Basic Introduction

- In the 1965, the MIT (Massachusetts Institute of Technology), AT&T Bell Labs, and General Electric developed an experimental time sharing operating system called Multics (Multiplexed Information & Computing Service).
- In 1969, Ken Thompson and Dennis Ritchie, other System engineers at AT&T's Bell Labs, created the UNICS (UNiplexed Information and Computing Service) on PDP – 7 (Programmed Data Processor)
- UNICS was developed in Assembly Language.
- In 1972, UNIX was rewritten in the C programming language and renamed as "UNIX" and moved to the PDP – 11/20.

Features of UNIX

UNIX offers several features like Multiuser, Multitasking, Inter Process Communication, Security, Portability, and Open Source.

1) Multiuser :

In Multiuser System, the same computer resources say hard disk, memory etc. are accessible to many users. The Users are given different terminals to operate from. A terminal is a keyboard and a monitor, which are the input and output devices for that user. All terminals are connected to main computer whose resources are available by all users. So a user at any of the terminals can use not only computer but also any peripherals that may be attached e.g. Printer, Scanner etc.

At the heart of UNIX installation, is host machine which is known as Server or Consol. The no of terminals connected to the host machine depends on no of ports that are present in its controller part.

There are several types of terminals which are as follows:

a) **Dumb Terminal**:

This terminal consists of a keyboard & a display unit with no memory or disk of its own. This terminal can never act as independent machine. If they are to be used, they have to be connected with Server machine.

b) Terminal Emulation:

A PC has an own microprocessor, memory and disk drive. By attaching this PC to a host via cable & running software from this PC to the host we can emulate work like a dumb terminal. However the memory & disk are used of PC but can't carry out any processing on its own. Like Dumb Terminal, it transmits its processing Job to the host machine. The Software that

makes a PC work like Dumb terminal is called Terminal Emulation S/W. for e.g. VTERM, XTALK.

c) Dial-in Terminal:

This terminal use telephone lines to connect with host machine. To communicate via telephone lines, it is necessary to attach unique modem to a terminal as well as to the host machine.

2) The Bulding-Block Approach

UNIX does not pack too many features in few tools, UNIX has developed a few hundred commands, each of which performs one simple job only.

3) The UNIX Toolkit

The UNIX toolkit contains general-purpose tools, text manipulation utilities (filters), compilers and interpreters, networked applications and system administration tools. UNIX also provides a choice of shells.

4) Pattern Matching

UNIX also provide pattern matching features. The * (metacharacter) is a special character used by the system to indicate that it can match a number of file names. For example chapter*. It will match find all file starting from chapter (chapter1, chapter2, chapter3 etc.).

Some of the most advanced and useful tools also use a special expression called a regular expression that is framed with characters from this set.

5) Multitasking:

UNIX is capable of carrying out more than one job at same time i.e. in UNIX, a single user can also run multiple tasks concurrently. It is usual for a user to edit a file, print another on the printer, send email to a friend and browse the World Wide Web – all without leaving any of the applications. The kernel is designed to handle a user's multiple needs. In such a multitasking situation, only one job runs in the foreground while the rest run in the background. You can switch jobs between background and foreground, suspend or even terminate them. Multitasking is an important component of the process management system.

6) Inter process Communication (IPS):

This feature allows user to pass on data, exchange mail or program to another users in networks. This communication may be within the Network of single main computer or between two or more such Computer Network.

7) Security:

UNIX allows sharing of data but to secure them it provides three levels of protection.

a) By assigning login name and password to individual user ensuring that no other user can have access to your work.
b) At file level, there are read, write and execute permission to each file. It decides who can access particular file, who can modify it and who can execute it.
We can apply this permission in combination to User, Group and Other.
c) By performing encryption, the file converts in unreadable format. So that even if someone succeeds in opening it, your secretes are safe.

