BIT 1106 OPERATING SYSTEMS

Course overview

Introduction

This course gives an overview of the main facilities provided by an operating system and also introduces the basic functions of two mainstream operating systems.

The whole course unit is composed of twelve lectures which are sub-divided into topics to support the student to demonstrate and show understanding and appreciation of the operating systems.

Objectives

By the end of this unit you should be able to:

1. describe the interface between hardware and software
2. describe the main facilities underlying the operating environment of a computer
3. demonstrates basic skills in a command-based operating system

CONTENTS

The following topics are contained in this module

Lecture One

Operating systems concepts and overview: Introduction, types of operating systems.

Lecture Two

Principles of Operating systems, Operating systems architecture

Lecture Three

Process Management: Process model, Process scheduling,

Lecture Four

Process Management: Process Synchronization and Communication (mutual exclusion, Semaphores, monitors),

Lecture Five

Thread Management

Lecture Six

Concurrency Control

Lecture Seven

Deadlocks.

Lecture Eight

Memory Management: Real Memory Management, Virtual Memory Management.

Lecture Nine

I/O Management

Lecture Ten

File Systems

Lecture Eleven

Protection and Security

Lecture Twelve

Linux System

Windows System

Each lecture contains the following structure

1.1 Introduction

1.2 Specific Objectives

1.3 Lecture Outline

1.4 Lecture

1.5 End of lecture activities (self –tests)

1.6 Summary

1.7 Suggestion for further reading

LECTURE ONE

1.1 Introduction

Welcome to the first lecture on Operating Systems. In this lecture we shall examine the basic organization of computer systems, Operating systems types, functions, history assembler, loader and compiler.

1.2 Specific objectives

At the end of this lecture you should be able:

- describe organization of computer systems
- describe the types of operating systems
- define operating system, functions, history
- define assembler, kernel, loader, compiler

1.3 Lecture Outline

1.4 Introduction

1.5 Operating System

1.5.1Definition of operating system

1.5.2 Functions of Operating System

1.5.3 Operating System as User Interface

1.5.4 Types of Operating systems

1.6 I/O System Management

1.7 Assembler

1.8Compiler

1.9 Loader

1.10 Kernel

1.11 History of Operating System

1.4 Introduction

An operating system act as an intermediary between the user of a computer and computer hardware. The purpose of an operating system is to provide an environment in which a user can execute programs in a convenient and efficient manner.

An operating system is a software that manages the computer hardware. The hardware must provide appropriate mechanisms to ensure the correct operation of the computer system and to prevent user programs from interfering with the proper operation of the system.

1.5 Operating System

1.5.1 Definition of operating system

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

A more common definition is that the operating system is the one program running at all times on the computer (usually called the kernel), with all else being applications programs.

An Operating system is concerned with the allocation of resources and services, such as memory, processors, devices and information. The Operating System correspondingly includes programs to manage these resources, such as a traffic controller, a scheduler, memory management module, I/O programs, and a file system.

In brief, an operating system is the set of programs that controls a computer. Some examples of operating systems are UNIX, Mach, MS-DOS, MS Windows, Windows/NT, OS/2 and MacOS.

The operating system is an important part of almost every computer system.A computer system can be divided roughly into four components: the hardware, the operating system and the application programs and the users. (see diagram below)

1.5.2 Functions of Operating System

1. Booting the computer

The process of starting or restarting the computer is known as booting. A cold boot is when you turn on a computer that has been turned off completely. A warm boot is the process of using the operating system to restart the computer.

2. Performs basic computer tasks

The operating system performs basic computer tasks, such as managing the various peripheral devices such as the mouse, keyboard and printers. For example, most operating systems now are plug and play which means a device such as a printer will automatically be detected and configured without any user intervention.

3. Provides a user interface

A user interacts with software through the user interface. The two main types of user interfaces are: *command line and a graphical user interface (GUI).* With a command line interface, the user interacts with the operating system by typing commands to perform specific tasks. An example of a command line interface is DOS (disk operating system). With a graphical user interface, the user interacts with the operating system by using a mouse to access windows, icons, and menus. An example of a graphical user interface is Windows 8 or Windows 10.

The operating system is responsible for providing a consistent application program interface (API) which is important as it allows a software developer to write an application on one computer and know that it will run on another computer of the same type even if the amount of memory or amount of storage is different on the two machines.

4. Handles system resources

The operating system handles system resources such as the computer's memory and sharing of the central processing unit (CPU) time by various applications or peripheral devices. Programs and input methods are constantly competing for the attention of the CPU and demand memory, storage and input/output bandwidth. The operating system ensures that each application gets the necessary resources it needs in order to maximise the functionality of the overall system.

