UNIT-1

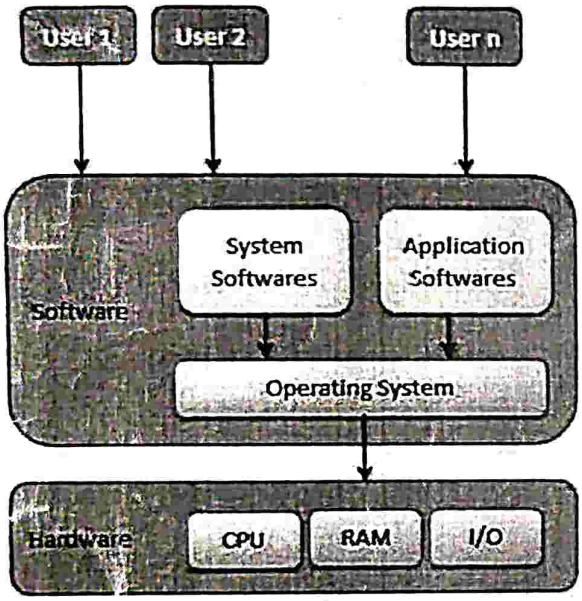
What is operating system?

An Operating System (OS) is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

Some popular Operating Systems include Linux Operating System, Windows Operating System, VMS, OS/400, AIX, z/OS, etc.

An operating system is a software programme required to manage and operate a computing device like smartphones, tablets, computers, supercomputers, web servers, cars, network towers, smartwatches, etc.

An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.



History and Evolution of Operating Systems

- The first operating system was created by General Motors in 1956 to run a single IBM central computer. In the 1960s, IBM was the first computer manufacturer to take on the task of developing operating systems and began distributing operating systems included in its computers.

- The first operating systems were developed in the 1950s, when computers could only run one program at a time. Later in the following decades, computers began to include more and more softwareprograms, sometimes called libraries, that came together to create the start of today's operating systems.

B. SUDARSHAN M.SC (CS), B. ED, D.EL.ED
gaddamsudarshan91@gmail.com

- In the late 1960s, the first version of the Unix operating system was developed. Written in programming language C, and available for free during its early years. Unix easily adapted to the new systems and quickly achieved wide acceptance.

- Many modern operating systems, including Apple OS X and all different versions of Linux, date back or rely on the Unix OS.

- Microsoft Windows was developed in response to an IBM request for an operating system to run its range of personal computers or PCs.

- The first operating system created by Microsoft was not called Windows, it was called MS-DOS and it was built in 1981 when it bought the 86-DOS operating system from Seattle Computer Products and modified it to meet IBM requirements.

- The Windows name was first used in 1985 when a graphical user interface was created and paired or joined with the MS-DOS.

- Today Apple, OS X, Microsoft Windows and the various forms of Linux (including Android) dominate the vast majority of the modern operating systems market, as we saw earlier.

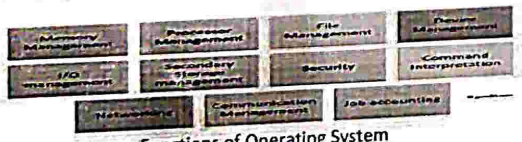
History Of OS

- Operating systems were first developed in the late 1950s to manage tape storage
- The General Motors Research Lab implemented the first OS in the early 1950s for their IBM 701
- In the mid-1960s, operating systems started to use disks
- In the late 1960s, the first version of the Unix OS was developed
- The first OS built by Microsoft was DOS. It was built in 1981 by purchasing the 86-DOS software from a Seattle company
- The present-day popular OS Windows first came to existence in 1985 when a GUI was created and paired with MS-DOS.

Functions of Operating System

Some typical operating system functions may include managing memory, files, processes, I/O system & devices, security, etc.

Below are the main functions of Operating System:



Functions of Operating System

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

In an operating system software performs each of the function:

1. **Process management:** Process management helps OS to create and delete processes. It also provides mechanisms for synchronization and communication among processes.
2. **Memory management:** Memory management module performs the task of allocation and de-allocation of memory space to programs in need of this resources.
3. **File management:** It manages all the file-related activities such as organization storage, retrieval, naming, sharing, and protection of files.
4. **Device Management:** Device management keeps tracks of all devices. This module also responsible for this task is known as the I/O controller. It also performs the task of allocation and de-allocation of the devices.
5. **I/O System Management:** One of the main objects of any OS is to hide the peculiarities of that hardware devices from the user.
6. **Secondary-Storage Management:** Systems have several levels of storage which includes primary storage, secondary storage, and cache storage. Instructions and data must be stored in primary storage or cache so that a running program can reference it.
7. **Security:** Security module protects the data and information of a computer system against malware threat and authorized access.
8. **Command interpretation:** This module is interpreting commands given by the and acting system resources to process that commands.
9. **Networking:** A distributed system is a group of processors which do not share memory, hardware devices, or a clock. The processors communicate with one another through the network.
10. **Job accounting:** Keeping track of time & resource used by various job and users.
11. **Communication management:** Coordination and assignment of compilers, interpreters, and another software resource of the various users of the computer systems.

Resource abstraction

- Resource abstraction is the process of "hiding the details of how the hardware operates, thereby making computer hardware relatively easy for an application programmer to use".
- Such an abstraction saves the programmer from needing to learn the details of both hardware interfaces.

. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
ddamsudarshan91@gmail.com

- One way in which the operating system might implement resource abstraction is to provide a single abstract disk interface which will be the same for both the hard disk and floppy disk.
- Such an abstraction saves the programmer from needing to learn the details of both hardware interfaces. Instead, the programmer only needs to learn the disk abstraction provided by the operating system.
- Typical resources include the central processing unit (CPU), computer memory, file storage, input/output (I/O) devices, and network connections.
- Abstraction means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation. Consider a real-life example of a man driving a car.

Types of Operating System (OS)

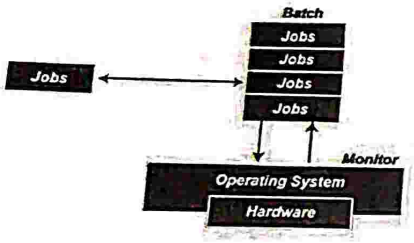
Following are the popular types of OS (Operating System):

- Batch Operating System
- Multitasking/Time Sharing OS
- Multiprocessing OS
- Real Time OS
- Distributed OS
- Network OS
- Mobile OS

Batch Operating System

Some computer processes are very lengthy and time-consuming. To speed the same process, a job with a similar type of needs are batched together and run as a group.

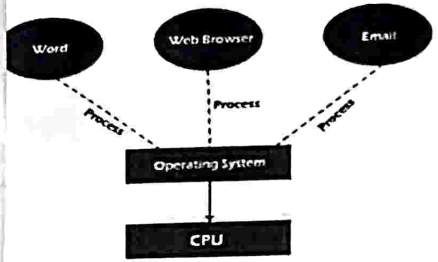
The user of a batch operating system never directly interacts with the computer. In this type of OS, every user prepares his or her job on an offline device like a punch card and submit it to the computer operator.



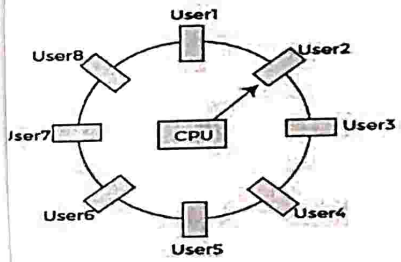
B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
geddamsudarshan91@gmail.com

ulti-Tasking/Time-sharing Operating systems

ne-sharing operating system enables people located at a different terminal(shell) to use a single computer stem at the same time. The processor time (CPU) which is shared among multiple users is termed as time aring.



the Time Sharing operating system, computer resources are allocated in a time-dependent fashion to veral programs simultaneously. Thus it helps to provide a large number of user's direct access to the main mputer. It is a logical extension of multiprogramming. In time-sharing, the CPU is switched among multiple rograms given by different users on a scheduled basis.



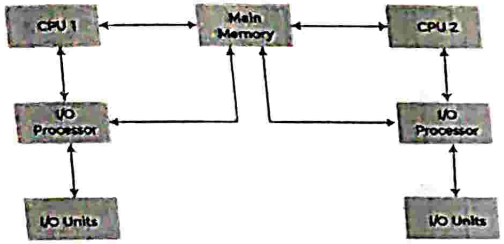
Timesharing in case of 8 users

time-sharing operating system allows many users to be served simultaneously, so sophisticated CPU heduling schemes and Input/output management are required.

ultiprocessing Operating System

Multiprocessing, Parallel computing is achieved. There are more than one processors present in the system ich can execute more than one process at the same time. This will increase the throughput of the system.

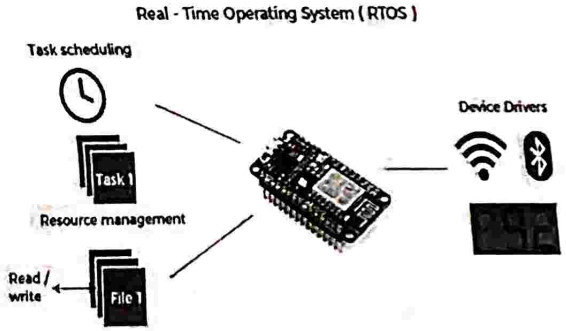
UDARSHAN M.SC (CS), B. ED, D.EL. ED
lamsudarshan91@gmail.com



Working of Multiprocessor System

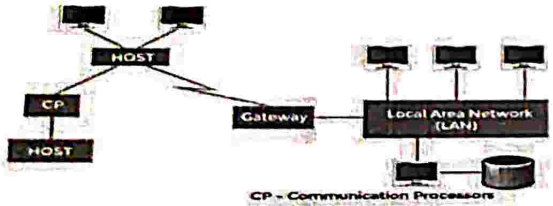
Real time OS

A real time operating system time interval to process and respond to inputs is very small. Examples: Military Software Systems, Space Software Systems are the Real time OS example.



Distributed Operating System

Distributed systems use many processors located in different machines to provide very fast computation to its users.

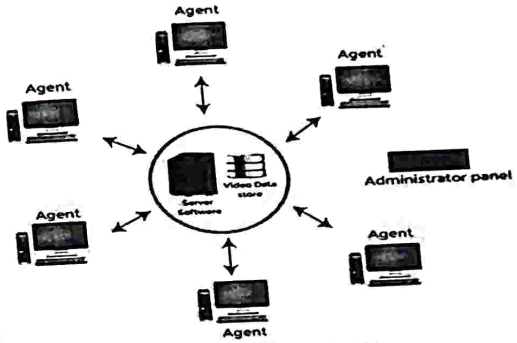


A Typical View of a Distributed System

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Network Operating System

Network Operating System runs on a server. It provides the capability to serve to manage data, user, groups, security, application, and other networking functions.



Network Operating Systems

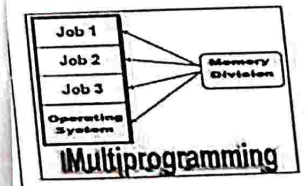
Mobile OS

Mobile operating systems are those OS which is especially that are designed to power smartphones, tablets, and wearables devices.

Some most famous mobile operating systems are Android and iOS, but others include BlackBerry, Web, and watchOS.

Multiprogramming Operating System

A multiprogramming operating system may run many programs on a single processor computer. If one program must wait for an input/output transfer in a multiprogramming operating system, the other programs are ready to use the CPU. As a result, various jobs may share CPU time. However, the execution of their jobs is not defined to be at the same time period.

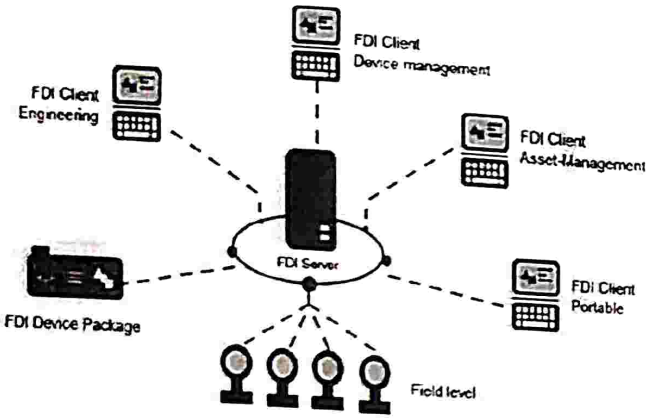


B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

There are mainly two types of multiprogramming operating systems. These are as follows:

1. **Multitasking Operating System**
2. **Multiuser Operating System**

It is an operating system that permits several users to utilize the programs that are concurrently running on a single network server. The single network server is termed as "Terminal server". "Terminal client" is a software that supports user sessions. Examples include UNIX, MVS, etc.



1.Multitasking Operating System

Multitasking, in an operating system, is allowing a user to perform more than one computer task (such as the operation of an application program) at a time. The operating system is able to keep track of where you are in these tasks and go from one to the other without losing information.

A multitasking operating system enables the execution of two or more programs at the same time. The operating system accomplishes this by shifting each program into and out of memory one at a time. When a program is switched out of memory, it is temporarily saved on disk until it is required again.

2.Multiuser Operating System

A Multi-user operating system is a computer operating system which allows multiple users to access the single system with one operating system on it.

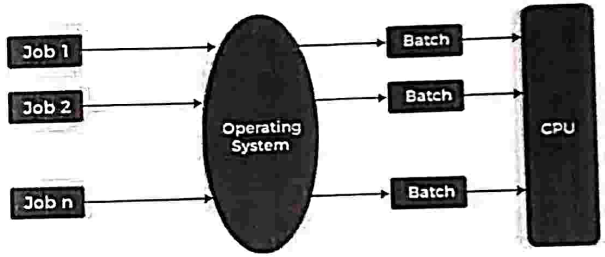
B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

A multiuser operating system allows many users to share processing time on a powerful central computer from different terminals. The operating system accomplishes this by rapidly switching between terminals, each of which receives a limited amount of processor time on the central computer.

The operating system changes among terminals so quickly that each user seems to have continuous access to the central computer. If there are many users on a system like this, the time it takes the central computer to reply can become more obvious.

Batch Operating System

Batch processing was very popular in the 1970s. The jobs were executed in batches. People used to have a single computer known as a mainframe. Users using batch operating systems do not interact directly with the computer. Each user prepares their job using an offline device like a punch card and submitting it to the computer operator



The batch operating system grouped jobs that perform similar functions. These job groups are treated as a batch and executed simultaneously. A computer system with this operating system performs the following batch processing activities:

1. A job is a single unit that consists of a preset sequence of commands, data, and programs.
2. Processing takes place in the order in which they are received, i.e., first come, first serve.
3. These jobs are stored in memory and executed without the need for manual information.
4. When a job is successfully run, the operating system releases its memory.

