UNIT 3 GRAPHICAL USER INTERFACE

Structure

3.0 INTRODUCTION

The **Graphical User Interface (GUI)** is one of the most revolutionary changes to occur in the evolution of modern computing system. In less than 10 years, the expectation of what the interaction between human and computer would be like was changed from a tense, character-oriented system to the more graphics-oriented system. This revolution has increased the accessibility and usability of computer systems to the general public.

The previous two units of this block dealt with the concepts of programming languages and operating system. In this unit we will look at another type of software which help users to interact with the system easily and to perform a complex task with little knowledge of operating system or memorized commands. The software providing such features supports **Modern User Interface** concept such as desktop metaphor which makes computers available to the majority of people who are either novice or non-programmers. The personal computer was invented for these users.

In this unit, we will discuss about several aspects of GUI, starting from common GUI terms, major components of GUI, its history and finally a popular package supporting GUI : MS-Windows, user interface.

3.1 OBJECTIVES

After going through this unit, you will be able to

- define what is GUI and how it is different from character oriented system
- define all the terms related with GUI, and
- identify important features of MS-WINDOWS.

3.2 WHAT IS GRAPHICAL USER INTERFACE?

The term "user interface" originated in the engineering environment in the late 1970s. Virtually every one who interacted directly with computers had been engineers and programmers, but a new kind of users were emerging : the non-programming user. These users often reacted more negatively to difficulties in dealing with a machine. New forms of interaction was needed new interfaces, were required attention flowed to "the user interface". With the introduction of the Macintosh in 1984, Apple Computer popularised the user interface as it is known today. Apple's user interface is now commonly referred to as a Graphical User Interface or GUI. The GUI has become associated with a common feature set available in a number of product offerings. Common features include:

secondary user-input devices. Usually a pointing device and typically a mouse. point and shoot functionality with screen menus that appear or disappear under pointing device-control.
windows that graphically display what the computer is doing. icons that represent files, directories and other application and system entitles. dialog boxes, button, sliders, check boxes and many other graphical metaphors that let the programmer and user tell the computer what to do and how to do it.

Today's GUIs have expanded basic functionalities to support not only graphics but also dimensions, colour, height, video and highly dynamic interaction. Modern user interfaces can simulate a very realistic view of a real, three dimensional world.

3.3 EVOLUTION OF THE HUMAN AN MACHINE INTERACTION

The primary means of communication with computers until recently has been through command-based interfaces. In command interfaces, users have to learn a large set of commands to get their job(s) done. In early computer systems paper tapes, cards and batch jobs were the primary means of communicating these commands to the computers. Later, time-sharing systems allowed the use of CRT terminals to interact/communicate with the computer. These early systems were heavily burdened by users trying to share precious computer resources such as CPU and peripherals.

The batch systems and time sharing led to command-driven user interfaces. Users had to memorise commands and options or consult a large set to user manuals. The early mainframe and minicomputer systems required a large set of instruction manuals on how to use the system. In some Systems, meaningful terms were used for command names to help the end user. But in other systems the end-user had to memorise several sequences of keystrokes to accomplish certain tasks.

Early users of computers were engineers and what we now call expert users; users that had a lot of interest in knowing more about the computer systems and the technology. Command line interfaces were acceptable by the majority of these users. In the 1970s, computers were introduced to a new class of users, secretaries, managers and non-technical people. These new users were less interested in learning computer technology and more interested in getting their jobs done through the machine. The command based

interfaces caused many of these new users to develop computer phobia. Imagine the thought of memorising commands made up of "Control-Alt-Del" to boot the system.

To make life easier for the end-user, a large collection of devices have been invented to control, monitor and display information. The early (and still widely used) peripherals are the keyboard and the video terminal. But, it was not until the late 70s, research projects at some universities led to the invention of pointing devices and windowing systems. The mouse and joystick were among some of the few pointing devices that were invented in this period. Also, research pioneers invented the notion of splitting the screen to allow multiple windows and direct manipulation of objects.

In the 70s, researchers designed powerful new workstations armed with graphical user-interfaces. The basic assumption of these new workstations was that one user could have a powerful desktop computer totally dedicated to that user's task. Thus, the computer is not only used to perform the task, but can also provide a much more intuitive and easy-to-use environment.

Instead of memorising commands to each stage, the user selects a command from a menu bar displaying a list of available commands. For example, figure 2 displays the menu bar. This menu bar displays a list of commands available such as File, Edit and Search. When the mouse is clicked on any One of these men commands the appropriate action is taken.

