MODULE 3 COMPUTER SOFTWARE

Unit 1 Computer Software (1)
Unit 2 Computer Software (2)

UNIT 1 COMPUTER SOFTWARE (1)

CONTENTS

5.0 INTRODUCTION

Computer hardware is driven by the software. The usefulness of the computer depends on the programs that are written to manipulate it. Computer software comes in different forms: the operating system, utility software, language translators and application

software. This unit therefore presents detailed discussions of each category of computer software.

6.0 OBJECTIVES

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

 identify the different types of computer software discuss the importance of each type of software.

3.0 MAIN CONTENT

3.1 Computer Software

The physical components of the computer are called the hardware while all the other resources or parts of the computer that are not hardware, are referred to as the software. The software is the set of programs that make the computer system active. In essence, the software is the set of programs that run on the computer.

Then, what is a program? A program is a series of coded instructions showing the logical steps the computer follows to solve a given problem.

3.2 Classification of Computer Software

The computer software could be divided into two major groups, namely system software (programs), and application software (programs).

3.2.1 System Software

This refers to the suits of programs that facilitate the optimal use of the hardware systems and/or provide a suitable environment for the writing, editing, debugging, testing and running of user programs. Usually, every computer system comes with a collection of these suits of programs which are provided by the hardware manufacturer.

3.2.2 Operating Systems

An operating system is a program that acts as an interface between a user of a computer and the computer hardware. The purpose of an operating system is to provide an environment in which a user may execute programs.

The operating system is the first component of the systems programs that interest us here. Systems programs are programs written for direct execution on computer hardware in order to make the power of the computer fully and efficiently accessible to applications programmers and other computer users. Systems programming is different from

application programming because the former requires an intimate knowledge of the computer hardware as well as the end users' needs. Moreover, systems programs are often large and more complex than application programs, although that is not always the case. Since systems programs provide the foundation upon which application programs are built, it is most important that systems programs are reliable, efficient and correct.

In a computer system the hardware provides the basic computing resources. The applications programs define the way in which these resources are used to solve the computing problems of the user. The operating system controls and coordinates the use of the hardware among the various systems programs and application programs for the various users.

The basic resources of a computer system are provided by its hardware, software and data. The operating system provides the means for the proper use of these resources in the operation of the computer system. It simply provides an environment within which other programs can do useful work.

We can view an operating system as a resource allocator. A computer system has many resources (hardware and software) that may be required to solve a problem: CPU time, memory space, file storage space, input/output devices, etc.

The operating system acts as the manager of these resources and allocates them to specific programs and users as necessary for their tasks. Since there may be many, possibly conflicting, requests for resources, the operating system must decide which requests are allocated resources to operate the computer system fairly and efficiently. An operating system is a control program. This program controls the execution of user programs to prevent errors and improper use of the computer.

Operating systems exist because they are a reasonable way to solve the problem of creating a usable computing system. The fundamental goal of a computer system is to execute user programs and solve user problems.

The primary goal of an operating system is convenience for the user. Operating systems exist because they are supposed to make it easier to compute with an operating system than without an operating system. This is particularly clear when you look at the operating system for small personal computers.
A secondary goal is the efficient operation of computer system. This goal is particularly important for large, shared multi-user systems. Operating systems can attain this goal. It is known that sometimes these two goals, convenience and efficiency, are contradictory.
While there is no universally agreed definition of the concept of an operating system, we offer the following as a reasonable starting point:

A computer's operating system (OS) is a group of programs designed to serve two basic purposes:

- To control the allocation and use of the computing system's resources among the various users and tasks.
- To provide an interface between the computer hardware and the programmer that simplifies and makes feasible the creation, coding, debugging, and maintenance of application programs.

Specifically, we can imagine that an effective operating system should accomplish all of the following:

- Facilitate creation and modification of program and data files through an editor program.
- Provide access to compilers to translate programs from high-level languages to machine language.
- Provide a loader program to move the complied program code to the computer's memory for execution.
- Provide routines that handle the intricate details of I/O programming.
- Assure that when there are several active processes in the computer, each will get fair and non-interfering access to the central processing unit for execution.
- Take care of storage and device allocation.
- Provide for long term storage of user information in the form of files.
- Permit system resources to be shared among users when appropriate, and be protected from unauthorised or mischievous intervention as necessary.

Though systems programs such as editor and translators and the various utility programs (such as sort and file transfer program) are not usually considered part of the operating system, the operating system is responsible for providing access to these system resources.