Types of Batch Operating System

There are mainly two types of the batch operating system. These are as follows:

1. Simple Batched System
2. Multi-programmed batched system

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Simple Batched System

- The user did not directly interact with the computer system for job execution in a simple batch operating system.
- However, the user was required to prepare a job that included the program, control information, and data on the nature of the job on control cards.
- The job was then submitted to the computer operator, who was usually in the form of a punch card.
- The program's output included results and registers and memory dumps in the event of a program error.
- The output appeared after some time that could take days, hours, and minutes.

Multi-programmed batched system

- In the multi-programmed batched system, jobs are grouped so that the CPU only executes one job at a time to improve CPU utilization.
- The operating system maintains various jobs in memory at a time. The operating system selects one job and begins executing it in memory.
- Finally, the job must wait for a task to complete, such as mounting a tape on an I/O operation.
- In a multiprogramming system, do not sit idle because the operating system switches to another task. When a job is in the wait state, and the current job is completed, the CPU is returned.

Time shared operating system

A time-shared operating system allows multiple users to share computers simultaneously. Each action or order at a time the shared system becomes smaller, so only a little CPU time is required for each user.

As the system rapidly switches from one user to another, each user is given the impression that the entire computer system is dedicated to its use, although it is being shared among multiple users.

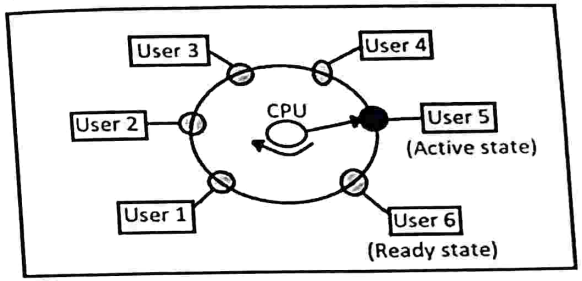
A time shared operating system uses CPU scheduling and multi-programming to provide each with a small portion of a shared computer at once. Each user has at least one separate program in memory.

A program loaded into memory and executes, it performs a short period of time either before completion or to complete I/O.This short period of time during which user gets attention of CPU is known as time slice, time slot or quantum. It is typically of the order of 10 to 100 milliseconds.

Time shared operating systems are more complex than multiprogrammed operating systems. In both, multiple jobs must be kept in memory simultaneously, so the system must have memory management and security.

To achieve a good response time, jobs may have to swap in and out of disk from main memory which now serves as a backing store for main memory. A common method to achieve this goal is virtual memory, a technique that allows the execution of a job that may not be completely in memory.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



In above figure the user 5 is active state but user 1, user 2, user 3, and user 4 are in **waiting state** whereas user 6 is in ready state.

1. **Active State –**
   The user's program is under the control of CPU. Only one program is available in this state.
2. **Ready State –**
   The user program is ready to execute but it is waiting for forit's turn to get the CPU. More than one user can be in ready state at a time.
3. **Waiting State –**
   The user's program is waiting for some input/output operation. More than one user can be in a waiting state at a time.

Operating Systems for personal computers

- Personal computer operating system provides a good interface to a single user.
- Personal computer operating systems are widely used for word processing, spreadsheets and Internet access.
- Personal computer operating system is made only for personal.
- You can say that your laptops, computer systems, tablets etc. are your personal computers and the operating system such as windows 7, windows 10, android, etc. are your personal computer operating system.
- And you can use your personal computer operating system for your personal purposes.
- for example, to chatting with your friends using some social media sites, reading some articles from internet, making some projects through Microsoft PowerPoint or any other, designing your website, programming something, watching some videos and movies, listening to some songs and many more.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Workstations and hand-held devices

A workstation is a special computer designed for technical or scientific applications.

Intended primarily to be used by a single user, they are commonly connected to a local area network and run user operating.

The term *workstation* has also been used loosely to refer to everything from a mainframe computer terminal to a PC connected to a network, but the most common form refers to the class of hardware offered by several current and defunct companies such as Microsystems, Silicon, Computer, DEC, HP, NeXT and IBM which opened the door for the 3D graphics animation revolution of the late 1990s.[1]

A handheld computer is a computer that can conveniently be stored in a pocket (of sufficient size) and used while you're holding it. Today's handheld computers, which are also called personal digital assistants (PDAs), can be divided into those that accept handwriting as input and those with small keyboards.

The original handheld that accepted handwriting was Apple's Newton, which was later withdrawn from the market. Today, the most popular handheld that accepts handwritten input is the PalmPilot from 3Com. Philips, Casio, NEC, Compaq, and other companies make handhelds with small keyboards.

Windows CE and EPOC are two of the most widely used operating systems in handheld computers.

Handheld computers are typically used for personal information manager (PIM) types of applications: maintaining schedules, keeping names and phone numbers, doing simple calculations, taking notes, and, with a modem, exchanging e-mail and getting information from the Web.

Keyboards have tiny keys that take getting used to. Those that handle handwriting also impose constraints and require some learning. Nevertheless, this class of computer is widely sold and appreciated by many users.

Process control and real time system

Process control:

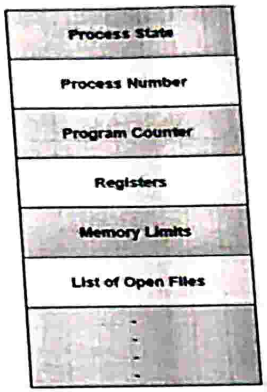
Process Control Block is a data structure that contains information of the process related to it. The process control block is also known as a task control block, entry of the process table, etc.

It is very important for process management as the data structuring for processes is done in terms of the PCB. It also defines the current state of the operating system.

Structure of the Process Control Block

The process control stores many data items that are needed for efficient process management. Some of these data items are explained with the help of the given diagram –

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



Process Control Block (PCB)

The following are the data items –

Process State:

This specifies the process state i.e. new, ready, running, waiting or terminated.

Process Number:

This shows the number of the particular process.

Program Counter:

This contains the address of the next instruction that needs to be executed in the process.

Registers:

This specifies the registers that are used by the process. They may include accumulators, index registers, stack pointers, general purpose registers etc.

List of Open Files:

These are the different files that are associated with the process

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

CPU Scheduling Information:

The process priority, pointers to scheduling queues etc. is the CPU scheduling information that is contained in the PCB. This may also include any other scheduling parameters.

Memory Management Information:

The memory management information includes the page tables or the segment tables depending on the memory system used. It also contains the value of the base registers, limit registers etc.

I/O Status Information:

This information includes the list of I/O devices used by the process, the list of files etc.

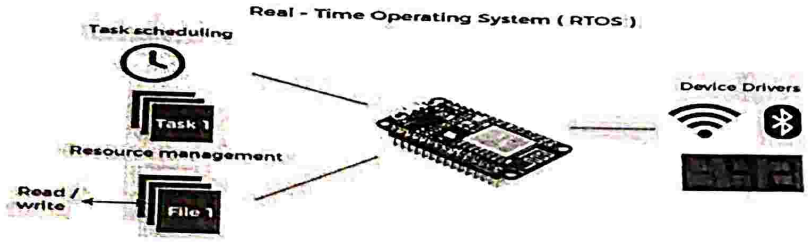
location.

Real-Time operating system

A real-time operating system (RTOS) is a special-purpose operating system used in computers that has strict time constraints for any job to be performed.

The benefits of real-time operating system are as follows-:

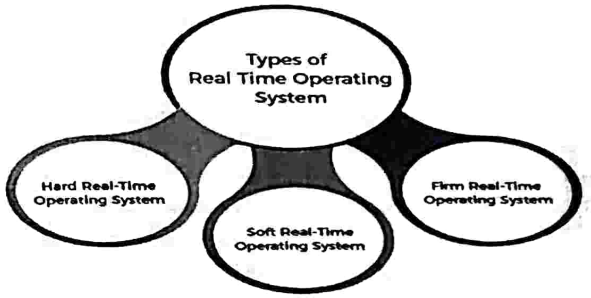
- Easy to layout, develop and execute real-time applications under the real-time operating system.
- The real-time working structures are extra compact, so those structures require much less memory space.
- In a Real-time operating system, the maximum utilization of devices and systems.
- Focus on running applications and less importance to applications that are in the queue.
- Since the size of programs is small, RTOS can also be embedded systems like in transport and others.
- These types of systems are error-free.
- Memory allocation is best managed in these types of systems.



B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Types of Real-time operating system

Following are the three types of RTOS systems are:



Hard Real-Time operating system:

In Hard RTOS, all critical tasks must be completed within the specified time duration, i.e., within the given deadline. Not meeting the deadline would result in critical failures such as damage to equipment or even loss of human life.

Soft Real-Time operating system:

Soft RTOS accepts a few delays via the means of the Operating system. In this kind of RTOS, there may be a closing date assigned for a particular job, but a delay for a small amount of time is acceptable. So, cut off dates are treated softly via means of this kind of RTOS.

Firm Real-Time operating system:

In Firm RTOS additionally want to observe the deadlines. However, lacking a closing date might not have a massive effect, however may want to purposely undesired effects, like a massive discount within the fine of a product.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

UNIT-2

Processor and User mode, kernel mode

Processor:

A processor is an integrated electronic circuit that performs the calculations that run a computer. A processor performs arithmetical, logical, input/output (I/O) and other basic instructions that are passed from an operating system (OS). Most other processes are dependent on the operations of a processor.

The terms processor, central processing unit (CPU) and microprocessor are commonly linked as synonyms. Most people use the word "processor" interchangeably with the term "CPU" nowadays, it is technically not correct since the CPU is just one of the processors inside a personal computer (PC).

The Graphics Processing Unit (GPU) is another processor, and even some hard drives are technically capable of performing some processing.

A processor (CPU) is the **logic circuitry that responds to and processes the basic instructions that drive a computer.** *The CPU is seen as the main and most crucial integrated circuitry (IC) chip in a computer, as it is responsible for interpreting most of computer's commands.*

A processor in a computer running Windows has two different modes:

1. *user mode*
2. *kernel mode.*

User Mode:

When a Program is booted up on an Operating system let's say windows, then it launches the program in user mode. And when a user-mode program requests to run, a process and virtual address space (address space for that process) is created for it by windows.

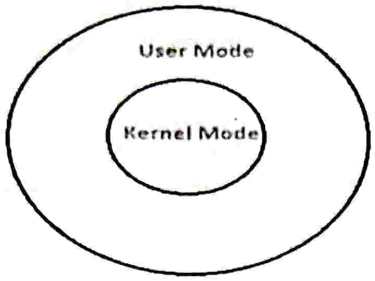
User-mode programs are less privileged that is user-mode applications are not allowed to access the system resources directly. For instance, if an application under user-mode wants to access system resources, it will have to first go through the Operating system kernel by using system calls.

Kernel Mode:

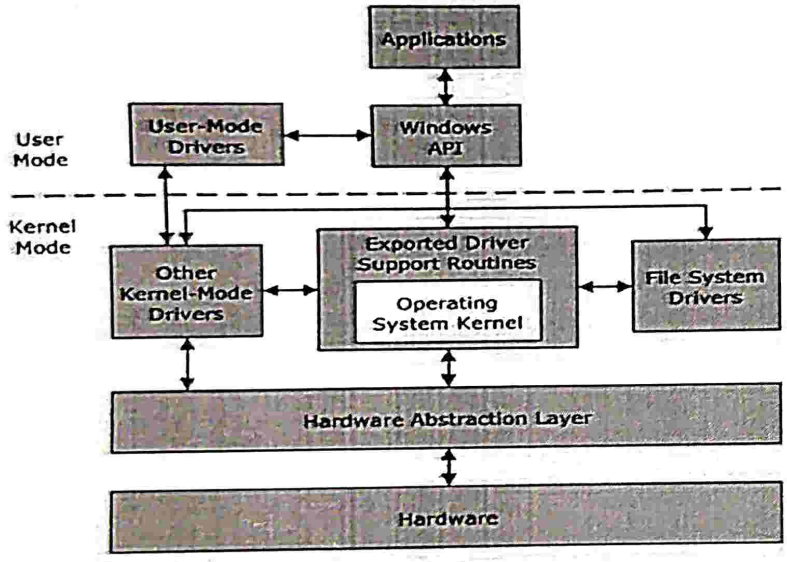
The kernel is the core program on which all the other operating system components rely, it is used to access the hardware components and schedule which processes should run on a computer system and when, and it also manages the application software and hardware interaction. Hence it is the most privileged program, unlike other programs it can directly interact with the hardware.

When programs running under user mode need hardware access for example like webcam, then first it has to go through the kernel by using a system call, and to carry out these requests the CPU switches from user mode to kernel mode at the time of execution. After finally completing the execution of the process the CPU again switches back to the user mode.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



This diagram illustrates communication between user-mode and kernel-mode components.



Difference Between Kernel mode and User mode

Criteria	Kernel Mode	User Mode
Kernel-mode vs User mode	In kernel mode, the program has direct and unrestricted access to system resources.	In user mode, the application program executes and starts out.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
Baddamsudarshan91@gmail.com

Criteria	Kernel Mode	User Mode
Interruptions	In Kernel mode, the whole operating system might go down if an interrupt occurs	In user mode, a single process fails if an interrupt occurs.
Modes	Kernel mode is also known as the master mode, privileged mode, or system mode.	User mode is also known as the unprivileged mode, restricted mode, or slave mode.
Virtual address space	In kernel mode, all processes share a single virtual address space.	In user mode, all processes get separate virtual address space.
Level of privilege	In kernel mode, the applications have more privileges as compared to user mode.	While in user mode the applications have fewer privileges.
Restrictions	As kernel mode can access both the user programs as well as the kernel programs there are no restrictions.	While user mode needs to access kernel programs as it cannot directly access them.
Mode bit value	The mode bit of kernel-mode is 0.	While; the mode bit of user-mode is 1.

System calls and system programs

A system call is a way for programs to interact with the operating system. A computer program makes a system call when it makes a request to the operating system's kernel. System call provides the services of the operating system to the user programs via Application Program Interface (API).