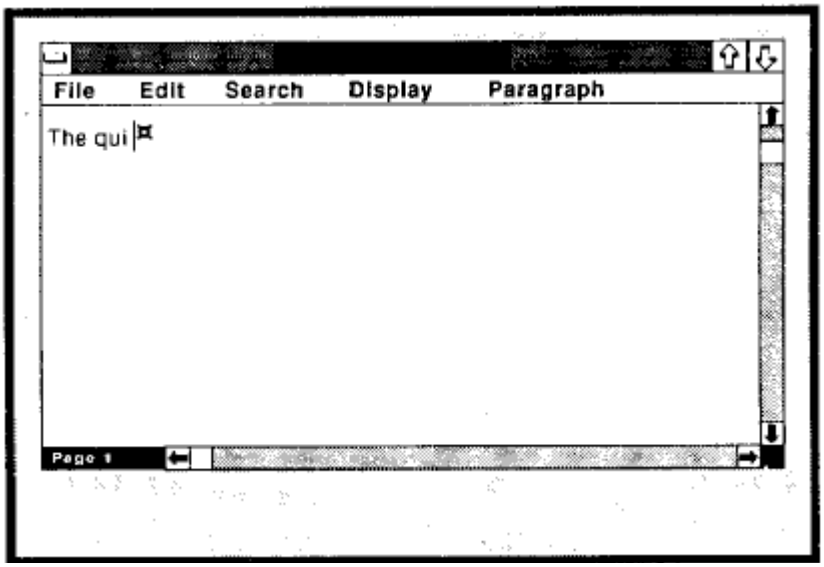

Fig. 1 : Menu Bar

Pull-down and pop-up menus are option commands available for each selection. Figure 2 shows the pull-down menu displayed when the Character menu item is selected. The user can then select from different character styles.

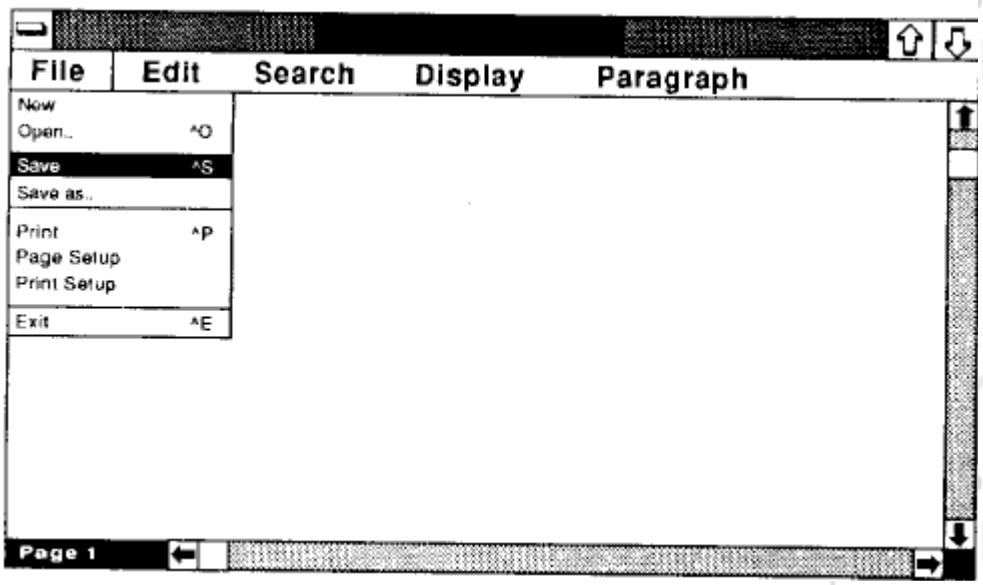

Fig. 2 : Pulldown Menu

Dialog boxes allow more complex interaction between the user and the computer. Dialog boxes employ a large collection of control objects such as buttons, scroll bars and editable boxes. For example, in figure 3, a dialog box is used to open file. This dialog box is composed of two buttons called Open and Close and edit box that allows a file name to be entered and a scroll region that allows navigation through the list of files and directories available on the disk. Clicking on the Open button causes the file to be viewed.

In graphical user-interfaces, textual data is not the only form of interaction. **Icons** represent concepts such as file **folders, waste baskets**, and **printers. Icons** symbolise words and concepts commonly applied in different situations. Figure 4 shows the **paint utility** with its palette composed of icons. Each one of these icons represents a certain type of painting behaviour. Once the pencil icon is clicked, for example, the cursor can behave as a pencil to draw lines. Application of icons to user-interface design are still being explored in new computer systems and softwares such as the NeXT computer user interface.

