CHAPTER 7: MANAGE OPERATING SYSTEM

Unit of learning code: ICT/CU/IT/CR/6/6

Related Unit of Competency in Occupational Standard: Manage Operating Systems

7.1 Introduction to the unit of learning

This unit covers competencies required to manage operating system. It involves Identifying fundamentals of operating system**,** Identifying concepts of process management concepts**,** Identifying concepts of Memory management**,** Identifying concepts of Input and Output devices**,** Identifying concepts of file management**,** Identifying Emerging trends in Operating system.

7.2 Summary of Learning Outcomes

1. Identifying Fundamentals of operating system
2. Identifying concepts of Process management concepts
3. Identifying concepts of Memory management
4. Identifying concepts of Input and Output devices
5. Identifying concepts of file management
6. Identifying Emerging trends in Operating system

7.2.1 Learning Outcome 1: Identifying fundamentals of Operating System.

7.2.1.1 Introduction to the learning outcome

It is critical for an ICT technician to master the tenets of Operating system. Ideally a computer system consists of programs and hardware components. Operating system is the main program that manages all of the hardware components and all other applications installed in the computer. Specifically, it controls every hardware device, main memory, every file, processing time, the computer users among others. In other words, without operating system the hardware component will not function. Understanding the fundamentals of Operating systems (Basic OS terms, concepts of OS, Structures of OS, Types of OS and functions of OS) serves as a good foundation for any ICT technician to be.

7.2.1.2 Performance Standard

7.2.1.1 Definition of Operating system is done

7.2.1.2 Concepts of operating system are identified.

7.2.1.3 Structures of operating system are described.

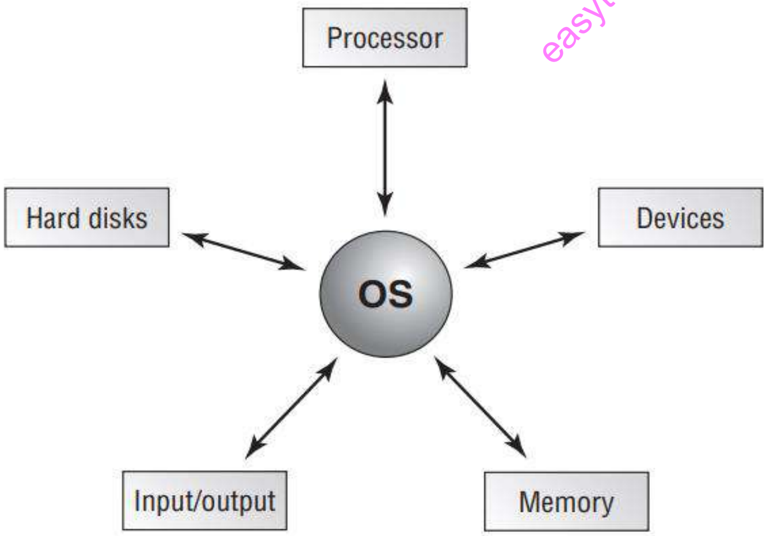
7.2.1.4 Types of operating system are identified.

7.2.1.5 Functions of operating system are identified.

7.2.1.3 Information Sheet

An ***Operating system*** is a set of instructions/program that manages a computer's hardware. It acts as an intermediary between the computer hardware and the user by providing a user interface through which one can interact with the hardware component. It also provides a basis for application programs to run. Majorly Operating systems are designed to be ***convenient*** or to be ***efficient*** or combination of the two (Abraham, 2013). OS is the centre through which the system hardware, other software, users, components work together.

Figure 184: OS vs Components



Source (CompTIA, 2013)

Identify concepts of Operating System

Operating systems are normally unique to their manufacturers and the hardware in which they are run. Modern sophisticated devices require that operating systems meet certain specific standards, consequently OS must perform the following main functions: -

1. Control of I/O operations
2. Job management
3. Memory management
4. Resource management
5. Error recovery

Operating system is a large and complex set of instruction created piece by piece. One can view it from several perspectives for example, one can focus on the services that the system provides or on the interface available to the users or its components and their interconnections. Most of the OS provides basic concepts and abstractions like processes, files and address spaces which are key for an ICT Technician to understanding them.

Processes

A process refers to a program in execution. It can also be looked at as a container that holds all the in information needed to run a program. All processes are associated with an ***address space*** and a list addressable memory location from 0 to a maximum value which can be read and written to by the process. The address space constitutes the program data, the executable program and the program stack. Each process requires a set of resources such as registers, list of open files, list of related processes and all other information needed to run a program successfully. Ideally a process is more than a program code (a combination of Program Code, Data and Execution status).

Files

All information stored in a computer must be in a file. There exists a number of file types such as; data files, program files, directory files, text files etc. Each file type stores a different type of information for example text files stores textual data while program files stores programs. thus, a file is simply a collection of data or information that has a name/identity usually called filename

System call

System calls are used for hardware services, to create or execute a process and for communicating with kernel services. It is a technique used by programs to interact with the Operating system. Ideally, a computer program makes a system call whenever it makes a request to the kernel.

Shell

A shell is a command line interface that enables the user to interact with the computer. Examples of shells include MS-DOS (For Windows Operating Systems), Command.com, csh, ksh and sh for (Unix operating Systems)

Kernel

Is the first portion of the operating system to load into a protected area in the memory so as not to be overwritten. It is responsible for key functions of the Operating system such as process management, memory management, file management, interrupt handling and disk drive management to mention a few.

Virtual Machines

VM is a software program that enables a host machine to support different operating systems just like a separate computer. It is created within another computing environment (host)

Evolution of operating systems

General Motors designed the first operating system in 1956 to operate a single IBM (International Business Machine) mainframe computer. Other owners of IBM mainframes followed suit and developed their own operating systems.

The earliest operating systems differed wildly from one computing device to the next and although they made it simpler to write programs, they did not allow programs to be used without a complete rewrite on more than one mainframe machine. IBM, the first computer manufacturer to develop an operating system began distributing operating systems with their computers in the early 1960s.

Other vendors such as GE, Digital Equipment Corporation, Xerox, Burroughs Corporation also released mainframe operating systems in early 1960s too.

In the late 1960s, the first version of the Unix Operating system was developed using C language. This version was freely available in the earliest stages and easily ported to new systems. Unix was rapidly accepted in the market. In fact, many modern OS such as Apple OS X and all Linux flavours trace their roots to Unix.

Microsoft Inc developed their version of operating system to run its range of personal computers in response to IBM development of their Operating System. The first OS built by Microsoft in 1981 was called MS-DOS. The name Windows was coined in 1985 when a graphical user interface (GUI) was created and paired with MS-DOS. Apple OS X, Microsoft Windows and the various forms of Linux command the vast majority of the modern Operating system market.

First Generation (1940’2 to early 1950’s)

The first computers were manufactured without any operating systems. In fact, they were manufactured to perform generally simple math calculations. All programming was done purely in machine language by wiring of plugboards to control the machine’s functions. Generally, the computers in generation were for solving simple math calculations.

Second Generation (1955 – 1965)

In the early 1950s, an operating system was introduced, called GMOS by General Motors produced for the 701 IBM computer. Since the data submitted was in batches, operating systems in the 1950's were called single-stream batch processing systems. The machines in this generation were referred to as mainframes and were used by professional operators in large computer rooms such as government agencies. The machines were costly.

The Third Generation (1965- 1980)

The computers manufactured during this era supported multiprogramming i.e. multiple jobs were executed simultaneously. These new developments allowed CPU to be busy nearly 100 percent of the time it was in use.

Minicomputers were introduced during this era and they created a whole new industry and the development of more portable computers.

The Fourth Generation (1980- Present day)

Manufacturing of personal computing device occurred during this time. These computers were very similar to the minicomputers developed in the third generation. The personal computers were affordable compared to the previous generations. Bill Gates and Paul Allen had a vision to take personal computing to the next level. They introduced MS-DOS (Microsoft Disk Operating System) in 1981 very effective but challenging to understand its cryptic commands to majority of users. Microsoft released several versions of Windows operating system such as Windows 95, Windows 98, Windows XP, Windows 7, Windows Vista, Windows 10 etc.

In 1980's, Steve jobs co-founder of Apple, created Apple Macintosh which was user friendly compared to MS-DOS.

In later years, a strong rivalry between the two companies developed however windows OS was rapidly developed and currently running in virtually all devices such as laptops, smartphones, ATM machines, motor vehicles and all our electronic devices. As the technology progresses so is the operating system.

Structures of Operating System

Earlier operating systems like UNIX and MS-DOS lacked a well-defined structure. For instance, absence of CPU Execution Mode (for user and Kernel) lead to system crashes in the event of errors. The OS structure can be described using the following six designs; monolithic systems, layered systems, microkernels, virtual machines, exokernels and client-server systems.

1. Monolithic systems

It is the oldest and most common architecture where operating systems run as a single program in kernel mode. The OS resides on kernel for anyone to execute. The functionality of the Operating Systems is invoked with simple ***function calls*** (system calls) within the kernel as a single large program.

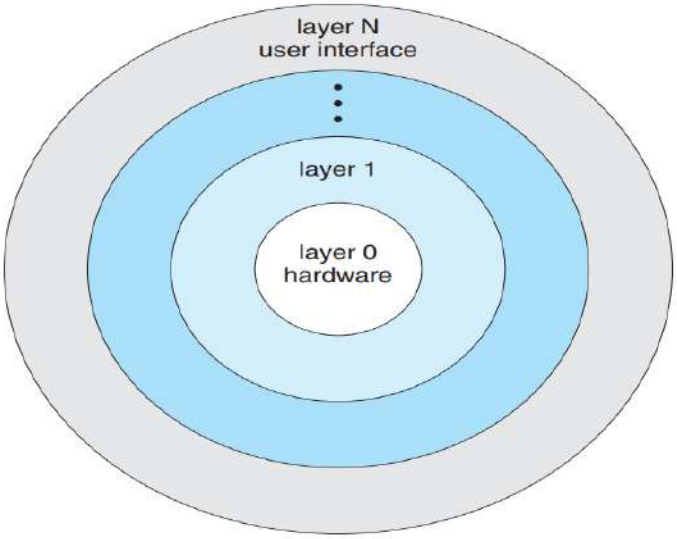
Many CPU has two modes of operation; the ***kernel mode*** (allows all the operating system instructions) and the ***user mode*** (for user program in which input/output devices are allowed/not allowed). The I/O device drivers are loaded into the kernel and become part and parcel of the kernel.

2. Layered Architecture

The layered approach was developed in 1960's. This architecture splits the OS into different layers thus increasing the modularity of the OS. The hardware layer serves as the bottom layer while the user interface serves as the highest layer. The layers are arranged in a way that each user functions and services of lower level layers only. Implementers can change the inner workings of the OS using the external interface of the routines. Once the 1st layer is debugged, its correct functioning can be assumed while the second layer is debugged up to layer N (Abraham P. G., 2013). In the event of an error during debugging process of a particular layer, the error is assumed to be on that layer since the layers below it is already debugged.

This approach has the advantage of simplicity of construction and debugging by programmers. However, the main disadvantage of this approach is it makes the OS less efficient compared to other implementations and it requires an appropriate definition of the various layers.

Figure 185: Layered OS Architecture



Source (Stallings, 2014)

3. Microkernels

This architecture structures the system by removing all non-essential parts of the kernel and implements them as system and user level programs thus a smaller kernel. This architecture has the advantage of providing a minimal process, memory management and a communication facility.

Microkernel provides communication between the client program and the various services running in user space. Message passing technique is used to provide communication in the OS. For instance, if a client program wants to access a file, it must interact with the file server. The client program and service never interact directly but indirectly through exchange of messages with microkernel.

This approach makes it easier to extend the operating system. All new services are added to user space and consequently do not require the kernel to be modified. The main advantage of this architecture is that it provides more security, reliability is easier to port from one hardware design to another.

4. Virtual machines

Virtual machine simulates the existence of a real machine within a physical machine. This architecture enables one to install different operating machines within the virtual environment. It supports running of several operating systems at once each on its virtual machine. Several concepts are used to ensure multiprogramming is implemented in virtual environment and it includes the control program (to create the virtual machine environment), conversation monitor (to support system application features), remote spooling communication system – RSCS (to enable virtual machine to transmit and receive file in distributed system) and the Interactive Problem Control System - IPCS

Table 46: Virtual memory architecture

User Program 1		User Program 1
Operating System 1		Operating system N
Virtual Machine		
Hardware		

5. Client/Server Model in Operating system

The Client/Server model supports implementation of most of the OS functions in user processes to request a service from a server. The server then responds accordingly. This model is a variation of the microkernel system where servers are located in the middle layer.

6. Exokernels

This architecture was developed in MIT (Massachusetts Institute of Technology). It provides. It provides protected access to the hardware through the use of abstraction layers and by putting everything into libraries. The main advantages of exokernels include explicit allocation of resources, separate protection and management, expose information among others. Complexity in design of exokernel interfaces and less consistency serves as its main drawback.

Identify types of Operating System

The major types of operating system include: -

Real-Time Operating System

It's a type of operating system in which the time interval to process and respond to inputs **(response time)** is so small that it controls the environment. These systems are used in situations where there are rigid time requirements on the operation of a processor or the flow of data and real-time systems can be used as a control device in a dedicated application. Real-time operating system is applied on medical imaging systems, air traffic control systems, robots, industrial control systems, scientific experiments, etc. Real-time systems can be **hard real-time systems** (guarantees that critical tasks complete on time)

Batch Operating system

This type of OS entails having an operator that takes similar jobs having same requirement and group them into batches for faster processing of jobs. The users of a batch operating system do not interact with the computer directly. Each user prepares their task on an off-line device such as punch cards and submits it to the computer operator. Then main draw backs of batch systems include; difficult in providing the desired priority, lack of interaction between the user and the job and CPU idle item.

Distributed Operating System

This type of operating system utilizes multiple central processors to serve multiple real-time applications and multiple users. The data processing jobs are distributed among the processors using various communication lines e.g. telephone lines and high-speed buses commonly referred to as loosely coupled systems (distributed systems). The processors sometimes referred to as sites, nodes etc may vary in size and functions.

The advantages of distributed systems include: -

- ✓ Reduction of delays in data processing
- ✓ In the event of one site failing in a distributed system the remaining sites will continue operating.
- ✓ Faster exchange of data
- ✓ Resource sharing facility

Time-sharing Operating System

This type of operating system enables many users located in different terminals to use a particular computer system at the same time. Processor’s time is shared among multiple users concurrently. The main objective of time-sharing OS is to minimize response time. Multiple jobs are executed by the CPU through switching between jobs frequently. This ensures user can submit their command and the response time is in few seconds.

Timesharing has the advantage of:

- ✓ Reducing CPU idle time
- ✓ Avoid duplication of software
- ✓ Quick response time

Its main drawbacks include:

- ✓ Data communication problems
- ✓ Security and integrity of user programs and data
- ✓ Reliability issues

Network Operating System

This type of operating system runs on server and it enables the server to manage networking functions in a networked environment. It allows users, groups, data, security, application to be shared in a Network. Examples of network operating system include Microsoft Windows Servers, Unix, Mac OS X, Novell Netware, Linux and BSD.

The main advantages include:

- ✓ Strengthened security
- ✓ Easy to upgrade new technologies
- ✓ Hardware can easily be integrated into the system
- ✓ Provides remote access to servers from different locations irrespective of the type of system.

The disadvantages include: -

- ✓ It requires regular maintenance and updates.
- ✓ Its costly to acquire and run a server system.
- ✓ When the central device fails (server) operations stops.

Identify functions of Operating System.

An Operating System apart from controlling the execution of application programs it has a number of functions which can be thought of as having three main objectives, namely, Efficiency, Convenience, and ability to evolve (its dynamic nature with respect to allowing effective development, testing and introduction of new system functions). The OS provides the following functions broadly viewed as user view and system view: -

- ✓ **Provides user interface.**

  Through the Operating System the user is able to interact with the hardware device. Earlier versions of OS had command-driven interface (Command Line Interface-CLI). However, with invention of MS-Windows every operating system provides GUI (Graphical User Interface).

- ✓ **Manage I/O Operations**

The OS manages input/output operations thus relieves the user from details of every input or output devices. In other words, it relieves the user from details of input or output devices connected in the computer system.

- ✓ **Manage file systems.**

In computer language a **file** is a logical concept that stores the user's data or program. Files are created using editor. The Operating System manages all the files created on the computer system by enabling tasks such as saving the file, retrieving the file and accessing the file conveniently etc.

s

- ✓ **Error control and detection**

When working with a computer, different types of errors are prone to be encountered. It's the duty of the Operating System to detect hardware errors as well as software errors. Memory access failure or sometimes device failure occur. Irrespective of the type of error it's the responsibility of the Operating System to detect the error and give an alert to the user regarding the error.

- ✓ **Resource management**

Computer resources are scarce and needed by most devices simultaneously. The OS performs resource allocation especially in the event of multiple processes running. Process creation, deletion is handled by OS. For a process to run, resources such as memory and CPU are required. It is the job of OS to source for available memory and CPU and allocate to the process in need of the resource. OS manages Inter Process Communication (IPC) and synchronization i.e. a situation where multiple processes are simultaneously running and require to communicate with other process for easy access to computer resource.

- ✓ **Storage management**

Main memory cannot accommodate every program or data. Secondary storage devices are required to augment the main memory in a computer system. The OS manages free space on storage devices; dictates how the devices will be accessed, how swapping will be done, performs disk scheduling techniques etc.

✓ **Process management**

A process is a unit of work in a computer system. It also known as a program in execution. An OS schedules processes and threads on the CPUs, creates and deletes system and user processes, suspending and resuming processes, process synchronization and inter-process communication.

✓ **Memory management**

Main Memory is a repository of accessing data shared by the CPU and input output devices quickly. CPU reads and writes instructions from main memory during the fetch execute cycle. For a program to be executed, it must be mapped to absolute addresses and loaded into main memory after which it will be executed and later on terminates to release the memory space for the next program to be executed.

The OS manages memory by keeping track of the parts of memory currently being used and by who, decides which processes and data to move into and out of memory and allocating and deallocating memory space in the computer system.

✓ **Program Control**

The OS controls all the programs running in the computer system. It protects one user’s program from another user’s program. The OS is capable of detecting any exception thrown by the system and notifies the user. In summary, the OS controls all the programs, user activities, system activities and I/O access.

✓ **Virtual Machine Manager**

OS acts as an abstraction that hides the complex details of hardware from users. It is very clumsy to interact with I/O devices, it is for this reason that the OS provides a layer on the actual hardware device on which it performs tasks for the user such that the user feels all the tasks are done by the hardware.

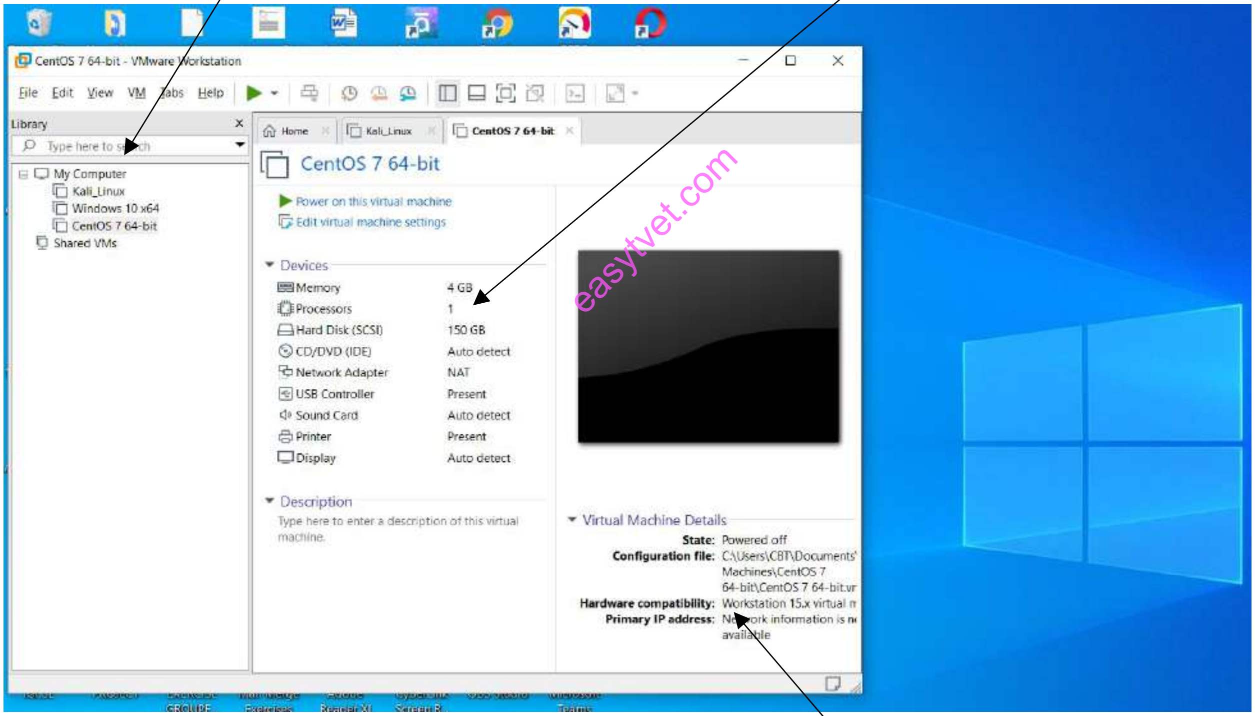
The OS implements virtual devices to lessen the burden of working with a lot of physical machines. The virtual machine also known as extended machine is implemented in the form of virtual processors, virtual memory and devices derived from the physical computer. Each user has an illusion that they are using a single machine.

Diagram shows an example of implementation of virtual machine using VM ware workstation running on a MS-Windows 10 Operating System. The Virtual machine has Kali Linux, Windows 10 OS and CentOS Operating system installed in the virtual platform.

Figure 186: Virtual Memory Implementation on Windows 10

Virtual Machines Implemented on the Physical PC

Virtual hardware components


CentOS 7 64-bit - VMware Workstation
File
Edit
View
VM
Tabs
Help
Library
Type here to search
My Computer
Kali_Linux
Windows 10 x64
CentOS 7 64-bit
Shared VMs
Home
Kali_Linux
CentOS 7 64-bit
Power on this virtual machine
Edit virtual machine settings
Description
Type here to enter a description of this virtual machine.

CentOS 7 64-bit

Devices

Memory	4 GB
Processors	1
Hard Disk (SCSI)	150 GB
CD/DVD (IDE)	Auto detect
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect

Virtual Machine Details

State: Powered off
Configuration file: C:\Users\CBT\Documents\ Machines\CentOS 7 64-bit\CentOS 7 64-bit.vr
Hardware compatibility: Workstation 15.x virtual n
Primary IP address: Network information is n available

Virtual Machine Details

Source (VMware work station, 2020)

✓ **Installation of operating systems**

As a computer technician it is critical that one understanding the basics of installing, configuring, optimizing, and upgrading an operating Systems

Before installing an Operating System, you need to check the system requirements and confirm whether your device can support the installation. OS usually have specific revisions made to the original code identified by a number called ***Version.*** It tells you how new the product is in relation to other versions of the product. Very minor revisions are indicated with an additional decimal point on the revision number. ***Source code*** defines how a piece of software works. An Operating system can be open source (OS can be examined and modified by users) or closed course (Users cannot modify or examine the code)

General Steps in Installing Windows Operating System

Each version of Microsoft Windows is installed on a computer using similar steps. However, there are steps in the installation process that differ slightly between the versions of Windows.