8) Portability:

UNIX is written in high level language (i.e. in C) making it easier to read, understand, change and move to other machines.

9) Open System:

UNIX has an open architecture using which user can add tools to the program by writing program and storing the executable in a separate area in this file system.

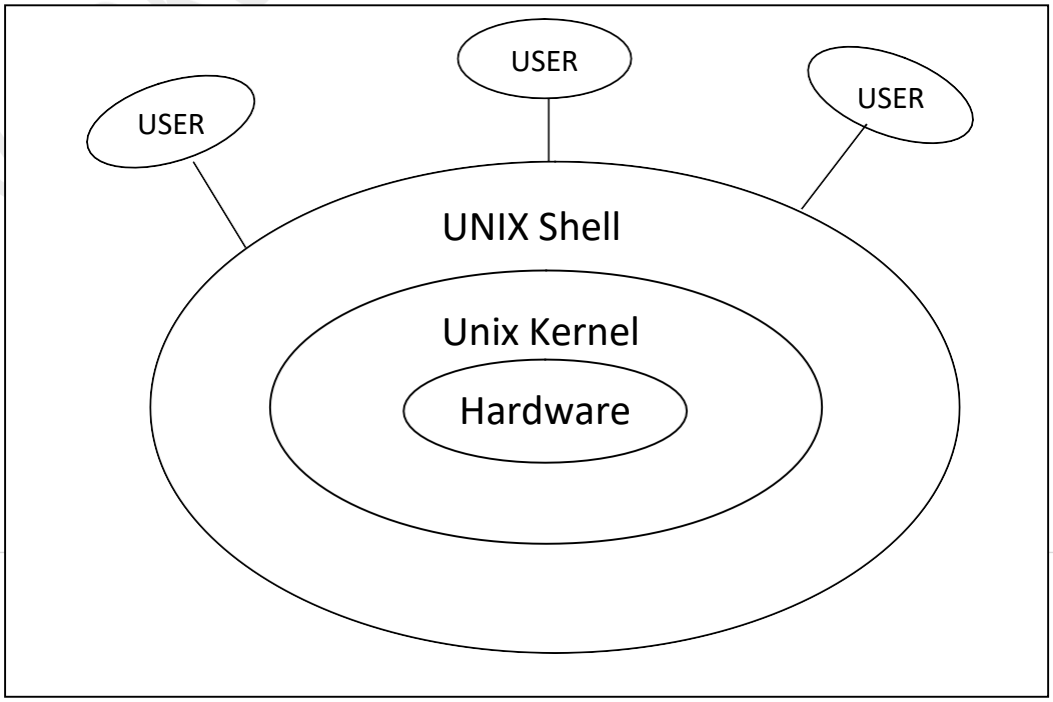
10) Programming Facility :

The Unix Shell is also a programming language. The Shell is able to be programmed. It is used to write Shell Script. Many of the system's functions can be controlled and automated by using these shell script.

11) Documentation:

UNIX documentation is very much important for Command reference and their configuration files. The online help facility available is the **man** command. Also, other resources are available on internet as ***newsgroups*** *on UNIX* where you can fire your queries related to Shell Programming or a Network Configuration.

UNIX Architecture or Organization



The functioning of UNIX is managed in three levels. On the outer level reside the application programs and other utilities, which speak our language.

At the heart of UNIX is the kernel, which interacts with the actual hardware in machine language.

Above figure shows the three layers of Unix OS. The Shell or Command Interpreter, which interprets the commands that we give and then conveys them to the kernel, which ultimately executes them.

