Note Junction

Best Note Provider

Note By: Roshan BiSt

Scanned with CamScanner

Unit-1

Operating System Overview:-

⊛Introduction and Definition:

Computer software can be divided into two main categories: application software and system software. Application software consists of the programs for performing tasks particular to the machines utilization. This software is designed to solve a particular problem for users. Spreadsheets, database systems, powerpoint etc. are its examples.

System software acts as an interface between the hardware of the computer and the application software that users need to run on the computer. Different versions of Windows, Linux etc are its example. The most important type of system software is Operating System (OS).

Defn → An Operating System (OS) is a collection of programs that acts as an interface between a user of a computer and the computer hardware.

⊛Two views/aspects of Operating System:

1) Operating system as an extended machine:-

The architecture of most computers at machine-language level is primitive and awkward to program, especially for input/output. So one of the major task of operating system is to hide the hardware and present programs with nice, clean, consistent and user friendly. We are not interested on how hardware components work together to perform our tasks, all we need is task to be completed with no complexity and overhead.

Thus all we need is the program should hide the truth about hardware and present simple and nice view. To achieve it all the operating system provides a variety of services that programs can obtain using special instructions called system calls.

Scanned with CamScanner

pp) Operating System as resource manager:-

The major function of any machine is how resources are managed. For example we constructed system which provides all necessary banking operations, Now one of the user needs to deposit amount while another user requires to print statement i.e, first user requires access to database while the second one too requires to access database as well as printer. Users are interested in getting their job done quicker anyhow. But the problem occurs with the system how to manage these requests. The process of managing all those requests/resources is called resource management and this is done by resource manager.

Hardware provides basic computing resources CPU, Memory, I/O devices etc. whereas operating system controls and co-ordinate the use of the hardware among various users.

⊛ Evolution/History of Operating System:-

The first true digital computer was designed by the English mathematician: Charles Babbage. But this computer was purely mechanical and did not have an operating system. later Babbage realized that he would need a software for his analytical engine, so he hired a young woman named Ada lovelace as the world's first programmer. Operating systems are closely tied to the architecture of the computers on which they run. So successive generations of computers describes what their operating systems were like.

1) The first generation of Computer (1945-55):-

→ Vaccum-Tubes were used for building machine.
→ A single group of people designed, built, programmed, operated and maintained each machine.
→ All programming was done in absolute machine code.
→ Programming languages and operating system was unknown.

Scanned with CamScanner

ii) The second generation of computer (1955-65):-

→ Transistors were used for building machine.
→ Batch systems were used.
→ Programs were written in FORTAN and assembly language.
→ Typical operating systems were used FMS(FORTON Monitor System) and IBSYS (IBM's Operating System).

iii) The third generation of computer (1965-1980):-

→ IC's were used for building machines.
→ Multiprogramming came into existence so that processors can be kept busy 100 percent of the time.
→ Operating systems: OS/360, MULTICS, CTSS were used.

iv) The fourth generation of computer (1980- Present):-

→ LSI's and VLSI's were used for building machines.
→ Microcomputers came into existence.
IBM PC: Operating System was MS-DOS.
IBM PC/AT: Operating System was MS-DOS.
Apple Macintosh: Operating System was Macintosh.
→ Introduction to Windows: Because of the influence of the success of Macintosh, Microsoft decided to build a GUI-based system which originally ran on top of MS-DOS and named it Windows. later on different versions of Windows like Windows-XP, Windows-7, Windows-8, Windows-10 came.

⊛ Types of Operating Systems:-

1) Mainframe Operating Systems:

→ The operating systems used in mainframe computers are called mainframe operating systems.
→ OS/390 is an example of Mainframe OS.
→ Mainframe OS have very high I/O capacity.
→ They provide three types of services batch, transaction processing and timesharing.

Scanned with CamScanner

Server Operating Systems:-

→ The operating systems which run on servers, which are either very large personal computers, workstations or even mainframes, are called server operating systems.

→ Linux, Windows 2000 etc. are its examples.

→ They serve multiple users at once over a network and then allow the users to share software and hardware resources.

→ They provide services like print service, file service or web service.