3.3 Types of Operating Systems

Modern computer operating systems may be classified into three groups, which are distinguished by the nature of interaction that takes place between the computer user and his or her program during its processing. The three groups are called batch, time-shared and real time operating systems.

3.3.1 Batch Processing Operating System

In a batch processing operating system environment users submit jobs to a central place where these jobs are collected into a batch, and subsequently placed on an input queue at

the computer where they will be run. In this case, the user has no interaction with the job during its processing, and the computer's response time is the turnaround time - the time from submission of the job until execution is complete, and the results are ready for return to the person who submitted the job.

3.3.2 Time Sharing Operating System

Another mode for delivering computing services is provided by time sharing operating systems. In this environment a computer provides computing services to several or many users concurrently on-line. Here, the various users are sharing the central processor, the memory, and other resources of the computer system in a manner facilitated, controlled, and monitored by the operating system. The user, in this environment, has nearly full interaction with the program during its execution, and the computer's response time may be expected to be no more than a few seconds.

3.3.3 Real Time Operating System

The third class of operating systems, the real time operating systems, are designed to service those applications where response time is of the essence in order to prevent error, misrepresentation or even disaster. Examples of real time operating systems are those which handle airlines reservations, machine tool control, and monitoring of a nuclear power station. The systems, in this case, are designed to be interrupted by external signal that require the immediate attention of the computer system.

In fact, many computer operating systems are hybrids, providing for more than one of these types of computing service simultaneously. It is especially common to have a background batch system running in conjunction with one of the other two on the same computer.

A number of other definitions are important towards gaining an understanding of operating systems:

3.3.4 Multiprogramming Operating System

A multiprogramming operating system is a system that allows more than one active user program (or part of user program) to be stored in main memory simultaneously.
Thus, it is evident that a time-sharing system is a multiprogramming system, but note that a multiprogramming system is not necessarily a time-sharing system. A batch or real time operating system could, and indeed usually does, have more than one active user program simultaneously in main storage. Another important, and all too similar, term is 'multiprocessing'.

A multiprocessing system is a computer hardware configuration that includes more than one independent processing unit. The term multiprocessing is generally used to refer to large computer hardware complexes found in major scientific or commercial applications.

A networked computing system is a collection of physically interconnected computers. The operating system of each of the interconnected computers must contain, in addition to its own standalone functionality, provisions for handing communication and transfer of program and data among the other computers with which it is connected.

A distributed computing system consists of a number of computers that are connected and managed so that they automatically share the job processing load among the constituent computers, or separate the job load as appropriate to particularly configured processors. Such a system requires an operating system which, in addition to the typical standalone functionality, provides coordination of the operations and information flow among the component computers.

The networked and distributed computing environments and their respective operating systems are designed with more complex functional capabilities. In a network operating system the users are aware of the existence of multiple computers, and can log in to remote machines and copy files from one machine to another. Each machine runs its own local operating system and has its own user (or users).

3.3.5 Distributed Operating System

A distributed operating system, in contrast, is one that appears to its users as a traditional uniprocessor system, even though it is actually composed of multiple processors. In a true distributed system, users should not be aware of where their programs are being run or where their files are located; that should all be handled automatically and efficiently by the operating system.

3.3.6 Network Operating Systems

Network operating systems are not fundamentally different from single processor operating systems. They obviously need a network interface controller and some low-level software to drive it, as well as programs to achieve remote login and remote files access, but these additions do not change the essential structure of the operating systems.

True distributed operating systems require more than just adding a little code to a uniprocessor operating system, because distributed and centralised systems differ in

critical ways. Distributed systems, for example, often allow programs to run on several processors at the same time, thus requiring more complex processor scheduling algorithms in order to optimize the amount of parallelism achieved.

3.4 Operating System Components

An operating system provides the environment within which programs are executed. To construct such an environment, the system is partitioned into small modules with a well-defined interface. The design of a new operating system is a major task. It is very important that the goals of the system be will defined before the design begins. The type of system desired is the foundation for choices between various algorithms and strategies that well be necessary.

A system as large and complex as an operating system can only be created by partitioning it into smaller pieces. Each of these pieces should be a well defined portion of the system with carefully defined inputs, outputs, and functions. Obviously, not all systems have the same structure. However, many modern operating systems share the system components outlined below.

3.4.1 Process Management

A process is the unit of work in a system. Such a system consists of a collection of processes, some of which are operating system processes, those that execute system code, and the rest being user processes, those that execute user code. All of those processes can potentially execute concurrently.