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.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Types of System Calls: There are 5 different categories of system calls –

1. **Process control:** end, abort, create, terminate, allocate and free memory.
2. **File management:** create, open, close, delete, read file etc.
3. Device management
4. Information maintenance
5. Communication

Examples of Windows and Unix System Calls –

	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()

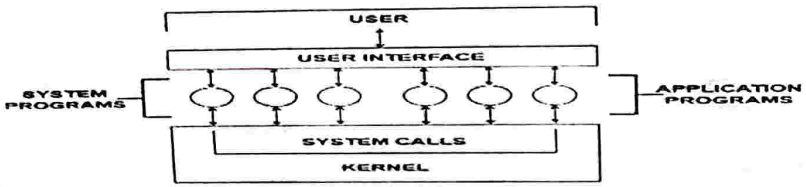
System Programming:

System Programming can be defined as the act of building Systems Software using System Programming Languages. According to Computer Hierarchy, one which comes at last is Hardware. Then it is Operating System, System Programs, and finally Application Programs

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Some examples of system program in O.S. are –

- Windows 10
- Mac OS X
- Ubuntu
- Linux
- Unix
- Android
- Anti-virus
- Disk formatting
- Computer language translators



System Programs can be divided into these categories:

1. **File Management –**
   A file is a collection of specific information stored in the memory of a computer system. File management is defined as the process of manipulating files in the computer system, its management includes the process of creating, modifying and deleting files.
   - It helps to create new files in the computer system and placing them at specific locations.
   - It helps in easily and quickly locating these files in the computer system.
   - It makes the process of sharing files among different users very easy and user-friendly.
   - It helps to store files in separate folders known as directories.

2. **Status Information –**
   Information like date, time amount of available memory, or disk space is asked by some users. Others providing detailed performance, logging, and debugging information which is more complex. All this information is formatted and displayed on output devices or printed. Terminal or other output devices or files or a window of GUI is used for showing the output of programs.

3. **File Modification –**
   For modifying the contents of files we use this. For Files stored on disks or other storage devices, we used different types of editors. For searching contents of files or perform transformations of files we use special commands.

4. **Programming-Language support –**
   For common programming languages, we use Compilers, Assemblers, Debuggers, and interpreters which are already provided to users. It provides all support to users. We can run any programming language. All languages of importance are already provided.

5. **Program Loading and Execution –**
   When the program is ready after Assembling and compilation, it must be loaded into memory for execution. A loader is part of an operating system that is responsible for loading programs and libraries. It is one of the essential stages for starting a program. Loaders, relocatable loaders, linkage editors, and Overlay loaders are provided by the system.

6. **Communications –**
   Virtual connections among processes, users, and computer systems are provided by programs. Users can send messages to another user on their screen, User can send e-mail, browsing on web pages, remote login, the transformation of files from one user to another.

System view of the process and resources

System View: The OS manages the hardware resources of a computer system. Resources include processors, memory, disks and other storage devices, network interfaces, I/O devices such as keyboards, mice and monitors, and so on. controls the sharing of resources among programs.

An operating system is a framework that enables user application programs to interact with system hardware. The operating system does not perform any functions on its own, but it provides an atmosphere in which various apps and programs can do useful work. The operating system may be observed from the point of view of the user or the system, and it is known as the user view and the system view. In this article, you will learn the views of the operating system.

Three views of an operating system

1.Application View: what services does it provide?

2.System View: what problems does it solve?

3. Implementation View: how is it built?

1.Application View of an Operating System

The OS provides an execution environment for running programs.

– The execution environment provides a program with the processor time and memory space that it needs to run.

execution environment provides interfaces through which a program can use networks, storage, I/O s, and other system hardware components. Interfaces provide a simplified, abstract view of hardware plication programs.

tem View:

OS manages the hardware resources of a computer system. Resources include processors, memory, disks l other storage devices, network interfaces, I/O devices such as keyboards, mice and monitors, and so on.

e operating system allocates resources among running programs. It controls the sharing of resources among rograms. The OS itself also uses resources, which it must share with application programs.

Implementation View:

The OS is a concurrent, real-time program. Concurrency arises naturally in an OS when it supports concurrent applications, and because it must interact directly with the hardware.

Process abstraction in operating system

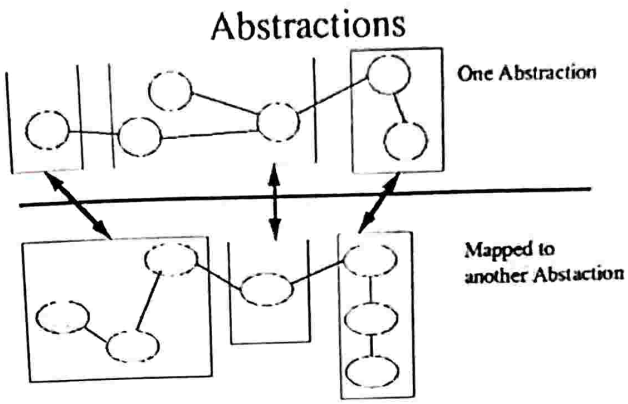
- ✓ An abstraction is software that hides lower-level details and provides a set of higher-level functions.
- ✓ An operating system transforms the physical world of devices, instructions, memory, and time into virtual world that is the result of abstractions built by the operating system.
- ✓ The fundamental operation of the operating system (OS) is to abstract the hardware to the programmer and user.
- ✓ The operating system provides generic interfaces to services provided by the underlying hardware.

There are three types of abstract:

- Descriptive
- informative
- critical

- ✓ Abstraction can be accomplished on functions, data, and processes.
- ✓ In functional abstraction, details of the algorithms to accomplish the function are not visible to the consumer of the function.
- ✓ The consumer of the function needs to only know the correct calling convention and have trust in the accuracy of the functional results.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



An example of process abstraction is the concurrency scheduler in a database system. A database system can handle many concurrent queries

Process hierarchy

Now-a-days all general-purpose operating systems permit a user to create and destroy processes. A process can create several new processes during its time of execution.

The creating process is called the Parent Process and the new process is called Child Process.

There are different ways for creating a new process. These are as follows –

- **Execution** – The child process is executed by the parent process concurrently or it waits till all children get terminated.
- **Sharing** – The parent or child process shares all resources like memory or files or children process shares a subset of parent's resources or parent and children process share no resource in common.

The reasons that parent process terminates the execution of one of its children are as follows –

- The child process has exceeded its usage of resources that have been allocated. Because of this there should be some mechanism which allows the parent process to inspect the state of its children process.
- The task that is assigned to the child process is no longer required.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

What is a Thread?

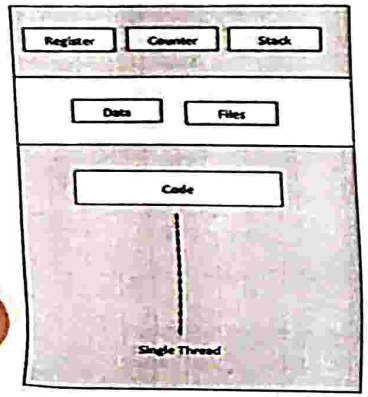
A thread is a path of execution within a process. A process can contain multiple threads.

A thread is a flow of execution through the process code, with its own program counter that keeps track of which instruction to execute next, system registers which hold its current working variables, and a stack which contains the execution history.

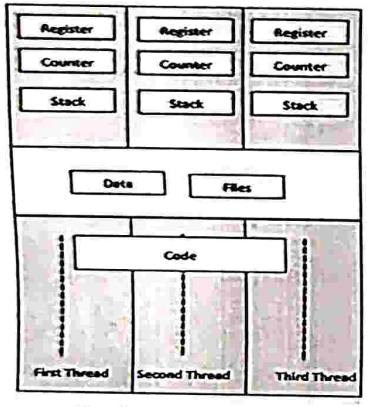
A thread shares with its peer threads few information like code segment, data segment and open files. When one thread alters a code segment memory item, all other threads see that.

A thread is also called a **lightweight process**. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.

Each thread belongs to exactly one process and no thread can exist outside a process. Each thread represents a separate flow of control. Threads have been successfully used in implementing network servers and web server. They also provide a suitable foundation for parallel execution of applications on shared memory multiprocessors. The following figure shows the working of a single-threaded and a multithreaded process.



Single Process P with single thread



Single Process P with three threads

Types of Thread:

Threads are implemented in following two ways –

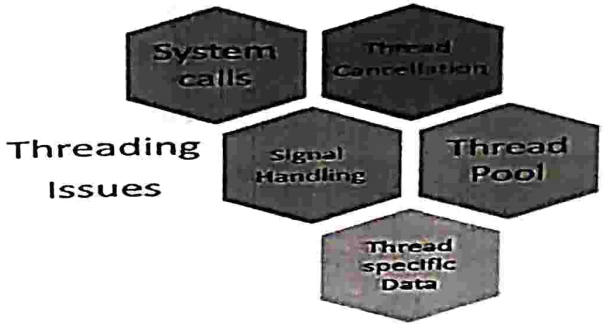
- **User Level Threads** – User managed threads.
- **Kernel Level Threads** – Operating System managed threads acting on kernel, an operating system core.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Threading Issues in OS

There are several threading issues when we are in a multithreading environment. In this section, we will discuss the threading issues with system calls, cancellation of thread, signal handling, thread pool and thread-specific data.

Along with the threading issues, we will also discuss how these issues can be deal or resolve to retain the benefit of the multithreaded programming environment.



Threading Issues are:

1.System Calls
2.Thread Cancellation
3.Signal Handling
4.Thread Pool
5.Thread Specific Data

1. The fork () and exec () System Calls:

The fork () and exec () are the system calls. The fork () call creates a duplicate process of the process that invokes fork (). The new duplicate process is called child process and process invoking the fork () is called the parent process. Both the parent process and the child process continue their execution from the instruction that is just after the fork ().

2. Thread cancellation:

Termination of the thread in the middle of its execution it is termed as 'thread cancellation'. Let us understand this with the help of an example. Consider that there is a multithreaded program which has let its multiple

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

threads to search through a database for some information. However, if one of the thread returns with the desired result the remaining threads will be cancelled.

Now a thread which we want to cancel is termed as target thread. Thread cancellation can be performed in two ways:

Asynchronous Cancellation: In asynchronous cancellation, a thread is employed to terminate the target thread instantly.

Deferred Cancellation: In deferred cancellation, the target thread is scheduled to check itself at regular interval whether it can terminate itself or not.

3. Signal Handling:

Signal handling is more convenient in the single-threaded program as the signal would be directly forwarded to the process. But when it comes to multithreaded program, the issue arrives to which thread of the program the signal should be delivered.

Let's say the signal would be delivered to:

- All the threads of the process.
- To some specific threads in a process.
- To the thread to which it applies
- Or you can assign a thread to receive all the signals.

4. Thread Pool:

When a user requests for a webpage to the server, the server creates a separate thread to service the request. Although the server also has some potential issues. Consider if we do not have a bound on the number of actives thread in a system and would create a new thread for every new request then it would finally result in

5. Thread Specific data:

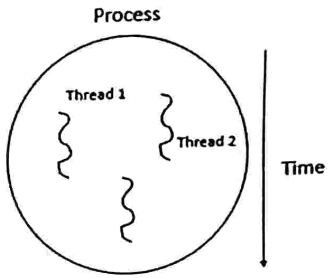
the specific data associated with the specific thread is referred to as thread-specific data.

Consider a transaction processing system, here we can process each transaction in a different thread. To determine each transaction uniquely we will associate a unique identifier with it. Which will help the system to identify each transaction uniquely.

What are thread libraries?

A thread is a lightweight of process and is a basic unit of CPU utilization which consists of a program counter, a stack, and a set of registers.

Given below is the structure of thread in a process –



A process has a single thread of control where one program can counter and one sequence of instructions is carried out at any given time. Dividing an application or a program into multiple sequential threads that run in quasi-parallel, the programming model becomes simpler.

Thread has the ability to share an address space and all of its data among themselves. This ability is essential for some specific applications.

Ways of implementing thread library:

There are two primary ways of implementing thread library, which are as follows –

- The first approach is to provide a library entirely in user space with kernel support. All code and data structures for the library exist in a local function call in user space and not in a system call.
- The second approach is to implement a kernel level library supported directly by the operating system. In this case the code and data structures for the library exist in kernel space.

Invoking a function in the application program interface for the library typically results in a system call to the kernel.

The main thread libraries which are used are given below –

- **POSIX threads** – Pthreads, the threads extension of the POSIX standard, may be provided as either a user level or a kernel level library.
- **WIN 32 thread** – The windows thread library is a kernel level library available on windows systems.
- **JAVA thread** – The JAVA thread API allows threads to be created and managed directly as JAVA programs.

Process Scheduling in Operating System

The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

Process scheduling is an essential part of a Multiprogramming operating systems. Such operating systems allow more than one process to be loaded into the executable memory at a time and the loaded process shares the CPU using time multiplexing.

There are three types of process scheduler.

1. **Long Term or job scheduler:**
   It brings the new process to the 'Ready State'. It controls ***Degree of Multi-programming***, i.e., number of process present in ready state at any point of time. It is important that the long-term scheduler make a careful selection of both IO and CPU bound process. IO bound tasks are which use much of their time in input and output operations while CPU bound processes are which spend their time on CPU. The job scheduler increases efficiency by maintaining a balance between the two.

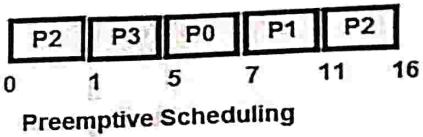
2. **Short term or CPU scheduler:**
   It is responsible for selecting one process from ready state for scheduling it on the running state. Note: Short-term scheduler only selects the process to schedule it doesn't load the process on running. Here is when all the scheduling algorithms are used. The CPU scheduler is responsible for ensuring there is no starvation owing to high burst time processes.
   Dispatcher is responsible for loading the process selected by Short-term scheduler on the CPU (Ready to Running State) Context switching is done by dispatcher only. A dispatcher does the following:
   1. *Switching context.*
   2. *Switching to user mode.*
   3. *Jumping to the proper location in the newly loaded program.*
3. ***Medium-term scheduler:***
   It is responsible for suspending and resuming the process. It mainly does swapping (moving processes from main memory to disk and vice versa). Swapping may be necessary to improve the process mix or because a change in memory requirements has overcommitted available memory, requiring memory to be freed up. It is helpful in maintaining a perfect balance between the I/O bound and the CPU bound. It reduces the degree of multiprogramming.

Preemptive and Non-Preemptive Scheduling

1. Preemptive Scheduling:
Preemptive scheduling is used when a process switches from running state to ready state or from the waiting state to ready state. The resources (mainly CPU cycles) are allocated to the process for a limited amount of time and then taken away, and the process is again placed back in the ready queue if that process still has CPU burst time remaining. That process stays in the ready queue till it gets its next chance to execute.

Algorithms based on preemptive scheduling are: Round Robin (RR), Shortest Remaining Time First (SRTF), Priority (preemptive version), etc.