Step One: Check hardware compatibility

Check the minimum hardware requirements and the compatibility test for your device. Microsoft provides a Windows compatibility product list for checking if the hardware in your computer is compatible with thee OS you want to install. Incompatibility might lead to unsuccessful installation of the Windows OS.

Step Two: Get a genuine MS-Windows CD, DVD, or USB thumb drive

Genuine copy of MS-OS installation disc with a product key is required at this step. The key is required to activate Windows after installation. If one has an OEM computer (it does not have a genuine windows CD, DVD or USB thumb drive), the Windows product key is positioned on the back or side of the computer. Additionally, you will reinstall Windows and all the necessary software using a hidden partition or a set of restore discs.

Step Three: Open the BIOS or CMOS setup

Insert the appropriate disk on the computer and change the boot order of the computer to either CD, DVD disc drive or USB thumb drive as the first boot device. Once inserted the BIOS will recognize and include the disk in the list. Save the changes and exit the BIOS.

Begin the Windows installation process by turning on the computer. Once the installation disk is detected, a message “Press any key to boot from CD/DVD” will appear press any key on the keyboard to have computer boot from the installation disk or drive.

Step Four: Select the type of Installation

One can perform a full installation, a repair or an upgrade installation. If installing for the first time chose Full installation, if it is to rectify an already installed machine select repair, if it is to transit to a higher/newer version of OS then select Upgrade option.

Step Five: Select the partition to install Windows onto

Select the main partition usually drive C or any other drive labeled “Unallocated partition”. However, if upgrading, select the existing installation of Windows on the hard drive. You will be prompted to select the option of “Format the hard drive to allow Windows OS to be installed”. Follow the basic steps and provide appropriate information when prompted such as configuring time zone, preferred language, name of the account you will use to access Windows

Step Six: Installation process completion

Once the installation process is complete, the computer will prompt you to log in after loading the Windows OS. You are now free to install necessary drivers to support necessary hardware.

General Steps in Installing Linux Operating System

The steps involved in installing Windows apply for Linux installation with slight variations.

7.2.1.4 Learning Activities

Learning Activity 1

You have been tasked to prepare a brief on the Operating systems used in your College computer laboratory. The management would like to know every basic detail regarding the operating system installed. Prepare a 1-page document that shows the computer name you were working on, the Operating system installed, its version and build, language configured, system manufacturer, system model, BIOS version, Processor type and speed, page file and RAM capacity.

- *Knowledge in relation to Performance Criteria given as content in the curriculum*

The trainee is expected to research and run ***dxdiag*** command in order to mine information regarding the performance criteria for this learning outcome.

- *Special instructions related to learning activities*

The learning activity can be used before teaching the learning outcome; trainees will be able to: Demonstrate their understanding of the operating system concepts and types of operating system.

OR

It can be used at the end of teaching the learning outcome; trainees will be able to:

1. Interact with a computing device and
2. Demonstrate their understanding of the operating system concepts and types of operating system.

Learning Activity 2

Case Study of MS Windows vs Linux

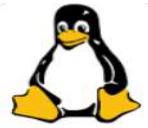
The trainer to implement virtual PC and install Linux OS as well as Windows OS together with the trainees. During the installation, the trainer should emphasize on the following aspects: -

a) Interface designs of the two Operating Systems
b) Memory management of the two Operating Systems
c) Performance of Linux OS as well as Windows OS
d) Security of Windows OS as well as Linux OS

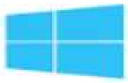
7.2.1.5 Self-Assessment

1. Associate the following icons with their appropriate Operating systems

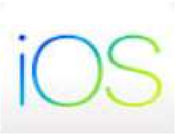
1.



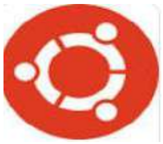
2.



3.



4.



2. When a user wants to open and work simultaneously on many windows on his system, what type of OS should be chosen? **………………………..**

3. Program which reads and interprets control statements and passes the signals to operating system is known as…………………….

4. Examine the following conditions and find appropriate operating system for them:

   a. In a LAN, users want to share some costly resources like laser printers.
   b. Multiple users on a system want quick response on their terminals.
   c. Washing machine.

4. Using relevant examples in each case distinguish System software and Application software

7.2.1.6 Tools, Equipment, Supplies and Materials

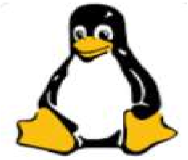
A functional computer with an installed OS

The Personal Computers should be connected to the Internet for research purposes

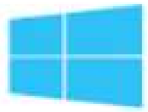
7.2.1.7 model answers to self-assessment

A. Associate the following icons with their appropriate Operating systems

1.



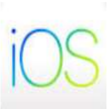
2.



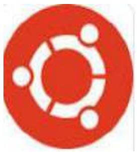
Windows

Linux

3.



4.



Ubuntu

iPhone OS

B. When a user wants to open and work simultaneously on many windows on his system, what type of OS should be chosen? ***Multi-tasking***

C. Program which reads and interprets control statements and passes the signals to operating system is known as…………… ***shell***

D. Examine the following conditions and find appropriate operating system for them:

   d. In a LAN, users want to share some costly resources like laser printers. – **Network Operating System**

   e. Multiple users on a system want quick response on their terminals. – **Real time OS**

   f. Washing machine **– Embedded OS such as Tizen in Linux**

E. Using relevant examples in each case distinguish System software and Application software

   System software is any computer software which manages and controls computer hardware so that application software can perform a task. Operating systems, such as Microsoft Windows, Mac OS X or Linux, are prominent examples of system software.

   Application software are programs that enable the end-user to perform specific, productive tasks, such as word processing or image manipulation.

7.2.1.7 References

Abraham, P. a. (2013). *Operating Systems Concepts.* United States of America: Wiley.

Abraham, P. G. (2013). *Operating Systems Concepts* . Hoboken: John Wiley & Sons, Inc.

7.2.2 Learning Outcome 2: Identifying concepts of process management.

7.2.2.1 Introduction to the learning outcome

A computer technician needs to understand how an operating system and application program works. To do so, one needs to grasp the concept of a process since, there are a number of user and system processes to be managed in a computer system.

One can view a process from two separate and potentially independent concepts: one relating to resource ownership and the other one relating to execution (Stallings, 2014). Resource ownership is where a process includes a virtual address space to hold the process image (a collection of data, program, stack and attributes defined in the PCB). A process controls or owns resources such as main memory, I/O channels, I/O devices and data for a while during execution.

Execution of a process entails following an execution path through one or more programs. This execution can be connected to other processes. It is evident that a process requires a resource during execution and in a computer system there are more than one process being executed thus the need to understand process management.

7.2.2.2 Performance Standard

7.2.2.2.1 Concepts of processing are identified and explained.

7.2.2.2.2 Process states are described.

7.2.2.2.3 Definition of ***Concurrency control*** and types is done.

7.2.2.2.4 Explanation of Process scheduling and types of schedulers is done.

7.2.2.2.5 Definition of Deadlocks.

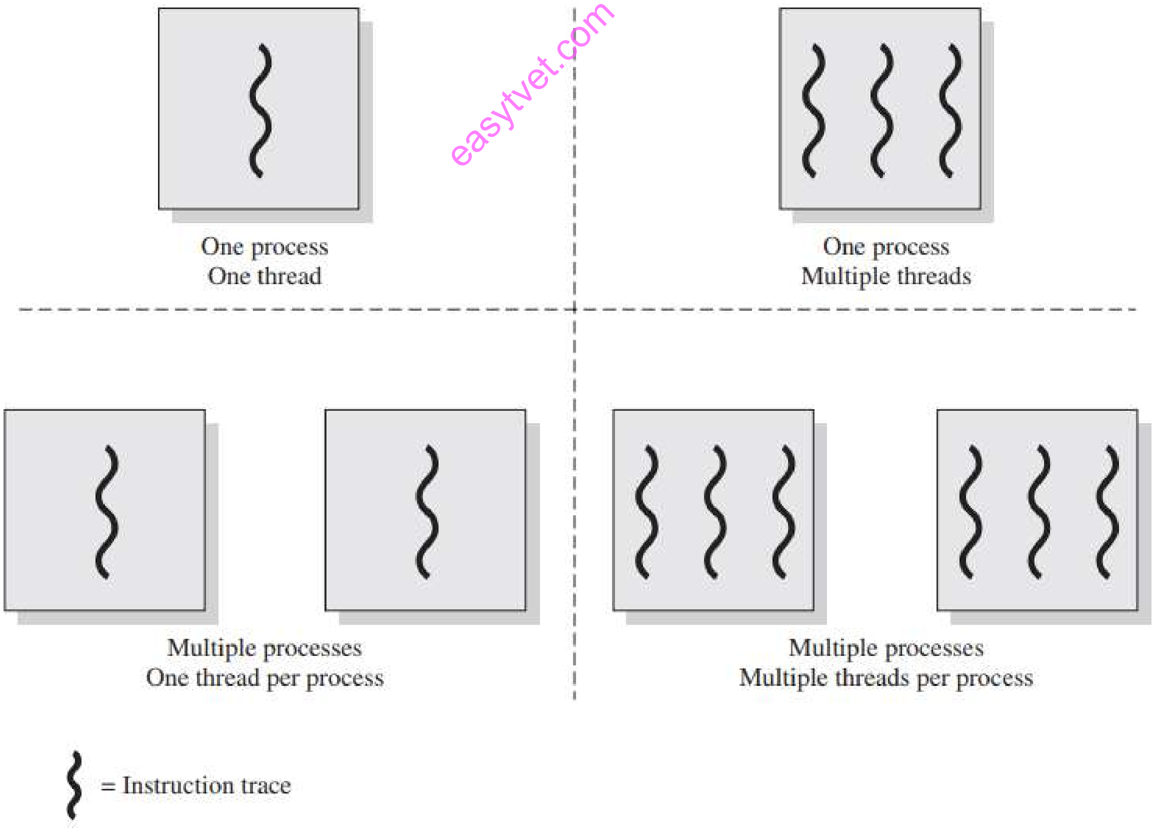
7.2.2.3 Information Sheet

Identify and explain concepts of processing

Process management entails creating and deleting processes and also providing mechanisms for processes to communicate and synchronize with each other in a computer system. ***A process*** therefore can be referred to as a job or the fundamental unit of work in an operating system. A job is simply a sequence of single programs. Interestingly the term ‘program’ and ‘job’ are used interchangeably. According to (Stallings, 2014), a ***thread*** is referred to as a unit of dispatch or a lightweight process. On the other hand, a process or a task is referred to as the unit of resource ownership

In the event the operating system is required to support multiple concurrent paths of execution within a single process it is referred to as **multithreading**. MS-DOS supports a single user process and a single thread. Other OS supports multiple user processes but only support one thread process at a time.

Figure 187: Threads and process using a diagram



Source (Stallings, 2014)

Process control block

For an OS to implement and control processes it requires some attributes associated with the process. These attributes are stored in a data structure called ***Process Control Block*** (PCB) also referred to as process descriptor. The collection of user program, stack, data section and the attributes form a ***process image.***

Whenever a process is created in the computer system the PCB is created too since it holds all information about the process needed for its control. The PCB contains: -

Process state

When a program is loaded into the memory and it becomes a process, it can be divided into four sections namely stack, heap, text and data.

Stack contains temporary data such as method/function parameters and return address; **Heap** is a dynamically allocated memory to a process during its runtime; **Text** contains current activity represented by the value of PC (Program Counter) and contents of the processor's registers; **Data** contains global and static variables

A process exists in various states such as New, Ready, Running, Waiting, Halted

Program counter

It indicates the address of the next instruction to be executed for that particular process.

CPU registers

Accumulators, index registers, general-purpose registers, stack pointers comprise the CPU register. In the event an interrupt occurs CPU registers and program counter information must be saved to allow a process to be continued correctly

CPU scheduling information

It contains scheduling parameters such as process priority, pointers to scheduling queues etc.

Memory management information

It contains information such as limit registers, value of the base and the page tables/segment tables.

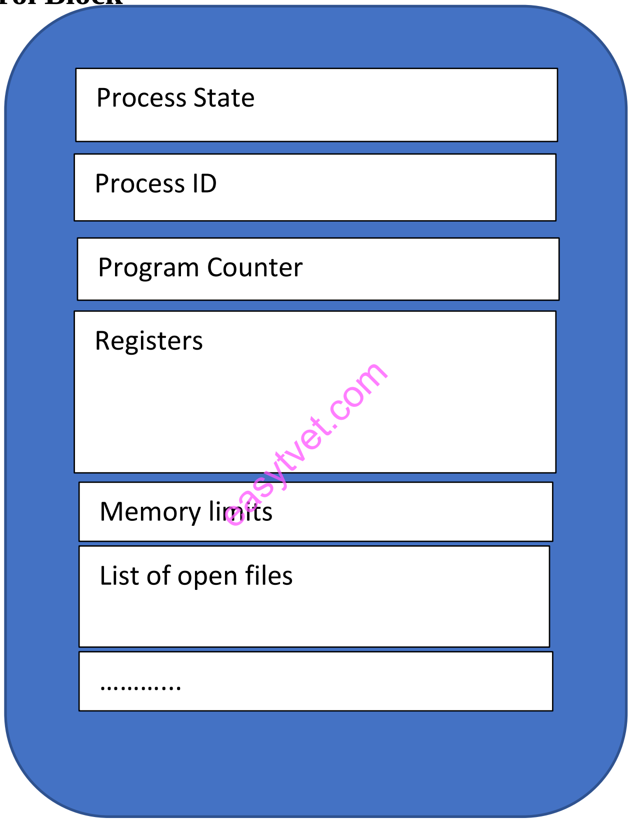
Accounting information

It contains the amount of CPU and real time used, account numbers, time limits, job or process numbers etc.

I/O status information

It contains the list of I/O devices allocated to a particular process and a list of open files etc.

Figure 188: Process Control Block



Process state

A process/task is created for a program that needs to be executed. The CPU executes instructions in sequence dictated by the changing values in the program counter register. A ***trace*** is referred to as listing the sequence of instructions that execute for that particular process. The life of a process is bounded by its creation and termination. As a process executes, it changes state as defined by the current activity of that process. The changing process states of a typical activity from creation to termination is as shown in figure. Typically, activities can be removed from memory when they are in the waiting, stopped or terminated states.

It is important to note that only one process can be running on any processor at any instant even though processes may be ready and waiting. A process may exist in one or more of the following states:

New: This is the state where a process is created. It is the transient state when the activity has just begun.

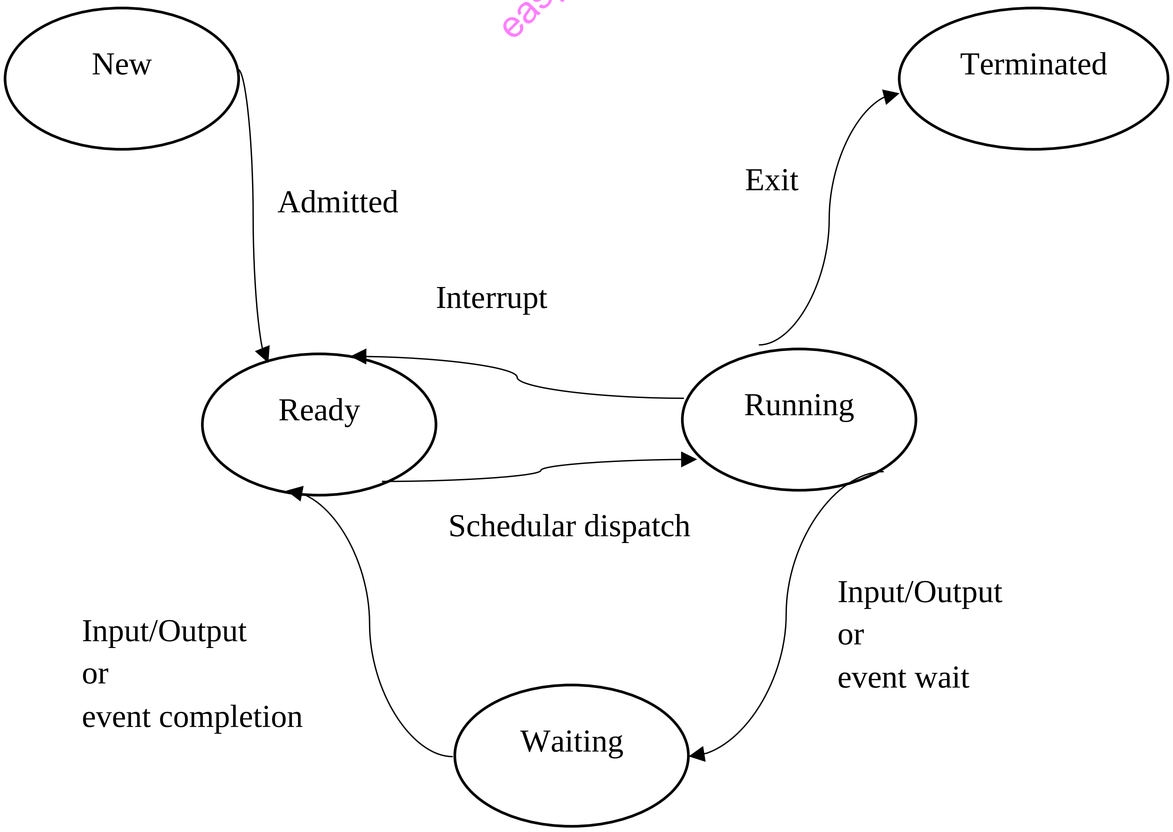
Running: This state entails execution of instructions. Activities execute until they are interrupted by another activity or user command. It is also referred to as Resumed state.

Waiting: It is a stop for an activity that is interrupted and ready to go into background mode. The process is waiting for some event to occur for example I/O completion, or user starts another application.

Ready: The process is waiting to be assigned a processor

Terminated: The process has completed the execution. Ideally the activities in this state disappeared from the user's view.

Figure 189: Diagram of Process State



Concurrency control and control types

Inter-process communication

Inter-process communication (IPC) facility provides a mechanism to allow processes to communicate and to synchronize their actions to prevent ***Race conditions*** *(a condition where several processes access and manipulate the same data concurrently and the outcome of the execution depends on the particular order in which the access takes place).*

To guard against this; certain mechanisms are necessary to ensure that only one process at a time can be manipulating the data. The mechanisms:

1. Critical Section Problem.

It is a segment of code in which a process may be changing common variables, writing a file, updating a table etc. Each process has its critical section while the other portion is referred to as Reminder section.

The important feature of the system is that when one has process is executing in its critical section no other process is to be allowed to execute its critical section. Each process must request permission to enter its critical section. A solution to critical section problem must satisfy the following:

- ✓ **Mutual Exclusion** ensures that if a process is executing its critical section, then no other process can be executing in their critical section.
- ✓ **Progress** ensures that if no process is executing in its critical section and that there exist some processes that wish to enter critical sections then those processes that are not executing in their remainder section can be allowed.
- ✓ **Bounded Waiting** ensures there must exist a bound on the number of times that other processes are allowed to enter their critical sections after a process has made a request to enter its critical section and before that request is granted. This helps prevent Busy waiting.

  NB: Busy waiting occurs when one process is being executed in its critical section and process 2 tests to confirm if process 1 is through. It is only acceptable technique when the anticipated waits are brief; otherwise it could waste CPU cycles.

Techniques used to handle Critical section problem (Synchronization)

1. Semaphores

The main principle behind Semaphores is that two or more processes can operate by means of simple signals such that a process can be forced to stop at a specified place until it has received a special signal. Any complex coordinative requirement can be satisfied by the appropriate structure of signals. For signaling special variables called semaphores are used.

To transmit a signal via semaphores, a process executes the primitive wait; if the corresponding signal has not yet been transmitted the process is suspended until transmission takes place.

A semaphore can be viewed as a variable that has an integer value upon which three main operations are defined: -

1. A semaphore may be initialized to a non-negative value
2. The wait operation decrements these semaphore value. If the value becomes negative the process executing the wait is blocked.
3. The signal operation increments the semaphore value. If the value is not positive then a process is blocked by await operation is unblocked

A binary semaphore accepts only two values 0 or 1. For both semaphore and binary semaphore a queue is used to hold processes waiting on the semaphore. A strong semaphore has a mechanism of removing processes from a queue e.g. using First In First Out (FIFO) policy. A weak semaphore doesn’t specify the order in which processes are removed from the queue. Example of implementation:

2. Message passing

Is a technique that provides a means for co-operating processes to communicate when they are not using shared memory environment. System calls are used by processes generally to send and receive messages such as: *send (receiver process; message) or Receive (sender process; message).*

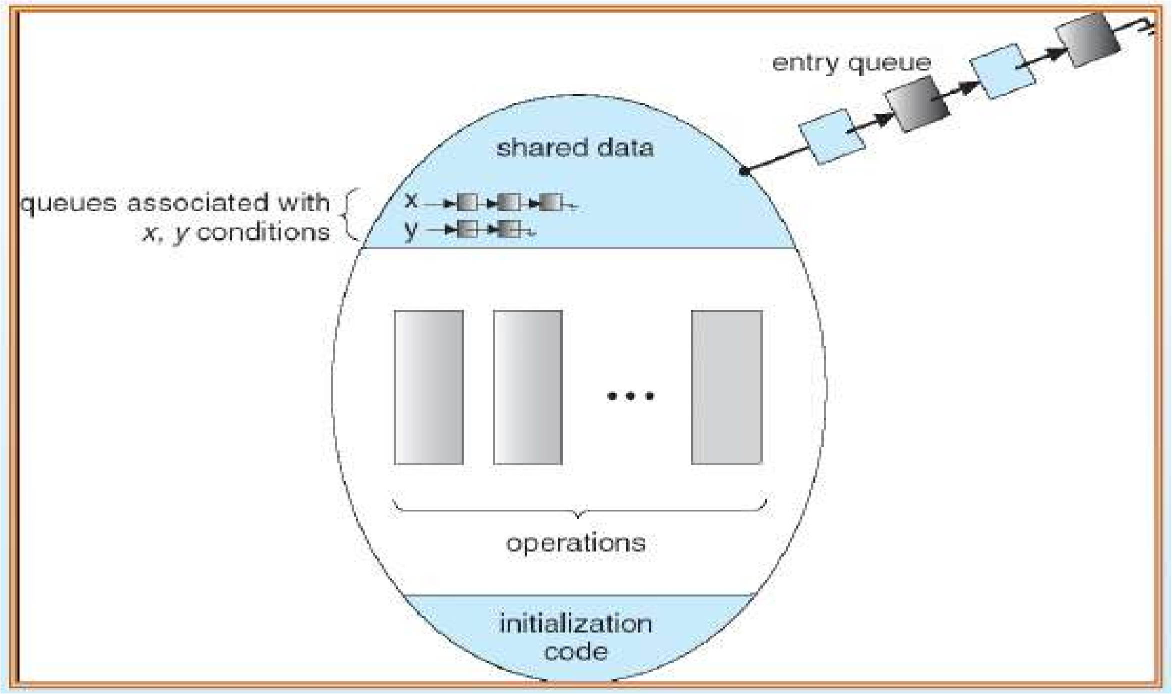
A blocking send must wait for receiver to receive the message. A non-blocking send enables the sender to continue with other processing even if the receiver has not yet received the message. This requires buffering mechanisms to hold messages until the receiver receives it. Acknowledgment protocols are used in distributed systems to pass message since communication can be flawed sometimes. However, on some personal computers passing can be flawless.

One complication in distributed systems with send / receive message passing is in naming processes unambiguously so that send and receive calls references the proper process. Process creation and destruction can be coordinated through some centralized naming mechanisms but this can introduce considerable transmission overhead as individual machines request permission to use new names.