The operating system is responsible for the following activities in connection with processes managed:

- The creation and deletion of both user and system processes The suspension and resumption of processes. The provision of mechanisms for process synchronisation
- The provision of mechanisms for deadlock handling.

3.4.2 Memory Management

Memory is central to the operation of a modern computer system. Memory is a large array of words or bytes, each with its own address. Interaction is achieved through a

sequence of reads or writes of specific memory address that the CPU fetches from and stores in memory.

In order for a program to be executed, it must be mapped to absolute addresses and loaded in to memory. As the program executes, it accesses program instructions and data from memory by generating these absolute is declared available, and the next program may be loaded and executed.

The operating system is responsible for the following activities in connection with memory management:

- Keeping track of which parts of memory are currently being used and by whom.
- Deciding which processes are to be loaded into memory when memory space becomes available.
- Allocating and de-allocating memory space as needed.

3.4.3 Secondary Storage Management

The main purpose of a computer system is to execute programs. These programs, together with the data they access, must be in main memory during execution. Since the main memory is too small to permanently accommodate all data and programs, the computer system must provide secondary storage to back-up main memory. Most modem computer systems use disks as the primary on-line storage of information, of both programs and data. Most programs, like compilers, assemblers, sort routines, editors, formatters, and so on, are stored on the disk until loaded into memory, and then use the disk as both the source and destination of their processing. Hence the proper management of disk storage is of central importance to a computer system.

The operating system is responsible for the following activities in connection with disk management:

- Free space management Storage allocation
- Disk scheduling.

3.4.4 I/O System

One of the purposes of an operating system is to hide the peculiarities of specific hardware devices from the user. For example, in Unix, the peculiarities of I/O devices are hidden from the bulk of the operating system itself by the I/O system. The I/O system consists of:

- A buffer caching system

- A general device driver code
- Drivers for specific hardware devices

Only the device driver knows the peculiarities of a specific device.

3.4.5 File Management

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

For the convenient use of the computer system, the operating system provides a uniform logical view of information storage. The operating system abstracts from the physical properties of its storage devices to define a logical storage unit, the file. Files are mapped, by the operating system, onto physical devices.

A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data. Data files may be numeric, alphabetic or alphanumeric. Files may be free-form, such as text files, or may be rigidly formatted. In general a file is a sequence of bits, bytes, lines or records whose meaning is defined by its creator and user. It is a very general concept.
The operating system is responsible for the following activities in connection with file management:

- The creation and deletion of files
- The creation and deletion of directory
- The support of primitives for manipulating files and directories The mapping of files onto disk storage. Backup of files on stable (non volatile) storage.

3.4.6 Protection System

The various processes in an operating system must be protected from each other's activities. For that purpose, various mechanisms are used to ensure that the files, memory segment, CPU and other resources can be operated on only by those processes that have gained proper authorisation from the operating system.

For example, memory addressing hardware ensures that a process can only execute within its own address space. The timer ensures that no process can gain control of the CPU without relinquishing it. Finally, no process is allowed to do its own I/O, to protect the integrity of the various peripheral devices.

Protection refers to a mechanism for controlling the access of programs, processes, or users to the resources defined by a computer controls to be imposed, together with some means of enforcement.

Protection can improve reliability by detecting latent errors at the interfaces between component subsystems. Early detection of interface errors can often prevent contamination of a healthy subsystem by a subsystem that is malfunctioning. An unprotected resource cannot defend against use (or misuse) by an unauthorised or incompetent user.

3.4.7 Networking

A distributed system is a collection of processors that do not share memory or a clock. Instead, each processor has its own local memory, and the processors communicate with each other through various communication lines, such as high speed buses or telephone lines. Distributed systems vary in size and function. They may involve microprocessors, workstations, minicomputers, and large general purpose computer systems.

The processors in the system are connected through a communication network, which can be configured in a number of different ways. The network may be fully or partially connected. The communication network design must consider routing and connection strategies, and the problems of connection and security.

A distributed system provides the user with access to the various resources the system maintains. Access to a shared resource allows computation speed-up, data availability, and reliability.

3.4.8 Command Interpreter System

One of the most important components of an operating system is its command interpreter. The command interpreter is the primary interface between the user and the rest of the system.

Many commands are given to the operating system by control statements. When a new job is started in a batch system or when a user logs in to a time-shared system, a program which reads and interprets control statements is automatically executed. This program is variously called (1) the control card interpreter, (2) the command line interpreter, (3) the shell (in UNIX), and so on. Its function is quite simple: get the next command statement, and execute it.

The command statement themselves deal with process management, I/O handling, secondary storage management, main memory management, file system access, protection, and networking.