Process	Arrival Time	CPU Burst Time (in millisec.)
P0	3	2
P1	2	4
P2	0	6
P3	1	4

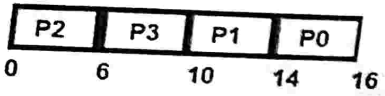


Preemptive Scheduling

2. Non-Preemptive Scheduling:
Non-preemptive Scheduling is used when a process terminates, or a process switches from running to the waiting state. In this scheduling, once the resources (CPU cycles) are allocated to a process, the process holds the CPU till it gets terminated or reaches a waiting state. In the case of non-preemptive scheduling does not interrupt a process running CPU in the middle of the execution. Instead, it waits till the process completes its CPU burst time, and then it can allocate the CPU to another process.

Algorithms based on non-preemptive scheduling are: Shortest Job First (SJF basically non preemptive) and Priority (non preemptive version), etc.

Process	Arrival Time	CPU Burst Time (in millisec.)
P0	3	2
P1	2	4
P2	0	6
P3	1	4



0 6 10 14 16

Non-Preemtive Scheduling

DIFFERENCE BETWEEN PREEMPTIVE SCHEDULING NON-PREEMPTIVE SCHEDULING:

PARAMETER	PREEMPTIVE SCHEDULING	NON-PREEMPTIVE SCHEDULING
Basic	In these resources (CPU Cycle) are allocated to a process for a limited time.	Once resources (CPU Cycle) are allocated to a process, the process holds it till it completes its burst time or switches to waiting state.
Interrupt	Process can be interrupted in between.	Process cannot be interrupted until it terminates itself or its time is up.
Starvation	If a process having high priority frequently arrives in the ready queue, a low priority process may starve.	If a process with a long burst time is running CPU, then later coming process with less CPU burst time may starve.
Overhead	It has overheads of scheduling the processes.	It does not have overheads.
exibility	Flexible	rigid
Cost	cost associated	no cost associated
CPU Utilization	In preemptive scheduling, CPU utilization is high.	It is low in non preemptive scheduling.
xamples	Examples of preemptive scheduling are Round Robin and Shortest Remaining Time First.	Examples of non-preemptive scheduling are First Come First Serve and Shortest Job First.

UNIT-3

PROCESS MANAGEMENT

What is process?

Process is the execution of a program that performs the actions specified in that program. It can be defined as an execution unit where a program runs. The OS helps you to create, schedule, and terminates the processes which is used by CPU. A process created by the main process is called a child process.

Process operations can be easily controlled with the help of PCB (Process Control Block). You can consider it as the brain of the process, which contains all the crucial information related to processing like process id, priority, state, CPU registers, etc.

What is Process Management?

- Process management involves various tasks like creation, scheduling, termination of processes, and a dead lock.
- Process is a program that is under execution, which is an important part of modern-day operating systems.
- The OS must allocate resources that enable processes to share and exchange information.
- It also protects the resources of each process from other methods and allows synchronization among processes.
- It is the job of OS to manage all the running processes of the system. It handles operations by performing tasks like process scheduling and such as resource allocation.

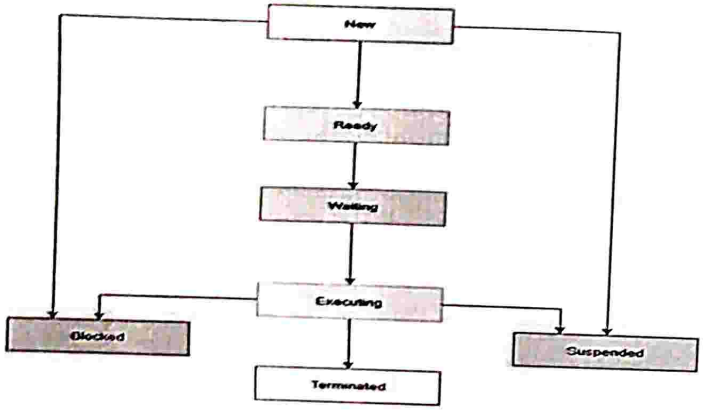
Process Architecture:

Process architecture Image

Here, Is an Architecture diagram of the Process

- **Stack:** The Stack stores temporary data like function parameters, returns addresses, and local variables.
- **Heap** Allocates memory, which may be processed during its run time.
- **Data:** It contains the variable.
- **Text:**
  Text Section includes the current activity, which is represented by the value of the Program Counter.

Process States:



Process States Diagram

A process state is a condition of the process at a specific instant of time. It also defines the current position of the process.

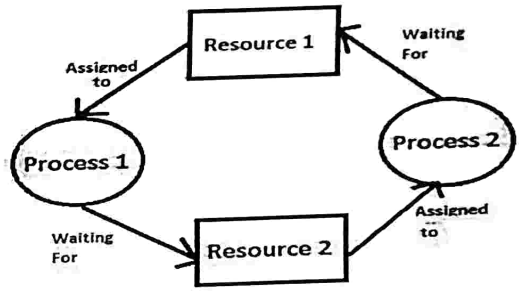
There are mainly seven stages of a process which are:

- *New:* The new process is created when a specific program calls from secondary memory/ hard disk to primary memory/ RAM a
- *Ready:* In a ready state, the process should be loaded into the primary memory, which is ready for execution.
- *Waiting:* The process is waiting for the allocation of CPU time and other resources for execution.
- *Executing:* The process is an execution state.
- *Blocked:* It is a time interval when a process is waiting for an event like I/O operations to complete.
- *Suspended:* Suspended state defines the time when a process is ready for execution but has not been placed in the ready queue by OS.
- *Terminated:* Terminated state specifies the time when a process is terminated

UDARSHAN M.SC (CS), B. ED, D.EL. ED
amsudarshan91@gmail.com

DEAD LOCK

Deadlock is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.
Consider an example when two trains are coming toward each other on the same track and there is only one track, none of the trains can move once they are in front of each other. A similar situation occurs in operating systems when there are two or more processes that hold some resources and wait for resources held by other(s). For example, in the below diagram, Process 1 is holding Resource 1 and waiting for resource 2 which is acquired by process 2, and process 2 is waiting for resource 1.



Deadlock can arise if the following four conditions hold simultaneously (Necessary Conditions)

1.Mutual Exclusion: Two or more resources are non-shareable (Only one process can use at a time)
2. Hold and Wait: A process is holding at least one resource and waiting for resources.
3.No Preemption: A resource cannot be taken from a process unless the process releases the resource.
4.Circular Wait: A set of processes are waiting for each other in circular form. What is deadlock? Explain necessary and sufficient condition for deadlock to occur

Necessary and sufficient conditions for Dead lock (or)

Deadlock Characterization

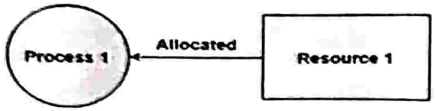
A deadlock happens in operating system when two or more processes need some resource to complete their execution that is held by the other process.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

A deadlock occurs if the four Coffman conditions hold true. But these conditions are not mutually exclusive. They are given as follows –

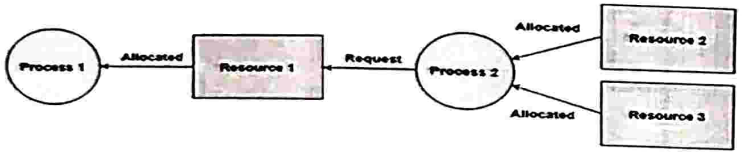
Mutual Exclusion

There should be a resource that can only be held by one process at a time. In the diagram below, there is a single instance of Resource 1 and it is held by Process 1 only.



Hold and Wait

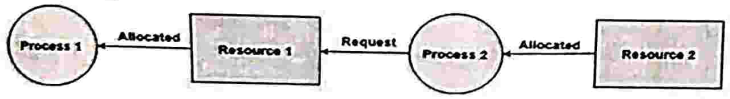
A process can hold multiple resources and still request more resources from other processes which are holding them. In the diagram given below, Process 2 holds Resource 2 and Resource 3 and is requesting the Resource 1 which is held by Process 1.



No Preemption

A resource cannot be taken away from process that is currently holding it.

A resource cannot be preempted from a process by force. A process can only release a resource voluntarily. In the diagram below, Process 2 cannot preempt Resource 1 from Process 1. It will only be released when Process 1 relinquishes it voluntarily after its execution is complete.

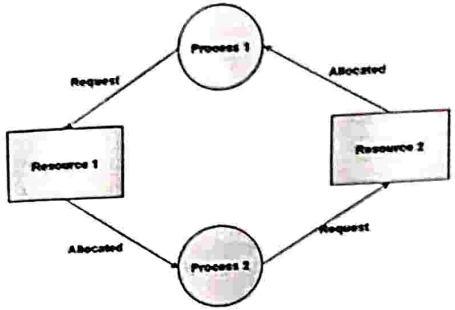


Circular Wait

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

A process is waiting for the resource held by the second process, which is waiting for the resource held by the third process and so on, till the last process is waiting for a resource held by the first process. This forms a

circular chain. For example: Process 1 is allocated Resource2 and it is requesting Resource 1. Similarly, Process 2 is allocated Resource 1 and it is requesting Resource 2. This forms a circular wait loop.



Deadlock prevention

It means that we design such a system where there is no chance of having a deadlock.

- **Mutual exclusion:**
  It can't be resolved as it is the hardware property. For example, the printer cannot be simultaneously shared by several processes. This is very difficult because some resources are not sharable.
- **Hold and wait:**
  Hold and wait can be resolved using the conservative approach where a process can start it and only if it has acquired all the resources.
- **Active approach:**
  Here the process acquires only requires resources but whenever a new resource requires it must first release all the resources.
- **Wait time out:**
  Here there is a maximum time bound until which a process can wait for other resources after which it must release the resources.
- **Circular wait:**
  In order to remove circular wait, we assign a number to every resource and the process can request only in the increasing order otherwise the process must release all the high number acquires resources and then make a fresh request.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

- **No pre-emption:**
  In no pre-emption, we allow forceful pre-emption where a resource can be forcefully pre-empted. The pre-empted resource is added to the list of resources where the process is waiting. The new process

  can be restarted only when it regains its old resources. Priority must be given to a process which is in waiting for state.

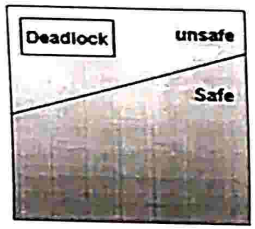
Deadlock avoidance

A deadlock avoidance algorithm dynamically examines the resources allocation state to ensure that a circular wait condition case never exists. Where the resources allocation state is defined by the of available and allocated resources and the maximum demand of the process.

deadlock avoidance is containing three states:

A state is safe if the system can allocate resources to each process (up to its maximum requirement) in some order and still avoid a deadlock. Formally, a system is in a safe state only, if there exists a safe sequence. So, a safe state is not a deadlocked state and conversely a deadlocked state is an unsafe state.

In an Unsafe state, the operating system cannot prevent processes from requesting resources in such a way that any deadlock occurs. It is not necessary that all unsafe states are deadlocks; an unsafe state may lead to a deadlock.



The above Figure shows the Safe, unsafe, and deadlocked state spaces

Let's understand it by an example:

Consider the following 3 process total resources are given for A= 6, B= 5, C= 7, D = 6

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Process	maximum				Allocation				Need			
	A	B	C	D	A	B	C	D	A	B	C	D
P1	3	3	2	2	1	2	2	1	2	1	0	1
P2	1	2	3	4	1	0	3	3	0	2	0	1
P3	1	3	5	0	1	2	1	0	0	1	4	0

First we find the need matrix by **Need= maximum – allocation**

Then find available **resources = total – allocated**

```
A   B   C  D( 6   5  7  6) -   A  B  C  D(  3  4  6  4
Available resources A B C D (3( 1   2)
```

Then we check whether the system is in deadlock or not and find the safe sequence of process.

P1 can be satisfied

```
   Available= P1 allocated + available
(1, 2, 2, 1) +( 3, 1, 1,2) = (4, 3, 3, 3)
```

P2 can be satisfied

```
   Available= P2 allocated + available
   (1, 0, 3, 3) + (4, 3, 3, 3) = (5, 3, 6, 6)
```

P3 can be satisfied

```
   Available= P3 allocated + available
   (1, 2, 1, 0) + (5, 3, 6, 6) = (6, 5, 7, 6)
```

So the system is safe and the safe sequence is **P1 → P2 → P3**

Detection and recovery

When the system is in deadlock then one method is to inform the operates and then operator deal with deadlock manually and the second method is system will automatically recover from deadlock. There are two ways to recover from deadlock:

- **Process termination:**
  Deadlock can be eliminated by aborting a process. Abort all deadlock process. Abort is processed at a time until the deadlock cycle is eliminated. This can help to recover the system from file deadlock.
- **Resources preemption:**
  To eliminate deadlock using resources preemption, we prompt the same resources pas processes and

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

give these resources to another process until the deadlock cycle is broken.
Here a process is partially rollback until the last checkpoint or and hen detection algorithm is executed.

Concurrent and Dependent process in Operating System

Concurrent processing is a computing model in which multiple processors execute instructions simultaneously for better performance. Concurrent means, which occurs when something else happens. The tasks are broken into sub-types, which are then assigned to different processors to perform simultaneously, sequentially instead, as they would have to be performed by one processor. Concurrent processing is sometimes synonymous with parallel processing.

It helps in techniques like coordinating execution of processes, memory allocation and execution scheduling for maximizing throughput.

Principles of Concurrency:

Both interleaved and overlapped processes can be viewed as examples of concurrent processes, they both present the same problems.
The relative speed of execution cannot be predicted. It depends on the following:

- The activities of other processes
- The way operating system handles interrupts
- The scheduling policies of the operating system

antages of Concurrency:

- **Running of multiple applications –**
  It enable to run multiple applications at the same time.
- **Better resource utilization –**
  It enables that the resources that are unused by one application can be used for other applications.
- **Better average response time –**
  Without concurrency, each application has to be run to completion before the next one can be run.
- **Better performance –**
  It enables the better performance by the operating system. When one application uses only the processor and another application uses only the disk drive then the time to run both applications concurrently to completion will be shorter than the time to run each application consecutively.

Drawbacks of Concurrency:

- It is required to protect multiple applications from one another.
- It is required to coordinate multiple applications through additional mechanisms.
- Additional performance overheads and complexities in operating systems are required for switching among applications.
- Sometimes running too many applications concurrently leads to severely degraded performance.