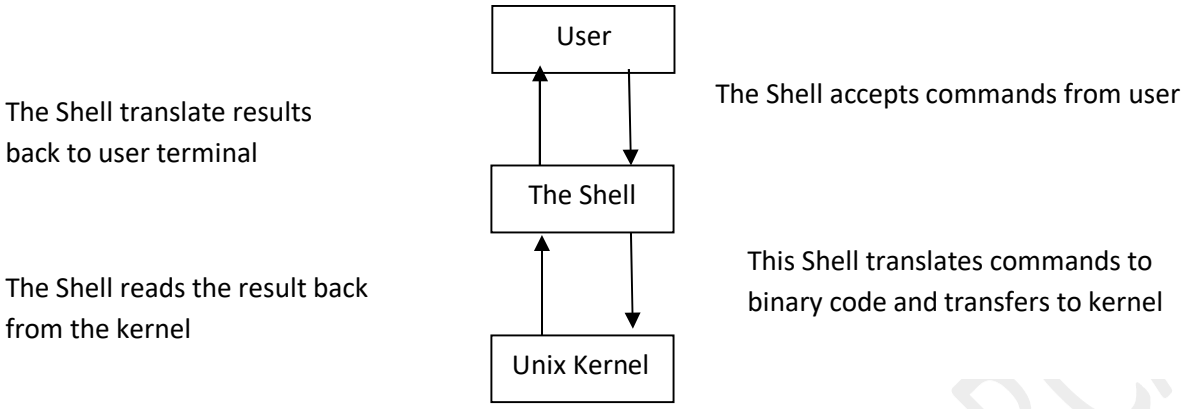
The Kernel is a collection of routines (Processes or functions) written is C. It is loaded into memory when system is booted & communicates directly with h/w. User Program that needs to access the h/w, uses the service of Kernel which perform the job on behalf of user. This Program access the kernel through a set of functions known as **System Call.**

The kernel manages files, carries out all the data transfer between the file system and the hardware, and also manages memory. The Scheduling of various programs running in memory or allocation of CPU time to all running programs also lies with kernel. It also handles any interrupts issued, as it is the entity that has direct dealing with the hardware.

The Kernel program is usually stored in a file called “Unix” whereas the shell program is in a file called “sh”. For each user working with UNIX at any time has different shell programs. Thus, at a particular point in time there may be several shells running in memory but only one kernel. This is because, at any instance Unix is capable of executing only one program as the other programs wait for their turn. And since it is the kernel which executes the program one kernel is sufficient. However, different users at different terminals are trying to seek kernel’s attention. And since the user interacts with the kernel through the shell, different shells are necessary.

What is Shell?

- Shell is nothing but Command Interpreter.
- Shell is most widely used Utility program on all Unix System.
- Shell is the key to interact with the Unix System.
- The Shell is itself a program that is written in ‘C’.
- It is loaded into memory when a user logs in and interacts with the user enables to communicate with the kernel.
- The Shell is also called as Command Interpreter because it interprets the commands from the user and passes the commands to the kernel.
- The Shell is responsible for returning the result of the commands to the terminal, a file, or another device such as a printer.
- Following figure shows



[Relationship of the Shell to UNIX]

A unique feature of Unix OS is that all UNIX command exists as Utility Program. This Program located as individual file in system directory such as /bin, /etc, /userbin. The Shell can be considered as a Master Utility Program which enable a user to gain access to all other utility & resources of the computer.

Types of Shell

The shell runs like any other program under the UNIX System. Due to this feature a no of shells have been developed in response to different needs of User. Some of the popular Shells are listed below.

1) Bourne Shell :

- This is one of the most widely use Shell in Unix World.
- It is developed by Steve Bourne at AT&T Bell Laboratories in late 1970.
- It is primary Unix command interpreter and comes along with every Unix System.
- It provides Dollar Prompt ($) on Unix installation is the trademark of the Bourne Shell.
- The executable file name is 'sh'.

2) C Shell:

- Mr. Bill Joy developed it at University of California. It has two advantages over Bourne Shell.
    1. It allows aliasing of commands. This is useful when lengthy commands are used and renamed. Instead of typing the entire command you can simply use the short alias at the command line.
    2. C Shell has a command history feature means previously typed command can be recalled. This is similar to the DOSKEY in MSDOS environment.
- It provides Percentage Prompt (%) on Unix installation is the trademark of C Shell.
- The executable file name is 'csh'.