Multiprocessor Operating Systems:-

→ If more than one processor is used to handle multiple tasks at a time and reduce systems work load then, this is called a multiprocessor operating system.

→ By connecting multiple CPUs into a single system we can get heavy computing power out of the machine.

→ Depending upon precisely how they are connected and what is shared, these systems are called parallel computers, multicomputers or multiprocessors.

PC Operating System:

→ The operating system used in Personal Computers are called PC operating systems.

→ They have comparatively less processing power and memory than mainframes.

→ Windows-7, Macintosh, Linux etc. are its examples.

→ They are widely used for word processing, spreadsheets and internet access.

Scanned with CamScanner

v) Real-time Operating Systems:

→ The operating systems that have time as the key parameter is called Real-time OS.
→ QNX, RTLinux, HART etc are its examples.
→ Real time OS has a well defined, fixed time constraints and processing must be done within the defined constraints.
→ They provide quick event response and thus meet the scheduling deadlines.

vi) Embedded Operating Systems:

→ The operating systems such as Palm OS and Windows CE (Consumer electronics) that run on embedded devices such as a palmtop computer or PDAs (Personal Digital Assistants) and mobile phones are called embedded operating systems.
→ These devices have generally low processing power, memory, size and battery life.

vii) Smart Card Operating Systems:

→ Smart cards are credit card-sized devices containing a CPU chip.
→ They have very low processing power and memory capabilities.
→ Resource management and protection are two of the main tasks of smart card operating systems.

viii) Multiprogramming Operating Systems:-

→ Multiprogramming is a technique to execute number of programs simultaneously in a single processor.
→ Multitasking and multiprocessing are two different forms of multiprogramming.
→ It manages computer related resources like CPU, memory, and I/O devices.
→ It provides high CPU utilization, efficient memory utilization and supports multiple simultaneous interactive users.

Scanned with CamScanner

⊛ Operating System Structures:-

i) Monolithic Systems:- In monolithic systems the operating system is written as a collection of procedures, each of which can call any of the other ones whenever it needs to. In this approach one first compiles all the individual procedures or files containing the procedures and then binds them all together into a single object file using the system linker. This organization suggests a basic structure for the operating system:

→ 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.

In this model, for each system call there is one service procedure that takes care of it. The utility procedures do things that are needed by several service procedures, such as fetching data from user programs.

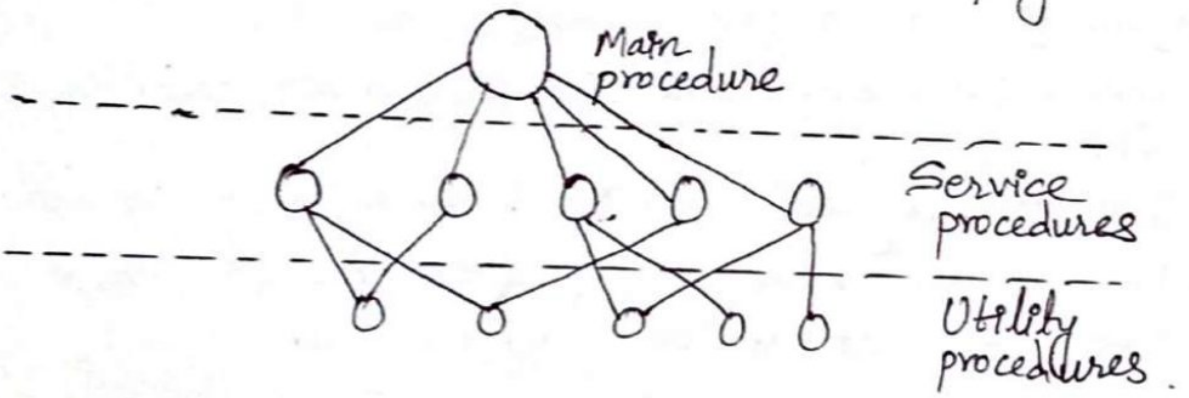

fig. A simple structuring model for monolithic systems.

ii) 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

Scanned with CamScanner

The table above was the structure of the layered systems.