3. Monitors

It is a construct of high synchronization characterized by a collection of operations specified by the programmer. It is made of declarations of variables and bodies of procedures or functions that implement operations of a given type. It cannot be used directly by the various processes; ensures that only one process at a time can be active within a monitor.

Figure 190: Schematic view of a monitor



Processes desiring to enter the monitor when it is already in use must wait. This waiting is automatically managed by the monitor. Data inside the monitor is accessible only to the process inside it. There is no way for processes outside the monitor to access monitor data. This is called ***information hiding.***

4. Event Counters

It is an integer counter that does not decrease. Introduced to enable process synchronization without use of mutual exclusion. It keeps track of number of occurrences of events of a particular class of related events. Some of the operations that allow processes to reference event counts include; ***Advance*** event, ***Read*** event and ***Await*** event values.

Advance event signals the occurrence of an event of the class of events represented by incrementing event count by 1.

Read event obtains value of E; because Advance (E) operations may be occurring during Read (E) it is only guaranteed that the value will be at least as great as E was before the read started.

Await event blocks the process until the value of E becomes at least V; this avoids the need for busy waiting.

Other techniques used include:

- ✓ Using Peterson's algorithm,
- ✓ Dekker's algorithm and
- ✓ Bakely's Algorithm

Process scheduling

Process scheduling is required to enable the CPU to execute one process at a time. There are various types of scheduler modules in the OS that execute at their appropriate time. Process scheduling is the process of switching the CPU among processes consequently making the computer more productive.

It is the basis of multi programmed Operating System. Normally several processes are kept in memory and when one process has to wait, the OS takes the CPU away from that process and gives the CPU to another process. The process scheduling modules of the OS include: -

- ✓ CPU Scheduler
- ✓ CPU dispatcher

CPU Schedular

The schedular selects a process from the *Ready* queue that is to be executed. Scheduling decisions may take place under the following conditions:

a) When a process terminates.

b) When a process switches from Running state to the Ready state (e.g. when an interrupt occurs)

c) When a process switches from the *Waiting* state to the *Ready* state (e.g. completion of I/O)

d) When process switches from *Running* state to *Waiting* state (e.g. Input/Output request).

Scheduling algorithm can either be:

(i) ***Non pre-emptive*** scheme where the process keeps CPU until when it releases it.

(ii) ***Pre-emptive*** scheme where the CPU can be taken from a process when it is in the middle of updating or processing data by another process.

Types of schedulers

The schedulers are classified according to their frequency of their use in the system. When a schedular is invoked frequently it is referred to as short-term schedular while when the use is after a long time it is referred to as long-term. Operating system has many schedulers but the three main schedulers are: -

SHORT-TERM SCHEDULER

The main objective of this type of schedular is to maximize CPU utilization. It allocates processes in the Ready queue to CPU for immediate processing. It works more frequently and faster because it selects a process quite often and executes the process only for a few seconds before it goes for input output operations.

MEDIUM-TERM

Occasionally the OS need to remove a process from the main memory as this process may be in need of some I/O operation. So, such processes may be removed (or suspended) from the main memory to the hard disk. Later on, these processes can be reloaded into the main memory and continued from where it was left earlier. This saving of suspended process is said to be rolled-out. And this swapping (in and out) is done by the medium-term scheduler.

LONG-TERM SCHEDULER

This scheduler is responsible for selecting the process from secondary storage device like a disk and loads them into the main memory for execution. At is also known as a **job scheduler.** The primary objective of long-term scheduler is to provide a balance of mix of job to increase the number of processes in a ready queue. Long term scheduler provides a good performance by selecting processes with a combination of I/O and CPU bound type. Long-term scheduler selects the jobs from the batch-queue and loads them into the memory. In memory, these processes belong to ready queue.

CPU Dispatcher

CPU dispatcher gives control of CPU to the process selected by the scheduler. Its functions include: -

- ✓ Switching to user mode
- ✓ Switching context

- ✓ Jumping to proper location in the user program to restart that program.

Scheduling levels

Three important levels of scheduling are considered.

a) High level scheduling (Job Scheduling) – determines which jobs should be allowed to compete actively for the resources of the system. Also called Admission Scheduling. Once admitted jobs become processes or groups of processes.

b) Intermediate level scheduling – determines which processes shall be allowed to compete for the CPU.

c) Low level scheduling – determines which ready process will be assigned the CPU when it next becomes available and actual assigns the CPU to this process.

Various CPU scheduling algorithms are available, thus in determining which is the best is for a particular situation, the following criteria have been suggested. They include:

(i) *CPU utilization* – keeps the CPU as busy as possible.

(ii) *Throughput* – it is the measure of work i.e. no of processes that are completed per time unit.

(iii) *Turnaround time* – The interval from time of submission to time of completion. It is sum of periods spent waiting to get into memory, waiting in the ready queue, executing in CPU and doing I/O.

(iv) *Waiting Time* – Sum of periods spent waiting in ready queue.

(v) *Response time* – Measure of time from submission of a request by user until the first response is produced. It is amount of time it takes to start responding but not time that it takes to output that response.

(vi) *I/O Bounded-ness* of a process – when a process gets the CPU, does it use the CPU only briefly before generating an I/O request?

(vii) *CPU Bounded-ness of* a process – when a process gets the CPU does it tend to use the CPU until its time quantum expires.

Scheduling Algorithms

Scheduling algorithms are either non-preemptive or preemptive. ***Non-preemptive*** algorithms are designed so that once a process enters the running state, it cannot be preempted until it completes its allotted time. Resources such as CPU cycles, are allocated to the process for the 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.

Preemptive scheduling is based on priority where a scheduler may preempt a low priority running process anytime a high priority process enters into a ready state.

When a method switches from running state to ready state or from waiting state to ready state, ***preemptive*** scheduling is used. That process stays in ready queue till it gets next chance to execute. Algorithms based on preemptive scheduling include: Shortest Remaining Time First (SRTF), Priority (preemptive version), Round Robin (RR).

Non-Preemptive scheduling is used when a process terminates, or switches from running to waiting state. Once the CPU has been allocated to a process, it will hold the CPU till it gets terminated or it reaches a waiting state. Algorithms based on this scheduling technique include; Priority (non- preemptive version) and Shortest Job First (SJF).

1. First Come First Served (FCFS) / First in First Out (FIFO) Scheduling

It is the process that request CPU first and is allocated the CPU first. Code for FCFS is simple to write and understand. Under FCFS Average waiting time is often quite long. FCFS is non-preemptive discipline (once a process has the CPU it runs to completion).

Example

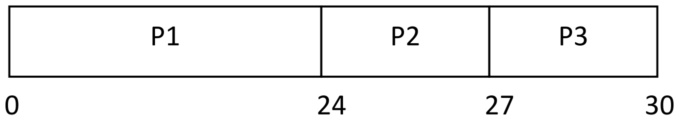
Processes arrive at time 0, with length of the CPU-burst time given in milliseconds.

Table 47: FCFS

Process	**Burst Time** **(Time of CPU use)**

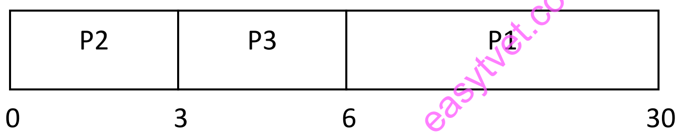
P1	24
P2	3
P3	3

If processes arrive in order P1, P2, P3; Using a Gantt chart.



Average waiting Time = (0 + 24 +27) / 3 = 17 ms

If processes arrive in order P2, P3, P1



Average waiting Time = (0 + 3 + 6) / 3 = 3 ms

From example one, the average waiting time under FCFS policy is generally not minimal and may vary substantially if processes CPU-Burst times vary greatly. FCFS algorithm is not ideal for time sharing systems.

2. Shortest Job First (SJF) Scheduling

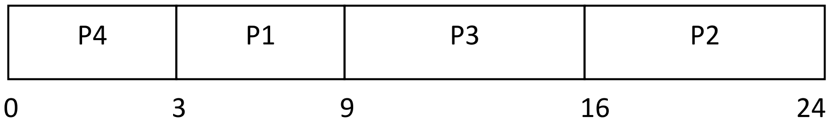
When the CPU is available it is assigned to the process that has the smallest next CPU burst. If two processes have the same length next CPU Burst, FCFS scheduling is used to break the tie. It reduces average waiting time as compared to FCFS.

Example

Table 48: SJF

Process	**Burst Time (ms)**
P1	6
P2	8
P3	7
P4	3

GANTT chart for SJF:



Average waiting Time = (0 + 3 + 16 + 9) / 4 = 7 ms

The main disadvantage of SJF is knowing the next CPU request and this information is not usually available. NB: SJF is non-preemptive in nature.

3. Shortest Remaining Tine (SRT) Scheduling

It is the preemptive counterpart of SJF and is useful in time sharing. Currently executing process is preempted when a process with a short CPU burst is ready in a queue.

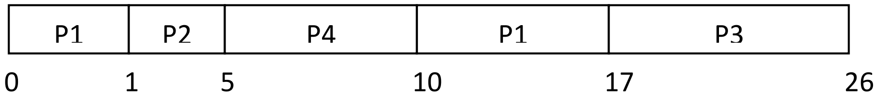
Example

Table 49: SRT

Process	**Arrival Time**	**Burst Time (ms)**
P1	0	8
P2	1	4

P3	2	9
P4	3	5

Gantt chart



Process	Waiting Time
P1	0 + (10 – 1) = 9
P2	(1 – 1) = 0
P3	17 – 2 = 15
P4	5 – 3 = 2

Average waiting time = (9 + 0 + 15 + 2) / 4 = 6.5ms

P1 is started at time 0 since it's the only process in the queue. P2 arrives at time 1. The remaining time for P1 (7ms) is larger than time for process P2 (4ms) so P1 is preempted and P2is scheduled. Average waiting time is less as compared to non-preemptive SJF.

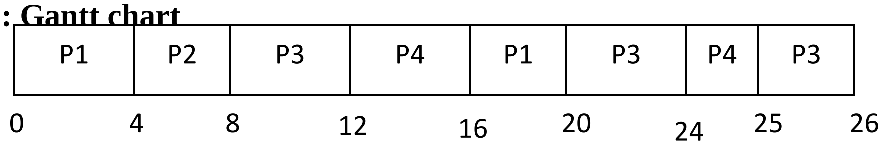
SRT has higher overhead than SJF since it must keep track of elapsed service time of the running job and must handle occasional preemptions.

4. Round Robin (RR) Scheduling

Similar to FCFS scheduling but processes are given a limited amount of CPU time called ***time slice*** or a ***quantum*** which is generally from 10 to 100ms. If the process has a CPU burst of less than 1-time Quantum, the process itself will release the CPU voluntarily. The processes ready queue is treated as a circular queue and CPU scheduler goes around the queue, allocating the CPU to each process for a time of up to 1-time quantum. Using processes in example above and Time Quantum of 4 Ms.

Gantt chart:

Table 50: Gantt chart



Average waiting time = (9 + 0 + 15 + 2) / 4 = 6.5ms

Process	Waiting Time
P1	0 + (16 – 4) = 12
P2	4 – 1 = 3
P3	(8 – 2) + (20 – 12) + (25 – 24) = 15
P4	(12 – 3) + (24 – 16) = 17

RR is effective in time sharing environments. The pre-emptive overheads are kept low by efficient context switching mechanisms and providing adequate memory for the processes to reside in main storage.

5. Priority Scheduling

SJF is a special case of general priority scheduling algorithms. It is the most common scheduling algorithms in batch systems. A priority is associated with each process and CPU is allocated to the process with highest priority. Equal priority processes are scheduled in FCFS order. Priorities are generally some fixed range of numbers e.g. 0 – 7 or 0 – 4095. Some system uses low numbers to represent low priority; others use low numbers for high priority. Priorities can be defined either internationally or externally.

a) For internally defined priorities the following measurable quantities are used – time limits, memory requirements, no of open files, ration of average I/O burst to CPU burst.

b) Externally defined priorities are set by criteria that are eternal to o/s e.g. importance of process, type and amount of funds being paid for computer use, department sponsoring the work and other political factors.

Priority scheduling can either be preemptive or non-preemptive. Major problem is indefinite blocking or starvation: low priority jobs may wait indefinitely for CPU. ***Aging*** technique is used to gradually increase the priority of process that waits in the system, for a long time.

For example, consider the Processes Pa, Pb, Pc and Pd with the following priority and execution time.

Process	**Arrival Time**	**Execute Time**	**Priority**	**Service Time**
Pa	0	5	1	9
Pb	1	3	2	6
Pc	2	8	1	14
Pd	3	6	3	0

Pd	Pb	Pa	Pc

0 6 9 14 22

The wait time for each process will be as follows: -

Process	**Wait Time: (Service Time- Arrival Time)**
Pa	9 - 0 = 9
Pb	6 – 1 = 5
Pc	14 – 2 = 12
Pd	0 – 0 = 0

The average Wait time is (9 + 5 + 12 + 0)/4 = 6.5

6. Multi-level Queue Scheduling

Used in a situation in which processes are easily classified into different groups e.g. *foreground* (interactive) processes and *background* (batch) processes. Ready queue is divided into these groups. Processes are permanently assigned to one queue generally based on some property of the process such as memory size, process priority or process type.

Each queue has its own scheduling algorithm e.g. foreground -> RR, Background ->FCFS. In addition, there must be scheduling between the queues which is commonly implemented as a fixed priority scheduling e.g. fore ground queue may have priority over back ground queue.

7. Multi-level Feedback queue scheduling

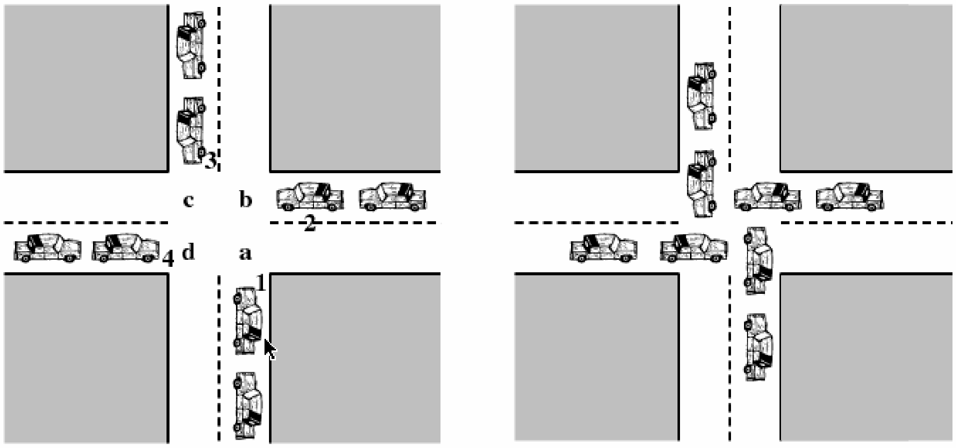
Allows a process to move between queues. The idea is to separate processes with different CPU burst characteristic. If a process uses too much CPU time it will be moved to a lower priority queue. This scheme leaves I/O bound and interactive processes in the highest priority queues. Similarly, a process that waits too long in a lower priority queue may be moved to a higher priority queue. This form of aging prevents starvation.

DEADLOCKS

Occurs when several processes compete for a finite number of resources i.e. a process is waiting for a particular event that will not occur. The event here may be resource acquisition and release.

Example of Deadlocks

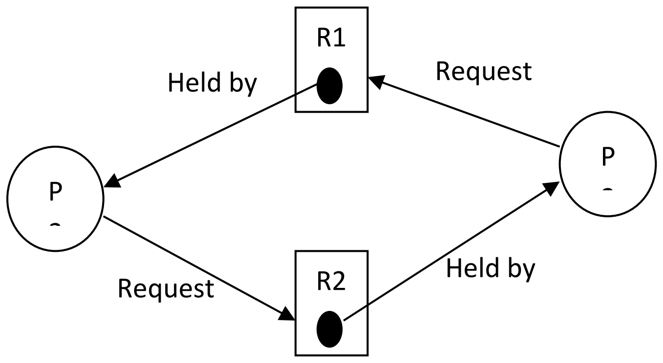
Figure 191: A traffic deadlock – vehicles in a busy section of the city.



(a) Deadlock possible

(b) Deadlock

Figure 192: A simple resource Deadlock



This system is deadlocked because each process holds a resource being requested by the other process and neither process is willing to release the resource it holds. This leads to deadlock.

1. Deadlock in spooling systems

A *spooling* system is used to improve system throughput by disassociating, a program from the slow operating speeds of devices such as printers e.g. lines of text are sent to a disk before printing starts. If disk space is small and jobs are many a dead lock may occur.

Deadlocks Characterization

Four necessary conditions for deadlock to exist:

1. *Mutual Exclusion* condition – a process claims exclusive control of resources they require.
2. *Wait* for Condition (Hold and Wait) –Processes hold resources already allocated to them while waiting for additional resources.
3. *No preemption* condition – resources cannot be removed from the processes holding them until the resources are used to completion.
4. *Circular wait* condition – A circular chain of processes exists in which each process holds one or more resources that are requested by the next process in the chain

Major areas of deadlock research in computing

a) Deadlock Detection

b) Deadlock Recovery

c) Deadlock Prevention

d) Deadlock Avoidance

Deadlock detection

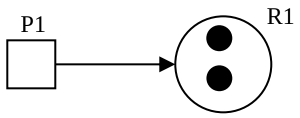
This is the process of determining that a deadlock exists and of identifying the processes involved in the deadlock i.e. determine if a circular wait exists. To facilitate detection of deadlocks resource allocation graphs are used which indicate resource allocation and requests. These graphs change as process request resources, acquire them and eventually release them to the OS.

Reduction of Resource Allocation Graphs is a technique used for detecting deadlocks i.e. processes that may complete their execution and the processes that will remain deadlocked are determined. If a process's resource request may be granted then we say that a graph may be reduced by that process (arrows connecting process and resource is removed).

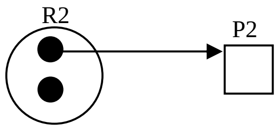
If a graph can be reduced by all its processes then the irreducible processes constitute the set of deadlocked processes in the graph.

NB: - the order in which the graph reductions are performed does not matter; the final result will always be the same.

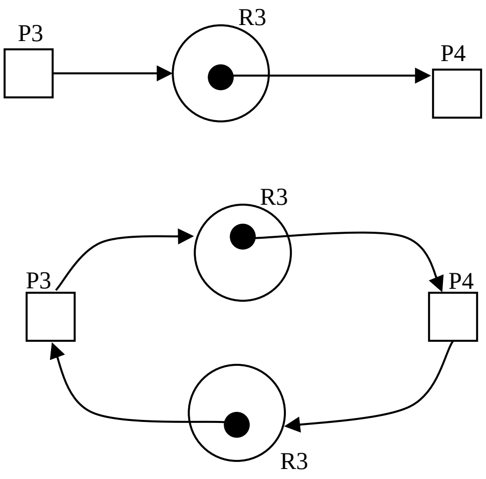
Notations of Resource Allocation and request graphs.



P1 is requesting a resource of type R1



A resource of type R2 has been allocated
to process P2



Process P3 is requesting resource R3 which has been allocated to process P4

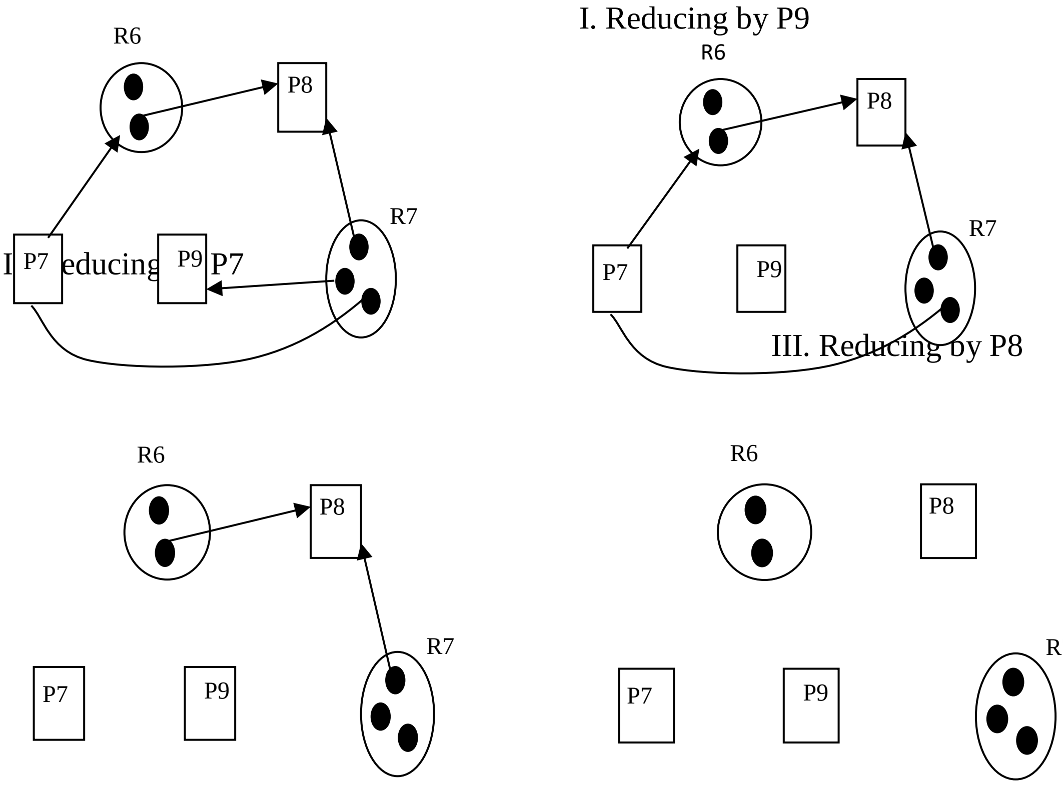
Circular wait (deadlock) process P5 has been allocated R5 which is requested by P6 that has been allocated R4; that is being

Figure 193: Notations of Resource Allocation

Graph Reduction

Given the resource allocation graph can you determine the possibility of a deadlock?

Figure 194: Graph Reduction



I. Reducing by P9

I educing P7

III. Reducing by P8

No deadlock since circular wait doesn't exist.

NB: Deadlock detection algorithm should be invoked at less frequent intervals to reduce overhead in computation time. If it is invoked at arbitrary points there may be many cycles in resource graph and it would be difficult to tell which of the many deadlocked processes "caused" the deadlock.

Deadlock recovery

Once a deadlock has been detected several alternatives exist:

a) *Ostrich Algorithm* (Bury head in the sand and assume things would just work out).

b) Informing the operator who will deal with it manually. Let the system recover automatically.

There are 2 options for breaking a deadlock:

1. Process termination – killing some processes/processes involved in the deadlock.

Methods available include:

a) Abort all deadlocked processes but at a great expense.

b) Abort one process at a time until the deadlock cycle is eliminated.

A scheduling algorithm may be necessary to identify the process to abort.

Factors that may determine which process is chosen.

- ✓ Priority of the process
- ✓ Number and type of resources the process has used (is it simple to preempt)
- ✓ No of resources it needs to complete.
- ✓ Time spent and time remaining.
- ✓ Is the process interactive or batch oriented?

2. Resource Preemption- Preempts some resources from processes and give these resources to other processes until the deadlock cycle is broken. Issues to be addressed include: -
   - ➢ Selecting a victim i.e. a resource to preempt.
   - ➢ Roll Back – a process that is preempted a resource needs to be rolled back to some safe state and restarted from that state once the deadlock is over.