3) Korn Shell:

- It is very powerful, and is a superset of Bourne Shell.
- The Korn Shell includes all the enhancements which are in the C Shell, like Command History & aliasing, and offers a few more features itself.
- It is developed by David Korn at AT&T Bell Laboratories.
- It provides Dollar Prompt ($).
- The executable file name is 'ksh'.

Features of Shell

1) Interactive Environment:

- The Shell allows user to create communication between User and Host UNIX System. This Communication terminates when user ends the Session.

2) Shell Script:

- The Shell contains internal command which can be utilized by the User for programming.
- The Shell Script is group of UNIX commands and executes as individual programming file.

3) I/O Redirection:

- **Output** Redirection is function of the Shell that redirects the Output of the program as an input to other file or command.
- Similarly, the Shell can make a program that accepts **input** from other than keyboard by redirecting its input from another Source like file.

4) Piping Mechanism:

- Piping Facility allows the output of one command to be used as input of another command.

  E.g. who | wc
  - Here, the output of 'who' command will the input of 'wc' command.

5) Meta character Facility (Filename Substitution) :

Shell recognizes *, . ,?, [..] as Special characters when reading the argument to a command line. Shell then perform filename expansion on this list before executing the requested command.

E.g. ls s*
- Here this command displays the list of all files begin with 's'.

6) Background Process:

- Multitasking facilities allows the user to run commands in background.
- This allows the commands to be processed while the user can proceed with other task.
- When a background task is completed, the user is notified.

7) Customize Environment:

- The Shell is working environment and gives facilities by which users can change according to their needs.

8) Shell Variable:

- The User can declare the variables to store the data which can be access anywhere in program.

What is kernel?

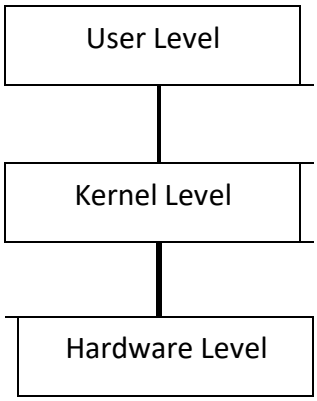
It is the heart of Operation System. The kernel assigns memory to each of the programs that are running, carries out all the data transfer between the file system and the hardware, partitions time fairly so that each program can get its job done, handles all I/O operations, all other low-level services, scheduling of various programs running in memory and so on.