This system had 6 layers as shown in table above. Layer 0 deals with the allocation of processors and multiple processes run on a single processor. Layer 1 did the memory management. It allocated space for processes in main memory and 512K word drum used for holding parts of processes. Layer 2 handeled communication between each process and the operator console. Layer 3 took care of managing the I/O devices and buffering the information streams to and from them. Layer 4 was where the user programs were found and the system operator process was located in layer 5.

(iii) Virtual machines:- This system originally called CP/CMS and later renamed VM/370 had multiprogramming and more convenient interface. The heart of this system, known as the virtual machine monitor, runs on the bare hardware and does the multiprogramming, providing several virtual machines to the next layer up. They are exact copies of the bare hardware, including kernel/user mode, I/O, interrupts and everything else the real machine has.

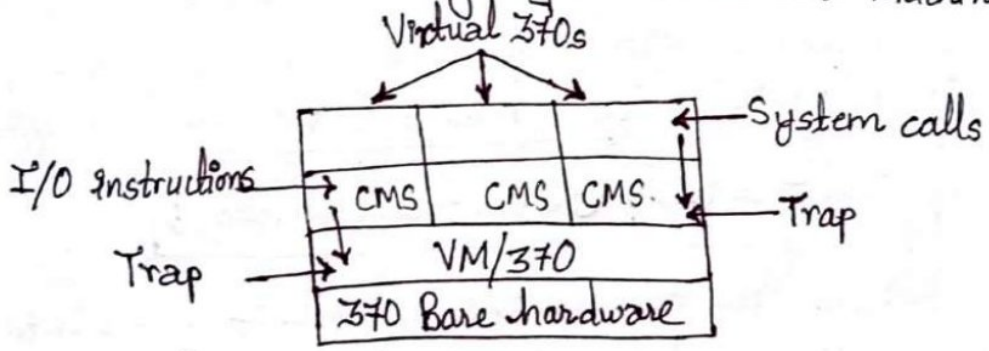

Fig. The structure of VM/370 with CMS.
CMS → stands for Conversational Monitor System.

(iv) Client Server Model:-

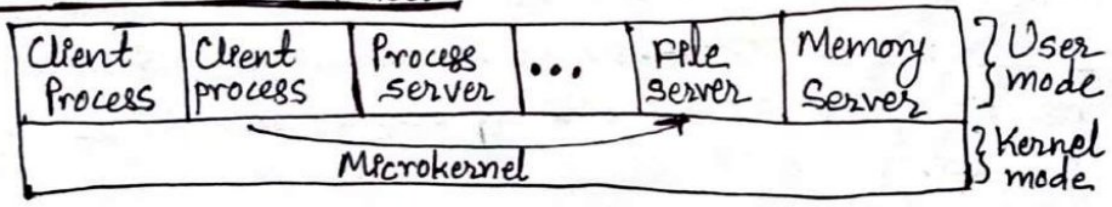

Fig. The client-server model.

Scanned with CamScanner

In this model to request a service, such as reading a block of a files, a user process (now known as client process) sends the request to a server process, which then does the work and sends back the answer. All the kernel handles the communication between clients and servers. By splitting the operating system up into parts each of which only handles one facet of the system.

⊛. System Calls:-

System calls are the interface between a process (i.e, user program) and the operating system. They are generally available as assembly language instructions, however languages like C that are defined to replace assembly language for systems programming allow system calls to be made directly via procedure calls.

To make a system call mechanism we need three parameters: the first parameter specifies file, the second parameter points to the buffer and the third parameter gives the number of bytes to read. like all system calls, it is invoked from a C program by calling a library procedure with the same name as the system call; read. A call from a C program looks like:

```
count = read (fd, buffer, nbytes);
```

The system call returns the number of bytes actually read in count. This value is normally the same as nbytes, but maybe smaller if end of file is encountered while reading. If the system call cannot be carried out, either due to an invalid parameter or a disk error, count is set to -1. Programs should always check the results of a system call to see if an error occured.

