Chapter 1: Introduction to Operating System.

1.1 Introduction to operating System

- An operating system (OS) is system software that manages computer hardware and software resources and provides common services for computer programs.
- An operating system is a collection of system programs that control computer and any other peripherals connected to it. The program that hides the truth about the hardware from the programmer and present and a nice simple view a named file that can be read & written as "operating system". Operating system shields the programmer from the interface, the abstraction offers by the operating system is slower & easier to use than the underlying hardware.
- An operating System is a layer of software on a bare hardware machine that performs two basic functions:
  i. Resource management.
  ii. Virtual machine management.

1. Resource Management

- A computer consists of a set of resources such as processors, memories, timers, disks, printers and many others.
- The Operating System manages these resources and allocates them to specific programs.
- As a resource manager, the Operating system provides the controlled allocation of the processors, memories, I/O devices among various programs.
- Moreover, multiple user programs are running at the same time, the processor itself is a resource and the Operating System decides how much processor time should be given for the execution of a particular user program.
- An operating system is a control program, a control program manages the execution of user program to prevent errors and improve use of computer. It is especially concerned with the operation and control of I/O devices. When a computer has multiple users, the operating system manages and protects the memory I/O devices. The operating system also keeps in trace that who is using which resource and to whom to the grant resource.

2. Virtual Machine Management

- The architecture (instruction set, memory, I/O, and bus structure) of most computers at the machine level language is primitive and awkward to program, especially for input/output operations.
- Users do not want to be involved in the programming of storage devices. Moreover, Operating System provides a simple, high-level abstraction such that these devices contain a collection of named files.
- Such files consist of the useful piece of information like a digital photo, email messages, or web page.
- Operating System provides a set of basic commands or instructions to perform various operations such as read, write, modify, save or close.
- Also, dealing with them is easier than directly dealing with hardware. Thus, Operating System hides the complexity of hardware and presents a beautiful interface to the users.

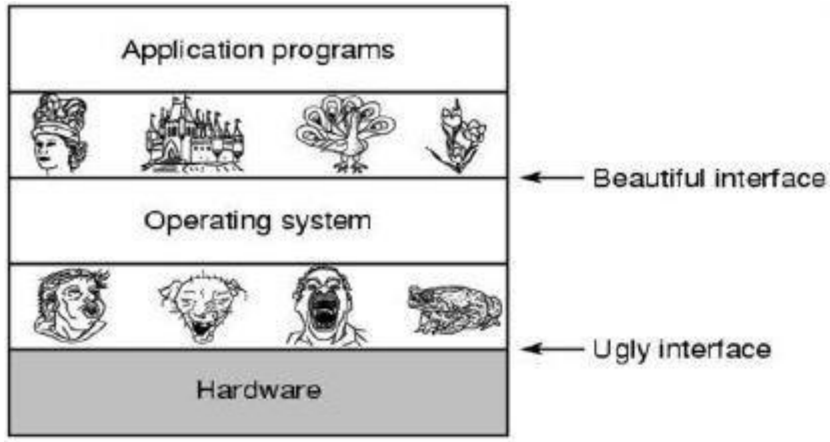


Figure. Operating Systems turn ugly hardware into beautiful abstractions.

- Just as the operating system shields (protect from an unpleasant experience) the programmer from the disk hardware and presents a simple file-oriented interface, it also conceals a lot of unpleasant business concerning interrupts, timers, memory management, and other low-level features.
- In each case, the abstraction offered by the operating system is simpler and easier to use than that offered by the underlying hardware.
- Moreover, in this view, the function of the operating system is to present the user with the equivalent of an **extended machine** or **virtual machine** that is easier to work with than the underlying hardware.
- The operating system provides a variety of services that programs can obtain using special instructions called system calls.

1.3 Types of Operating System

1.3.1 Batch Processing:

In the batch system requires the grouping of similar jobs which consists of programs data and system commands. The instructions, data and some controlled information are submitted to the computer operator in the form of job. The users are not allowed to interact with the computer system. Thus, the programs that do not require interaction are well served by batch OS.

Since the jobs are executed in the FIFO manner, the batch OS require very simple CPU scheduling techniques. In addition, batch system allow only one user program to reside in the memory at a time, thus memory management is also very simple affairs in batch OS.

Since only one program is in execution at a time any time critical device management is not required, which simplifies the I/O management. Since the files are accessed in a serial manner, a concurrency control mechanism is not required, making the file management also very simple matter in batch OS.