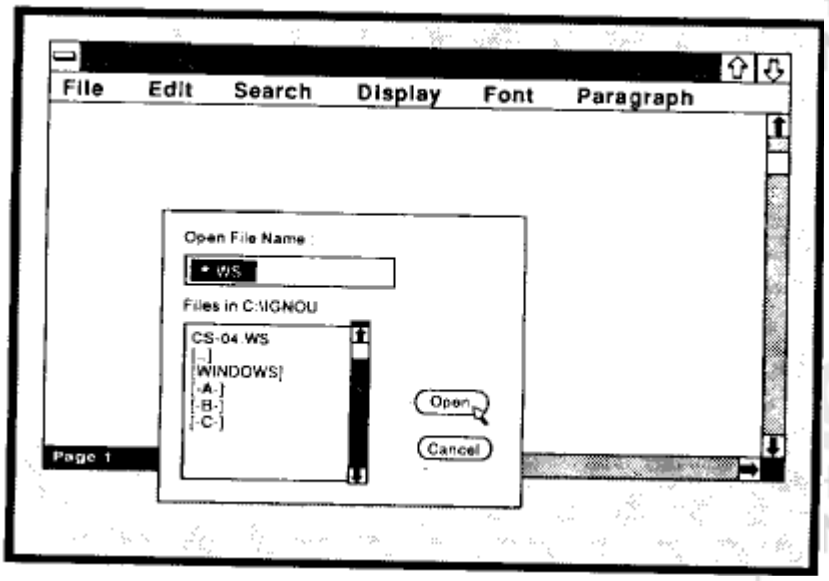

Fig. 3 : Dialog Box

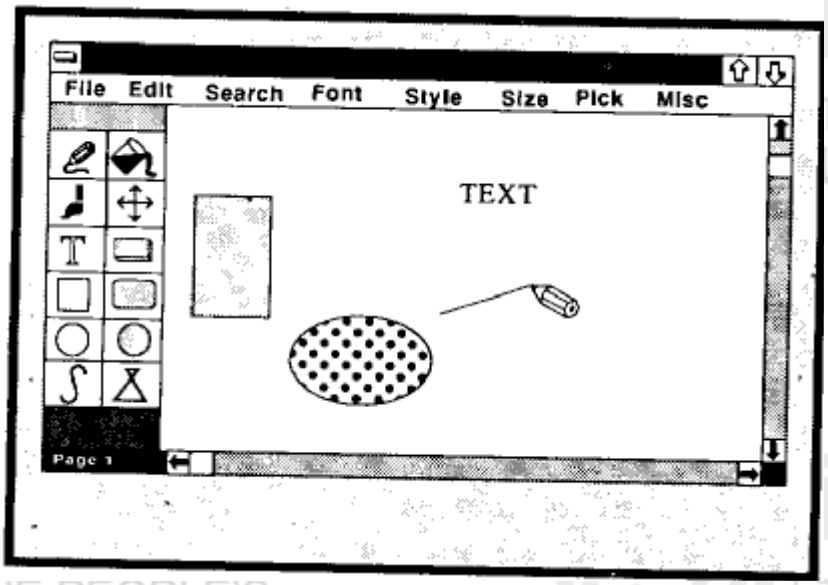

Fig. 4 : Paint Palette

The idea of metaphors has brought the computer closer to the natural environment of the end-user. The concept of physical metaphor paradigm developed by Alan Kay, initiated most of the research for graphic user interfaces based on new programming approach called object-oriented programming. Discussion on this subject is beyond this unit. This will be covered in detail in C++ and object oriented programming

course, offered in the 3rd year of the MCA programme. The physical metaphor is a way of saying that the visual displays of a computer system should present the images of real physical objects.

For example, the wastepaper basket icon can be use to discard objects from the system by simply dragging the unwanted objects into the wastepapers basket, as in real life. The desktop metaphor probably has been the most famous paradigm. Because of the large set of potential office users, this metaphor can have the most dramatic effect. In this paradigm, the computer presents information and objects as they would appear and behave in an office, using icons for folders, in-baskets, out-baskets and calendars.

3.4 COMMON GRAPHICAL USER INTERFACE TERMS

This section presents a list of terms used commonly with the graphical user interface (GUI).

1. Pointing devices

Pointing devices allow users to point at different parts of the screen. Pointing devices can be used to invoke a command from a list of commands, presented in a menu. They can also be used to manipulate objects on the screen by

- selecting objects on the screen
- moving objects around the screen, or
- merging several objects into another object

Since 1960s, a diverse set of tools have been used as pointing devices include the light pen, joystick, touch sensitive screen and the popularity of the mouse is due to optimal coordination of hand and easier tracking of the cursor on the screen.