➢ Starvation – guarantee that resources will not always be preempted from some process.

Deadlock prevention

Aims at getting rid of conditions that cause deadlock. Methods used include:

✓ Denying Mutual exclusion – should only be allowed on non-shareable resources. For shareable resources e.g. read only files the processes' attempting to open it should be granted simultaneous access. Thus, for shareable we do not allow mutual exclusion.

✓ Denying Hold and wait (wait for condition) – To deny this we must guarantee that whenever a process requests a resource it does not hold any other resources. Protocols used include:

i) Requires each process to request and be allocated all its resources before it begins execution. While waiting for resources to be available it should not hold any resource – may lead to serious waste or resources.

ii) A process only requests for a resource when it has none or it has to release all resources before getting a resource.

Disadvantages of these protocols

✓ Starvation is possible – A process that needs several popular resources may have to wait indefinitely because at least one of the resources that it needs is always allocated to some other process.

✓ Resource utilization is low since many of the resources may be allocated but unused for a long period.

✓ Denying "No preemption" - If a process that is holding some resource requests another resource that cannot be immediately allocated to it (i.e. it must wait) then all resources currently being held are preempted. The process will be started only when it can regain its old resources as well as the new ones that it is requesting.

✓ Denying Circular wait - All resources are uniquely numbered and processes must request resources in linear ascending order. It has been implemented in many OS but has some difficulties i.e.

- Addition of new resources required rewriting of existing program so as to give the unique numbers.

- Jobs requiring resources in a different order from that one implemented by o/s; it required resources to be acquired and held possibly long before they are actually used (leads to waste)
- Affects a user's ability to freely and easily write applications code.

Deadlock avoidance

Dijkstra's Bankers Algorithm is the widely used deadlock avoidance technique. Its goal is to improve less stringent (constraining) conditions than in deadlock prevention in an attempt to get better resource utilization. Avoidance does not precondition the system to remove all possibility of deadlock to loom, but whenever a deadlock is approached it is carefully side-stepped.

NB: it is used for same type of resources e.g. tape drives or printers.

Dijkstra's Bankers Algorithm says allocation of a resource is only done when it results in a safe state rather than in unsafe states. A safe state is only in which the total resource situation is such that all users would eventually be able to finish. An unsafe state is one that might eventually lead to a deadlock.

Example of a safe state

Assume a system with 12 equivalent tape drives and 3 users sharing the drives:

Table 51: State I

State I

Users	Current Loan	Maximum Need
User (A)	1	4
User (B)	4	6
User (C)	5	8
Available	2	

The state is 'safe' because it is still possible for all 3 users to finish i.e. the remaining 2 drives may be given to users (B) who may run to completion after which six tapes would be released for user (A) and user (C). Thus, the key to a state being safe is that there is at least one way for all user to finish.

Example of an unsafe state

Table 52: State II

Users	Current Loan	Maximum Need
User (A)	8	10
User (B)	2	5
User (C)	1	3
Available	1	

A 3-way deadlock could occur if indeed each process needs to request at least one more drive before releasing any drives to the pool.

NB: An unsafe state does not imply the existence of a deadlock of a deadlock. What an unsafe state does imply is simply that some an unfortunate of events might lead to a deadlock.

Example of safe state to unsafe state transition

Table 53: State III (Safe)

Users	Current	Maximum
User (1)	1	4
User (2)	4	6
User (3)	5	8
Available	2	

If User (3) requests an additional resource

Table 54: State IV (Unsafe)

Users	Current	Maximum
	Loan	Need
User (A)	1	4

User (B)	4	6
User (C)	6	8
Available	1	

Therefore, State IV is not necessarily deadlocked but the state has gone from a safe one to an unsafe one. Thus, Dijkstra's Bankers Algorithm, the mutual exclusion, wait-for and No-preemption conditions are allowed but processes do not claim exclusive use of the resources they require.

Weaknesses in the Banker's Algorithm

Some serious weaknesses that might cause a designer to choose another approach to the deadlock problem.

- Requires that the population of users remain fixed
- Requires that users state their max need in advance. Sometimes this may be difficult.
- Requires that the Banker grant all requests within a finite time.
- Requires a fixed no of resources to allocate and this cannot be guaranteed due to maintenance etc.
- The algorithm requires that clients (i.e. jobs) to repay all loans (i.e. return all resources) within a finite time.
-

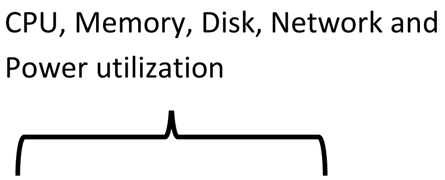
7.2.2.4 Learning Activities

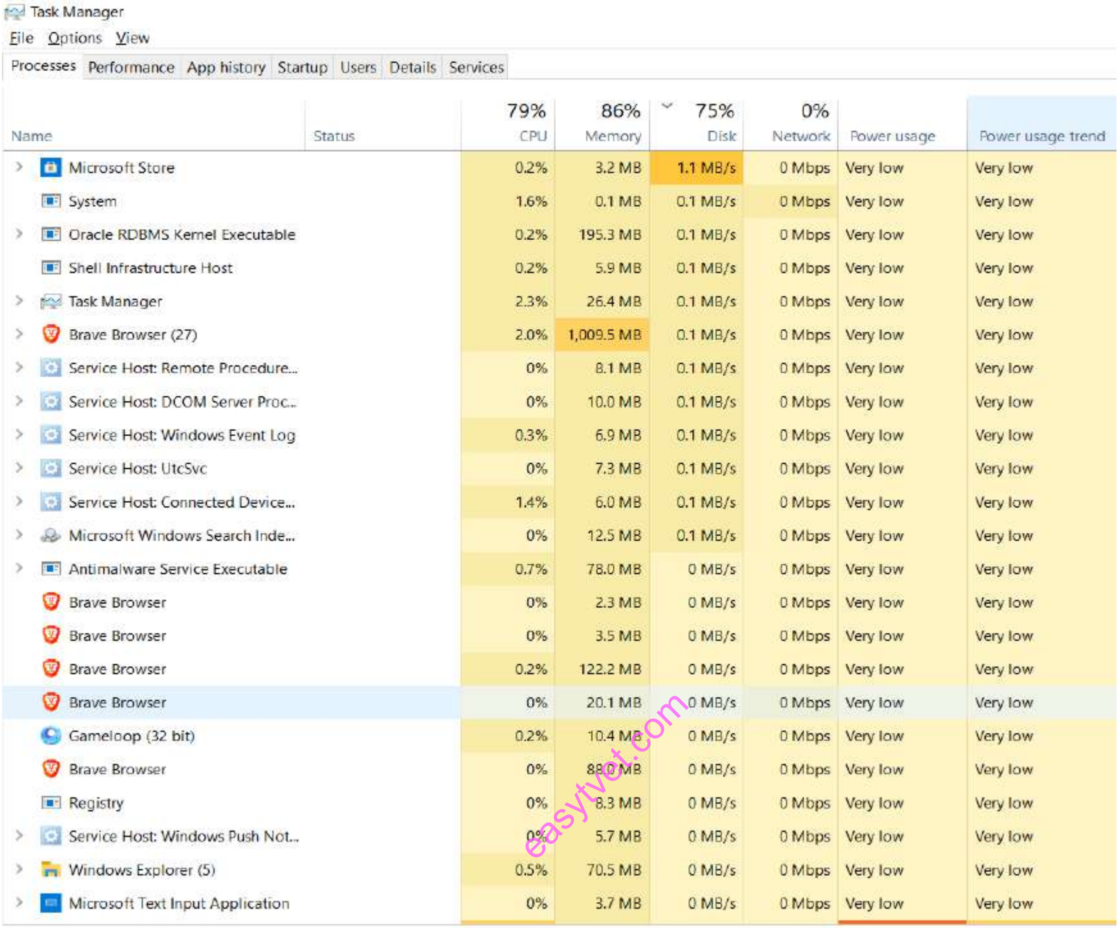
Viewing Status of all processes running in a Windows 10 installed computer system

Steps

1. Press "***Ctrl + Alt + Delete***" and then choose "Task Manager". Alternatively press "***Ctrl + Shift + Esc"*** to directly open task manager.
2. To view a list of processes that are running on your computer, click "Processes". Scroll down to view the list of hidden and visible programs.

Figure 195: Task Manager




Task Manager
File Options View
Processes Performance App history Startup Users Details Services

Name	Status	79% CPU	86% Memory	75% Disk	0% Network	Power usage	Power usage trend
Microsoft Store		0.2%	3.2 MB	1.1 MB/s	0 Mbps	Very low	Very low
System		1.6%	0.1 MB	0.1 MB/s	0 Mbps	Very low	Very low
Oracle RDBMS Kernel Executable		0.2%	195.3 MB	0.1 MB/s	0 Mbps	Very low	Very low
Shell Infrastructure Host		0.2%	5.9 MB	0.1 MB/s	0 Mbps	Very low	Very low
Task Manager		2.3%	26.4 MB	0.1 MB/s	0 Mbps	Very low	Very low
Brave Browser (27)		2.0%	1,009.5 MB	0.1 MB/s	0 Mbps	Very low	Very low
Service Host: Remote Procedure...		0%	8.1 MB	0.1 MB/s	0 Mbps	Very low	Very low
Service Host: DCOM Server Proc...		0%	10.0 MB	0.1 MB/s	0 Mbps	Very low	Very low
Service Host: Windows Event Log		0.3%	6.9 MB	0.1 MB/s	0 Mbps	Very low	Very low
Service Host: UtcSvc		0%	7.3 MB	0.1 MB/s	0 Mbps	Very low	Very low
Service Host: Connected Device...		1.4%	6.0 MB	0.1 MB/s	0 Mbps	Very low	Very low
Microsoft Windows Search Inde...		0%	12.5 MB	0.1 MB/s	0 Mbps	Very low	Very low
Antimalware Service Executable		0.7%	78.0 MB	0 MB/s	0 Mbps	Very low	Very low
Brave Browser		0%	2.3 MB	0 MB/s	0 Mbps	Very low	Very low
Brave Browser		0%	3.5 MB	0 MB/s	0 Mbps	Very low	Very low
Brave Browser		0.2%	122.2 MB	0 MB/s	0 Mbps	Very low	Very low
Brave Browser		0%	20.1 MB	0 MB/s	0 Mbps	Very low	Very low
Gameloop (32 bit)		0.2%	10.4 MB	0 MB/s	0 Mbps	Very low	Very low
Brave Browser		0%	88.2 MB	0 MB/s	0 Mbps	Very low	Very low
Registry		0%	8.3 MB	0 MB/s	0 Mbps	Very low	Very low
Service Host: Windows Push Not...		0%	5.7 MB	0 MB/s	0 Mbps	Very low	Very low
Windows Explorer (5)		0.5%	70.5 MB	0 MB/s	0 Mbps	Very low	Very low
Microsoft Text Input Application		0%	3.7 MB	0 MB/s	0 Mbps	Very low	Very low

3. Check the description and the name to identify each process. Check the “memory” column to see the memory capacity consumed by each process.
4. You can get additional information by searching online for the process name.
5. You can right-click on any active process and choose “End process” to end the program.
6. Press “***Windows + S”*** to open the search pane. Type the name of the program you are trying to find and it should appear on the returned results. Right click the file in search results and click “Open file location”. Browse the folder to see the program that the process belongs to.
7. If you cannot find any information about the program, use the file name to search online. Sites such as processLibrary.com can give you information on whether the program is a virus, adware, or a spyware.
8. One can disable startup programs to boost your computer speed.

7.2.2.5 Self-Assessment

A. What is the first step when installing Windows onto a system that doesn't already have a functioning operating system?

B. Where is the best place to find Windows hardware compatibility information?

C. Describe the method used to add new hardware to a Windows 10 system if Plug and Play does not work?

D. There are several algorithms used to allocate processor time to processes to ensure optimum utilization of the processor and fair distribution of processor time among competing processes. Describe the following algorithms and using the given table, graphically illustrate the differences in the processor scheduling using the THREE algorithms.

   a. First Come First Served (**FCFS**)

   b. Shortest Process Next **(SPN)**

   c. Shortest Remaining Time **(SRT)**

Process	**Arrival Time**	**Required Service**
A	**0**	**3**
B	**2**	**6**
C	**4**	**4**
D	**5**	**5**
F	**8**	**2**

7.2.2.6 Tools, Equipment, Supplies and Materials

- Functional computer
- Windows operating system
- Note pad

7.2.2.7 Model answers to self-assessment

A. What is the first step when installing Windows onto a system that doesn't already have a functioning operating system? ***We Partition the disk then format it to prepare it to store files***

B. Where is the best place to find Windows hardware compatibility information?

Manufacturer's website

C. Describe the method used to add new hardware to a Windows 10 system if Plug and Play does not work?

1. Run ***devmgmt.msc***
2. Select the hardware device you want to add e.g. Network adapters
3. then click Action menu and select scan for hardware changes

D. There are several algorithms used to allocate processor time to processes to ensure optimum utilization of the processor and fair distribution of processor time among competing processes. Describe the following algorithms and using the given table, graphically illustrate the differences in the processor scheduling using the THREE algorithms.

a. First Come First Served (**FCFS**)

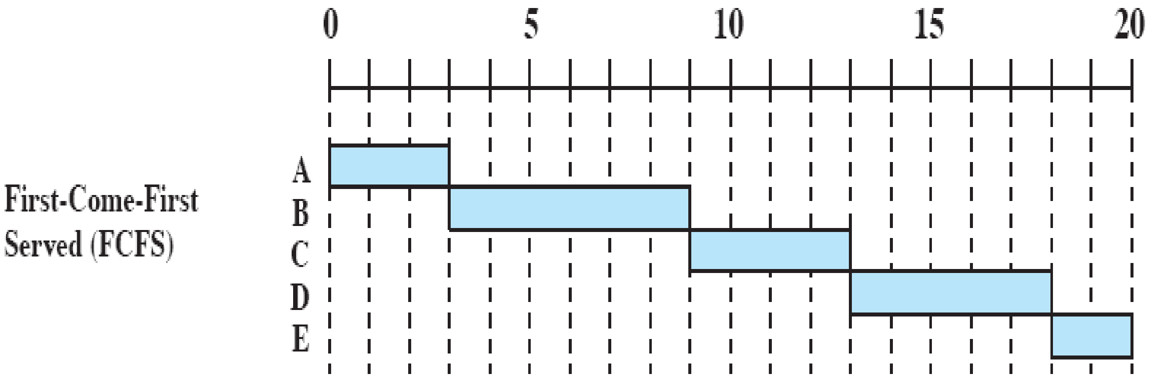
b. Shortest Process Next **(SPN)**

c. Shortest Remaining Time **(SRT)**

Process	**Arrival Time**	**Required Service**
A	**0**	**3**
B	**2**	**6**
C	**4**	**4**
D	**5**	**5**
F	**8**	**2**

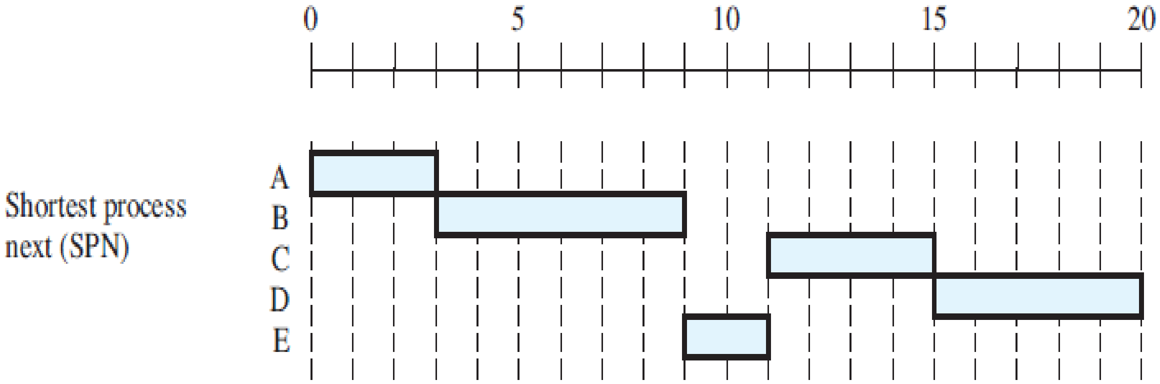
FCFS (First Come First Served)

- Each process joins the Ready queue
- When the current process ceases to execute, the process which has been in the Ready queue the longest is selected



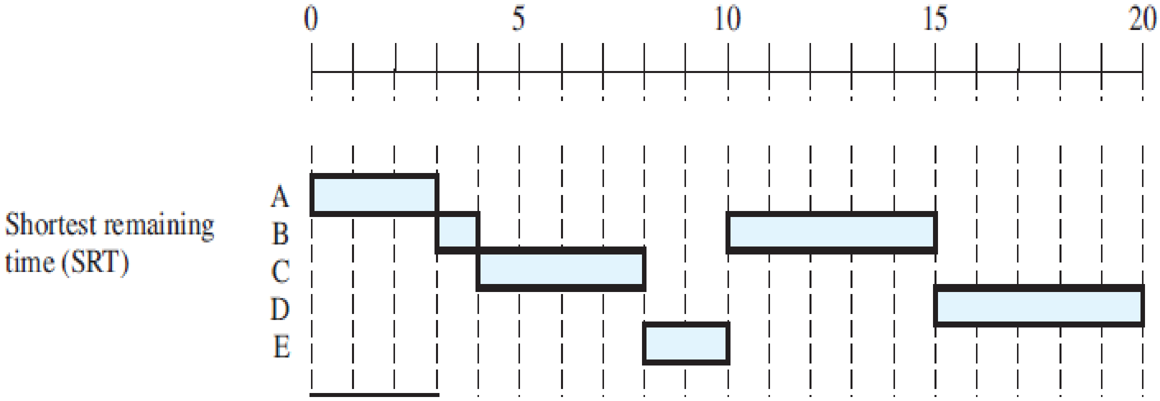
SPN (Shortest Process Next)

- Non-preemptive policy
- Process with shortest expected processing time is selected next
- Short process jumps ahead of longer processes



Shortest Remaining Time (SRM)

- Preemptive version of shortest process next policy
- Must estimate processing time and choose the shortest



7.2.2.7 References

Abraham, P. a. (2013). *Operating Systems Concepts*. United States of America: Wiley.

Abraham, P. G. (2013). *Operating Systems Concepts* . Hoboken: John Wiley & Sons, Inc.

CompTIA. (2013). *CompTIA A+*. (Vol. 2013, Issue December 2). http://certification.comptia.org/getCertified/certifications/a.aspx

Li, A. W. A. and K. (n.d.). *Virtual Memory Primitives for User Programs.pdf*.

Maccabe, A., Bridges, P., Brightwell, R., & Riesen, R. (n.d.). *Recent Trends in Operating Systems and their Applicability to*.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2017). *OPERATING SYSTEM ls s s i t*.

Stallings, W. (2014). Operating Systems: Internals E Designs Principles. In *Journal of Chemical Information and Modeling*.

7.2.3 Learning Outcome 3: Identifying concepts of Memory management.

7.2.3.1 Introduction to the learning outcome

Memory is key to the operation of a modern computer system. A computer memory consists of a large array of words or bytes, each with its own logical address. Ideally a program resides on a disk as a binary executable file. For a program to be executed it must be brought into memory and placed within a process. Depending on the memory management in use the process may be moved between disk and memory during its execution. Learning memory management basics, empowers a computer technician to understand how programs utilize memory during execution.

7.2.3.2 Performance Standard

7.2.3.2.1 Definition of memory management is done.

7.2.3.2.2 Objectives of memory management are identified.

7.2.3.2.3 Memory management techniques are identified.

7.2.3.2.4 Memory management policies are identified.

7.2.3.3 Information sheet

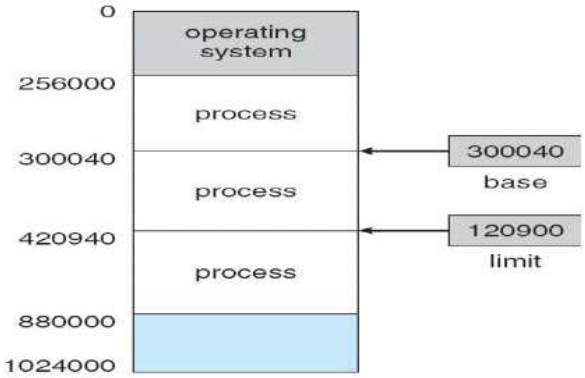
Memory management

Memory management is the functionality of an operating system which handles or manages primary memory and moves processes back and forth between main memory and disk during execution. Memory management keeps track of each and every memory location, regardless of either it is allocated to some process or it is free. It checks how much memory is to be allocated to processes. It decides which process will get memory at what time. It tracks whenever some memory gets freed or unallocated and correspondingly it updates the status.

A program resides in a disk as a binary executed file and must be brought into memory and placed within a process for it to be executed. Collection of processes on the disk that are waiting to be brought into memory for execution forms the input queue. Two registers a base and a limit are used to provide protection. The base register holds the smallest legal physical memory address; the limit register specifies the range of the size. For example, if a base register holds

300040 and the limit register is 120900, then the program can legally access all addresses from 300040 through 420939 inclusive as shown in figure 13

Figure 196: A base and limit register defining a logical address space



Instructions and data to memory addresses can be done during the following times: -

- ✓ **Compile time** where the process will reside in order to generate the absolute code
- ✓ **Load time** where the compiler generates a re-locatable code.
- ✓ **Execution time** in the event a process is moved during its execution from one memory segment to another

Objectives of memory management

✓ To provide a detailed description of various ways of organizing memory hardware.
✓ To discuss various memory-management techniques, including paging and segmentation.
✓ To provide a detailed description of the Intel Pentium, which supports both pure segmentation and segmentation with paging.

Memory management techniques

Fixed partitioning: this partitioning approach divided into a fixed number of partitions such that one process can be loaded into one partition at the same time. Strengths of this approach: easy to implement it and slandered method as a partitioning solution. Weaknesses of this approach includes; -

- ✓ It is limited to largest partition size
- ✓ Insufficient use due to the internal fragmentation
- ✓ One must know the maximum number of active processes that can run

- ✓ The degree of multiprogramming is limited by the number of partitions
- ✓ Memory is wasted in the partition
- ✓ It must translate relative address to physical address.
- ✓ It is fixed in terms of the size of the task

Dynamic partitioning

Partitions are created dynamically, each process loaded into a partition has exactly the same size as the process.

Strength of this approach include:

- ✓ Ensures more efficient use of the main memory and no internal fragmentation.

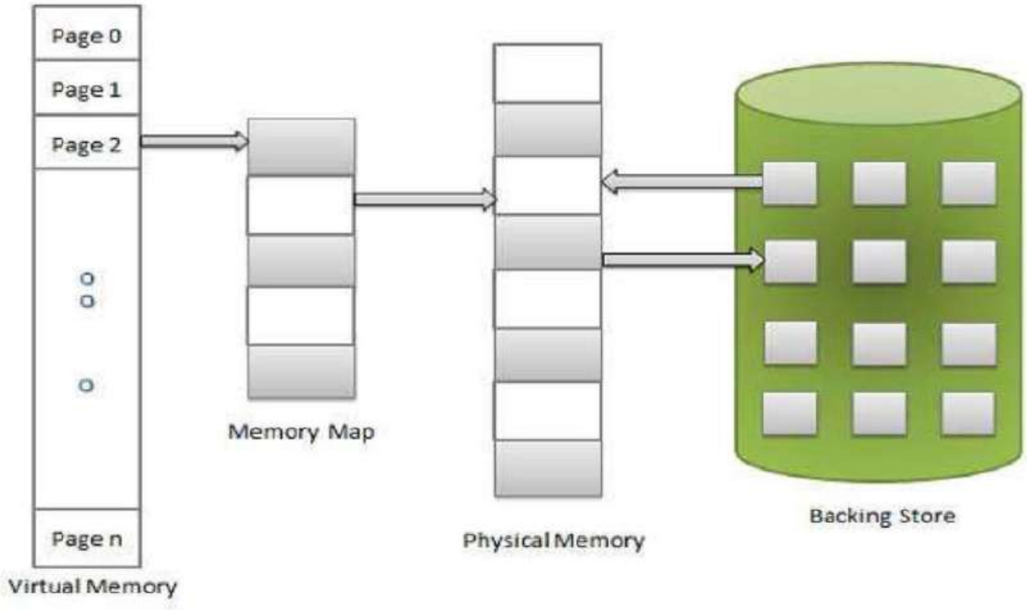
Weaknesses of this approach include:

- ✓ Inefficient use of processor because of the need for compaction and external fragmentation.