Issues of Concurrency:

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

- **Non-atomic –**
  Operations that are non-atomic but interruptible by multiple processes can cause problems.
- **Race conditions –**
  A race condition occurs of the outcome depends on which of several processes gets to a point first.
- **Blocking –**
  Processes can block waiting for resources. A process could be blocked for long period of time waiting for input from a terminal. If the process is required to periodically update some data, this would be very undesirable.
- **Starvation –**
  It occurs when a process does not obtain service to progress.
- **Deadlock –**
  It occurs when two processes are blocked and hence neither can proceed to execute.

The Critical Section Problem

- Critical Section is the part of a program which tries to access shared resources. That resource may be any resource in a computer like a memory location, Data structure, CPU or any IO device.
- The critical section cannot be executed by more than one process at the same time; operating system faces the difficulties in allowing and disallowing the processes from entering the critical section.
- The critical section problem is used to design a set of protocols which can ensure that the Race condition among the processes will never arise.
- In order to synchronize the cooperative processes, our main task is to solve the critical section problem. We need to provide a solution in such a way that the following conditions can be satisfied.

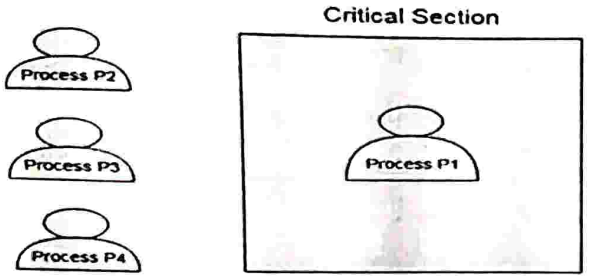
Any solution to the critical section problem must satisfy the following requirements:

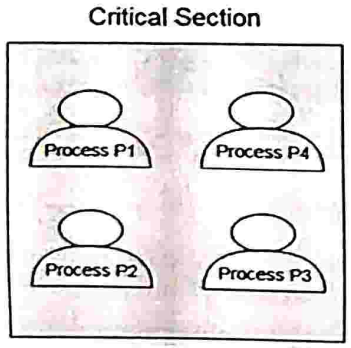
Primary:

1. **Mutual Exclusion**

   Our solution must provide mutual exclusion. By Mutual Exclusion, we mean that if one process is executing inside critical section then the other process must not enter in the critical section.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com





2. **Progress**

   Progress means that if one process doesn't need to execute into critical section then it should not stop other processes to get into the critical section.

Secondary:

1. **Bounded Waiting**

We should be able to predict the waiting time for every process to get into the critical section. The process must not be endlessly waiting for getting into the critical section.

2. **Architectural Neutrality**

   Our mechanism must be architectural natural. It means that if our solution is working fine on one architecture then it should also run on the other ones as well.

Semaphores in Operating System

Semaphores are integer variables that are used to solve the critical section problem by using two atomic operations, wait and signal that are used for process synchronization.

The definitions of wait and signal are as follows –

- **Wait**

  The wait operation decrements the value of its argument S, if it is positive. If S is negative or zero, then no operation is performed.

```
wait(S)
{
  while (S<=0);

  S--;
}
```

- **Signal**

  The signal operation increments the value of its argument S.

```
signal(S)
{
  S++;
}
```

Types of Semaphores:

There are two main types of semaphores i.e. counting semaphores and binary semaphores. Details about these are given as follows –

- **Counting Semaphores**

These are integer value semaphores and have an unrestricted value domain. These semaphores are used to coordinate the resource access, where the semaphore count is the number of available resources. If the resources are added, semaphore count automatically incremented and if the resources are removed, the count is decremented.

- **Binary Semaphores**

The binary semaphores are like counting semaphores but their value is restricted to 0 and 1. The wait operation only works when the semaphore is 1 and the signal operation succeeds when semaphore is 0. It is sometimes easier to implement binary semaphores than counting semaphores.

Methods for Interprocess Communication

Inter-process communication (IPC) is set of interfaces, which is usually programmed in order for the programs to communicate between series of processes. This allows running programs concurrently in an Operating System. These are the methods in IPC:

1. **Pipes (Same Process) –**
   This allows flow of data in one direction only. Analogous to simplex systems (Keyboard). Data from the output is usually buffered until input process receives it which must have a common origin.
2. **Names Pipes (Different Processes) –**
   This is a pipe with a specific name it can be used in processes that don't have a shared common process origin. E.g. is FIFO where the details written to a pipe is first named.
3. **Message Queuing –**
   This allows messages to be passed between processes using either a single queue or several message queue. This is managed by system kernel these messages are coordinated using an API.
4. **Semaphores –**
   This is used in solving problems associated with synchronization and to avoid race condition. These are integer values which are greater than or equal to 0.
5. **Shared memory –**
   This allows the interchange of data through a defined area of memory. Semaphore values have to be obtained before data can get access to shared memory.
6. **Sockets –**
   This method is mostly used to communicate over a network between a client and a server. It allows for a standard connection which is computer and OS independent

Process Synchronization in Operating System

Process Synchronization means coordinating the execution of processes such that no two processes access the same shared resources and data. It is required in a multi-process system where multiple processes run together, and more than one process tries to gain access to the same shared resource or data at the same time.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Changes made in one process aren't reflected when another process accesses the same shared data. It is necessary that processes are synchronized with each other as it helps avoid the inconsistency of shared data.

On the basis of synchronization, processes are categorized as one of the following two types:

- **Independent Process :** Execution of one process does not affects the execution of other processes.
- **Cooperative Process :** Execution of one process affects the execution of other processes.

Process synchronization problem arises in the case of Cooperative process also because resources are shared in Cooperative processes.

For example: A process P1 tries changing data in a particular memory location. At the same time another process P2 tries reading data from the same memory location. Thus, there is a high probability that the data being read by the second process is incorrect.

Classical problems of Synchronization

we will see number of classical problems of synchronization as examples of a large class of concurrency-control problems. In our solutions to the problems, we use semaphores for synchronization, since that is the traditional way to present such solutions. However, actual implementations of these solutions could use mutex locks in place of binary semaphores.

The following problems of synchronization are considered as classical problems:

1. **Bounded-buffer (or Producer-Consumer) Problem**
2. **Readers and Writers Problem**

1.Bounded-buffer (or Producer-Consumer) Problem:

Bounded Buffer problem is also called producer consumer problem.
The Producer-Consumer problem is a classical multi-process synchronization problem, that is we are trying to achieve synchronization between more than one process.

- This problem is generalized in terms of the Producer-Consumer problem.
- Solution to this problem is, creating two counting semaphores "full" and "empty" to keep track of the current number of full and empty buffers respectively.
- Producers produce a product and consumers consume the product, but both use of one of the containers each time.

2.Readers and Writers Problem:

Suppose that a database is to be shared among several concurrent processes. Some of these processes may want only to read the database, whereas others may want to update (that is, to read and write) the database.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

We distinguish between these two types of processes by referring to the former as readers and to the latter as writers. Precisely in OS we call this situation as the readers-writers problem. Problem parameters:

- One set of data is shared among a number of processes.
- Once a writer is ready, it performs its write. Only one writer may write at a time.
- If a process is writing, no other process can read it.
- If at least one reader is reading, no other process can write.
- Readers may not write and only read.

UNIT-4

MEMORY MANAGEMENT

Memory Management in Operating System

- In a multiprogramming computer, the operating system resides in a part of memory and the rest is used by multiple processes.
- The task of subdividing the memory among different processes is called memory management.
- Memory management is a method in the operating system to manage operations between main memory and disk during process execution.
- The main aim of memory management is to achieve efficient utilization of memory.

Physical and virtual Address Space

Physical Address space:

- An address seen by the memory unit (i.e., the one loaded into the memory address register of the memory) is commonly known as a "Physical Address".
- A Physical address is also known as a Real address.
- The set of all physical addresses corresponding to these logical addresses is known as Physical address space.
- A physical address is computed by MMU.
- The run-time mapping from virtual to physical addresses is done by a hardware device Memory Management Unit (MMU).
- The physical address always remains constant.

Logical Address space:

- An address generated by the CPU is known as "Logical Address".
- It is also known as a Virtual address.
- Logical address space can be defined as the size of the process. A logical address can be changed.

Memory allocation strategies

To gain proper memory utilization, memory allocation must be allocated efficient manner. One of the simplest methods for allocating memory is to divide memory into several fixed-sized partitions and each partition contains exactly one process. Thus, the degree of multiprogramming is obtained by the number of partitions.

In the operating system, the following are four common memory management techniques.

Single contiguous allocation: Simplest allocation method used by MS-DOS. All memory (except some reserved for OS) is available to a process.
Partitioned allocation: Memory is divided into different blocks or partitions. Each process is allocated according to the requirement.
Paged memory management: Memory is divided into fixed-sized units called page frames, used in a virtual memory environment.
Segmented memory management: Memory is divided into different segments (a segment is a logical grouping of the process' data or code). In this management, allocated memory doesn't have to be contiguous.
Most of the operating systems (for example Windows and Linux) use Segmentation with Paging. A process is divided into segments and individual segments have pages.

In **Partition Allocation**, when there is more than one partition freely available to accommodate a process's request, a partition must be selected. To choose a particular partition, a partition allocation method is needed. A partition allocation method is considered better if it avoids internal fragmentation.
When it is time to load a process into the main memory and if there is more than one free block of memory of sufficient size then the OS decides which free block to allocate.

There are different Placement Algorithm:

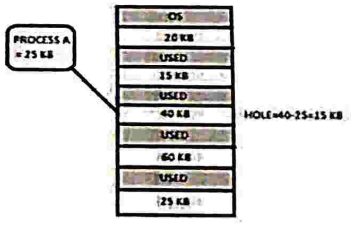
A. First Fit

B. Best Fit

C. Worst Fit

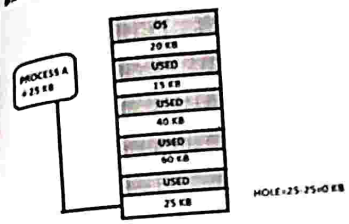
D. Next Fit

1. First Fit: In the first fit, the partition is allocated which is the first sufficient block from the top of Main Memory. It scans memory from the beginning and chooses the first available block that is large enough. Thus, it allocates the first hole that is large enough.

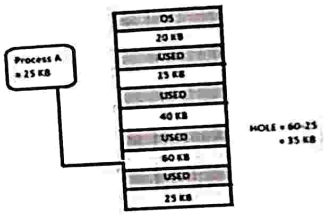


2. Best Fit Allocate the process to the partition which is the first smallest sufficient partition among the free available partition. It searches the entire list of holes to find the smallest hole whose size is greater than or equal to the size of the process.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



3. Worst Fit Allocate the process to the partition which is the largest sufficient among the freely available partitions available in the main memory. It is opposite to the best-fit algorithm. It searches the entire list of holes to find the largest hole and allocate it to process.



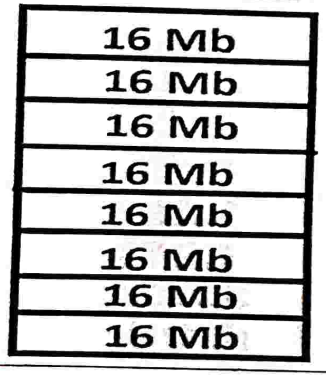
3. **Next Fit:** Next fit is similar to the first fit but it will search for the first sufficient partition from the last allocation point.

Fixed Partitioning and Variable Partitioning

1. **Fixed Partitioning :**

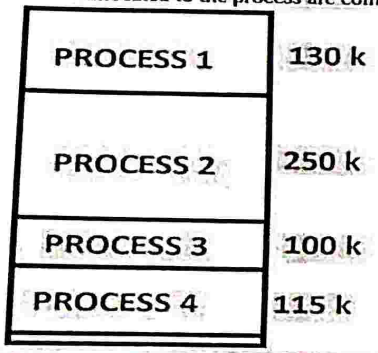
Multi-programming with fixed partitioning is a contiguous memory management technique in which the main memory is divided into fixed sized partitions which can be of equal or unequal size.
Whenever we have to allocate a process memory then a free partition that is big enough to hold the process is found. Then the memory is allocated to the process. If there is no free space available then the process waits in the queue to be allocated memory. It is one of the oldest memory management techniques which is easy to implement.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



2. **Variable Partitioning :**

Multi-programming with variable partitioning is a contiguous memory management technique in which the main memory is not divided into partitions and the process is allocated a chunk of free memory that is big enough for it to fit. The space which is left is considered as the free space which can be further used by other processes. It also provides the concept of compaction. In compaction the spaces that are free and the spaces which not allocated to the process are combined and single large memory space is made.



B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Difference between Fixed Partitioning and Variable Partitioning:

S.NO.	Fixed partitioning	Variable partitioning
1.	In multi-programming with fixed partitioning the main memory is divided into fixed sized partitions.	In multi-programming with variable partitioning the main memory is not divided into fixed sized partitions.
2.	Only one process can be placed in a partition.	In variable partitioning, the process is allocated a chunk of free memory.
3.	It does not utilize the main memory effectively.	It utilizes the main memory effectively.
4.	There is presence of internal fragmentation and external fragmentation.	There is external fragmentation.
5.	Degree of multi-programming is less.	Degree of multi-programming is higher.
6.	It is more easier to implement.	It is less easier to implement.
7.	There is limitation on size of process.	There is no limitation on size of process.

Paging in memory management

Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory. This scheme permits the physical address space of a process to be non-contiguous.

- Logical Address or Virtual Address (represented in bits): An address generated by the CPU
- Logical Address Space or Virtual Address Space (represented in words or bytes): The set of all logical addresses generated by a program
- Physical Address (represented in bits): An address actually available on a memory unit
- Physical Address Space (represented in words or bytes): The set of all physical addresses corresponding to the logical addresses

Example:

- If Logical Address = 31 bits, then Logical Address Space = 2^{31} words = 2 G words (1 G = 2^{30})
- If Logical Address Space = 128 M words = $2^7 * 2^{20}$ words, then Logical Address = $\log_2 2^{27}$ = 27 bits

B.SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

- If Physical Address = 22 bits, then Physical Address Space = 2^{22} words = 4 M words (1 M = 2^{20})
- If Physical Address Space = 16 M words = $2^4 * 2^{20}$ words, then Physical Address = $\log_2 2^{24}$ = 24 bits

Segmentation in memory management

In Operating Systems, Segmentation is a memory management technique in which the memory is divided into the variable size parts. Each part is known as a segment which can be allocated to a process.