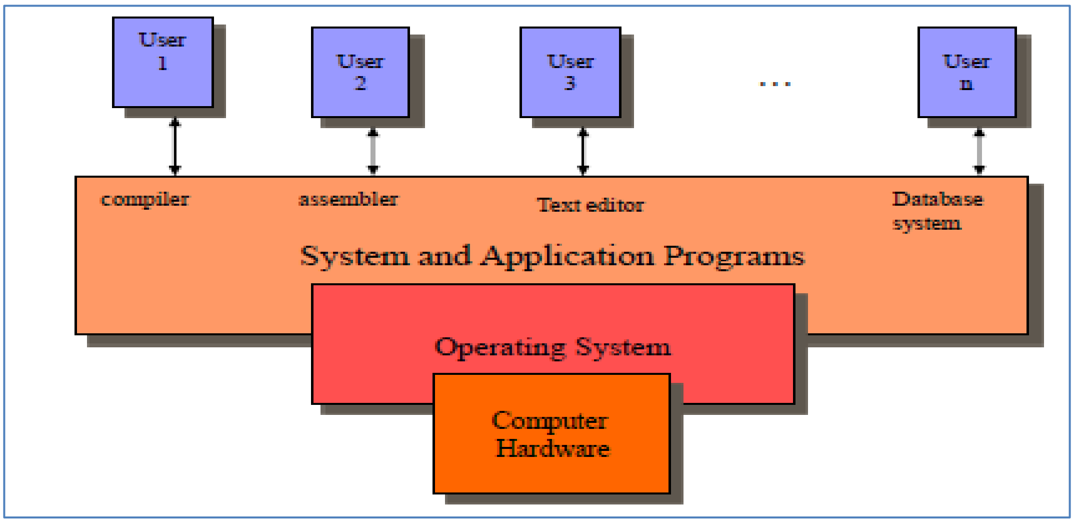
5. **Provides file management**

The operating system also handles the organisation and tracking of files and directories (folders) saved or retrieved from a computer disk. The file management system allows the user to perform such tasks as creating files and directories, renaming files, coping and moving files, and deleting files. The operating system keeps track of where files are located on the hard drive through the type of file system. The type two main types of file system are File Allocation table (FAT) or New Technology File system (NTFS)

1.5.3 Operating System as User Interface

Every general purpose computer consists of the hardware, operating system, system programs and application programs. The hardware consists of memory, CPU, ALU, I/O devices, peripheral device and storage device. System program consists of compilers, loaders, editors, OS etc. The application program consists of business program, database program.

The diagram below shows the conceptual view of a computer system.



Every computer must have an operating system to run other programs. The operating system and coordinates the use of the hardware among the various system programs and application program for a various users. It simply provides an environment within which other programs can do useful work.

The operating system is a set of special programs that run on a computer system that allow it to work properly. It performs basic tasks such as recognizing input from the keyboard, keeping track of files and directories on the disk, sending output to the display screen and controlling a peripheral devices.

OS is designed to serve two basic purposes:

1. It controls the allocation and use of the computing system‘s resources among the various user and tasks.

2. It provides an interface between the computer hardware and the programmer that simplifies and makes feasible for coding, creation, debugging of application programs.

1.5.4 Types of Operating systems (Classification)

Operating Systems are classified into different types depending on their capability of processing

1. Single User and Single Task operating systems is for use by a single user for a standalone single computer for performing a single task .operating system for personal computers are single-user operating systems. For example, if the user is editing a document, then a document cannot be printed on the printer simultaneously. Single user operating systems are simple operating system designed to manage one task at a time. Eg Ms Dos
2. Single User and Multitasking operating system allows execution of more than one task or process concurrently. For this, the processor time is divided amongst different tasks. This division of time is also called time sharing. The processor switches rapidly between processes. For example, the user can listen to music on the computer while writing an article using word processor software. The user can switch between the applications and also transfer data between them. Windows versions are examples of multitasking operating systems.
3. Multiuser operating systems is used in computer networks that allow same data and applications to be accessed by multiple users at the same time. The users can also communicate with each other. Linux, Unix, and windows 10 are examples of multiuser operating systems.
4. Multiprocessing operating systems have two or more processors for a single running process. Processing takes place in parallel and is also called parallel processing. Each processor works on different parts of the same task, or, on two or more different tasks. Since execution takes place in parallel, they are used for high speed execution, and to increase the power of computer. Linux, UNIX and Windows 7 are examples of multiprocessing operating systems.
5. Real time operating systems are designed to respond to an event within a predetermined time. These operating systems are used to control processes. Processing is done within a time constraint. Operating systems monitors the events that affect the execution of process and respond accordingly. They are used to respond to queries in areas like medical imaging system, industrial control systems etc. Lynx operating systems is an example of real time operating systems.