Virtual memory

A storage allocation scheme in which secondary memory can be addressed as though it were part of main memory. Virtual memory allows the execution of processes which are note completely available in memory. The addresses a program may use to reference memory are distinguished from the addresses the memory system uses to identify physical storage sites, and program-generated addresses are translated automatically to the corresponding machine addresses. Virtual memory is the separation of user logical memory from physical memory. The size of virtual storage is limited by the addressing scheme of the computer system and by the amount of secondary memory available and not by the actual number of main storage locations.

Figure 197: Virtual Memory View



Thrashing

At any given time, only few pages of any process are in main memory and therefore more processes can be maintained in memory. Furthermore, time is saved because unused pages are not swapped in and out of memory. However, the OS must be clever about how it manages this scheme. In the steady state practically, all of main memory will be occupied with process's pages, so that the processor and OS have direct access to as many processes as possible. Thus, when the OS brings one page in, it must throw another out. If it throws out a page just before it is used, then it will just have to get that page again almost immediately. Too much of this leads to a condition called **Thrashing**. The system spends most of its time swapping pages rather than executing instructions.

Causes of Thrashing:

1. **High degree of multiprogramming:** If the number of processes keeps on increasing in the memory than number of frames allocated to each process will be decreased. So, less number of frames will be available to each process. Due to this, page fault will occur more frequently and more CPU time will be wasted in just swapping in and out of pages and the utilization will keep on decreasing.

For example:

Let free frames = 400

Case 1: Number of process = 100

Then, each process will get 4 frames.

Case 2: Number of process = 400

Each process will get 1 frame.

Case 2 is a condition of thrashing, as the number of processes are increased, frames per process are decreased. Hence CPU time will be consumed in just swapping pages.

2. **Lacks of Frames**: If a process has a smaller number of frames then less pages of that process will be able to reside in memory and hence more frequent swapping in and out will be required. This may lead to thrashing. Hence sufficient number of frames must be allocated to each process in order to prevent thrashing.

Recovery of Thrashing:

1. Do not allow the system to go into thrashing by instructing the long-term scheduler not to bring the processes into memory after the threshold.
2. If the system is already in thrashing then instruct the mid-term schedular to suspend some of the processes so that we can recover the system from thrashing.

Memory Allocation Techniques

To improve both utilization of the CPU and the speed of computer's response to its users, it is important for the OS to manage its resource. One of that resource is the memory. The easiest way to manage it is by the fooling list of techniques: -

Overlays

The main problem in fixed partitioning is the size of a process has to be limited by the maximum size of the partition, which means a process can never be span over another. To solve this problem, earlier systems have used solution called Overlays.

The concept of overlays is that whenever a process is running it will not use the complete program at the same time, it will use only some part of it. Whatever part you required, you load it on once the part is done, then you just unload it, means just pull it back and get the new part you required and run it. Overlays is the process of transferring a block of program code or other data into internal memory, replacing what is already stored. Sometimes it happens that compared to the size of the biggest partition, the size of the program will be even more, then, in that case, you should go with overlays.

In conclusion, overlay is a technique to run a program that is bigger than the size of the physical memory by keeping only those instructions and data that are needed at any given time. It divides the program into modules in such a way that not all modules need to be in the memory at the same time.

Advantages

- Reduce memory requirement
- Reduce time requirement

Disadvantages

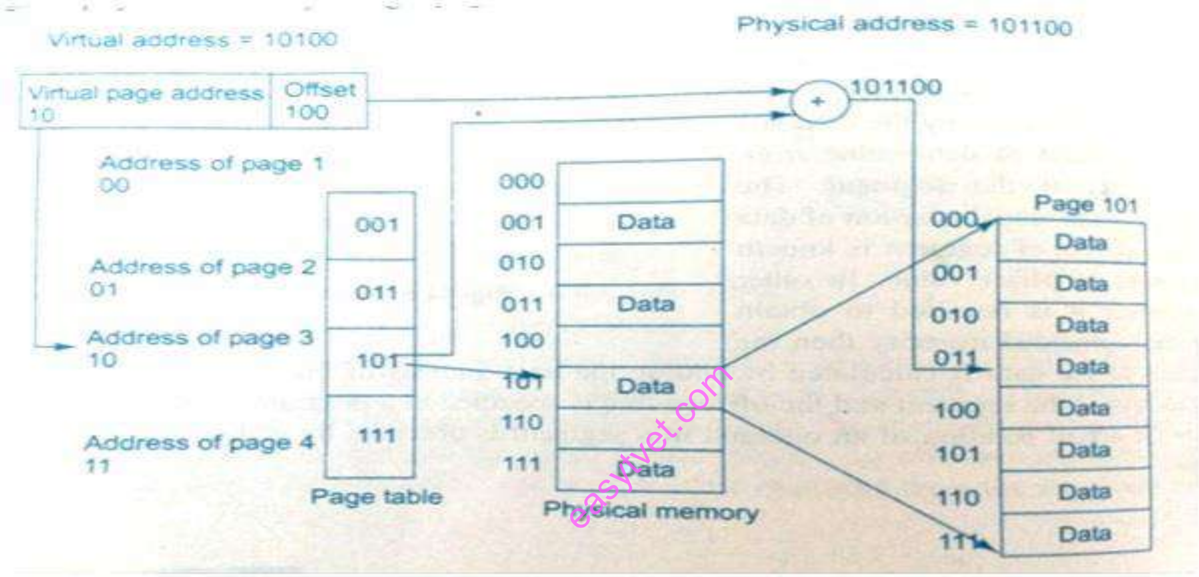
- Overlap map must be specified by programmer
- Overlapped module must be completely disjoint
- Programmer must know memory requirement
- Programming design of overlays structure is complex and not possible in all cases

Paging

Paging is a technique in which the main memory of computer system is organized in the form of equal sized blocks called pages. In this technique, the address of occupied pages of physical memory are stored in a table, which is known as page table. Paging enables the operating system to obtain data from the physical memory location without specifying lengthy memory address in the instruction.

In this method, the virtual address is used to map the physical address of the data. The length of virtual address is specified in the instruction and is smaller than physical address of the data. It consists the address of page called *virtual page* in the page table and the *offset value* of the actual data in the page.

Figure 198: Virtual Address Space

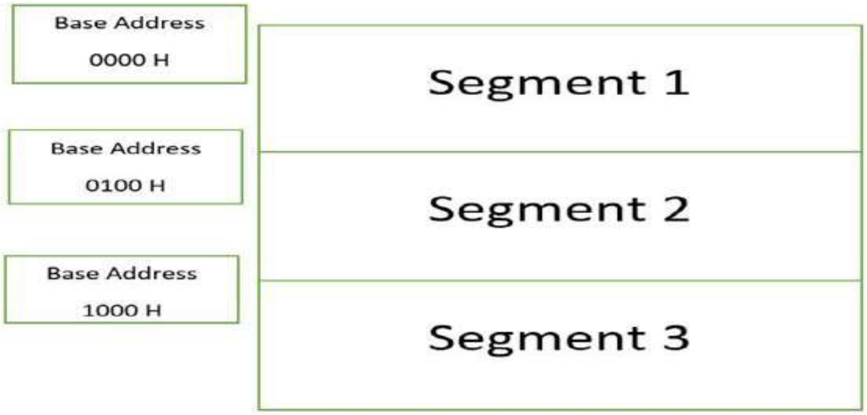


The above figure shows how the virtual address is used to obtain the physical address of an occupied page of physical memory using a page table.

Segmentation

Segmentation refers to the technique of dividing the physical memory space into multiple blocks. Each block has specific length and is known as a *segment*. Each segment has a starting address called the base address. The length of the segment determines the available memory space in the segment.

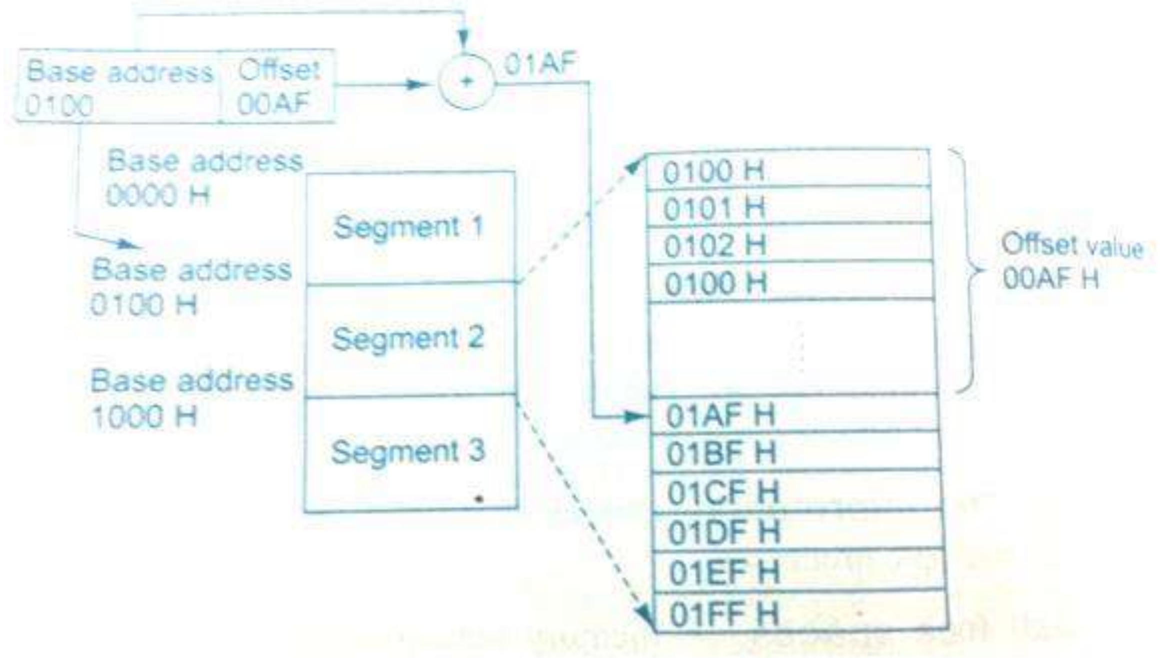
Figure 199: Organization of segment in memory unit



The location of data values stored in the segment can be determined by the distance of actual position of data value from base address of the segment. **Displacement or Offset value** is the distance between the actual position of data and the base address of segment. In other words, when there is a need to obtain data from required segmented memory then the actual address of data is calculated by adding the base address of the segment with offset value.

The base address of the segment and the offset value is specified in a program instruction itself. The following figure shows how the actual position of an operand in a segment is obtained by adding the base address and offset value.

Figure 200: Base address and offset value



Swapping

User programs do not remain in main memory until completion. In some systems one job occupies the main storage once. That job runs then when it can no longer continue it relinquishes both storage and CPU to the next job. Thus, the entire storage is dedicated to one job for a brief period, it is then removed (i.e. swapped out or rolled out) and next job is brought in (swapped in or rolled in). A job will normally be swapped in and out many times before it is completed. It guarantees reasonable response times

Memory management policies

Memory management handles primary memory and moves processes back and forth between memory and disk during execution. It keeps track of each and every memory location irrespective of whether a process has been allocated to it or not. It checks how much memory is to be allocated to a process(s). The decision of when to allocate memory to a process rests with memory management policies.

The runtime mapping from virtual address to physical address is done by the memory management unit (MMU) which is a hardware device. It converts a virtual address to physical address by adding a value in the base register to every address generated by a user process.

Main memory management policy includes swapping and segmentation.

Swapping entails managing space on the swap device, swapping processes into and out of main memory. Swap device is simply a block device in a configurable section of a disk. Consequently, the swap space is allocated in form of contiguous blocks instead of single block allocation for files. This allocation space is transitory (temporary) since a process that resides in the swap space will eventually migrate to main memory. The kernel allocates contiguous space on the swap device without regard for fragmentation because faster I/O transfer is critical.

The Kernel maintains free space for the swap device in a table called the ***map*** (An array where each entry consists of an address of an allocatable resource and the number of resource units available). Consequently, the kernel interprets the address and units according to the type of map.

At first, a map contains one entry that indicates the address and the total number of resources and then responds appropriately as the allocation increases as shown in figure 18, 19 and 20:

Figure 201: Sample Map: Interpreted as 1000 free blocks starting from address 1

Address	Units
1	1000

Figure 202: State of the Map after allocation 100 units

Address	Units
101	900

Figure 203: State of the map after allocating 50 units

Address	Units
151	850

The kernel finds the proper position in the map address every time it frees resources using the following possible mechanisms: -

a) The freed resources fill a hole in the map completely i.e. the kernel combines the newly freed resources and the existing two entries into one entry in the table.

b) The freed resources partially fill a hole but are not contiguous to any resources in the map thus a new entry is created and inserted in the proper position.

c) The freed resource partially fills a hole and the kernel adjusts the address and units filed of an appropriate entry to account for the resources just freed.

Swapping Processes Out

Swapping out occurs when the kernel requires space in memory under the following situations:

✓ When the fork system call must allocate space for a child process.

✓ When the brk system call increases the size of a process.

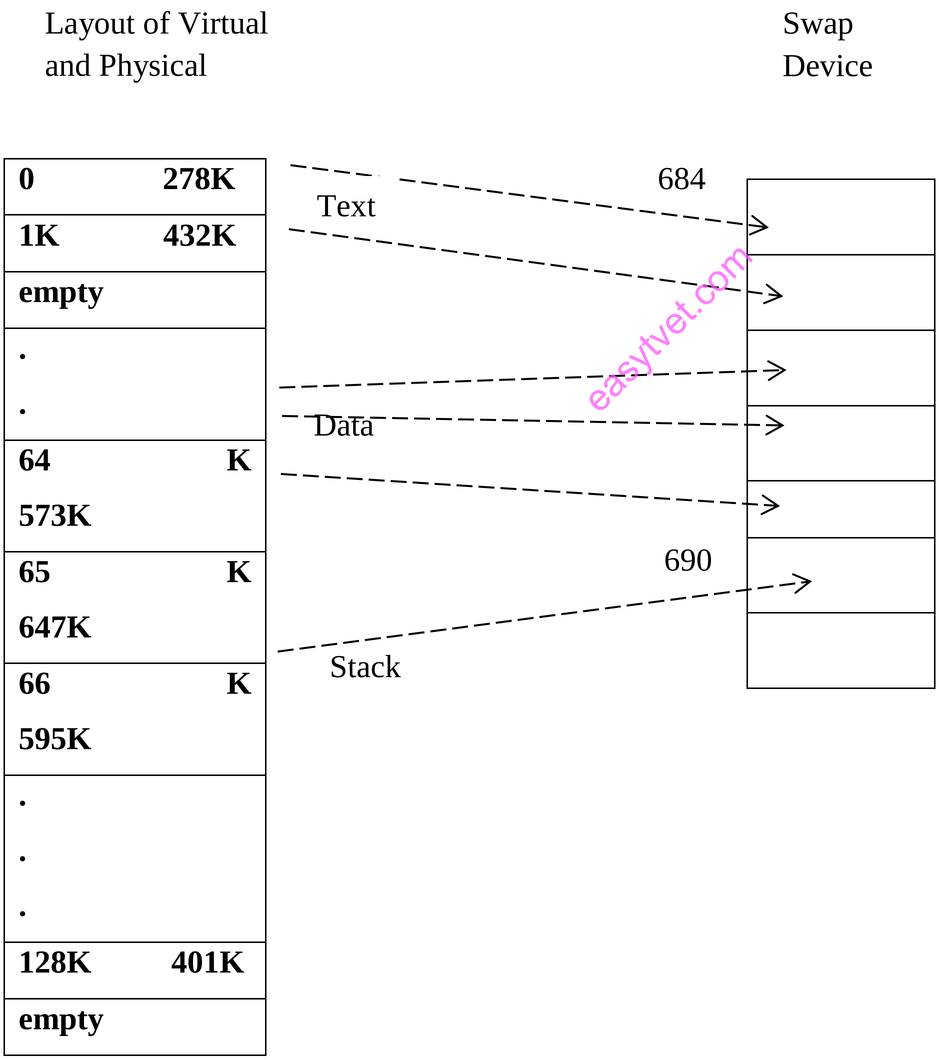
✓ When a process becomes larger by the natural growth of its stack.

✓ When the kernel wants to free space in memory for processes it had previously swapped out and should now swap in.

The kernel decrements the reference count of each region in the process and swaps the region out if its reference count drops to 0 every time it decides a process is eligible for swapping. It allocates space on a swap device and locks in memory thus prevents swapping it out while the current swap operation is in session. At the same time, it saves the swap address of the region in the map.

The buffer cache is circumvented when the kernel swaps data between the swap device and user address space. In the event memory is organized in pages, the data to be swapped out is most likely to be discontiguous in physical memory. The kernel gathers the page addresses of data to be swapped out, and the disk driver may use the collection of page addresses to set up the I/O. The swapper waits for each I/O operation to complete before swapping out other data.

Figure 204:Mapping the image of a process onto a swap device



NB: The virtual address space is depicted as a linear array of page table entries, disregarding the fact that each region usually has a separate page table.

Demand paging

A kernel that implements a demand paging algorithm (swapping memory pages between the main memory and a swap unit) can support machines whose memory architectures are based on pages and whose CPU has restart-able instructions.

Demand paging systems free processes from size confines otherwise imposed by the amount of physical memory available on a machine. However, the kernel has a limit on the virtual size of a process depending on the amount of virtual memory the machine can address. Demand paging is transparent to the user except for the virtual size allowed to a process. The Text portion (contains frequently called subroutines and program loops) forms a small subset of the total space of the entire process. This is referred to as the principle of locality.

Segmentation refers to the technique of dividing the physical memory space into multiple blocks. Each block has specific length and is known as a segment. Each segment has a starting address called the base address. The length of the segment determines the availability of memory space in the segment.

Process Address Space

It is a set of logical addresses that a process references in its code. Addresses can range from 0 to 0x7fffffffff when 32-bit addressing is in use; that is, 2^31 possible numbers, for a total theoretical size of 2 gigabytes. The OS maps the logical addresses to physical addresses during memory allocation to the program through: -

a) Physical address space where the loader generates the addresses at the time when a program is loaded into the main memory.

b) Relative address space where during compilation the compiler converts symbolic addresses into relative addresses

c) Symbolic addresses space where the variable names, constants and instruction labels are the basic elements of the symbolic address space.

The set of all physical addresses corresponding to logical addresses is referred to as physical address space while the set of all logical addresses generated by a program is referred to as logical address space.

Fetch Policy

The fetch policy determines when a page should be brought into main memory. It uses either demand paging or pre-paging. For demand paging, a page will be brought into main memory when a reference is made to a location on that page. If this is chosen by virtual memory subsystem when a process is first started, there will be a flurry of page faults. The more the pages brought in the less likely occurrence of page faults, due to the principle of locality (the most recently visited pages will be most likely to be visited again). The pages are already in main memory.

For pre-paging, pages are brought in even when they are not requested. This method usually takes advantage of the characteristic of secondary memory devices. For example, a disk where pages are stored on a track consecutively, pages aught to be read along the track where the read head of the disk has to pass. It is often an effective method since related instructions are more likely to be neighbors and thus stored adjacently. This policy is used when a process is first started up to avoid frequent page fault.

Placement policy

The placement policy determines where in real memory a process piece should to reside. This policy is an important issue only in pure segmentation systems, because with paging, a page is the unit for main memory and virtual memory management and placement is usually irrelevant. This is because the address translation hardware and the main memory access hardware can perform their functions for any page-frame combination with equal efficiency. Simple algorithms can be used in a pure segmentation, where a segment requires a contiguous space whose size is at least same as the size of the segment. Algorithms such as first-fit algorithm

Page Replacement Policy

This policy is used to find location of page on a disk and also find a free page frame. It works by following the following algorithms:

1. Find the location of a free page on a disk
2. Find a free page frame
    1. When a free page is found, use it.
    2. Otherwise, select a page frame using the page replacement algorithm
    3. Write the selected page to the disk and update any necessary tables
3. Read the requested page from the disk
4. Restart the instruction.

Some of the page replacement strategies used include: -

1. Random page replacement where a page is chosen randomly
2. The principle of optimality where a page that will not be used most of the time in the future is replaced
3. First in First out (FIFO) strategy where a page that has been in primary memory for long is replaced
4. Least Recently Used (LRU) strategy where a page that has not been used for a long time is replaced
5. Least Frequently Used (LFU) strategy where a page that is used least often is replaced

7.2.3.4 Learning Activities

Learning Activity One

Configure Virtual Memory in a Personal Computer

The size of virtual memory can sometimes be adjusted by a computer user to improve system performance and augment the existing RAM in a computer. Using MS-Windows Operating System, discover if you can change the size of virtual memory, how to do so, and the minimum and maximum recommended sizes of virtual memory for the computer you are using. Cite the steps required to retrieve the desired information. NB: Do not change the size of your virtual memory settings. This learning activity is for research purposes only. While you are at it determine:

a) Your system configuration:
b) Examine and observe the memory demand of an executing process

c) Examine the effect of increased demand for memory resources

Learning Activity Two

Compare TWO processors currently being produced for laptop computers. Use standard industry benchmarks for your comparison and briefly list the advantages and disadvantages of each. You can compare different processors from different manufacturers (such as Intel and AMD) or from same manufacturer (such as two Intel processors) or different processors.

7.2.3.5 Self-Assessment

A. Discuss the following memory placement strategies

i. First fit

ii. Best fit

iii. Worst fit

B. System software is computer software designed to operate the computer hardware and to provide a platform for running application software. Elaborate on the following types of system software:

i. Device drivers

ii. The operating system

iii. Utility software

iv. Window systems

C. In memory management and file naming convention what do the following file extension stands for:

i. File.bak

ii. File.c

iii. File.txt

iv. File.zip

v. File.o

D. Differentiate between Paging and Segmentation as applied in memory management

E. Describe demand paging and prepaging as used in virtual memory page fetch policies. Justify the less common in practice.