Scanned with CamScanner

⊛. Handling System Calls / System call flow with block diagram

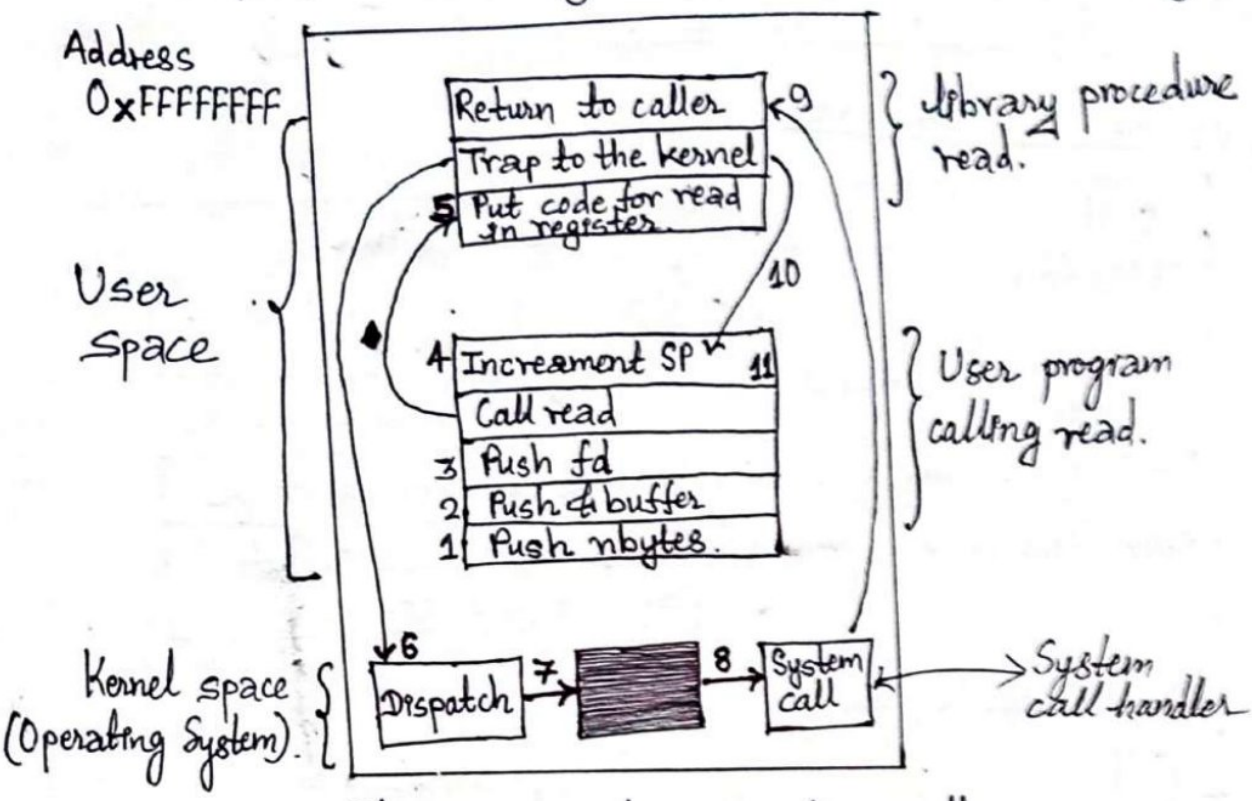

Fig. Steps in making system call

→ For calling the read library procedure the calling program first pushes the parameters onto the stack (steps 1-3 in fig). C compilers push the parameters onto the stack in reverse order. The first and third parameters are passed by value, but the second parameter is passed by refrence.

→ Then the library procedure is called. (step 4).

→ The library procedure puts the system call number in a place where the OS expects it, such as a register. (step 5)

→ The library procedure then executes a TRAP instruction to switch from user mode to kernel mode. (step 6).

→ Started kernel mode examines the system call number and then dispatches to the correct system call handler. (step 7).

→ The system call handler runs (step 8).

→ Once system call handler completes its work control is returned to the user space library procedure. (step 9).

→ Then kernel is trapped. (step 10).

→ Finally stack pointer (SP) is incremented to remove the parameters pushed. (step 11).

Scanned with CamScanner

⊛. Types of System calls / System calls for process, file and directory management.

i) 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 execute and return status.

ii) 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 buffer
n = write (fd, buffer, nbytes)	Write data from a buffer into file.
position = seek (fd, offset, whence)	Move a file pointer
s = stat (name, &buf)	Get a file's status information.

iii) 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.