6. Embedded operating systems is embedded in a device in the rom. they are specific to a device and are less resource intensive. They are used in appliances like microwaves, washing machines, traffic control systems etc.

1.6 I/O System Management

One of the purposes of an operating system is to hide the peculiarities of specific hardware devices from the user.

The module that keeps track of the status of devices is called the I/O (input/output) traffic controller. Each I/O device has a device handler that resides in a separate process associated with that device. The I/O subsystem consists of

1. A memory management component that includes buffering, caching and spooling.
2. A general device driver interface.

Drivers are provided for specific hardware devices.

1.7 Assembler

Input to an assembler is an assembly language program. Output is an object program plus information that enables the loader to prepare the object program for execution. At one time, the computer programmer had at his disposal a basic machine that interpreted, through hardware, certain fundamental instructions. He would program this computer by writing a series of ones and zeros (machine language), place them into the memory of the machine.

1.8 Compiler

The high level languages – examples are FORTRAN, COBOL, ALGOL and PL/I – are processed by compilers and interpreters. A compiler is a program that accepts a source program in a high-level language and produces a corresponding object program. An interpreter is a program that appears to execute a source program as if it was machine language. The same name (FORTRAN, COBOL etc) is often used to designate both a compiler and its associated language.

1.9 Loader

A loader is a routine that loads an object program and prepares it for execution. There are various loading schemes: absolute, relocating and direct-linking. In general, the loader must load, relocate, and link the object program. Loader is a program that places programs into memory and prepares them for execution. In a simple loading scheme, the assembler outputs the machine language translation of a program on a secondary device and a loader is placed in core. The loader places into memory the machine language version of the user‘s program and transfers control to it. Since the loader program is much smaller than the assembler, this makes more core services available to user‘s program.

1.10 Kernel

The heart of the operating system is called the *kernel*. This is the collection of functions that perform the basic services such as scheduling applications. CPUs typically have (at least) two execution modes: *user* mode and *kernel* mode. User applications run in user mode. The kernel runs in kernel mode. Kernel mode is also called *supervisor* mode or *privileged* mode. A kernel can be contrasted with a shell, the outermost part of an operating system that interacts with user commands. Kernel and shell are terms used more frequently in Unix operating systems than in IBM mainframe or Microsoft Windows systems.

It typically makes these facilities available to application processes through inter-process communication mechanisms and system calls. Operating system tasks are done differently by different kernels, depending on their design and implementation. While monolithic kernels execute all the operating system code in the same address space to increase the performance of the system, microkernels run most of the operating system services in user space as servers, aiming to improve maintainability and modularity of the operating system. A range of possibilities exists between these two extremes

1.11 Activities

1. Point out the differences between the kernel of early operating systems and the kernel of current operating systems.
2. Explain how the kernel of current operating systems makes it easier to add new components or modify existing ones, which is useful for distributed systems
3. Discuss the history of Operating systems

1.12 Self-Assessment

1. Define Operating System ,Assembler, Loader, Compiler
2. Explain various function of operating system
3. Explain I/O system Management
4. Outline the history of operating systems

1.12 Summary

- Every general purpose computer consists of the hardware, operating system, system programs and application programs
- A loader is a routine that loads an object program and prepares it for execution. There are various loading schemes: absolute, relocating and direct-linking. In general, the loader must load, relocate, and link the object program
- A compilers is a program that accepts a source program ‖ in a high-level language‖ and produces a corresponding object program.
- The kernel is the central component of most computer operating systems; it is a bridge between applications and the actual data processing done at the hardware level
- An Operating system is concerned with the allocation of resources and services, such as memory, processors, devices and information. The Operating System correspondingly includes programs to manage these resources, such as a traffic controller, a scheduler, memory management module, I/O programs, and a file system.

1.13 Suggestion for further reading

1. Abraham Silberschatz, Greg Gagne, Peter Baer Galvin(2002):Operating System Concepts, Wiley
2. Andrew S. Tanenbaum, Modern Operating Systems 3 e, (c) 2008 Prentice-Hall

1.14 Extracted from

1. Prof. Harsh Bhor,Prof. UdayRote,Prof. UmeshShinde Operating Systems (K.J.S.I.E.I.T SION MUMBAI)
2. Anita GoelComputer Fundamentals,Pearson Education India
3. Basic functions of an operating system. http://www.hsc.csu.edu.au/info_tech/compulsory/os/4014/basic_functions.htm