7.2.3.6 Tools, Equipment, Supplies and Materials

- Physical PC
- Memory Modules from different manufacturers, different types and sizes
- Antistatic wrist wrap
- Internet for research purposes

7.2.3.7 Model answers to self-assessment

A. Discuss the following memory placement strategies

i. First fit

- Scans memory form the beginning and chooses the first available block that is large enough
- Fastest
- May have many processes loaded in the front end of memory that must be searched over when trying to find a free block

ii. Best fit

- Chooses the block that is closest in size to the request
- Worst performer overall
- Since smallest block is found for process, the smallest amount of fragmentation is left memory compaction must be done more often

iii. Worst fit

- More often allocate a block of memory at the end of memory where the largest block is found
- The largest block of memory is broken up into smaller blocks
- Compaction is required to obtain a large block at the end of memory

B. System software is computer software designed to operate the computer hardware and to provide a platform for running application software. Elaborate on the following types of system software:

i. Device drivers

Provides basic functionality to operate and control the hardware connected to or built into the computer such as computer BIOS and device firmware.

ii. The operating system

It allows the parts of a computer to work together by performing tasks like transferring data between memory and disks or rendering output onto a display device. It also provides user interface through which users interact with the system as well as a platform to run high-level system software and application software.

iii. Utility software

This software helps to analyze, configure, optimize and maintain the computer.

iv. Window systems

They are components of a graphical user interface (GUI), and more specifically of a desktop environment, which supports the implementation of window managers, and provides basic support for graphics hardware, pointing devices such as mice, and keyboards. The mouse cursor is also generally drawn by the windowing system.

C. In memory management and file naming convention what do the following file extension stands for:

i. File.bak

Backup of another document often created automatically by the operating system or various programs such as AutoCAD; Windows may create BAK files for the System.ini and Win.ini files.

ii. File.c

Header file in C programming

iii. File.txt

Standard text document that contains unformatted text; recognized by any text editing or word processing program; can also be processed by most other software programs.

iv. File.zip

File compressed or "zipped" using Zip compression, a common type of compression in which every file in the archive is compressed separately; supported by most file compression/decompression programs.

v. File.o

Object file produced by a C compiler; often created instead of a program file during the development process.

D. Differentiate between Paging and Segmentation as applied in memory management.

Paging is a technique in which physical memory is broken into blocks of the same size called pages (size is power of 2, between 512 bytes and 8192 bytes) while Segmentation is a technique to break memory into logical pieces where each piece represents a group of related information (segments).

E. Describe demand paging and preparing as used in virtual memory page fetch policies. Justify the less common in practice.

Demand paging – is where relevant pages are loaded as page faults occur. It loads pages when page fault occurs (fetch on demand) and it is more common than preparing.

Pre paging - try to load pages for a process before they're accessed. Wastes I/O bandwidth if pages are loaded unnecessarily and worse if an unnecessary page kicks out a necessary page. Pre-paging brings in more pages than needed at the moment. I/O is improved due to reading large chunks but waste bandwidth if pages aren't used. Especially bad if we eject pages in working set in order to pre-fetch unused pages. Rarely used in practice.

7.2.3.8 References

Abraham, P. a. (2013). *Operating Systems Concepts.* United States of America: Wiley.

Abraham, P. G. (2013). *Operating Systems Concepts* . Hoboken: John Wiley & Sons, Inc.

CompTIA. (2013). *CompTIA A+*. (Vol. 2013, Issue December 2). http://certification.comptia.org/getCertified/certifications/a.aspx

Li, A. W. A. and K. (n.d.). *Virtual Memory Primitives for User Programs.pdf*.

Maccabe, A., Bridges, P., Brightwell, R., & Riesen, R. (n.d.). *Recent Trends in Operating Systems and their Applicability to*.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2017). *OPERATING SYSTEM ls s s i t*.

Stallings, W. (2014). Operating Systems: Internals E Designs Principles. In *Journal of Chemical Information and Modeling*.

7.2.4 Learning Outcome 4 Identifying concepts of Input and Output devices

7.2.4.1 Introduction to the learning outcome

Input Output devices (I/O) provides communication between a processing system and the users. It is critical to understand how signals or data is received by a computer system and how they are processed and released for display using output devices. As a technician mastering this concept enables you to set up a computer system, setup a video conferencing, troubleshoot I/O devices among others.

7.2.4.2 Performance Standard

7.2.4.2.1 Definition of input and output devices is done.

7.2.4.2.2 Objectives of input/output device management are identified

7.2.4.2.3 Concepts of input and output devices are identified.

7.2.4.2.4 Input/output devices software are explained.

7.2.4.2.5 Description of disk and disk operations are done.

7.2.4.2.6 Explanation of computer clock system is done.

7.2.4.2.7 Computer terminals are identified.

7.2.4.2.8 Virtual devices are defined.

7.2.4.3 Information Sheet

Input and output devices definition.

Computers operate many types of devices including transmission devices such as network cards, human-interface devices such as keyboard, mouse and storage devices such as tapes and disks etc.

Input devices

It is a device that transfers information outside the computer system to an internal storage location, such as system RAM, video RAM, flash memory, or disk storage (CompTIA, 2013).

Without input devices, computers would be unable to change state from their originally manufactured intent.

Examples of input devices include Mouse, Keyboard, Barcode reader, Scanner, Digital camera, Microphone, Joystick, Biometric devices, Touch screens, Webcam, Graphic tablet etc.

Output devices

These devices are used to put information from the computer onto a screen or paper. It receives data from a computer and then translates that data into another form. Examples include braille reader, printer, monitor, headphones, computer speakers, projector, Sound card, video card, speech generating device, GPS (Global Positioning System) device etc.

Concepts of input and output devices.

I/O devices are categorized into

a. Human readable: - these devices are suitable for communicating with the computer user such as video display, printers, keyboard etc.

b. Machine readable: - these devices are suitable for communicating with electronic equipment e.g. tape drives and disk, sensors, controlled and actuators.

c. Communication devices: -these devices are suitable for communicating with remote devices

The I/O devices differ in terms of; Data rate/data transfer rates, Application (different use in the system), Unit of transfer (as a byte, character or unit of large blocks), Data representation (encoding schemes), Error conditions of the devices.

Device controllers

The functions of the device controllers is to move data between devices and main memory usually performed by the CPU. The Kernel module is the one that controls a device driver.

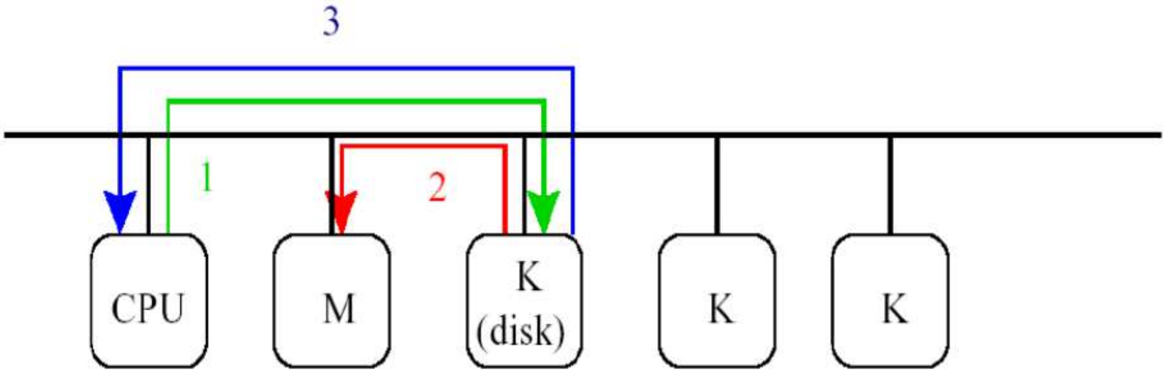
It can be viewed as the call interface designed to handle several basic categories of hardware devices. A computer system contains a number of I/O devices and their respective controllers for instance Network card, Graphics adapter, Disk controller, CD/DVD -ROM controller, USB, Sound card, serial port.

Direct Memory Access (DMA input/output)

DMA is a special control unit provided to allow transfer of a block of data directly between an external device and the main memory without the continuous intervention of the CPU.

DMA can be used with either interrupt software and polling. This technique is useful on devices such as disks where large number of bytes of information is transferred in a single I/O operation. Interaction with a device controller is managed through device drivers thus part of operating system. In Linux OS for instance devices are accessible through the /dev file system.

Figure 205: DMA Operations



DMA Operations

1. CPU issues DMA request to controller
2. Controller directs data transfer
3. Controller interrupts CPU

Input/Output device software.

The goal of the I/O software is to ensure device independence. It should be possible to write programs that can access any I/O device without necessarily specifying the device in advance. It is the duty of I/O software to take care of the challenges caused by the fact that the I/O devices are different and require very different command sequences to read or write whether its an IDE disk or a SCSI disk.

The I/O software should also manage uniform naming of files. The name of a file or device should be a string or integer not dependent on the device in any manner. For example, in UNIX OS all disks can be integrated into the file system hierarchy in arbitrary ways as such the user need not be aware of which name corresponds to which device i.e. all devices are addressed in a similar manner using path name.

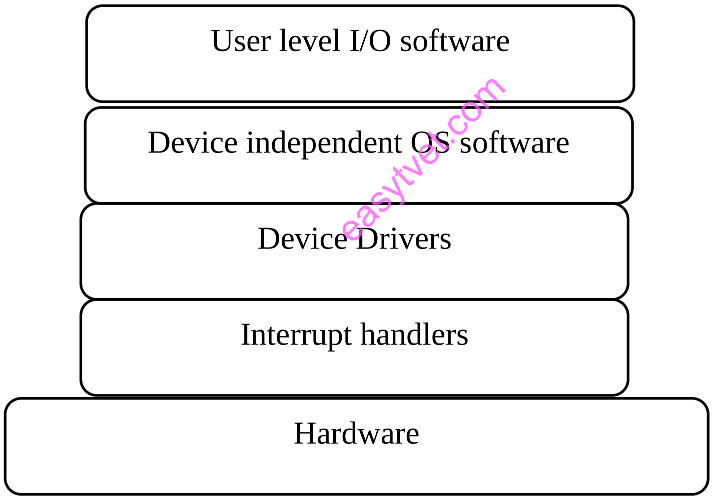
I/O software performs error handling. It ensures errors are handled close to the hardware as possible. Once the controller discovers an error it corrects immediately. If it can’t manage then the device should handle it

I/O software enables buffering especially when data that come off a device cannot be stored directly in its final destination. Packets coming from a network need a temporary storage before being examined and accepted by the device

I/O enables sharing of devices and also supports dedicated devices. A disk can be accessed and used by multiple users because of the I/O software.

Layers of the I/O software system

Figure 206: Layers of the I/O software system



Hardware constitutes the Input/ output device using a controller to communicate to a motherboard e.g. IDE, SCSI etc.

Interrupt handlers

Interrupts should be hidden from user by the operating system, so that as little of the operating system as possible knows about them. The ideal way to hide them is to have the driver starting an I/O operation block until the I/O has completed and the interrupt occurs. The driver can block itself by doing a down on a semaphore, a wait on a condition variable or a receive on a message.

Device drivers

Each I/O device attached to computer requires device specific code for controlling it. It is generally written by the device manufactures and delivered via a disk or via manufacturers website. Each device driver usually handles one device type, or one class of related devices. OS classify drivers as block devices (e.g. disks) or character devices (e.g. keyboard).

Device-independent I/O software

In spite of some of the I/O software being device specific, a large fraction of it is device independent. The rudimentary functions of the I/O software is to perform the I/O functions that are common to all devices and to provide a uniform interface to the user-level software thereby enabling uniform interfacing for device drivers, buffering, error reporting, release and allocation of dedicated devices and providing a device-independent block size.

- ✓ Input and output software layers

The I/O software design must ensure device independent where it should be possible to write programs that can access any I/O device without having to specify the device in advance. For instance, a program that reads a file as an input should be able to read a file on a flash disk, on a hard disk, or on a DVD-ROM, without having to alter the program for each different device.

The I/O software layers are categorized as: -

Hardware: This layer includes actual hardware and hardware controller which interact with the device drivers to ensure the hardware device is functional.

User Level Libraries: This layer provides simple interface to the user program to perform input and output functions e.g., stdio is a library provided by C and C++ programming languages.

Kernel Level Modules: This layer provides device driver to interact with the device controller and device independent I/O modules used by the device drivers.

Interrupts

The CPU hardware has an inbuilt mechanism that the CPU senses after executing every instruction. When the CPU detects that a controller has asserted a signal on the interrupt request line, the CPU saves small amount of state such as the current value of the instruction pointer and it jumps to the interrupt handler routine at a fixed address in memory.

Interrupt handler determines the cause of the interrupt, performs the required processing and executes a return from interrupt instruction to return to return the CPU to the execution state prior to the interrupt.

Sophisticated interrupt handling features need to be implemented in modern Operating System. Features such as ability to defer interrupt handling during critical processing, efficient way of dispatching properly interrupt handler for a device, multilevel interrupts to distinguish low level and high-level interrupts.

CPUs can use non-maskable interrupt (reserved for unrecoverable memory errors) or maskable interrupt (can be turned off by the CPU before the execution of critical instruction sequences that must not be interrupted). Maskable interrupt is used by device controllers to request service.

- ✓ Principles of input and output software
- ✓ Input and output software layers

Description of disk and disk operations.

- ✓ Structure

Disk provides the largest storage of computer system. It can be considered the I/O device common to each and every computer. Disk come in various sizes and speeds. Information inside the disk can be stored optically or magnetically. Magnetic tape was the earlier secondary storage device however the access time is much slower than for disks. Tapes are currently used for backup.

Modern disk drives are addressed as large one-dimensional arrays of logical blocks, where the logical block is the smallest unit of transfer. The Operating system, the nature of the I/O channel and disk controller hardware determines the actual details of disk I/O operation.

The basic unit of information storage is a sector stored on a flat, circular, media disk. The media spins close to one or more read/write heads which moves from the inner section of the disk to the outer section. During operation the disk rotates at a constant speed. The head must be positioned at the desired track and at the beginning of the desired sector on that track electronically. In summary, the main components of hard disk drive include: -

Platters: These are disk like structures present on the hard disk, stacked one above the other and store the data

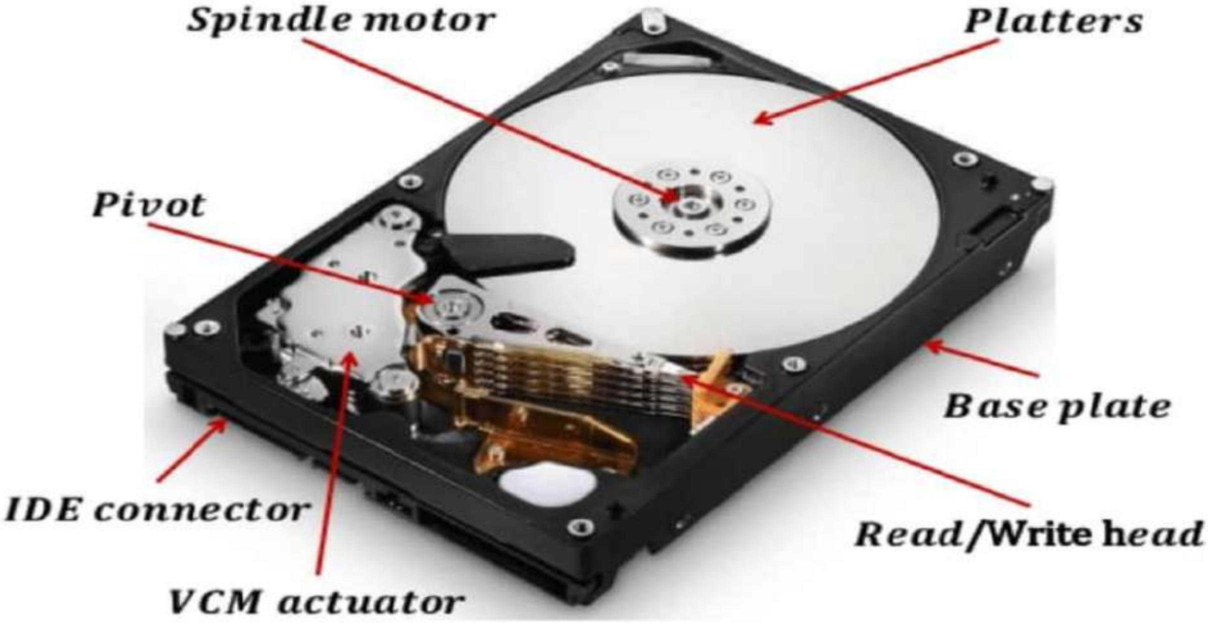
Actuator: It is a device, consisting of the read-write head that moves over the hard disk con to save or retrieve information

Spindle: It is the spinning shaft on which holds the platters in a fixed position such that it is feasible for the read/write arms to get the data on the disks

Head: it is a device on the arm of the hard drive that reads or writes data on the magnetic platters mounted on the surface of the drive

Cylinder are the circular tracks on the platters of the disk drive at equal distances from the center

Figure 207: Hard Disk



Sources: Info-savvy

- ✓ Operations

During operation the disk rotates at constant speed and to read or write, the head must be positioned at the desired track and sector. Selecting a track entails moving the head in a movable-head system and the time it takes to position the head at the track is known as **seek time.**

Upon selection of the track, the disk controller waits until the appropriate sector rotates to line up with the head. This is known as **rotational delay/rotational latency**.

The sum of the seek time and the rotational delay gives us the **access time** i.e. the time it takes to get into position to read or write.

Once the head is in position, the read write operation is then performed as the sector moves under the head a process known as **transfer time** i.e. (the data transfer portion of the operation; the time required for the transfer is the transfer tim**e)**

Seek time is the time needed to move the disk arm to the required track. It consists of two key components: the initial startup time and the time taken to traverse the tracks that have to be crossed once the access arm is up to speed.

On average disks have a rotational delay of about 2ms. However floppy disks by virtue of rotating between 300 and 600rpm have an average delay of between 50 and 100 ms. The transfer time to or from the disk depends on the rotation speed of the disk as displayed by the formula below:

$T = b/rN$

where

T = transfer time

b = number of bytes to be transferred

N = number of bytes on a track

r = rotation speed, in revolutions per second

✓ Disk arm scheduling algorithms

The amount of head needed to satisfy a series of I/O request affects the disk performance. If desired disk drive and controller are available, the request can be serviced immediately. If a device or controller is busy, any new requests for service will be placed on the queue of pending requests for that drive. When one request is completed, the operating system chooses which pending request is to be serviced next.

Examples of scheduling algorithms include: -

a. First In First Out (FIFO)

First Come, First Served scheduling algorithm (FCFS). It is an algorithm that processes items from the queue in a sequential order. This strategy has the advantage of being fair, because every request is honored in the order received. With FIFO, if there are only a few processes that require access and if many of the requests are to clustered file sectors, then we can hope for good performance. To improve throughput and queue length the disk can take advantage of priority (PRI) and last in first out transactions processing systems.

b. Shortest Seek Time First (SSFT)

SSFT uses a policy that selects the disk I/O request that requires the least movement of the disk arm from its current position. The main challenge is there may always be new requests arriving that will be chosen before an existing request SSTF provides better performance than FCFS algorithm. Under heavy load, SSTF can prevent distant request from ever being serviced, a phenomenon known as **starvation**. SSTF scheduling is essentially a form of shortest job first scheduling. SSTF scheduling algorithm are not very popular because of two reasons.

1. Possible existence of Starvation
2. It increases higher overheads.

c. SCAN

This algorithm used the head start at track 0 and move towards the highest numbered track, servicing all requests for a track as it passes the track. The service direction is then reserved and the scan proceeds in the opposite direction, again picking up all requests in order. SCAN algorithm is guaranteed to service every request in one complete pass through the disk. SCAN algorithm behaves almost identically with the SSTF algorithm. The SCAN algorithm is also called elevator algorithm.

d. Circular-SCAN (C-SCAN)

C SCAN Scheduling Algorithm restricts scanning to one direction only. Implying, when the last track has been visited in one direction, the arm is returned to the opposite end of the disk and the scan begins again. This reduces the maximum delay experienced by new requests.

e. LOOK

This algorithm starts with the head moving in one direction. It satisfies the request for the closest track in that direction. When there is no more request in the direction, the head is traveling, reverse direction and repeat. This algorithm is similar to innermost and outermost track on each circuit.

f. Circular LOOK (C-LOOK)

It is an enhanced version of both SCAN and LOOK disk scheduling algorithms. It uses the idea of wrapping the tracks as a circular cylinder as C-SCAN algorithm but the seek time is better than C-SCAN algorithm. Both C-SCAN and C-LOOK are used to avoid starvation and services requests more uniformly. In C-LOOK algorithm, the head services requests only in one direction(either left or right) until all the requests in this direction are not serviced and then jumps back to the farthest request on the other direction and service the remaining requests which gives a better uniform servicing as well as avoids wasting seek time for going till the end of the disk.

Disk Management

One of the functions of the Operating system is to manage all disks connected to a computer system.

Disk formatting is one of the activities performed during disk management. One can perform: -

a) Physical formatting or low-level formatting.

b) Logical Formatting

Physical Formatting

A disk must be formatted before storing data. Disk must be divided into sectors that the disk controllers can read/write. Low level formatting files the disk with a special data structure for each sector usually done at the factory.

Data structure consists of three fields: header, data area and trailer. Header and trailer contain information used by the disk controller. Sector number and Error Correcting Codes (ECC) contained in the header and trailer. For writing data to the sector – ECC is updated. For reading data from the sector – ECC is recalculated.

Logical Formatting

After disk is partitioned, the logical formatting follows where the operating system stores the initial file system data structures onto the disk.

- RAM disk

Boot Block

When a computer system is powered up or rebooted, a program in read only memory executes as the diagnostic check is done first. Stage 0 boot program is executed. Boot program reads the first sector from the boot device and contains a stage-1 boot program. Sometimes a boot sector may not contain a boot program. For a PC to boot from hard disk, the boot sector must contain a partition table. The code in the boot ROM instructs the disk controller to read the boot blocks into memory and then begins executing that code.

Swap Space Management

Swap space management is low level task of the operating system with the main of providing the best throughput for the virtual memory system.

Swap-Space Use

The operating system needs to release sufficient main memory to bring in a process that is ready to execute. Operating system uses this swap space in various way. Paging systems may simply store pages that have been pushed out of main memory. Unix operating system allows the use of multiple swap space are usually put on separate disks, so the load placed on the I/O system by paging and swapping can be spread over the systems I/O devices.

Swap Space Location

Swap space resides in two places: Separate disk partition or Normal file System. If the swap space is simply a large file within the file system, normal file system routines can be used to create it, name it and allocate its space. This is easy to implement but also inefficient. External fragmentation can greatly increase swapping times.

Catching is used to improve the system performance. Block of information is cached in the physical memory, and by using special tools to allocate physically continuous blocks for the swap file.

Swap space can be created in a separate disk partition. No file system or directory structure is placed on this space. A separate swap space storage manager is used to allocate and deallocate the blocks. This manager uses algorithms optimized for speed. Internal fragmentation may increase. Some operating systems are flexible and can swap both in raw partitions and in file system space.

- RAID

Redundant Array of Independent Disks is a way of combining the storage power of more than one hard disk for a special purpose such as increased performance or fault tolerance (CompTIA, 2013).

RAID is more commonly done with SCSI drives, but it can be done with IDE drives too.

A RAID System should provide: -

- ✓ *Reliability*: How many disk faults can the system tolerate?
- ✓ *Availability*: How available is the system for actual use?
- ✓ *Performance*: How good is the response time? How high is the throughput (rate of processing work)?
- ✓ *Capacity*: Given a set of N disks each with B blocks, how much useful capacity is available to the user?

There are several types of RAID. The following are the most commonly used RAID levels:

RAID 0 - disk striping.

Data is written across multiple drives, so one drive can be reading or writing while the next drive's read-write head is moving. This makes for faster data access. However, if any one of the drives fails, all content is lost. Technically it is not RAID, since it doesn't provide fault tolerance.

RAID 1 - disk mirroring. This is a technique of producing fault tolerance by writing all data simultaneously to two separate drives.

when one drive fails, the other drive contains all the data and can be switched to. However, disk mirroring doesn't help access speed, and it is

costly compared to having a single drive.