Scanned with CamScanner

⊛. The Shell:

The shell is the UNIX command interpreter. Although it is not a part of operating system, it makes heavy use of many operating system features and thus serves as a good example of how the system calls can be used. It is also the primary interface between a user sitting at his terminal and the operating system, unless the user is using a graphical user interface. Many shells exist, including sh, csh, ksh and bash.

When any user logs in, a shell is started up. The shell has the terminal as standard input and standard output. It starts out by typing the prompt, a character such as a dollar sign, which tells the user that the shell is waiting to accept a command. If the user now types date then the shell creates a child process and run the data program as the child. While the child process is running, the shell waits for it to terminate. When the child finishes, the shell types the prompt again and tries to read the next input line.

→ The user can specify that standard output be redirected to a file. For example:

date > file

→ Similarly standard input can be redirected to file as:

sort < file1 > file 2.

This invokes the sort program with input token from file 1 and send the sorted output to file 2.

⊛. Open Source Operating Systems:-

The operating systems that are open to all, anyone can get the source code and make desirable changes are called open-source operating systems.

For example:- Android is open source, so phones of different manufacturer have different look and feel to it.

Scanned with CamScanner

The code that google provides is called stock code. Manufacturers change it according to their needs and add functionalities to their phones to make them different from others. Like Samsung's newly released note 4 has support for stylus, lg g3 has its own awesome multitasking or the camera features. Following are the different open source operating systems for computer available in the market:-

i) Cosmos → This is an open source operating system written mostly in C# programming language. Its full form is open source managed operating system. Till 2016, Cosmos did not intend to be a fully fitted with features but this system allowed other developers to easily build their own OS. It also hid the inner workings of the hardware from the developers thus providing data abstraction.

ii) Free DOS → This was a free operating system developed for systems compatible with IBM PC computers. Free DOS provides a complete environment to run legacy software and other embedded systems. It can be booted from a floppy disk or USB flash drive. Free DOS is licensed under GNU (General Public License) and contains free and open source software.

iii) Genode → Genode is also free as well as open source. It contains a microkernel layer and different user components. Genode can be used as an operating system for computers, tablets etc. as required. As it has a small code system it is also used as a base for virtualisation, interprocess communication, software development etc.

iv) Ghost OS → This is a free, open source operating system developed for personal computers. It started as a research project and development to contain various advanced features like graphical user, interface, C library etc. It has features like multiprocessing and multitasking and is based on Ghost kernel. Most of the programming in Ghost OS is done by C++.

Scanned with CamScanner

⊛. Functions of OS:-

this is out of micro-syllabus but it is imp and maybe asked in exam

i) Security:- The operating system uses password protection to protect user data and similar other techniques. It also prevents unauthorized access to programs and user data.

ii) Control over system performance:- It monitors overall system health to help improve performance. It records the response time between service requests and system response to have a complete view of the system health. It provides information needed to troubleshoot problems and help to improve performance.

iii) Error detecting aids:- Operating system constantly monitors the system to detect errors and avoid the malfunctioning of computer system.

iv) Coordination between other software and users:- Operating systems also coordinate and assign interpreters, compilers, assemblers and other software to the various users of the computer systems.

v) Memory Management:- The operating system manages the Primary memory or main memory. It keeps tracks of primary memory, i.e, which bytes of memory are used by which user program. In multiprogramming, the OS decides the order in which ~~OS~~ process are granted access to memory, and for how long. It allocates the memory to a process when the process requests it and ~~del~~ deallocates the memory when the process has terminated.

vi) Process management:- In a multiprogramming environment, the OS decides the order in which processes have access to the processor, and how much processing time each process has. This function of OS is called process scheduling.

Scanned with CamScanner

vii) Device Management:- An OS manages device communication via their respective drivers. It keeps tracks of all devices connected to system. It decides which process gets access to a certain device and for how long. It allocates devices when needed and deallocates devices when they are no longer required.

viii) File Management:- A file system is organized into directories for efficient or easy navigation and usage. OS keeps track of where information is stored, user access settings and status of every file and more...

Scanned with CamScanner