The problems with Batch Systems are as follows:
- Lack of interaction between the user and the job.
- CPU is often idle, because the speed of the mechanical I/O devices is slower than the CPU.
- Difficult to provide the desired priority

1.3.2 Multi-Processing:

A computer's capability to process more than one task simultaneously is called *multiprocessing*. A multiprocessing operating system is capable of running many programs simultaneously, and most modern network operating systems (NOSs) support multiprocessing. These operating systems include Windows NT, 2000, XP, and UNIX.

A multiprocessing system uses more than one processor to process any given workload, increasing the performance of a system's application environment beyond that of a single processor's capability. Collections of processors arranged in a loosely coupled configuration and interacting with each other over a communication channel have been the most common multiprocessor architecture.

Advantage:

- Increase throughput
- Economy of scale
- Increased reliability

Disadvantage:

- If one processor fails then it will affect in the speed
- Multiprocessor systems are expensive
- Complex OS is required
- Large main memory required.

1.3.3 Time Sharing Operating System

Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. The main difference between Multiprogrammed Batch Systems and Time-Sharing Systems is that in case of multiprogrammed batch systems, the objective is to maximize processor use, whereas in Time-Sharing Systems, the objective is to minimize response time. Multiple jobs are executed by the CPU by switching between them, but the switches occur so frequently. Thus, the user can receive an immediate response. For example, in a transaction processing, the processor executes each user program in a short burst or quantum of computation. That is, if n users are present, then each user can get a time quantum. When the user submits the command, the response time is in few seconds at most.

Advantages

- Provides the advantage of quick response
- Avoids duplication of software
- Reduces CPU idle time

Disadvantages

- Problem of reliability
- Question of security and integrity of user programs and data
- Problem of data communication

1.3.4 Real Time System:

The time taken by the system to respond to an input and display of required updated information is termed as the response time. A real-time system is defined as a data processing system in which the time interval required to process and respond to inputs is so small that it controls the environment. It is a multitasking operating system that aims at executing real-time applications. Real-time operating systems often use specialized scheduling algorithms so that they can achieve a deterministic nature of behavior. The main object of real-time operating systems is their quick and predictable response to events. They either have an event-driven design or a time-sharing one. An event-driven system switches between tasks based of their priorities while time-sharing operating systems switch tasks based on clock interrupts

- Hard real-time systems guarantee that critical tasks complete on time. In hard real-time systems, secondary storage is limited or missing and the data is stored in ROM. In these systems, virtual memory is almost never found.
- Soft real-time systems are less restrictive. A critical real-time task gets priority over other tasks and retains the priority until it completes. Soft real-time systems have limited utility than hard real-time systems. For example, multimedia, virtual reality, Advanced Scientific Projects like undersea exploration and planetary rovers, etc.

1.3.5 Network Operating System

A Network Operating System runs on a server and provides the server the capability to manage data, users, groups, security, applications, and other networking functions. The primary purpose of the network operating system is to allow shared file and printer access among multiple computers in a network, typically a local area network (LAN), and a private network or to other networks. Certain standalone operating systems, such as Microsoft Windows NT and Digital's OpenVMS, come with multipurpose capabilities and can also act as network operating systems. Some of the most well-known network operating systems include Microsoft Windows Server 2003, Microsoft Windows Server 2008, Linux and Mac OS X.

Mainly there are two types of network operating.

Peer-to-peer network operating systems allow users to share resources and files located on their computers and to access shared resources found on other computers. In a peer-to-peer network, all computers are considered equal; they all have the same privileges to use the resources available on the network. Peer-to-peer networks are designed primarily for small to medium local area networks. Windows for Workgroups is an example of the program that can function as peer-to-peer network operating systems.

Client/server network operating systems allow the network to centralize functions and applications in one or more dedicated file servers. The file servers become the heart of the system, providing access to resources and providing security. The workstations (clients) have access to the resources available on the file servers. The network operating system allows multiple users to simultaneously share the same resources irrespective of physical location. Novell Netware and Windows 2000Server are examples of client/ server network operating systems

1.3.6 Distributed Operating System

A distributed operating system is an operating system that runs on several machines. Its purpose is to provide a useful set of services, generally to make the collection of machines behave more like a single machine. Distributed operating systems typically run cooperatively on all machines whose resources they control. These machines might be capable of independent operation, or they might be usable merely as resources in the distributed system. In some architectures, each machine is an equally powerful peer as all the others. In other architectures, some machines are permanently designated as master or are given control of particular resources. In yet others, elections or other selection mechanisms are used to designate some machines as having special roles, often controlling roles.