RAID 5 Combines the benefits of both RAID 0 and RAID 1. It uses a parity block distributed across all the drives in the array and strips the data across the disks. That way, if one drive fails, the parity information can be used to recover what was on the failed drive.

Ideally a minimum of three drives is required.

Computer clock system.

Most computing devices have hardware clocks and timers that provide three basic functions:

1. Give the current time

2. Give the elapsed time

3. Set a timer to trigger operation X at time T

These functions are used heavily by the operating system, and also by time sensitive applications. Programmable interval timer measures elapsed time and trigger operations.

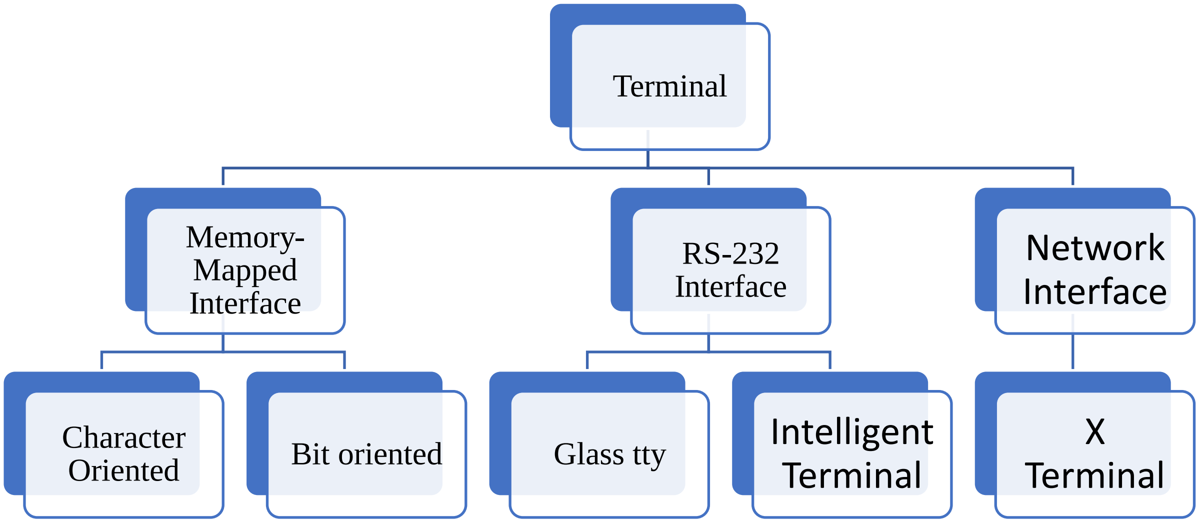
Computer terminals.

Terminals provide a means via which devices communicate with the computer consistently. Computer terminals can be referred to as an electronic or electromechanical hardware device that is used for entering data into, and displaying data from, a computer or a computing system.

- ✓ **Terminal hardware**

Based on how the operating system communicates with them, as well as their actual hardware features, terminals can be divided into three different categories namely: Memory-mapped terminals (consists of a keyboard and a display hardwired to the computer), RS-232 Interface terminals (terminals that interface the computer via a serial interface eg modems, mainframes) and network interface terminals (eg the X terminal)

Figure 208: Terminal Types



Memory-Mapped Terminals

These are an integral part of the computers. They consist of a display and a keyboard. Memory mapped displays are interfaced via a special memory called a video RAM, which forms part of the computer's address space and is addressed by the CPU the same way as the rest of memory. These terminals use a chip called video controller on the video RAM card to pull bytes out of the video RAM and generate the video signal used to drive the display. Displays are usually one of two types: CRT monitors or flat panel displays.

A ***CRT monitor*** generates a beam of electrons that scans horizontally across the screen, painting lines on it. Typically, the screen has 480 to 1200 lines from top to bottom, with 640 to 1920 points per line. These points are referred to as ***pixels***.

The ***Flat panel monitor*** uses a thin panel design instead of a traditional cathode ray tube (CRT) design. The screens are much lighter and thinner, and can be much more portable than traditional televisions and monitors. The video controller signal modulates the intensity of the electron beam, determining whether a given pixel will be light or dark. Color monitors have three beams, for red, green, and blue, which are modulated independently.

RS-232 Terminals

RS-232 terminals are devices that communicates using a serial interface, one bit at a time. These terminals use either a 9-pin or 25-pin connector, of which one pin is used for transmitting data, one pin is for receiving data, and one pin is ground. The other pins are for various control functions, most of which are not used.

To send a character to an RS-232 terminal, the computer must transmit it 1 bit at a time, prefixed by a start bit, and followed by 1 or 2 stop bits to delimit the character. A parity bit which provides rudimentary error detection may also be inserted preceding the stop bits, although this is commonly required only for communication with mainframe systems.

Common transmission rates are 14,400 bits/sec (for fax) and 56,000 bits/sec (for data). RS-232 terminals are commonly used to communicate with a remote computer using a modem and a telephone line.

- ✓ **Terminal software**

Terminal software makes it possible for devices to function in a computing system. For instance the keyboard and the display are almost independent devices thus can be split into distinct drivers.

Input software

The main task is to collect input from the keyboard and pass it to user programs when they read from the terminal. Its job is to accept input and pass it upward unmodified by getting a sequence of ASCII codes. The can either use raw mode (Non canonical) or cooked mode (Canonical) to read the sequence of inputs.

Output software

RS-232 terminals use output buffers associated with each terminal to display content. The buffers may come from the same pool as the input buffers, or be dedicated, as with input. When programs write to the terminal, the output is first copied to the buffers. Similarly, output from echoing is also copied to the buffers. After all the output details have been copied to the buffers (or the buffers are full), the first character is output, and the driver goes to sleep. When the interrupt comes in, the next character is output, and so on.

Virtual devices.

In operating systems, a virtual device refers to a device file which has no physical hardware associated with it. It only exists in software form. It makes the real computer believe that a particular hardware exists when it really does not.

- ✓ Objectives of virtual device

The main objective of the virtual device is to create a virtual environment where a technician can use to fix an error in the operating system or a bug/virus detected by supposing that an external device is monitoring it.

There exist several virtual devices such as RAM, loopback device, pipe, socket, video and media to mention a few.

These devices use virtual device drivers to function. These drivers allow more than one application to access the same memory or physical device without conflict of interest.

Pipe device requires a single address at the other end, socket requires one address and one port number at the other end to communicate.

- ✓ Spooling

A spool is a buffer that holds output for a device, such as a printer, that cannot accept interleaved data streams. The spooling system copies the queued spool files to the printer one at a time. system daemon process is used to handle spooling in some operating systems while in other operating systems kernel thread is used.

- ✓ Buffering

A buffer is a memory area that stores data temporarily while they are transferred between two devices or between a device or an application. Buffering is done for three reasons.

1. To cope with a speed mismatch between the producer and consumer of a data stream.
2. To adapt between devices that have different data transfer sizes.
3. To support copy semantics for application I/O.

- ✓ Caching

A cache is region of fast memory that holds copies of data. Access to the cached copy is more efficient than access to the original memory.

Caching and buffering are two distinct functions, but sometimes a region of memory can be used for both purposes.

7.2.4.4 Learning Activities

Learning Activity One

In this learning activity, trainees consider a number of computing devices to determine what types of inputs and outputs they use. Groups are assigned to a computing device and based on a trainer-provided definition of input and output, list the inputs and outputs of their device.

Earlier in the activity students are prompted to focus on more obvious physical inputs and outputs (e.g. a keyboard as an input or a screen as an output) but later discussions lead students to consider less obvious examples (e.g. that a touch screen is both an input and output, or the fact that the Internet can serve as both input and output). Throughout the learning activity the trainer records inputs and outputs that are identified on a T-Chart at the front of the room (on the whiteboard).

To conclude the lesson trainees, examine common activities they do on a computing device and select the inputs and outputs used for that activity from the chart.

Learning Objectives

1. Identify the inputs and outputs of common computing devices
2. Select the inputs and outputs used to perform common computing tasks

Brainstorming Inputs and Outputs

Group students in to 2 or 3 then distribute copies of Input and Output activity guide to each group

Project a document where the trainer will record all th inputs and outputs students brainstorm in this activity.

Possible Responses

Inputs Keyboard, mouse, other buttons, camera, microphone,

Outputs Screen, Speakers, Printer, Plotters

Have trainees list all of the possible inputs and outputs to a laptop.

Possible Responses

Inputs Keyboard, touchpad, USB port, trackpad, camera, microphone, Wi-Fi, Bluetooth

Outputs Screen, Speakers, Wi-Fi, Bluetooth

Have trainees list all of the possible inputs and outputs to a smartphone by asking them the following questions:

1. How does the phone know where it is?
2. How does Apple know that you got the phone wet?
3. How does the phone know it has to shut down when it is too hot?
4. How does the phone know to turn off the screen when you put the phone up to your ear?

5. How does the phone know to switch from playing music through the speakers to playing music through the headphones?

7.2.4.5 Self-Assessment

A. A hard disk with **200 tracks** has disk schedulers that receives the read and write requests. The requested tracks, in the order received by the disk scheduler, are **55, 58, 39, 18, 90, 160, 150, 38, 184**. Assuming that currently the read/write head is on track 100, use diagrams to show how the requests will be serviced using the following disk scheduling policies; **First In First Out (FIFO)**, **Shortest Service Time First (SSTF)** and the **CSCAN**

7.2.4.6 Tools, Equipment, Supplies and Materials

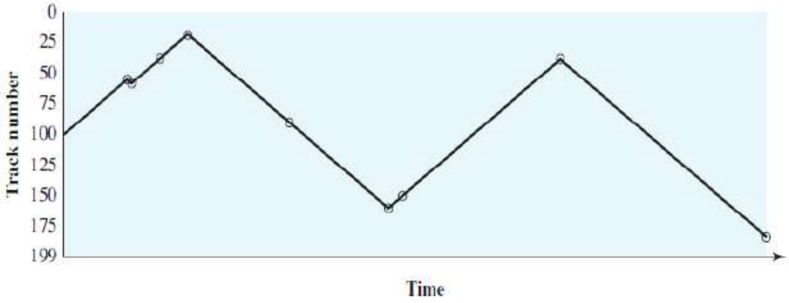
- A functional laptop
- A functional Personal Computer
- Internet
- Activity guide

7.2.4.7 Model answers to self-assessment

A. A hard disk with **200 tracks** has disk schedulers that receives the read and write requests. The requested tracks, in the order received by the disk scheduler, are **55, 58, 39, 18, 90, 160, 150, 38, 184**. Assuming that currently the read/write head is on track 100, use diagrams to show how the requests will be serviced using the following disk scheduling policies; **First In First Out (FIFO)**, **Shortest Service Time First (SSTF)** and the **CSCAN**

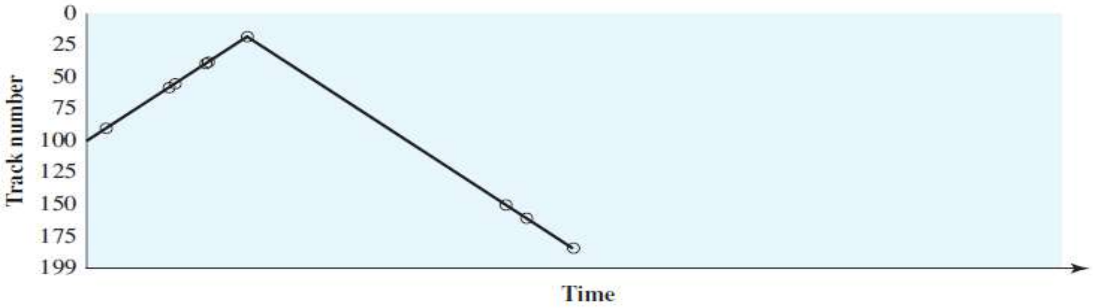
First in First Out (FIFO)

- The policy requires that an arriving process joins the rear of the queue
- Each process waits for its turn to process i.e. they are admitted into the processor in the order in which they arrived



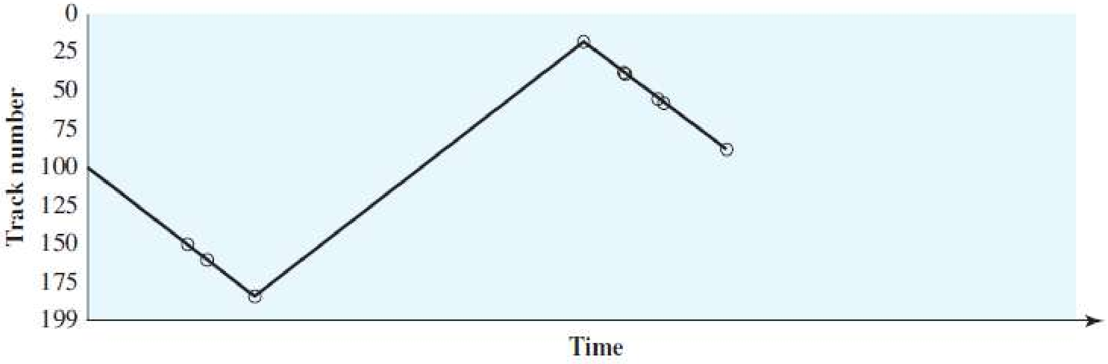
Shortest Service Time First (SSTF)

- The SSTF policy is to select the disk I/O request that requires the least movement of the disk arm from its current position.
- Thus, we always choose to incur the minimum seek time.



C-SCAN

- The C-SCAN (circular SCAN) policy restricts scanning to one direction only
- Thus, when the last track has been visited in one direction, the arm is returned to the opposite end of the disk and the scan begins again.
- This reduces the maximum delay experienced by new requests.



7.2.4.7 References

Abraham, P. a. (2013). *Operating Systems Concepts.* United States of America: Wiley.

Abraham, P. G. (2013). *Operating Systems Concepts* . Hoboken: John Wiley & Sons, Inc.

CompTIA. (2013). *CompTIA A+.* (Vol. 2013, Issue December 2). http://certification.comptia.org/getCertified/certifications/a.aspx

Li, A. W. A. and K. (n.d.). *Virtual Memory Primitives for User Programs.pdf*.

Maccabe, A., Bridges, P., Brightwell, R., & Riesen, R. (n.d.). *Recent Trends in Operating Systems and their Applicability to*.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2017). *OPERATING SYSTEM ls s s i t*.

Stallings, W. (2014). Operating Systems: Internals E Designs Principles. In *Journal of Chemical Information and Modeling*.

7.2.5 Learning Outcome 5 Identifying concepts of file management.

7.2.5.1 Introduction to the learning outcome

In the computer world, one needs to master how to efficiently manage computer files including how to find, download, rename, recover, copy and organize files. This learning outcome helps one learn how to administer computer system to correctly handle digital data. This improves the overall function of an organization's workflow.

7.2.5.2 Performance Standard

7.2.5.2.1 Definition of file system management is done.

7.2.5.2.2 system concepts are identified.

7.2.5.2.3 Objectives of file management are identified.

7.2.5.2.4 File access methods are identified.

7.2.5.2.5 Description of directory implementation is done.

7.2.5.2.6 File allocation techniques are identified.

7.2.5.2.7 File protection and security are identified.

7.2.5.3 Information Sheet

- Definitions of terms, Methods, Processes/ procedures/ guidelines, Illustrations (photographs, pictures, videos, charts, plans, digital content links, simulations links) and case studies

- Provides additional information sources related to the learning outcome e.g. books, web links

File system management.

File system is concerned with managing secondary storage space particularly disk storage in computer system. It consists of two distinct parts:

- ✓ A collection of files each storing related data
- ✓ A directory structure which organizes and provide information about all the files in the system.

A file is a named collection of related information that is recorded on secondary storage. It is basically an abstract data type defined and implemented by the operating system. Sometimes referred to a sequence of logical records such as a byte, a line, or a more complex data item.

File system concepts.

- ✓ *File Naming*

  A file is named for convenience of its human users and a name is usually a string of characters e.g. "MemoryFile.docx". When a file is named it becomes independent of the process, the user and even the system that created it i.e. another user may edit the same file and specify a different name.

- ✓ File Structure

A **file** has certain defined structure according to its type namely; Text file, Source file, executable file and object file.

Terms associated with the use of file include;

Field which is the basic element of data and it contains single value

Record is a collection of related fields that can be treated as a unit

Database is a collection of related data and the database itself consists of one or more types of files.

- ✓ File Types

OS should recognize and support different files types so that it can operate and file in reasonable ways. Files types are implemented by including it as part of file name. The name is split into 2 parts – a *Name* and an *Extension* usually separated by a period character. The extension indicates the type of file and type of operations that can be on that file.

File Type	**Usual extensions**
Executable	.exe , .com
Source Code	.c, .p, .pas, .bas, .vbp, .prg, .java
Text	.txt, .rtf, .doc
Graphics	.bmp, .gpg, .mtf
Spreadsheet	.xls
Archive	.zip, .arc.rar (compressed files)

- ✓ File Management Systems

It is the set of system software that provides services to users and applications in the use of files. The main objective of file management system is:

- ✓ To meet data management needs and requirements of the user.
- ✓ To guarantee validity of the data in the file.
- ✓ To optimize performance both from the system point of view in terms of overall throughput
- ✓ To eliminate potential for lost or destroyed data.
- ✓ To provide I/O support for multiple users n the case of multiple user systems
- ✓ To provide input/output support for a variety of storage device types among others

- ✓ File structure

At the lowest level, device drivers communicate directly with peripheral devices or their controllers or channels. The driver is responsible for starting I/O operations on a device and processing the I/O request operation. Typical devices for file operations include disk and tape drives. Device driver can be thought of as a translator thus it considered part of the operating system. File organization module knows the type of file allocation used and the location of the file and it translates the numbered logical block address to physical addresses (contains the data) for the basic file system to transfer. File organization includes the free-space manager which tracks unallocated space and avails these blocks to the module whenever requested.

The logical file system responsible for protection and security, uses the directory structure to provide the file organization module with the information the latter needs given a symbolic file name. To create a new file the logical file system reads the appropriate directory into memory, updating it with the new entry and writes it back to the disk. Upon finding the file the associated information like owner, size, access permission and data block locations are copied into a table in memory (open file table). Open file table constitutes information about all the currently opened files.

The first reference to a file causes the directory structure to be searched and the directory entry for this file to be copied into the table of opened files. The index into this table is returned to the user program and all further references are made through the index rather than with a symbolic name.

The index name varies with the OS; Windows refers to it as file handle, Unix as a file descriptor and other systems as a file control block.

Once the file is closed by all users that had opened it, the updated file information is copied back to the disk-based directory structure.

File system mounting

A file system must be mounted before it can be availed to processes on the system. The system is given the name of the device and the location within the file structure at which to attach the file system (mount point). The OS verifies that the device has a valid file system then notes in its directory structure that a file system is mounted at the specified mount point. This arrangement ensures the OS traverses its directory structure switching among file systems appropriately.

- ✓ Attributes

A part from the name other attributes of files include

Size – amount of information stored in the file

Location – Is a pointer to a device and to the location of file on the device

Protection – Access control to information in file (who can read, write, execute and so on)

Time, *Date* and *User* identification –This information is kept for creation, last modification and last use. These data can be useful for protection, security and usage monitor.

Volatility – frequency with which additions and deletions are made to a file

Activity – Refers to the percentage of file's records accessed during a given period of time.

File Implementation

It is concerned with issues such as file storage and access on the most common secondary storage medium the hard disk. It explores ways to allocate disk space, to recovers freed space, to track the locations of data and to interface other parts of the o/s to the secondary storage.

Directory implementation

The selection of directory allocation and directory management algorithms has a large effect on the efficiency, performance and reliability of the file system.

Algorithms used:

- ✓ *Hash Table* – A linear list stores the directory entries but a hash data structure is used. The hash table takes a value computed from the file name and returns a pointer to the file name in the linear list.
- ✓ *Linear* is the simplest method of implementing a directory. It uses a linear list of filenames with pointers to the data blocks. It requires a linear search to find a particular entry.

Free Space Management

To keep track of free disk space the system maintains a free space list which records all disk blocks that are free; those not allocated to some file or directory. To create a file, we search the free space list for the required amount of space and allocate that space to the new file. This space is then removed from the free space list. Techniques used to manage disk space include: -

- ✓*Bit vector* – Free space list is implemented as a bit map or bit vector. Each block is represented by 1 bit. If the block is free, the bit is 1; if the block is allocated, the bit 0.
- ✓*Linked List* – links together all the free disk blocks, keeping a pointer to the first free block in a special location on the disk and caching it in memory. This first block contains a pointer to the next free disk block and so on.
- ✓*Grouping* – Is a modification of free list approach and stores the addresses of n free blocks in the first free block. The first n-1 of these blocks are actually free blocks and so on. The importance of this implementation is that the addresses of a large number of free blocks can be found quickly unlike in the standard linked list approach.
- ✓*Counting* – it takes advantage of the fact that generally several contiguous blocks may be allocated or freed simultaneously, particularly when space is allocated with contiguous allocation algorithm or through clustering. Thus, rather than keeping a list of n-free disk addresses we can keep the address of the first free block and the number n of free contiguous blocks that follow the first block. Although each entry requires more space than would a simple list address; the overall list will be shorter as long as the count is generally greater than 1.

Efficiency and Performance

This is essential when it comes to files and directory implementation since disks tends to be a major bottleneck in system performance (they are the slowest main computer component). To improve performance, disk controllers are provided to include enough local memory. This creates on-board cache that may be sufficiently large to store an entire track at a time.

File Sharing

In a multi user system there is almost a requirement for allowing files to be shared among a number of users. Two issues arise: -

- ✓*Access rights* – should provide a number of options so that the way in which a particular file is accessed can be controlled. Access rights assigned to a particular file include: Read, execute, append, update, delete.
- ✓*Simultaneous access* – a discipline is required when access is granted to append or update a file to more than one user. An approach can allow a user to lock the entire file when it is to be updated.

File Operations

A File can be manipulated by operations such as:

- ✓ Open – prepare a file to be referenced
- ✓ Copy – create another version of file with a new name.
- ✓ Destroy – Remove a file
- ✓ Rename – change name of a file.
- ✓ List – Print or display contents.
- ✓ Move – change location of file
- ✓ Close – prevent further reference to a file until it is reopened
- ✓ Create – Build a new file

Individual data items within the file may be manipulated by operations such as:

- ✓ Read – Input a data item to a process from a file
- ✓ Write – Output a data item from a process to a file
- ✓ Update – modify an existing data item in a file
- ✓ Insert – Add a new data item to a file
- ✓ Delete – Remove a data item to a file

- ✓ Truncating – delete some data items but file retains all other attributes

File Structure

Refers to internal organization of the file. File types may indicate their structure. Certain files must conform to a required structure that is understood by the host OS. Some Operating Systems have file systems that do support multiple structure while others impose (and support) a minimal number of file structures e.g. MS DOS and UNIX.

UNIX considers each file to be a sequence of 8-bit bytes. Macintosh OS supports a minimal no of file structure and it expects executables to contain 2 parts – a resource fork and a data fork. Resource fork contains information of importance to user e.g. labels of any buttons displayed by program.

File access methods.

To grasp the concept of file access method it is crucial that we understand the concept of file organization. File organization refers to the manner in which records of a file are arranged on secondary storage device.

The most popular schemes of file organization include: -

Direct access where records are directly (randomly) accessed by their physical addresses on a direct access by storage device (DASD). The application user places the records on DASD in any order appropriate for a particular application.

Sequential access where records are placed in physical order. The next record is the one that physically follows the previous record. It is used for storing and accessing records in magnetic tape.

