computation speed-up
    - Increased data availability
    - Enhanced reliability

3.1.7 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.

3.1.8 Command-Interpreter System

- It is an interface between the user and the operating system.
- Many commands are given to the operating system by control statements.
- The commands themselves deal with:
    - process creation and management
    - I/O handling
    - secondary-storage management
    - main-memory management
    - file-system access
    - protection
    - networking

- 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.

3.2 Operating System Services

- **Program execution** – system capability to load a program into memory and to run it. The program must be able to end its execution, either normally or abnormally indicating error).
- **I/O operations** – since user programs cannot execute I/O operations directly such as to rewind a tape drive, or to blank the CRT screen), 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*** in which packets of information is are moved from between processes by the operating system..
- **Error detection** – The operating system should take an appropriate action to ensure correct computing by detecting errors in the CPU and memory hardware (such as memory error of power failure), in I/O devices (such as parity error on tape, lack of paper in the printer) , or in user programs (such as attempt to access an illegal memory location).

Additional Operating System Functions

Additional functions exist not for helping the user, but rather for ensuring efficient system operations.

- **Resource allocation** – allocating resources (such as CPU, main memory, and file storage) to multiple users or multiple jobs running at the same time.
- **Accounting** – keep track of and record which users use how many , how much and what kinds of computer resources. This information could be used for account billing or for accumulating usage statistics that might be used by researchers who wish to reconfigure the system to improve computing services.
- **Protection** – ensuring that all access to system resources is controlled. Security of the system from outsiders is also important.

3.4 System Programs

- System programs provide a convenient environment for program development and execution. It can be divided into:
  - **File management or manipulation** : these programs create, delete, copy, rename, print, and generally manipulate files and directories.
  - **Status information** : Some programs ask the system for date, time, amount of disk or memory space, number of users .. etc. This information is formatted and printed to a terminal or file or output device.
  - **File modification** : Several text editors may be available to create and modify the content of files stored on disk or tapes.
  - **Programming language support** : Compilers, assemblers, interpreters for common programming language (such as C ,

C++ , Java .. etc) are provided to the user with the operating system.

- **Program loading and execution** : Once the program is compiled, it must be loaded into memory to be executed. Debugging systems for either higher-level languages or machine language are needed also.
- **Communications** : Programs that provide the mechanism for creating virtual connections among processes, users, different computer systems. They allow the user to send messages to another screens, browse the web, send electronic-mail messages, to log in remotely or to transfer files from one machine to another.
- **Application programs** : Programs that solve common problems or perform common operations (such as web browsers, spreadsheets, word processors, database systems .. etc).

3.7 System Design and Implementation

Problems when designing and implementing an operating system.

3.7.1 System Design Goals

The first problem is to define the goals and the specifications of the system. At a higher level, the design of the system will be affected by the choice of hardware and type of system (such as batch, single user, multiuser, distributed .. etc). The requirement can be divided into :-

- **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.

Different requirements will result in different systems. Single user system differ from multiuser, multi-access operating system

3.7.2 Mechanisms and Policies

- **Mechanisms** determine **how** to do something, **policies** decide **what** will be done.
- **For example**, a timer construct is a mechanism for ensuring CPU protection, but deciding how long the timer is to be set for a particular user is a policy decision**.**
- The separation of policy from mechanism is a very important principle, it allows maximum flexibility if policy decisions are to be changed later.
- Policy decisions must be made for all resource-allocation and scheduling problems

3.7.3 System Implementation

- Traditionally written in assembly language, operating systems can now be written in higher-level languages (UNIX, OS/2, and WINDOWS NT are mainly written in C..
- 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. (*MS-DOS* Written in Intel 8088 assembly language, thus it is available only for the Intel family. The *UNIX* operating system, on the other hand , which is written mostly in C, is available on a number of different CPU including Pentium, Motorola, Ultra Sparc, Compaq).

3.8 System Generation (SYSGEN)

- Operating systems are designed to run on any of a class of machines at variety of sites with a variety of peripheral configurations; the system must be configured or generated for each specific computer site (this process sometimes known as **system generation SYSGEN**)
- SYSGEN program reads from a given file, or ask the operator of the system for information concerning the specific configuration of the hardware system or probes the hardware directly to determine what components are there.
- After the operating system is generated, it must be made available for the use by the hardware.
- The hardware must know where the kernel is and how to load it.
  - ***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.