Internal Command and External Command

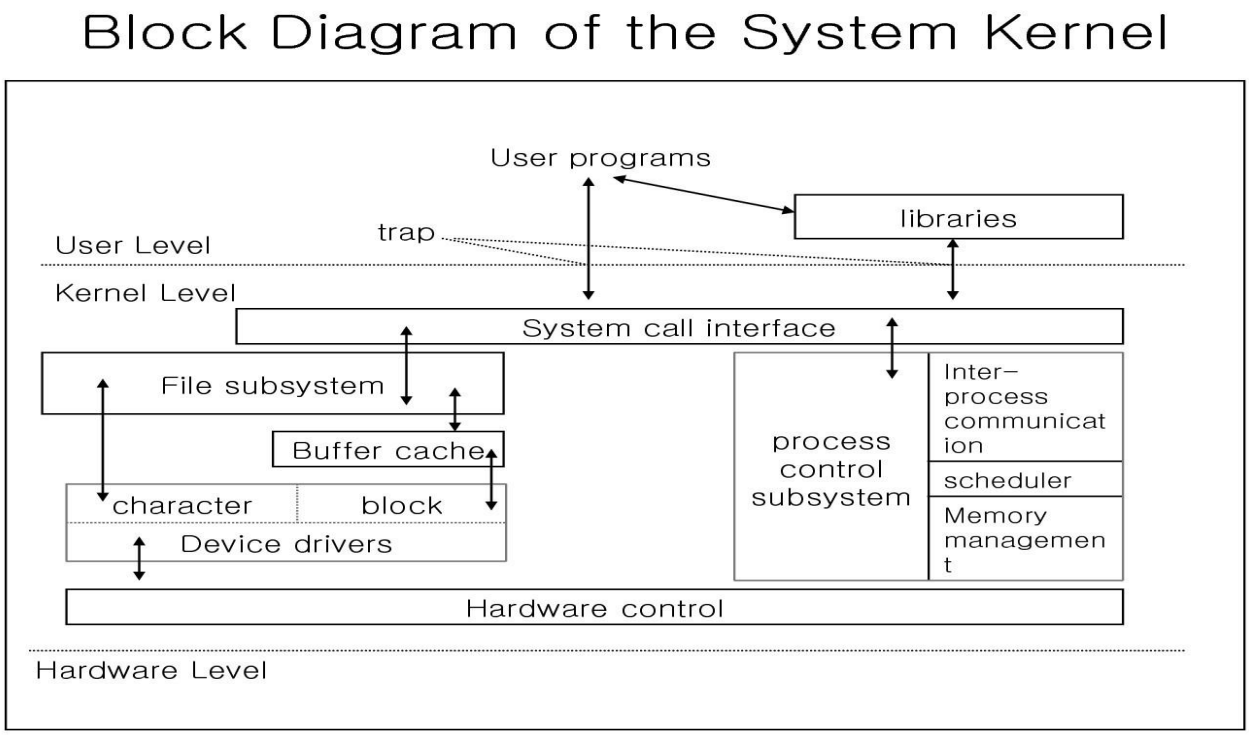
- The command built into the Shell is known as Internal Command. The Shell does not create a separate process to run internal command. They are directly run by Shell.
  E.g. cd, echo, pwd
- A command stored in /bin directory is known as External Command. Each command requires the Shell to create a new Process.
  E.g. ls, cat, date

Kernel Architecture

The Unix Operating System has three levels.



The above figure gives a top view of the Operating System. The User Level includes all the user programs, libraries and system call interface using which the users interact with the kernel. Following figure shows block diagram of kernel as follows:



The Kernel has the file subsystem and process control subsystem. Also the kernel has three components within the process control subsystem 1) *inter-process communication, 2) scheduler 3) memory management*.

The ***File Subsystem*** of the kernel, it uses *Device drivers* to actually read or write disk files. 1) It uses the *Buffer Cache* (high speed memory), which regulates data flow between the kernel and the secondary storage device. The Buffer Cache interacts with *Block Device* drivers to initiate the data transfer to and from the kernel, for block devices like disks and tapes, *Character Device* drivers are used to interact with character devices like terminals and any device which is not a Block Device, without buffer cache interaction. 2) *Device drivers* are the set of kernel modules that control the operations on peripheral devices by making them appear as random storage devices to the rest of the system.

The ***Process Control Subsystem*** and the File Subsystem interact with each other whenever a process needs an executable file to be loaded into memory for execution. The process control subsystem is responsible for inter-process communication, CPU Scheduling to processes, and memory management. 1) The ***Memory management*** module of the process control subsystem controls the allocation of memory to processes. Thus if the kernel finds that the main memory space is not sufficient for the processes, then the kernel moves the processes between the main memory and secondary memory. 2) The ***Scheduler*** module of the process control subsystem allocates the CPU to processes. This Scheduling of the CPU is done in turns, where each process is

given a time quantum. The moment the time quantum expires for any process, the next high priority process will be run. The previous process will again get CPU time as and when it becomes next high priority process. 3) ***Inter-process Communication*** is implemented using advanced features of piping between processes.

The ***Hardware control*** consists of modules for handling interrupts caused by devices like disk, tapes or terminals.