Indexed Sequential access where records are arranged in logical sequence according to a key contained in each record. The system maintains an index containing the physical addresses of certain principal records. Indexed sequential records may be accessed sequentially in key order or they may be accessed directly by a search through the system created index. It is usually used in disk.

Partitioned access is a file of sequential sub files. Each sequential sub file is called a member. The starting address of each member is stored in the file directory. Partitioned files are often used to store program libraries.

File allocation techniques.

Deals with how to allocate disk space to different files so that the space is utilized effectively and files can be accessed quickly. Methods used include: -

1. Contiguous Allocation

This is where files are assigned to contiguous areas of secondary storage. A user specifies in advance the size of area needed to hold a file to be created. If the desired amount of contiguous space is not available the file cannot be created.

Advantages

- ✓ Speeds up access since successive logical records are normally physically adjacent to one another
- ✓ File directories are relatively straight forward to implement for each file merely retain the address of start of file and its length.

Disadvantages

- ✓ Difficult to find space for a new file especially if the disk is fragmented into a number of separate holes (block). To prevent this, a user can run disk defragmenter (a repackaging routine)
- ✓ Another difficulty is determining how much space is needed for a file since if too little space is allocated the file cannot be extended or cannot grow.

Some Operating systems use a modified contiguous allocation scheme where a contiguous chunk of space is allocated initially and then when that amount is not large enough, another chunk of contiguous space, an extent is added.

2. Non-Contiguous Allocation

Files do tend to grow or shrink overtime and because users rarely know in advance how large a file will be, non-contiguous allocation is critical. Techniques used include:

- ✓ *Linked Allocation*

It solves all problems of contiguous allocation. Each file is a linked list of disk blocks, the disk blocks may be scattered anywhere on the disk. The directory contains a pointer to the first and last disk block of the file. Each Block contains a pointer to the next block.

Its main advantage is that it eliminates external fragmentation. However, it is very effective only for sequential access files i.e. where you have to access all blocks but space may be wasted for pointers since they take 4 bytes.

The solution is to collect blocks into multiples clusters rather than blocks.

The linked allocation is not very reliable since loss or damage of pointer would lead to failure

b) Indexed Allocation

It tries to support efficient direct access which is not possible in linked allocation. Pointers to the blocks are brought together into one location (index block). Each file has its own index block, which is an array of disk block addresses. Its advantage is that it supports direct access without suffering from external fragmentation. The disadvantages include wasted space due to pointer overhead of the index block and the challenge of determining how large the index block would be.

The mechanisms to deal with this include: -

- ✓ **Linked scheme** – index block is normally one disk block; to allow large files we may link together several index blocks.
- ✓ **Multilevel index** – use a separate index block to point to the index blocks which point to the files blocks themselves
- ✓ **Combined scheme** – keeps the first say 15 pointers of the index block in the file's index block. The first 12 of these pointers point to direct blocks that contain addresses of blocks that contain data of the file. The next 3 pointers point to indirect blocks which may contain address to blocks containing data.

File protection and security.

Computer system contains many objects, and they need to be protected from tampering or misuse. Objects such as memory, CPU time, and I/O devices or software such as files, programs and semaphores need to be safe guarded at all times. Access right is permission to perform an operation on an object. Processes execute in domains and may use any of the access rights in the domain to access and manipulate objects.

File protection improves reliability by detecting any errors at the interfaces between component systems. Early detection is critical in prevention of corruption of a file. A protection-oriented system provides a mechanism for the enforcement of the policies governing utilization of the resources. The main role of protection in a computer system is to provide a way for the enforcement of the policies governing resource use. Some policies are fixed in the design of the system, while others are formulated by the management of a system. Others can be defined by individual users to protect their own files and programs.

The principle of least privilege dictates that programs, users and even systems be given just enough privileges to perform their tasks so that failure or compromise of a component does minimum damage. This results in a more secure computing environment. Policy decisions concerning protection can be implemented by the access matrix. To change the contents of the access-matrix entries requires three additional operations: copy, owner and control.

- ✓ Access control

Access rights defines the access control. To achieve a secure environment access matrix (a model for protection is used. Access-matrix scheme provides us with the mechanism for specifying a variety of policies i.e. it ensures that a process executing in a particular domain can access only objects within that domain. The permissions allowed for the files include Read, Write and Execute

- ✓ Audit trail

It is a record of the changes that have been made to database file. It is a mechanism that traces the detailed transactions relating to any process in the OS. It provides support documentation and history that is used to authenticate security and operational actions

CASE STUDY

In this case study trainees will form groups and carry out a case study resulting in an oral presentation with slides.

General Requirements

1. The presentation should be in English.
2. All members of the group must participate in preparing the presentation.
3. The presentation should be 20 to 25 minutes.
4. You should prepare and use slides PowerPoint or Google slides.
5. Each group should bring their own laptop to the presentation.
6. All members of the group must talk during the presentation.
7. Not all members of the group must talk for an equal amount of time.
8. Every presentation will end with a discussion.
9. The first slide should include the following:

    a. Title of the presentation.

    b. Course name, Course code and year

    c. Names of the group members participating in the presentation in alphabetical order.

10. The presentation should end with a list of all referenced material such as URLs to websites, articles etc.

CASES

Table 55: OS CASES

Case	Operating system	Description
1	Linux	Linux is a Unix-like and mostly POSIX-compliant computer operating system (OS) assembled under the model of free and open-source software development and distribution.
2	VxWorks	VxWorks is a real-time operating system (RTOS) designed for use in embedded systems requiring real-time, deterministic performance and, in many cases, safety and security certification, for industries, such as aerospace and defense, medical devices, industrial equipment, robotics, energy, transportation, network infrastructure, automotive, and consumer electronics.
3	Android	Android is a mobile operating system (OS) currently developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such

Case	Operating system	Description
		as smartphones and tablets.
4	Contiki	Contiki is an operating system for networked, memory-constrained systems with a focus on low-power wireless Internet of Things devices. Extant uses for Contiki include systems for street lighting, sound monitoring for smart cities, radiation monitoring, and alarms. It is open-source software released under a BSD license.
5	TinyOS	TinyOS is an embedded, component-based operating system and platform for low-power wireless devices, such as those used in wireless sensor networks (WSNs), smartdust, ubiquitous computing, personal area networks, building automation, and smart meters.

Suggested outline of the presentation

This is just a suggestion for topics to include in your presentation and you may choose to exclude or rearrange parts of the list and/or include other topics you find relevant in the context of your case study.

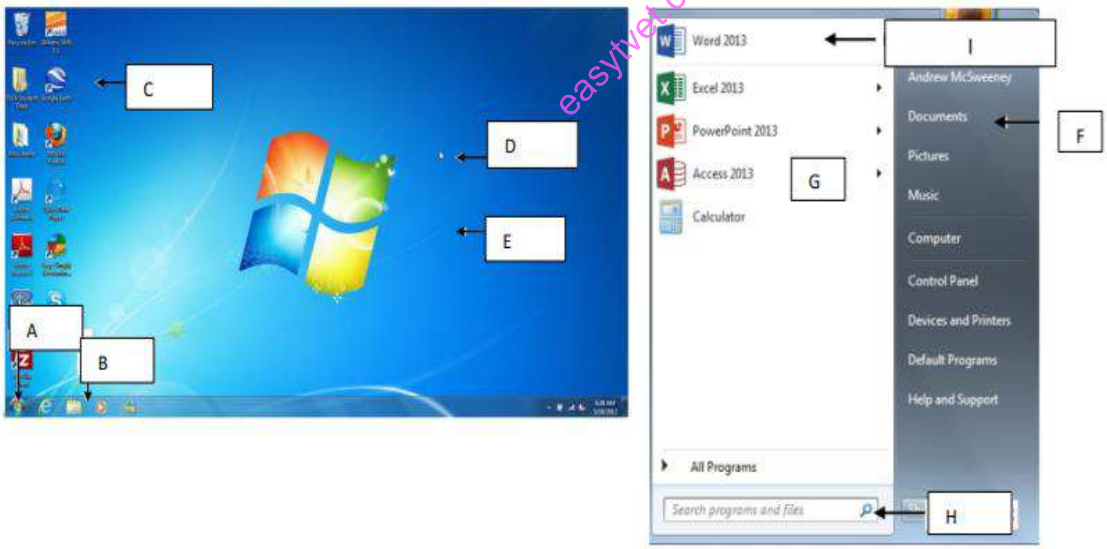
1. Title slide (mandatory)
    a. Title
    b. Course name, course code and year (2018)
    c. Names of the group members participating in the presentation in alphabetical order.
2. History, background and overall purpose of the OS.
3. Is it
    a. Embedded OS?
    b. Real-time OS?
    c. Other OS?
4. Concurrent programming/execution.
    a. Processes, threads or similar?

   b. Inter-process communication or similar?
   c. Synchronization?
5. Scheduling algorithm(s)?
6. Power management?
7. Memory management?
8. Example(s) of real-world usage?
9. List of references (mandatory).

7.2.5.4 Learning Activities

A. Match the following terms to the images shown below. Then state their purpose.

Useful system folders, Desktop, Desktop icons, Mouse pointer, Taskbar, Start button, Recently used programs, Pinned area, Search box

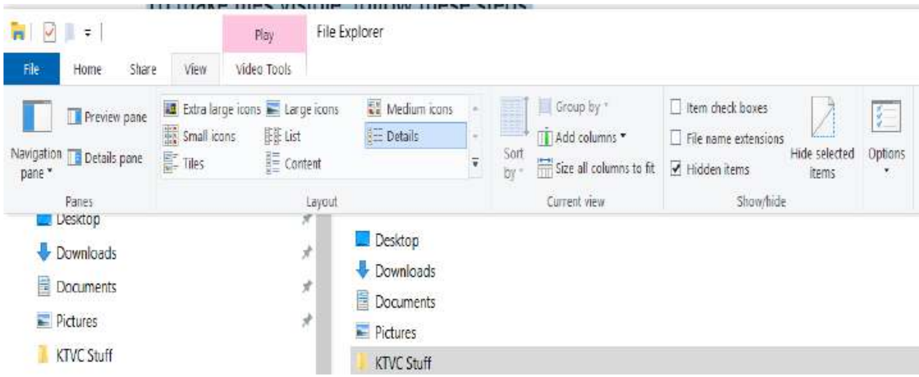


B. Activity to make a file visible in Windows 10 OS

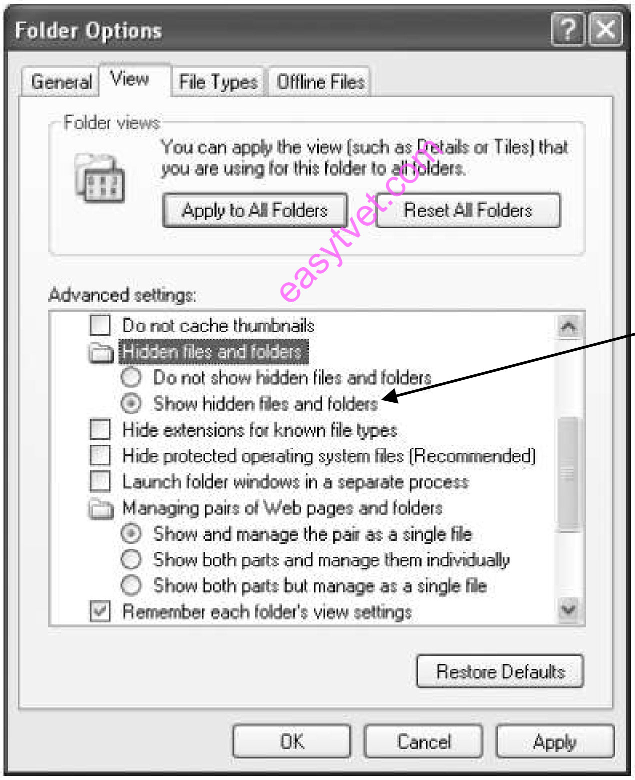
Follow these steps:

1. Open Windows Explorer.
2. Select the View tab, and scroll until you find the Hidden items checkbox.

Check to hide/Uncheck to hide



3. Alternatively Right click the Folder. Once the Folder Options dialog box is shown click on Hidden files and folders.


Folder Options

General | View | File Types | Offline Files

Folder views

You can apply the view (such as Details or Tiles) that you are using for this folder to all folders.

Apply to All Folders | Reset All Folders

Advanced settings:

- □ Do not cache thumbnails
- Hidden files and folders
  - ○ Do not show hidden files and folders
  - ● Show hidden files and folders
- □ Hide extensions for known file types
- □ Hide protected operating system files (Recommended)
- □ Launch folder windows in a separate process
- Managing pairs of Web pages and folders
  - ● Show and manage the pair as a single file
  - ○ Show both parts and manage them individually
  - ○ Show both parts but manage as a single file
- ☑ Remember each folder's view settings

Restore Defaults

OK | Cancel | Apply

Click to display the hidden files and folders in the Computer

4. Check the box to hide. Uncheck the box to unhide a file

C. *File protection and security activity*

Study the computer laboratories and offices in your learning institution and asses the following details: -

No	**Physical Security Checks for rooms with Computers**		
	Observation Checkpoints	Yes	No
1	Has the room or facility been constructed with full-height walls?		
2	Does each secure room or facility have low visibility (e.g., no unnecessary signs)?		
3	Has the room or facility been constructed with a fireproof ceiling?		
4	Are there two or fewer doorways?		
5	Are doors solid and fireproof?		
6	Are doors equipped with locks?		
7	Are window openings to secure areas kept as small as possible?		
8	Are keys and combinations to door and window locks secured responsibly?		
9	Have alternatives to traditional lock and key security measures (e.g., bars, anti-theft cabling, magnetic key cards, and motion detectors) been considered?		
10	Have both automatic and manual fire equipment been properly installed?		
11	Are personnel properly trained for fire emergencies?		
12	Are acceptable room temperatures always maintained (i.e., between 50- and 80-degrees Fahrenheit)?		
13	Are acceptable humidity ranges always maintained (i.e., between 20 and 80 percent)?		
14	Are eating, drinking, and smoking regulations in place and enforced?		
15	Has all non-essential, potentially flammable, material (e.g., curtains and stacks of computer paper) been removed from secure areas?		
16	Are up-to-date records of all equipment brand names, model names, and serial numbers kept in a secure location?		
17	Have qualified technicians (staff or vendors) been identified to repair critical equipment if and when it fails?		

18	Has contact information for repair technicians (e.g., telephone numbers, customer numbers, maintenance contract numbers) been stored in a secure but accessible place?		
19	Has all equipment been labeled in a covert way that only authorized staff would know to look for (e.g., inside the cover)?		
20	Have security staff been provided up-to-date lists of personnel and their respective access authority?		
21	Are surge protectors used with all equipment?		
22	Are Uninterruptible Power Supplies (UPSs) in place for critical systems?		
23	Has consideration been given to the use of electrical outlets so as to avoid overloading?		
24	Are all paper copies of sensitive information shredded before being discarded?		
25	Are photocopiers, fax machines, and scanners kept in open view?		
26	Are all paper copies of sensitive information shredded before being discarded?		
27	Is everything, including passwords, encrypted before leaving user workstations?		
28	Is the identity of information recipients verified before transmission?		
29	Is the handling of sensitive information restricted to authorized personnel?		
30	Does staff know to communicate security concerns immediately?		

In your opinion how secure is your learning institution in terms of file protection and security?

At the end of the activity the trainee is expected to write a comprehensive observation report based on the self-assessment on the physical security checks in the trainees' learning Institutions Computer Labs

7.2.5.5 Self-Assessment

A. Describe the term the Desktop as used in computing?

B. Identify file attributes available to files on a FAT32 partition

C. Name the system tool used to configure Virtual memory

D. Which of the following partitions is specifically the partition from which the operating system boots?

A. Active partition

B. Logical partition

C. Primary partition

D. Extended partition

E. How does the distinction between kernel mode and user mode function as a basic form of protection (security)?

7.2.5.6 Tools, Equipment, Supplies and Materials

- Computer lab
- Written permission to conduct the exercise from the trainer charge
- Internet.
-

7.2.5.7 Model answers to self-assessment

A. Describe the term the Desktop as used in computing?

It is the virtual desk upon which all of your other programs and utilities run

B. Identify file attributes available to files on a FAT32 partition

Hidden, Read Only, Archive, System. Encryption and Compression are found in NTFS.

C. Name the system tool used to configure Virtual memory

System control panel or the Performance tab.

D. Which of the following partitions is specifically the partition from which the operating system boots?

A. Active partition

B. Logical partition

C. Primary partition

D. Extended partition

The operating system boots from the active partition. An active partitions must be primary partitions, even though a primary partition does not have to be active partition (a single hard drive can support up to four primary partitions.

E. How does the distinction between kernel mode and user mode function as a basic form of protection (security)?

The distinction between kernel mode and user mode provides a basic form of protection in the following manner. Certain instructions can only be executed when the CPU is in kernel mode thus, hardware devices could be accessed only when the program is executing in kernel mode. Control over when interrupts could be enabled or disabled is also possible only when the CPU is in kernel mode. Consequently, the CPU has very limited capability when executing in user mode, consequently enforcing protection of critical resources.

7.2.5.7 References

Abraham, P. a. (2013). *Operating Systems Concepts*. United States of America: Wiley.

Abraham, P. G. (2013). *Operating Systems Concepts* . Hoboken: John Wiley & Sons, Inc.

CompTIA. (2013). *CompTIA A+*. (Vol. 2013, Issue December 2). http://certification.comptia.org/getCertified/certifications/a.aspx

Li, A. W. A. and K. (n.d.). *Virtual Memory Primitives for User Programs.pdf*.

Maccabe, A., Bridges, P., Brightwell, R., & Riesen, R. (n.d.). *Recent Trends in Operating Systems and their Applicability to*.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2017). *OPERATING SYSTEM ls s s i t*.

Stallings, W. (2014). Operating Systems: Internals E Designs Principles. In *Journal of Chemical Information and Modeling*.

7.2.6 Learning Outcome 6 Identifying Emerging trends in Operating system.

7.2.6.1 Introduction to the learning outcome

Computing is a dynamic field. Devices used 10 years ago have been replaced by better versions. Same applies to operating systems as well as application programs. A collection of recent and innovative works in the field Network Security, Cryptography, Cloud Computing, Big Data Analytics, Data Mining & Data Warehouse, Communication, Nanotechnology, VLSI, Image Processing etc. requires OS that can support them. In this learning guide we are going to explain these emerging trends in Operating Systems, challenges they pose and how to cope with these challenges. In particular the lightweight kernels, hypervisors, microkernels, modular kernels and building systems with a single system image.

7.2.6.2 Performance Standard

7.2.6.2.1 Explanation of emerging trends is done.

7.2.6.2.2 Challenges of emerging trends are identified.

7.2.6.2.3 Ways of coping with emerging trends are identified.

7.2.6.3 Information Sheet

Operating systems Emerging trends.

Microsoft Windows has been the dominant operating system for a while. The need to support large number of changing standards in virtually every sector requires that we invest more on the emerging OS trends to remain relevant. Companies developing large systems require operating system that can support the environment. Operating System that can support scalability and at the same time lightweight in terms of storage requirements.

Microkernels

They provide the features required to create processes, run them and communicate between processes in a computer system. They do run in privileged mode. They enable services that would initially be part of monolithic OS to be implemented in server processes e.g. file system.

Microkernels ensure that OS services sends request messages to server processes rather than make direct calls to OS. The main advantages of microkernels include: -

- ✓ Provision of security through provision of a much smaller privileged code base

- ✓ Easy implementation since there are fewer services that must be implemented in microkernel. This gives users advantage of implementing services that are optimized to the needs of specific applications.
- ✓ Its possible to run full Operating system as a server on top of the microkernel eg Unix server that ran on top of the Mach microkernel

The disadvantage of the microkernels is that it incurs 2overheads for the traditional system calls

Hypervisors

Hypervisors were developed to run full featured operating systems as applications in physical computers. Hypervisors construct virtual machines for each operating system you install in a physical machine. It virtualizes hardware its goal being to run multiple Operating systems. Oracle VMware is a good example of an application that enables one to create virtual machines. Hypervisor provides an API for a virtual machine so that the machine is accessed using procedure calls instead of direct manipulation of registers.

Lightweight Kernels (Operating Systems)

Lightweight kernels are based on a partitioned system architecture. The nodes on the system are partitioned into groups to support different functional needs. The partitions include service nodes to support logins, I/O nodes to support access to storage devices and compute nodes for running computations. Lightweight kernels allow for specialization thus different hardware may run different software (Maccabe et al., n.d.). Examples include Catamount, Blue Gene/L

Linux

Linux has recently emerged as the platform of choice for developing parallel scientific applications. It provides a wide range of familiar services such as library writer as a result, developers can reach a large percentage of their potential market when the build those applications in Linux. Linux has the advantage of providing a large collection of device drivers meaning it can run on virtually any platform one might want to use. Even Oracle 19c is implemented on a Linux platform.

Challenges of emerging trends.

With emerging trends like; cloud technology which offers architectural solutions, resource virtualization, performance optimization, privacy and security; Internet of Things (IOT) which supports intelligent and self-configuring embedded devices and sensors interconnected globally.

Cloud users and service providers face a lot of challenges like encrypted data search, auditing, security management and privacy

Coping with emerging trends

One way of coping with challenges of cloud computing is through implementation of protocols that facilitate big data streaming from Internet of Things to the cloud and Quality of Service

7.2.6.4 Learning Activities

Research on the following trends: -

Distributed File Systems

Distributed Virtual Disks

RAID in xFS

7.2.6.5 Self-Assessment

A. Describe Block chain technology and Internet of things with respect to emerging trends in ICT

B. What is the significance of hypervisors in the world of computing?

7.2.6.6 Tools, Equipment, Supplies and Materials

- Internet
- Personal Computer
-

7.2.6.7 model Answers to self-assessment

A. Describe Block chain technology and Internet of things with respect to emerging trends in ICT

The Internet of Things (IoT) is an emerging movement of products with integrated Wi-Fi and network connectivity capabilities. Homes, Cars, appliances, and other products can now connect to the Internet, thus enhancing experience with activities done within the IoT environment. IoT enables a hands-free approach with simple commands issued to objects to execute.

Block chain technology is a distributed ledger or a distributed database where multiple instances of data records are held in several places and can be accessed only through cryptographic codes.

C. What is the significance of hypervisors in the world of computing?

Hypervisors were developed to run full featured operating systems as applications in physical computers. They aid to construct virtual machines for each operating system you install in a physical machine.

7.2.6.8 References

Abraham, P. a. (2013). *Operating Systems Concepts.* United States of America: Wiley.

Abraham, P. G. (2013). *Operating Systems Concepts* . Hoboken: John Wiley & Sons, Inc.

CompTIA. (2013). *CompTIA A+.* (Vol. 2013, Issue December 2). http://certification.comptia.org/getCertified/certifications/a.aspx

Li, A. W. A. and K. (n.d.). *Virtual Memory Primitives for User Programs.pdf.*

Maccabe, A., Bridges, P., Brightwell, R., & Riesen, R. (n.d.). *Recent Trends in Operating Systems and their Applicability to*.

Qurat-ul-Ain, Malik & Iqbal, M. & Khan, Nauman & Nauman, & Hamza, Khan & Haider, Ali. (2010). Modern Trends Used In Operating Systems For High Speed Computing Applications. International Journal on Computer Science and Engineering. 2.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2017). *OPERATING SYSTEM ls s s i t*.

Stallings, W. (2014). Operating Systems: Internals E Designs Principles. In *Journal of Chemical Information and Modeling*.