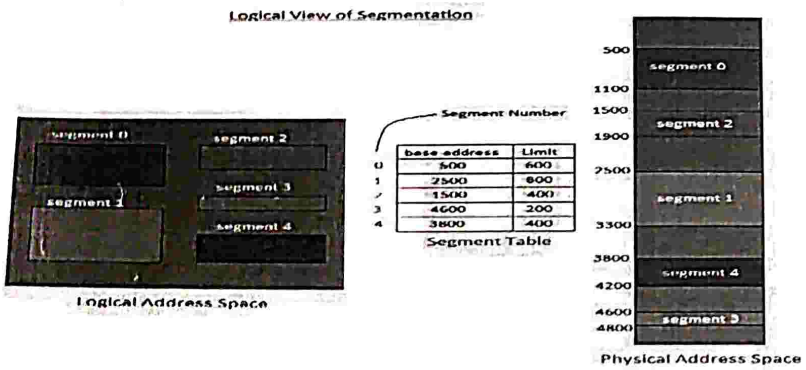
There are types of segmentation:

1. **Virtual memory segmentation –**
   Each process is divided into a number of segments, not all of which are resident at any one point in time.
2. **Simple segmentation –**
   Each process is divided into a number of segments, all of which are loaded into memory at run time, though not necessarily contiguously.

There is no simple relationship between logical addresses and physical addresses in segmentation. A table stores the information about all such segments and is called Segment Table.

Segment Table – It maps two-dimensional Logical address into one-dimensional Physical address. It's each table entry has:

- **Base Address:** It contains the starting physical address where the segments reside in memory.
- **Limit:** It specifies the length of the segment.



B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
addamsudarshan91@gmail.com

Virtual Memory

Virtual Memory is a storage scheme that provides user an illusion of having a very big main memory. This is done by treating a part of secondary memory as the main memory.

In this scheme, User can load the bigger size processes than the available main memory by having the illusion that the memory is available to load the process.

Instead of loading one big process in the main memory, the Operating System loads the different parts of more than one process in the main memory.

By doing this, the degree of multiprogramming will be increased and therefore, the CPU utilization will also be increased.

Advantages of Virtual Memory

1. The degree of Multiprogramming will be increased.
2. User can run large application with less real RAM.
3. There is no need to buy more memory RAMs.

Disadvantages of Virtual Memory

1. The system becomes slower since swapping takes time.
2. It takes more time in switching between applications.
3. The user will have the lesser hard disk space for its use.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
addamsudarshan91@gmail.com

UNIT-5

FILE AND I/O MANAGEMENT, OS SECURITY

What is a file?

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.

File management

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

Normally files are organised into directories to ease their use. 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 (non volatile) storage.

Input and output devices are components that form part of the computer system. These devices are controlled by the operating system. Input devices such as keyboard, mouse, and sensors provide input signals such as commands to the operating system.

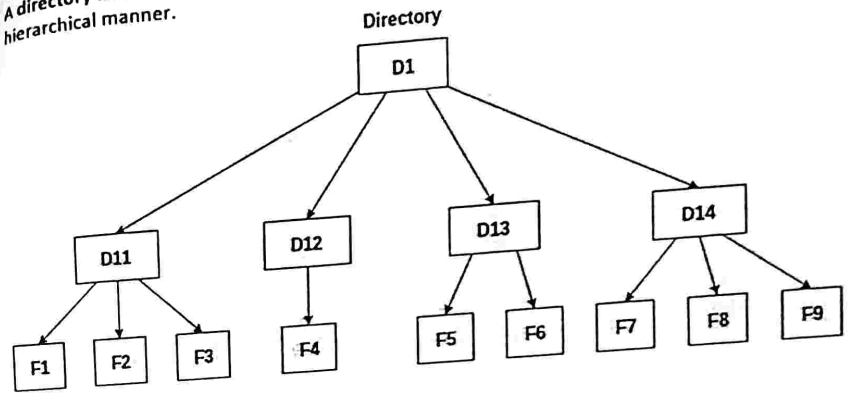
These commands received from input devices instruct the operating system to perform some task or control its behaviour. Output devices such as monitors, printers and speakers are the devices that receive commands or information from the operating system.

In the earlier unit, we had studied the memory management of primary memory. The physical memory, as we have already seen, is not large enough to accommodate all of the needs of a computer system. Also, it is not permanent. Secondary storage consists of disk units and tape drives onto which data can be moved for permanent storage. Though there are actual physical differences between tapes and disks, the principles involved in controlling them are the same, so we shall only consider disk management here in this unit.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

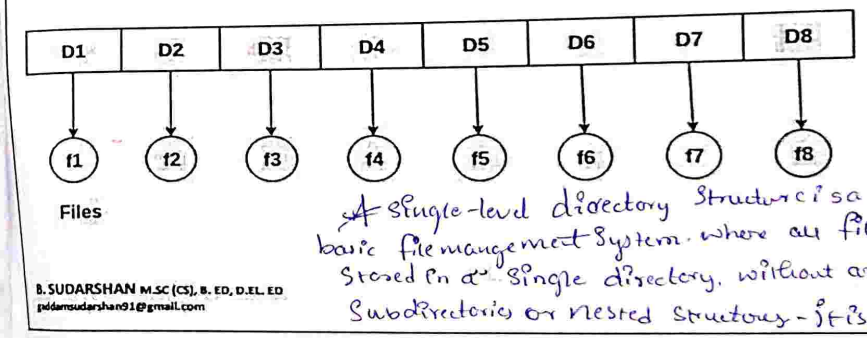
Structures of Directory in Operating System

A directory is a container that is used to contain folders and files. It organizes files and folders in a hierarchical manner.



There are several logical structures of a directory, these are given below.

- **Single-level directory –**
  The single-level directory is the simplest directory structure. In it, all files are contained in the same directory which makes it easy to support and understand.
  A single level directory has a significant limitation, however, when the number of files increases or when the system has more than one user. Since all the files are in the same directory, they must have a unique name. if two users call their dataset test, then the unique name rule violated.

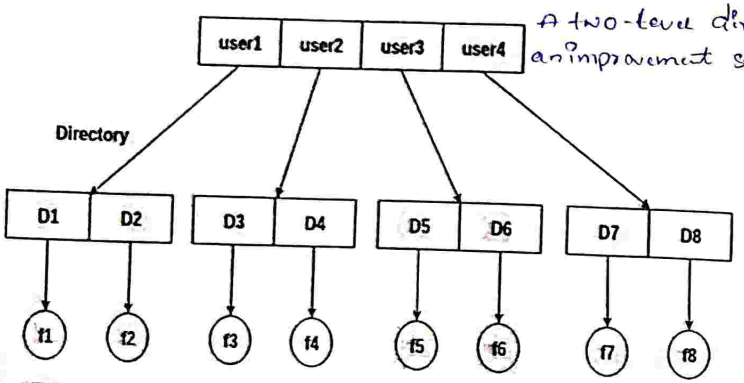


A single-level directory structure is a basic file management system where all files are stored in a single directory, without any subdirectories or nested structures. It's simple approach but can become inefficient as the number of files increases.

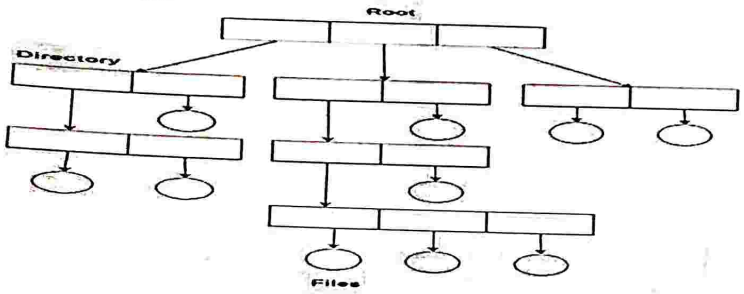
B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

- **Two-level directory –**
  As we have seen, a single level directory often leads to confusion of files names among different users. the solution to this problem is to create a separate directory for each user.
  In the two-level directory structure, each user has their own *user files directory (UFD)*. The UFDs have similar structures, but each lists only the files of a single user. system's *master file directory (MFD)* is searches whenever a new user id=s logged in. The MFD is indexed by username or account number, and each entry points to the UFD for that user.

A two-level directory structure is an improvement over a



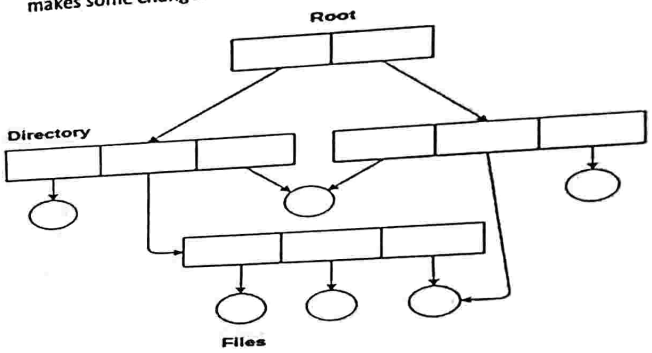
- **Tree-structured directory –**
  Once we have seen a two-level directory as a tree of height 2, the natural generalization is to extend the directory structure to a tree of arbitrary height.
  This generalization allows the user to create their own subdirectories and to organize their files accordingly.



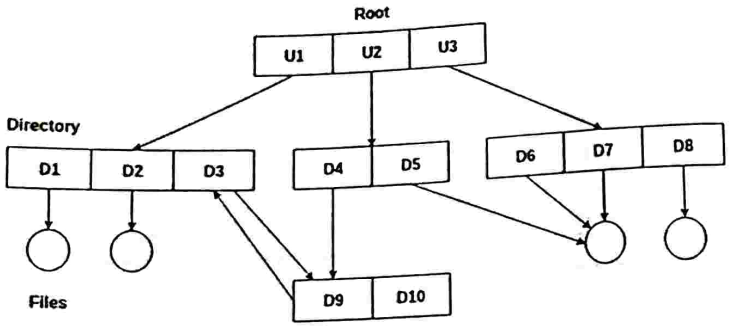
A tree structure is the most common directory structure. The tree has a root directory, and every file in the system has a unique path.

- **Acyclic graph directory –**
  An acyclic graph is a graph with no cycle and allows us to share subdirectories and files. The same file or subdirectories may be in two different directories. It is a natural generalization of the tree-structured directory.
  It is used in the situation like when two programmers are working on a joint project and they need to access files. The associated files are stored in a subdirectory, separating them from other projects and files of other programmers since they are working on a joint project so they want the subdirectories to be into their own directories. The common subdirectories should be shared. So here we use Acyclic directories.

  It is the point to note that the shared file is not the same as the copy file. If any programmer makes some changes in the subdirectory it will reflect in both subdirectories.



- **General graph directory structure –**
  In general graph directory structure, cycles are allowed within a directory structure where multiple directories can be derived from more than one parent directory.
  The main problem with this kind of directory structure is to calculate the total size or space that has been taken by the files and directories.



File operations in Operating System

A file is a collection of related information that is recorded on secondary storage. Or file is a collection of logically related entities. From user's perspective a file is the smallest allotment of logical secondary storage.

The name of the file is divided into two parts as shown below:

- name
- extension, separated by a period.

Here is a list of some operations that can be carried out on a file –

- Creating a new file
- Opening an existing file
- Reading file contents
- Searching data on a file
- Writing into a new file
- Updating contents to an existing file
- Deleting a file
- Closing a file

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Files attributes and its operations:

Attributes	Types	Operations
Name	Doc	Create
Type	Exe	Open
Size	Jpg	Read
Creation Data	Xis	Write
Author	C	Append
Last Modified	Java	Truncate
protection	class	Delete
		Close

File Allocation Methods

The allocation methods define how the files are stored in the disk blocks. There are three main disk space or file allocation methods.

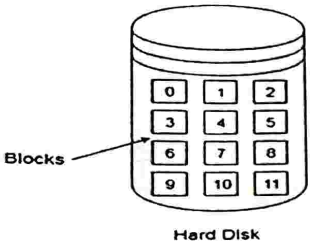
- Contiguous Allocation
- Linked List Allocation
- Indexed Allocation

1. Contiguous Allocation:

If the blocks are allocated to the file in such a way that all the logical blocks of the file get the contiguous physical block in the hard disk then such allocation scheme is known as contiguous allocation.

In the Image shown below, there are three files in the directory. The starting block and the length of each file are mentioned in the table. We can check in the table that the contiguous blocks are assigned to each file as per its need.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



File Name	Start	Length	Allocated Blocks
abc.text	0	3	0,1,2
video.mp4	4	2	4,5
jtp.docx	9	3	9,10,11

Directory

Contiguous Allocation

Advantages

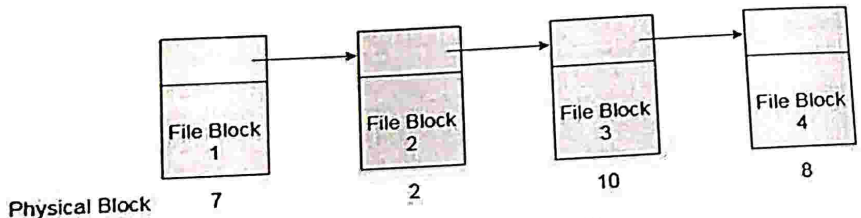
1. It is simple to implement.
2. We will get Excellent read performance.
3. Supports Random Access into files.

Disadvantages

1. The disk will become fragmented.
2. It may be difficult to have a file grow.

2. Linked List Allocation:

Linked List allocation solves all problems of contiguous allocation. In linked list allocation, each file is considered as the linked list of disk blocks. However, the disks blocks allocated to a particular file need not to be contiguous on the disk. Each disk block allocated to a file contains a pointer which points to the next disk block allocated to the same file.



Linked List Allocation

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Advantages

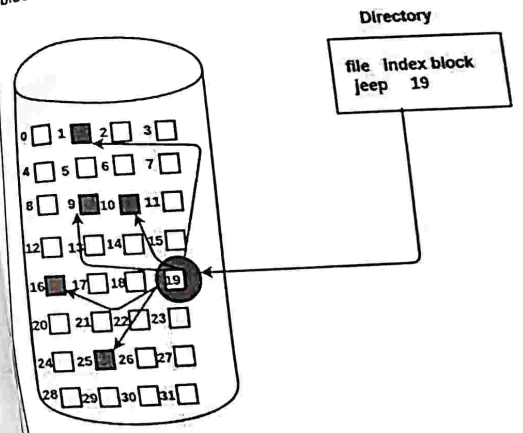
1. There is no external fragmentation with linked allocation.
2. Any free block can be utilized in order to satisfy the file block requests.
3. File can continue to grow as long as the free blocks are available.
4. Directory entry will only contain the starting block address.

Disadvantages

1. Random Access is not provided.
2. Pointers require some space in the disk blocks.
3. Any of the pointers in the linked list must not be broken otherwise the file will get corrupted.
4. Need to traverse each block.