Advantages

- Sharing of resources.
- Reliability.
- Communication.
- Computation speedup

1.4 Operating system components

An operating system provides the environment within which programs are executed. To construct such an environment, the system is partitioned into small modules with a well-defined interface. The design of a new operating system is a major task. It is very important that the goals of the system be will defined before the design begins. The type of system desired is the foundation for choices between various algorithms and strategies that will be necessary.

A system as large and complex as an operating system can only be created by partitioning it into smaller pieces. Each of these pieces should be a well-defined portion of the system with carefully defined inputs, outputs, and function. Obviously, not all systems have the same structure. However, many modern operating systems share the system components outlined below.

- Process management
- I/O management
- Main Memory management
- File & Storage Management
- Protection
- Networking
- Protection
- Command Interpreter

1.4.1 Process Management System

The CPU executes a large number of programs. While its main concern is the execution of user programs, the CPU is also needed for other system activities. These activities are called processes. A process is a program in execution. Typically, a batch job is a process. A time-shared user program is a process. A system task, such as spooling, is also a process. For now, a process may be considered as a job or a time-shared program, but the concept is actually more general.

In general, a process will need certain resources such as CPU time, memory, files, I/O devices, etc., to accomplish its task. These resources are given to the process when it is created. In addition to the various physical and logical resources that a process obtains when it is created, some initialization data (input) may be passed along. For example, a process whose function is to display on the screen of a terminal the status of a file, say F1, will get as an input the name of the file F1 and execute the appropriate program to obtain the desired information.

We emphasize that a program by itself is not a process; a program is a passive entity, while a process is an active entity. It is known that two processes may be associated with the same program, they are nevertheless considered two separate execution sequences.

A process is the unit of work in a system. Such a system consists of a collection of processes, some of which are operating system processes, those that execute system code, and the rest being user processes, those that execute user code. All of those processes can potentially execute concurrently.

The operating system is responsible for the following activities in connection with processes managed.

- The creation and deletion of both user and system processes
- The suspension is resumption of processes.
- The provision of mechanisms for process synchronization
- The provision of mechanisms for deadlock handling.

1.4.2 Memory Management System

Memory is central to the operation of a modern computer system. Memory is a large array of words or bytes, each with its own address. Interaction is achieved through a sequence of reads or writes of specific memory address. The CPU fetches from and stores in memory.

In order for a program to be executed it must be mapped to absolute addresses and loaded in to memory. As the program executes, it accesses program instructions and data from memory by generating these absolute is declared available, and the next program may be loaded and executed.

In order to improve both the utilization of CPU and the speed of the computer's response to its users, several processes must be kept in memory. There are many different algorithms depends on the particular situation. Selection of a memory management scheme for a specific system depends upon many factor, but especially upon the hardware design of the system. Each algorithm requires its own hardware support.

The operating system is responsible for the following activities in connection with memory management.

- Keep track of which parts of memory are currently being used and by whom.
- Decide which processes are to be loaded into memory when memory space becomes available.

- Allocate and deallocate memory space as needed.

1.4.3 Secondary Storage Management System

The main purpose of a computer system is to execute programs. These programs, together with the data they access, must be in main memory during execution. Since the main memory is too small to permanently accommodate all data and program, the computer system must provide secondary storage to backup main memory. Most modem computer systems use disks as the primary on-line storage of information, of both programs and data. Most programs, like compilers, assemblers, sort routines, editors, formatters, and so on, are stored on the disk until loaded into memory, and then use the disk as both the source and destination of their processing. Hence the proper management of disk storage is of central importance to a computer system.

There are few alternatives. Magnetic tape systems are generally too slow. In addition, they are limited to sequential access. Thus, tapes are more suited for storing infrequently used files, where speed is not a primary concern.

The operating system is responsible for the following activities in connection with disk management

- Free space management
- Storage allocation
- Disk scheduling.

1.4.4 I/O System

One of the important jobs of OS is to manage various I/O devices including mouse, keyboard, touch pad etc. I/O system required to take an application I/O request and send it to physical device, then take whatever response come back from device and send it to application. The I/O system consists of:

- A buffer caching system
- A general device driver code
- Drivers for specific hardware devices.

Only the device driver knows the peculiarities of a specific device.

1.4.5 File Management system

File management is one of the most visible services of an operating system. Computers can store information in several different physical forms; magnetic tape, disk, and drum are the most common forms. Each of these devices has its own characteristics and physical organization.