3. Indexed Allocation:

In this scheme, a special block known as the Index block contains the pointers to all the blocks occupied by a file. Each file has its own index block. The ith entry in the index block contains the disk address of the ith file block. The directory entry contains the address of the index block as shown in the image:



Advantages:

- This supports direct access to the blocks occupied by the file and therefore provides fast access to the file blocks.
- It overcomes the problem of external fragmentation.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Disadvantages:

- The pointer overhead for indexed allocation is greater than linked allocation.
- For very small files, say files that expand only 2-3 blocks, the indexed allocation would keep one entire block (index block) for the pointers which is inefficient in terms of memory utilization. However, in linked allocation we lose the space of only 1 pointer per block.

Device Management in Operating System

Device management in an operating system means controlling the Input/Output devices like disk, microphone, keyboard, printer, magnetic tape, USB ports, camcorder, scanner, other accessories, and supporting units like supporting units control channels.

A process may require various resources, including main memory, file access, and access to disk drives, and others. If resources are available, they could be allocated, and control returned to the CPU. Otherwise, the procedure would have to be postponed until adequate resources become available.

The system has multiple devices, and in order to handle these physical or virtual devices, the operating system requires a separate program known as an ad device controller. It also determines whether the requested device is available.

The fundamentals of I/O devices may be divided into three categories:

1. **Boot Device**
2. **Character Device**
3. **Network Device**

1. Boot Device

It stores data in fixed-size blocks, each with its unique address. For example- Disks.

2. Character Device

It transmits or accepts a stream of characters, none of which can be addressed individually. For instance, keyboards, printers, etc.

3. Network Device

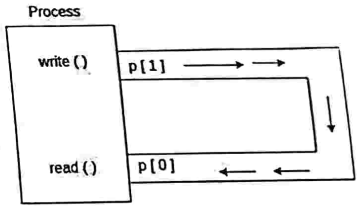
It is used for transmitting the data packets.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Pipes

A pipe is a connection between two processes, such that the standard output from one process becomes the standard input of the other process. In UNIX Operating System, Pipes are useful for communication between related processes (inter-process communication).

- Pipe is one-way communication only i.e we can use a pipe such that One process write to the pipe, and the other process reads from the pipe. It opens a pipe, which is an area of main memory that is treated as a ***"virtual file"***.
- The pipe can be used by the creating process, as well as all its child processes, for reading and writing. One process can write to this "virtual file" or pipe and another related process can read from it.
- If a process tries to read before something is written to the pipe, the process is suspended until something is written.
- The pipe system call finds the first two available positions in the process's open file table and allocates them for the read and write ends of the pipe.



Syntax in C language:

```
int pipe(int fds[2]);
```

Parameters :

fd[0] will be the fd(file descriptor) for the read end of pipe.
fd[1] will be the fd for the write end of pipe.

Returns : 0 on Success.

Buffer

A buffer is provided by the operating system to the system portion of the main memory.

The ***buffer*** is an area in the main memory used to store or hold the data **temporarily**. In other words, buffer temporarily stores data transmitted from one place to another, either between two devices or an application. The act of storing data temporarily in the buffer is called **buffering**.

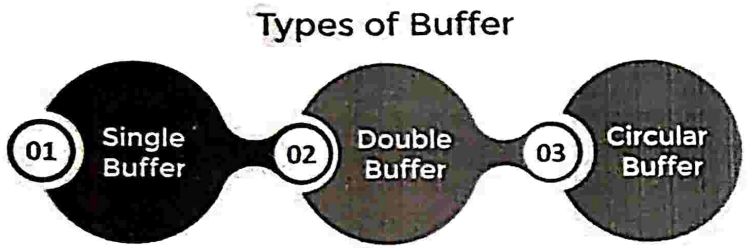
B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Purpose of Buffering

You face buffer during watching videos on YouTube or live streams. In a video stream, a buffer represents the amount of data required to be downloaded before the video can play to the viewer in real-time. A buffer in a computer environment means that a set amount of data will be stored to preload the required data before it gets used by the CPU.

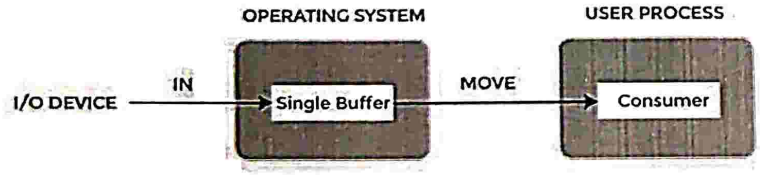
Types of Buffering

There are three main types of buffering in the operating system, such as:



1. Single Buffer

In Single Buffering, only one buffer is used to transfer the data between two devices. The producer produces one block of data into the buffer. After that, the consumer consumes the buffer. Only when the buffer is empty, the processor again produces the data.

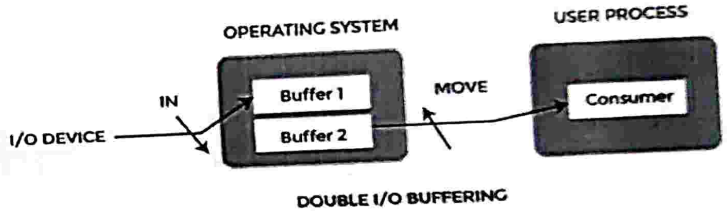


SINGLE I/O BUFFERING

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

2. Double Buffer

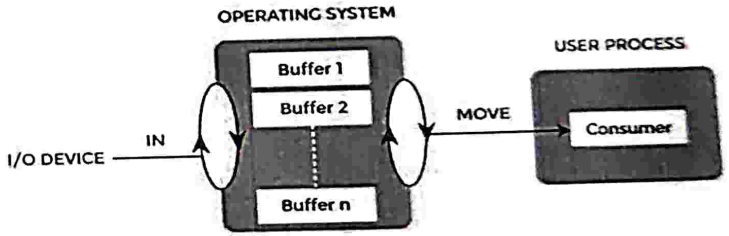
In Double Buffering, two schemes or two buffers are used in the place of one. In this buffering, the producer produces one buffer while the consumer consumes another buffer simultaneously. So, the producer not needs to wait for filling the buffer. Double buffering is also known as buffer swapping.



DOUBLE I/O BUFFERING

3. Circular Buffer

When more than two buffers are used, the buffers' collection is called a circular buffer. Each buffer is being one unit in the circular buffer. The data transfer rate will increase using the circular buffer rather than the double buffering.

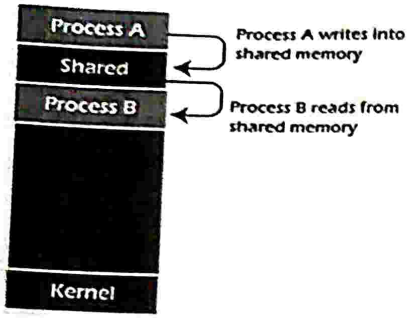


CIRCULAR I/O BUFFERING

Shared Memory

Shared memory is a memory shared between two or more processes. Each process has its own address space; if any process wants to communicate with some information from its own address space to other processes, then it is only possible with IPC (inter-process communication) techniques. Shared memory is the fastest inter-process communication mechanism. The operating system maps a memory segment in the address space of several processes to read and write in that memory segment without calling operating system functions.

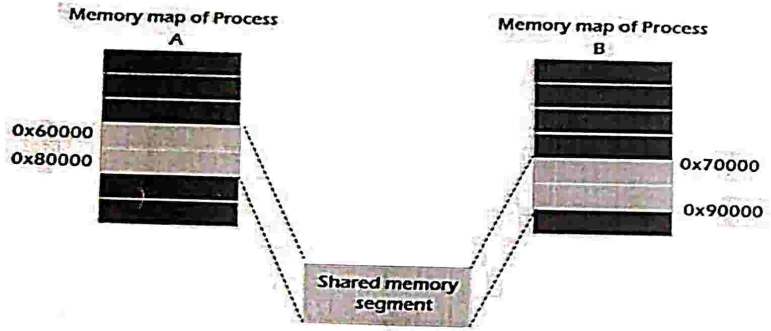
B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



For applications that exchange large amounts of data, shared memory is far superior to message passing techniques like message queues, which require system calls for every data exchange. To use shared memory, we have to perform two basic steps:

1. Request a memory segment that can be shared between processes to the operating system.
2. Associate a part of that memory or the whole memory with the address space of the calling process.

A shared memory segment is a portion of physical memory that is shared by multiple processes. In this region, processes can set up structures, and others may read/write on them. When a shared memory region is established in two or more processes, there is no guarantee that the regions will be placed at the same base address. Semaphores can be used when synchronization is required.



For example, one process might have the shared region starting at address 0x60000 while the other process uses 0x70000. It is critical to understand that these two addresses refer to the exact same piece of data. So

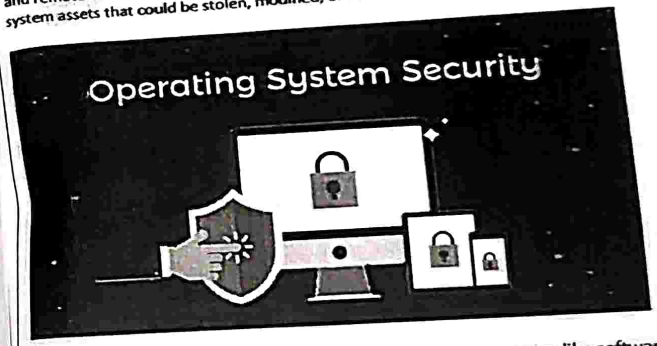
B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

storing the number 1 in the first process's address 0x60000 means the second process has the value of 1 at 0x70000. The two different addresses refer to the exact same location.

Security policy mechanism

Every computer system and software design must handle all security risks and implement the necessary measures to enforce security policies. At the same time, it's critical to strike a balance because strong security measures might increase costs while also limiting the system's usability, utility, and smooth operation. As a result, system designers must assure efficient performance without compromising security.

The process of ensuring OS availability, confidentiality, integrity is known as operating system security. OS security refers to the processes or measures taken to protect the operating system from dangers, including viruses, worms, malware, and remote hacker intrusions. Operating system security comprises all preventive-control procedures that protect any system assets that could be stolen, modified, or deleted if OS security is breached.



Security refers to providing safety for computer system resources like software, CPU, memory, disks, etc. It can protect against all threats, including viruses and unauthorized access. It can be enforced by assuring the operating system's **integrity, confidentiality,** and **availability.** If an illegal user runs a computer application, the computer or data stored may be seriously damaged.

System security may be threatened through two violations, and these are as follows:

1. Threat

A program that has the potential to harm the system seriously.

2. Attack

A breach of security that allows unauthorized access to a resource.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

There are two types of security breaches that can harm the system: malicious and accidental. Malicious threats are a type of destructive computer code or web script that is designed to cause system vulnerabilities that lead to back doors and security breaches. On the other hand, Accidental Threats are comparatively easier to protect against.

Security may be compromised through the breaches. Some of the breaches are as follows:

1. Breach of integrity

This violation has unauthorized data modification.

2. Theft of service

It involves the unauthorized use of resources.

3. Breach of confidentiality

It involves the unauthorized reading of data.

4. Breach of availability

It involves the unauthorized destruction of data.

5. Denial of service

It includes preventing legitimate use of the system. Some attacks may be accidental.

Protection in Operating System

Protection is especially important in a multiuser environment when multiple users use computer resources such as CPU, memory, etc. It is the operating system's responsibility to offer a mechanism that protects each process from other processes. In a multiuser environment, all assets that require protection are classified as objects, and those that wish to access these objects are referred to as subjects. The operating system grants different 'access rights' to different subjects.

A mechanism that controls the access of programs, processes, or users to the resources defined by a computer system is referred to as protection. You may utilize protection as a tool for multi-programming operating systems, allowing multiple users to safely share a common logical namespace, including a directory or files.

It needs the protection of computer resources like the software, memory, processor, etc. Users should take protective measures as a helper to multiprogramming OS so that multiple users may safely use a common logical namespace like a directory or data. Protection may be achieved by maintaining confidentiality, honesty and availability in the OS. It is critical to secure the device from unauthorized access, viruses, worms, and other malware.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Various needs of protection in the operating system are as follows:

1. There may be security risks like unauthorized reading, writing, modification, or preventing the system from working effectively for authorized users.
2. It helps to ensure data security, process security, and program security against unauthorized user access or program access.
3. It is important to ensure no access rights' breaches, no viruses, no unauthorized access to the existing data.
4. Its purpose is to ensure that only the systems' policies access programs, resources, and data.

Authentication and internal access authorization

Authentication is the process of verifying the identity of user or information. User authentication is the process of verifying the identity of user when that user logs into a computer system.

There are different types of authentication systems which are: –

1. Single-Factor authentication: –This was the first method of security that was developed. On this authentication system, the user has to enter the username and the password to confirm whether that user is logging in or not. Now if the username or password is wrong, then the user will not be allowed to log in or access the system.

Advantage of the Single-Factor Authentication System: –

- It is a very simple to use and straightforward system.
- it is not at all costly.
- The user does not need any huge technical skills.

The disadvantage of the Single-Factor Authentication

- It is not at all password secure. It will depend on the strength of the password entered by the user.
- The protection level in Single-Factor Authentication is much low.

2. Two-factor Authentication: – In this authentication system, the user has to give a username, password, and other information. There are various types of authentication systems that are used by the user for securing the system. Some of them are: –wireless tokens, virtual tokens. otp and more.

Advantage of the Two-Factor Authentication

- The Two-Factor Authentication System provides better security than the Single-factor Authentication system.
- The productivity and flexibility increase in the two-factor authentication system.
- The Two-Factor Authentication prevents the loss of trust.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Disadvantages of Two-Factor Authentication

- It is time-consuming.

3. Multi-Factor authentication system: – In this type of authentication, more than one factor of authentication is needed. This gives better security to the user. Any type of keylogger or phishing attack will not be possible in a Multi-Factor Authentication system. This assures the user, that the information will not get stolen from them.

The advantage of Multi-Factor Authentication System are: –

- No risk of security.
- No information could get stolen.
- No risk of any key-logger activity.
- No risk of any data getting captured.

The disadvantage of Multi-Factor Authentication System are: –

- It is time-consuming.
- it can rely on third parties.

The main objective of authentication is to allow authorized users to access the computer and to deny access to the unauthorized users. Operating Systems generally identifies/authenticates users using following 3 ways : Passwords, Physical identification, and Biometrics. These are explained as following below.