For convenient use of the computer system, the operating system provides a uniform logical view of information storage. The operating system abstracts from the physical properties of its storage devices to define a logical storage unit, the file. Files are mapped, by the operating system, onto physical devices.

A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data. Data files may be numeric, alphabetic or alphanumeric. Files may be free-form, such as text files, or may be rigidly formatted. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by its creator and user. It is a very general concept.

The operating system implements the abstract concept of the file by managing mass storage device, such as types and disks. Also, files are normally organized into directories to ease their use. Finally, when multiple users have access to files, it may be desirable to control by whom and in what ways files may be accessed.

The operating system is responsible for the following activities in connection with file management:

- The creation and deletion of files
- The creation and deletion of directory
- The support of primitives for manipulating files and directories
- The mapping of files onto disk storage.
- Backup of files on stable (nonvolatile) storage.

1.4.6 Protection System

The various processes in an operating system must be protected from each other's activities. For that purpose, various mechanisms which can be used to ensure that the files, memory segment, CPU and other resources can be operated on only by those processes that have gained proper authorization from the operating system.

For example, memory addressing hardware ensure that a process can only execute within its own address space. The timer ensure that no process can gain control of the CPU without relinquishing it. Finally, no process is allowed to do its own I/O, to protect the integrity of the various peripheral devices.

Protection refers to a mechanism for controlling the access of programs, processes, or users to the resources defined by a computer controls to be imposed, together with some means of enforcement.

Protection can improve reliability by detecting latent errors at the interfaces between component subsystems. Early detection of interface errors can often prevent contamination of a healthy subsystem by a subsystem that is malfunctioning. An unprotected resource cannot defend against use (or misuse) by an unauthorized or incompetent user.

1.4.7 Networking

A networking system of a computer OS that is designed of networking use. Networking system is an OS with file, task and job management. However in some earlier OS, it was a separate component that enhance a basic, non- networking OS by adding networking capabilities. NOS is designed primarily to support workstation, PC and in some instances, older terminal that are connected to LAN. Networking system allows to share file and printer access among multiple computers in a Network, to enable the sharing of data

1.4.8 Command Interpreter System

One of the most important component of an operating system is its command interpreter. The command interpreter is the primary interface between the user and the rest of the system.

Many commands are given to the operating system by control statements. When a new job is started in a batch system or when a user logs-in to a time-shared system, a program which reads and interprets control statements is automatically executed. This program is variously called (1) the control card interpreter, (2) the command line interpreter, (3) the shell (in Unix), and so on. Its function is quite simple: get the next command statement and execute it.

The command statement themselves deal with process management, I/O handling, secondary storage management, main memory management, file system access, protection, and networking.

In the following sections of this Chapter we show four important components of the operating system. There are process management, file organization, input/output, and memory management.

1.6 Operating System Services

1.6.1 System Calls

A system call is how a program requests a service from an operating system's kernel. This may include hardware related services (e.g. accessing the hard disk), creating and executing new processes, and communicating with integral kernel services (like scheduling). System calls provide an essential interface between a process and the operating system.

A system call is a mechanism that is used by the application program to request a service from the operating system. They use a machine-code instruction that causes the processor to change mode. An example would be from supervisor mode to protected mode. This is where the operating system performs actions like accessing hardware devices or the memory management unit. Generally, the operating system provides a library that sits between the operating system and normal programs.

System calls provide the interface between a process and the operating system. Most operations interacting with the system require permissions not available to a user level process, e.g. I/O performed with a device present on the system, or any form of communication with other processes requires the use of system calls.

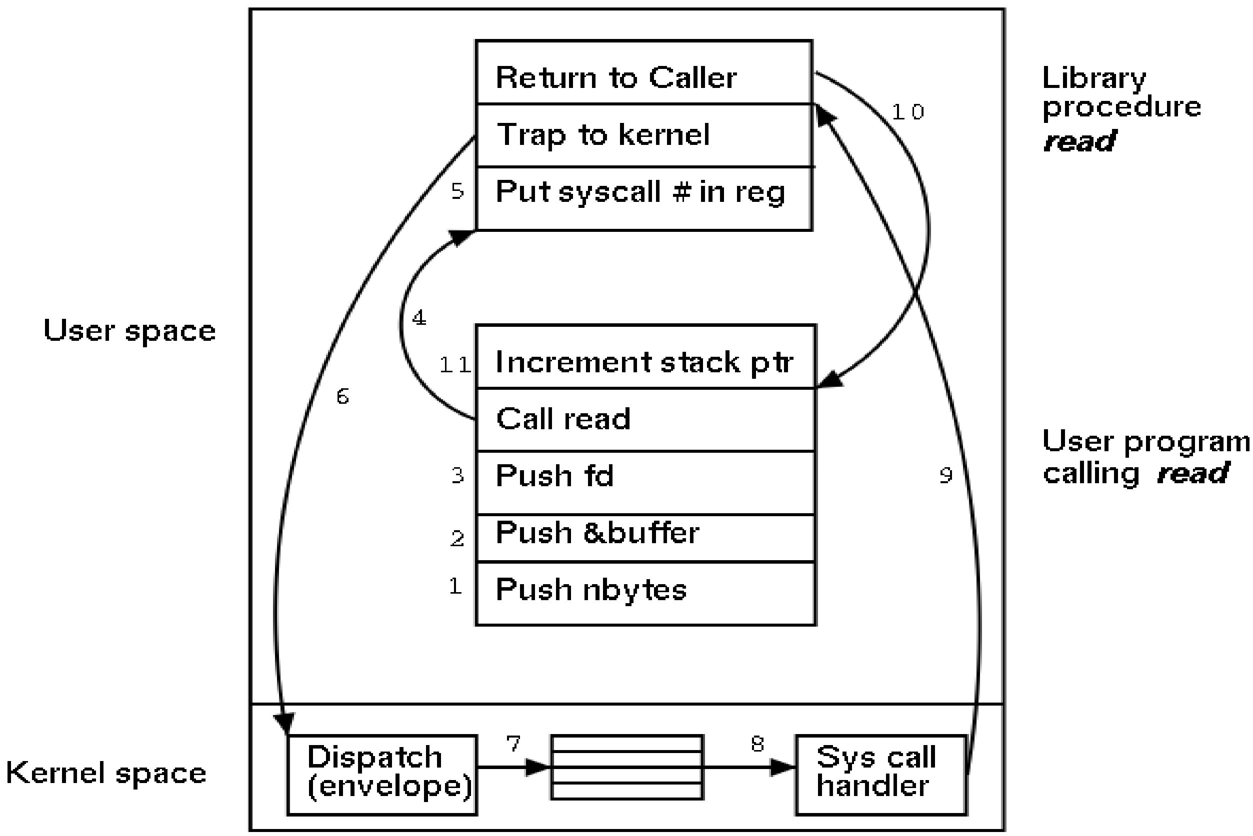
Services Provided by System Calls:

1. Process creation and management
2. Main memory management
3. File Access, Directory and File system management
4. Device handling(I/O)
5. Protection
6. Networking, etc.

Example of read () system call

read (fd, buffer, nbytes);

The system call (and the library procedure) return the number of bytes actually read in count. This value is normally the same as nbytes, but may be smaller, if, for example, end-of-file is encountered while reading.
It is explained in below diagram:



__Compiled By__: Er. Pradip Khanal
Lecturer, ACEM

SERVICES	WINDOWS	UNIX
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Manipulation	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()
Protection	SetFileSecurity() InitlializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

1.6.2 Shell and Kernel

Shell and the kernel are the parts of this Operating system. These both parts are used for performing any operation on the system. When a user gives his command for performing any operation, then the request will go to the shell parts, the shell parts is also called as the interpreter which translate the human program into the machine language and then the request will be transferred to the kernel that means shell is just as the interpreter of the commands which converts the request of the user into the machine language.

Kernel is also known as heart of operating system and every operation is performed by using the kernel, when the kernel receives the request from the shell then this will process the request and display the results on the screen. The various types of operations those are performed by the kernel are as followings: -

- It controls the state of the process means it checks whether the process is running or process is waiting for the request of the user.

- Provides the memory for the processes those are running on the system means kernel runs the allocation and de-allocation process, first when we request for the service then the kernel will provide the memory to the process and after that it also release the memory which is given to a process.

- The kernel also maintains a time table for all the processes those are running means the kernel also prepares the schedule time means this will provide the time to various process of the CPU and the kernel also puts the waiting and suspended jobs into the different memory area.

- When a kernel determines that the logical memory doesn't fit to store the programs. Then he uses the concept of the physical memory which will store the programs into temporary manner i.e. virtual memory.

- Kernel also maintains all the files those are stored into the computer system and the kernel also stores all the files into the system as no one can read or write the files without any permission. So that the kernel system also provides us the facility to use the passwords and also all the files are stored into the particular manner.

As we have learned there are many programs or functions those are performed by the kernel but the functions those are performed by the kernel will never be shown to the user. And the functions of the kernel are transparent to the user.