1.Passwords :
Passwords verification is the most popular and commonly used authentication technique. A password is a secret text that is supposed to be known only to a user. In password based system, each user is assigned a valid username and password by the system administrator.

System stores all username and Passwords. When a user logs in, its user name and password is verified by comparing it with stored login name and password. If the contents are same then the user is allowed to access the system otherwise it is rejected.

2.Physical Identification :
This technique include machine readable badges(symbols), card or smart cards. In some companies, badges are required for employees to gain access to the organization's gate. In many system, identification is combined with the use of password i.e the user must insert the card and then supply his /her password. This kind of authentication is commonly used with ATM. Smart card can enhance this scheme by keeping the user password within the card itself. This allow the authentication without storage of password in the computer system. The loss of such card can be dangerous.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

3.Biometrics :
This method of authentication is based on the unique biological characteristics of each user such as finger prints, voice or face recognition, signatures and eyes.

Introduction to android operating system

What is Android OS?

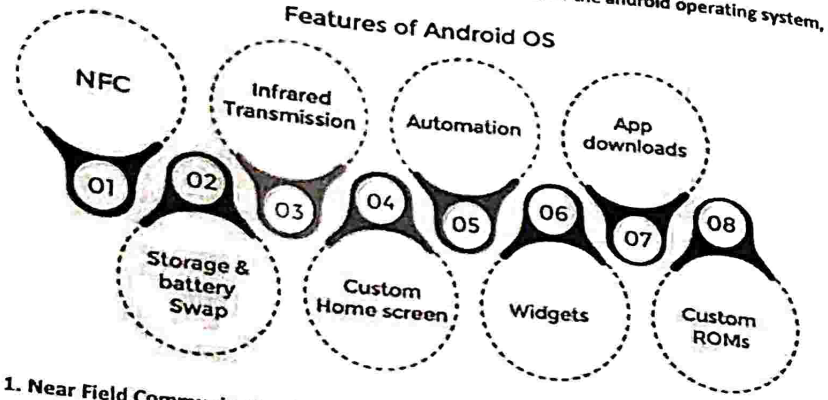
- Android OS is an operating system that was developed by Google for use on mobile devices.
- This means that it was designed for systems with little memory and a processor that isn't as fast as desktop processors.
- While keeping the limitations in mind, Google's vision for Android is that it would have a robust set of programming APIs and a very responsive UI.
- In order to facilitate this vision, they created an abstraction layer, which allows application developers to be hardware agnostic in their design.
- Android is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touch screen mobile devices such as smart phones and tablets.
- Android is developed by a partnership of developers known as the Open Handset Alliance and commercially sponsored by Google. It was disclosed in November 2007, with the first commercial Android device, the HTC Dream, launched in September 2008.
- It is free and open-source software. Its source code is Android Open Source Project (AOSP), primarily licensed under the Apache License.
- However, most Android devices dispatch with additional proprietary software pre-installed, mainly Google Mobile Services (GMS), including core apps such as Google Chrome, the digital distribution platform Google Play and the associated Google Play Services development platform.

- About 70% of Android Smartphone runs Google's ecosystem, some with vendor-customized user interface and some with software suite, such as ***TouchWiz***and later ***One UI*** by Samsung, and ***HTC Sense***.
- Competing Android ecosystems and forksinclude Fire OS (developed by Amazon) or LineageOS. However, the "Android" name and logo are trademarks of Google which impose standards to restrict "uncertified" devices outside their ecosystem to use android branding.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Features of Android Operating System

Below are the following unique features and characteristics of the android operating system, such as:



1. Near Field Communication (NFC)

Most Android devices support NFC, which allows electronic devices to interact across short distances easily. The main goal here is to create a payment option that is simpler than carrying cash or credit cards, and while the market hasn't exploded as many experts had predicted, there may be an alternative in the works, in the form of Bluetooth Low Energy (BLE).

2. Infrared Transmission

The Android operating system supports a built-in infrared transmitter that allows you to use your phone or tablet as a remote control.

3. Automation

The ***Tasker*** app allows control of app permissions and also automates them.

4. Wireless App Downloads

You can download apps on your PC by using the Android Market or third-party options like *AppBrain*. Then it automatically syncs them to your Droid, and no plugging is required.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

5. Storage and Battery Swap

Android phones also have unique hardware capabilities. Google's OS makes it possible to upgrade, replace, and remove your battery that no longer holds a charge. In addition, Android phones come with SD card slots for expandable storage.

6. Custom Home Screens

While it's possible to hack certain phones to customize the home screen, Android comes with this capability from the get-go. Download a third-party launcher like ***Apex, Nova***, and you can add gestures, new shortcuts, or even performance enhancements for older-model devices.

7. Widgets

Apps are versatile, but sometimes you want information at a glance instead of having to open an app and wait for it to load. Android widgets let you display just about any feature you choose on the home screen, including weather apps, music widgets, or productivity tools that helpfully remind you of upcoming meetings or approaching deadlines.

8. Custom ROMs

Because the Android operating system is open-source, developers can twist the current OS and build their versions, which users can download and install in place of the stock OS. Some are filled with features, while others change the look and feel of a device. Chances are, if there's a feature you want, someone has already built a custom ROM for it.

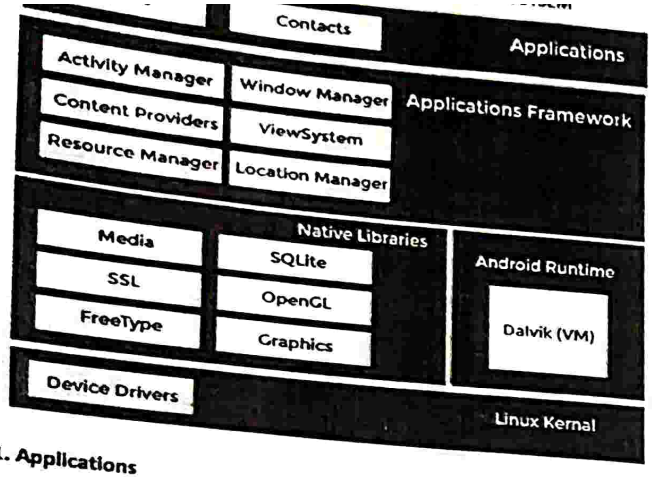
Architecture of Android OS

The android architecture contains a different number of components to support any android device needs. Android software contains an open-source Linux Kernel with many C/C++ libraries exposed through application framework services.

Among all the components, Linux Kernel provides the main operating system functions to Smartphone and Dalvik Virtual Machine (DVM) to provide a platform for running an android application. An android operating system is a stack of software components roughly divided into five sections and four main layers, as shown in the below architecture diagram.

- Applications
- Application Framework
- Android Runtime
- Platform Libraries
- Linux Kernel

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com



1. Applications

An application is the top layer of the android architecture. The pre-installed applications like camera, gallery, home, contacts, etc., and third-party applications downloaded from the play store like games, chat applications, etc., will be installed on this layer.

It runs within the Android run time with the help of the classes and services provided by the application framework.

2. Application framework

Application Framework provides several important classes used to create an Android application. It provides a generic abstraction for hardware access and helps in managing the user interface with application resources. Generally, it provides the services with the help of which we can create a particular class and make that class helpful for the Applications creation.

It includes different types of services, such as activity manager, notification manager, view system, package manager etc., which are helpful for the development of our application according to the prerequisite.

The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications. The Android framework includes the following key services:

- **Activity Manager:** Controls all aspects of the application lifecycle and activity stack.
- **Content Providers:** Allows applications to publish and share data with other applications.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

- **Resource Manager:** Provides access to non-code embedded resources such as strings, colour settings and user interface layouts.
- **Notifications Manager:** Allows applications to display alerts and notifications to the user.
- **View System:** An extensible set of views used to create application user interfaces.

3. Application runtime

Android Runtime environment contains components like core libraries and the Dalvik virtual machine (DVM). It provides the base for the application framework and powers our application with the help of the core libraries.

Like ***Java Virtual Machine*** (JVM), ***Dalvik Virtual Machine*** (DVM) is a register-based virtual machine designed and optimized for Android to ensure that a device can run multiple instances efficiently.

It depends on the layer Linux kernel for threading and low-level memory management. The core libraries enable us to implement android applications using the standard ***JAVA*** or ***Kotlin*** programming languages.

4. Platform libraries

The Platform Libraries include various C/C++ core libraries and Java-based libraries such as Media, Graphics, Surface Manager, OpenGL, etc., to support Android development.

- **app:** Provides access to the application model and is the cornerstone of all Android applications.
- **content:** Facilitates content access, publishing and messaging between applications and application components.
- **database:** Used to access data published by content providers and includes SQLite database, management classes.
- **OpenGL:** A Java interface to the OpenGL ES 3D graphics rendering API.
- **os:** Provides applications with access to standard operating system services, including messages, system services and inter-process communication.
- **text:** Used to render and manipulate text on a device display.
- **view:** The fundamental building blocks of application user interfaces.
- **widget:** A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
- **WebKit:** A set of classes intended to allow web-browsing capabilities to be built into applications.
- **media:** Media library provides support to play and record an audio and video format.
- **surface manager:** It is responsible for managing access to the display subsystem.
- **SQLite:** It provides database support, and FreeType provides font support.
- **SSL:** Secure Sockets Layer is a security technology to establish an encrypted link between a web server and a web browser.

5. Linux Kernel

Linux Kernel is the heart of the android architecture. It manages all the available drivers such as display, camera, Bluetooth, audio, memory, etc., required during the runtime.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

The Linux Kernel will provide an abstraction layer between the device hardware and the other android architecture components. It is responsible for the management of memory, power, devices etc. The features of the Linux kernel are:

- **Security:** The Linux kernel handles the security between the application and the system.
- **Memory Management:** It efficiently handles memory management, thereby providing the freedom to develop our apps.
- **Process Management:** It manages the process well, allocates resources to processes whenever they need them.
- **Network Stack:** It effectively handles network communication.
- **Driver Model:** It ensures that the application works properly on the device and hardware manufacturers responsible for building their drivers into the Linux build.

What is an Application Framework?

- An Android application framework is a software toolkit that enables app developers to piece together a finished product that meets the requirements of its proprietor.
- A framework provides the bones of an application, to be fleshed out with graphics, animation, special features and functionality.
- Application frameworks are designed to simplify the app development process, and make it easy to manage, modify and fix bugs down the road.
- It's necessary to distinguish between SDK (*software development kit*) and frameworks. SDK contains software development tools, including frameworks.
- Different Android app development frameworks share many of the same attributes, along with unique features that make them better suited to certain types of projects.
- The application framework you select should be chosen with the specific requirements of your project in mind.

With so many Android app frameworks available, finding those that stand apart and meet the needs of developers and stakeholders can be challenging. We used the following criteria to sort the sheep from the goats:

- **Cross-platform capability:** Cross-platform frameworks enable developers to use the same codebase across all operating systems, meaning a single app will have the same look, feel and user experience on any device. Plus, any updates and fixes are universally reflected automatically. Even if you are primarily interested in creating an Android app, using a framework designed to accommodate multiple platforms can save time and money down the road.
- **Popularity among developers:** App developers want to work with frameworks for Android development that facilitate the creative process, with minimal speed bumps along the way. They also want to be able to test, troubleshoot and fix bugs in the least complicated and most time-efficient ways.
- **User interface:** As apps become more sophisticated and complex, being able to customize framework elements to achieve the best user experience for every project is fundamental to high-end application development.

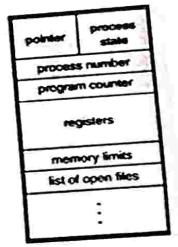
B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Android process management

Android process management is similar to that of Linux at a low level, but the Android Runtime provides a layer of abstraction to help keep often used processes in memory as long as it can. This is done using some memory management techniques that are not common.

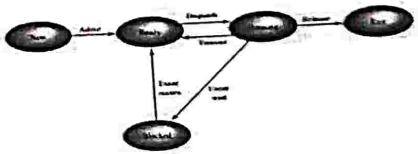
In this article I investigate the way that processes are managed and how the actual startup of a process differs from a typical operating system. This involves using an existing process called the zygote, which is at the most basic level a "warmed-up" virtual machine. I will also investigate when and how processes are finally killed.

Process management in a typical operating system involves many complex data structures and algorithms, but doesn't go much beyond the level managing the typical process data structure. Android is similar in that at the base level the control structures look the same. Similar to this:



Process Control Block

This data structure is managed by a standard process management, which is something like this:



An Android application framework is a software toolkit that enables app developers to piece together a finished product that meets the requirements of its proprietor. A framework provides the bones of an application, to be fleshed out with graphics, animation, special features and functionality. Application

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

Frame works are designed to simplify the app development process, and make it easy to manage, modify and fix bugs down the road. It's necessary to distinguish between SDK (*software development kit*) and frameworks. SDK contains software development tools, including frameworks.

Different Android app development frameworks share many of the same attributes, along with unique features that make them better suited to certain types of projects. The application framework you select should be chosen with the specific requirements of your project in mind.

Mobile applications are becoming increasingly sophisticated, interactive, and user-friendly as companies line up to join the mobile revolution. To meet demand, application framework developers have stepped up their game, providing superior technologies for app development.

Android App Development Framework

1. Flutter

Overview:

Flutter is Google's contribution to cross-platform application frameworks. Its first iteration, called Sky, was designed to run on the Android operating system. Flutter was released in 2017, making it one of the newer application development frameworks.

2. React Native

Overview:

Face book created React Native in 2015 as an open-source, cross-platform framework. It can be used to develop apps for iOS, Android, UWP and Web. With React Native, developers can build mobile apps using React and JavaScript, along with native Android app development capabilities.

This is the best Android app development framework in TATEEDA GLOBAL's opinion. Some of the best applications developed with React Native are Airbnb and Discord.

3. Xamarin

Overview:

Founded in 2011, Xamarin is owned by Microsoft and has been around for nearly a decade, making it one of the oldest application development frameworks. The platform provides developers with libraries and tools specifically compiled for building apps on Android, iOS, macOS, and others.

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com

4. Apache Cordova

Apache Cordova (formerly PhoneGap) is a free and open-source Android framework that enables hybrid app development in many basic web-development languages and technologies, like JavaScript, HTML5, and CSS3. It allows building multi-platform apps with a single code base, e.g.Appcelerator Titanium.

5. Ionic

Ionic is an MIT-certified, free, and open-source app development framework that uses a combination of HTML5 + CSS3 + JavaScript for cross-platform application development. It allows developing interactive hybrid apps and PWAs (Progressive Web Applications.)

B. SUDARSHAN M.SC (CS), B. ED, D.EL. ED
gaddamsudarshan91@gmail.com