2. Bit-mapped displays

As memory chips get denser and cheaper, bit displays are replacing character-based display screens. Bit-mapped display made up of tiny dots (pixels) that are independently addressable and much finer resolution than character displays. Bit-mapped displays have advantages over character displays. One of the major advantages include graphic manipulation capabilities for vector and raster graphics manipulation capabilities for vector and raster graphics, which present information in the final form on paper (also called WYSIWYG: What You See Is What You Get).

3. Windows

When a screen is split into several independent regions, each one is called a window. Several applications can display results simultaneously in different windows. Figure 5 presents a screen with two windows.

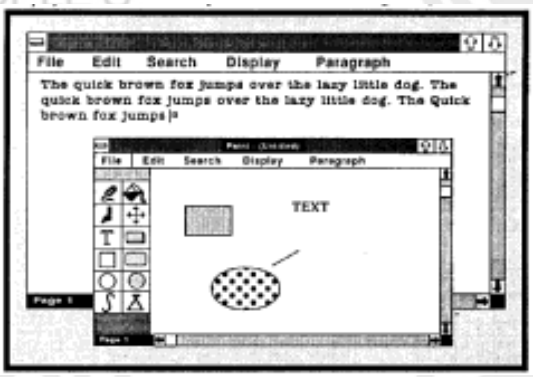

Figure 5 : Screen with two windows

The end-user can switch from one application to another or share data between applications. Windowing systems have capabilities to display windows either tiled or over-lapped, Figures 6 and 7. Users can organise the screen by resizing the window or moving related windows closer.

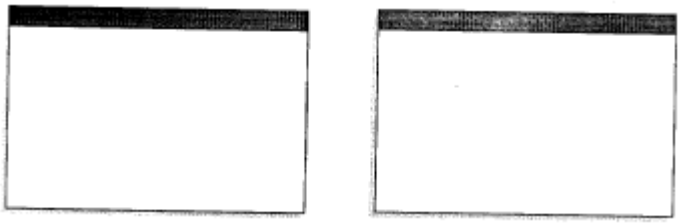

Figure 6 : Tiled Windows

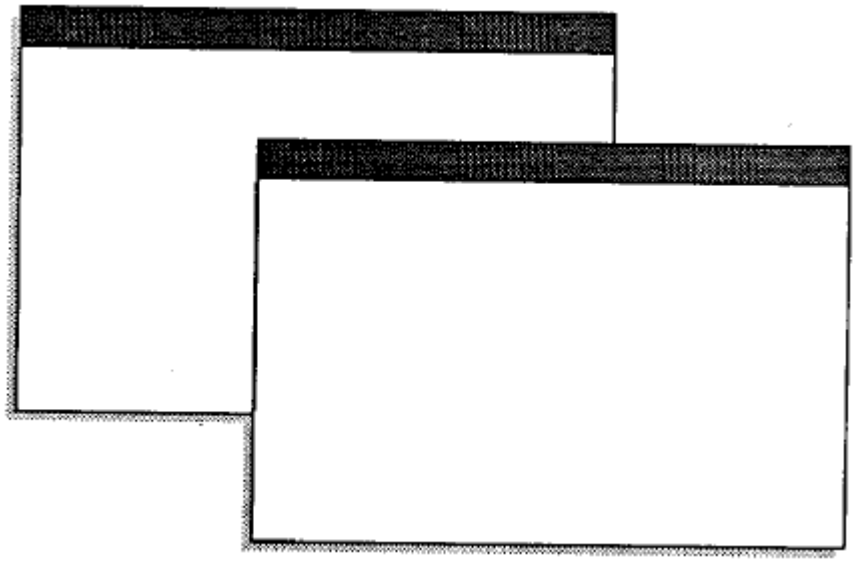

Figure 7 : Overlapped Windows

4. Menus

A menu displays a list of commands available within an application (figure 1). From the menu, the end-user can select operations such as File, Edit or Search. Instead of remembering commands at each stage, a menu can be use to provide a list of items. Each menu item can be either a word or an icon representing a command or a function. Menu item can be invoked by moving the cursor on the menu item and selecting the item by clicking the mouse.

When a menu item is invoked it could cause other menus, called pull-down menus, to appear. Pull-down menus (figure 2) are used to present a group of related commands or options for a menu item. Figure 2 presents the File pull-down menu.

5. Dialog boxes

Dialog boxes are used to collect information from the user or to present information to the user. For example, when printing a file, (figure 8) a dialog box is displayed to get additional information.

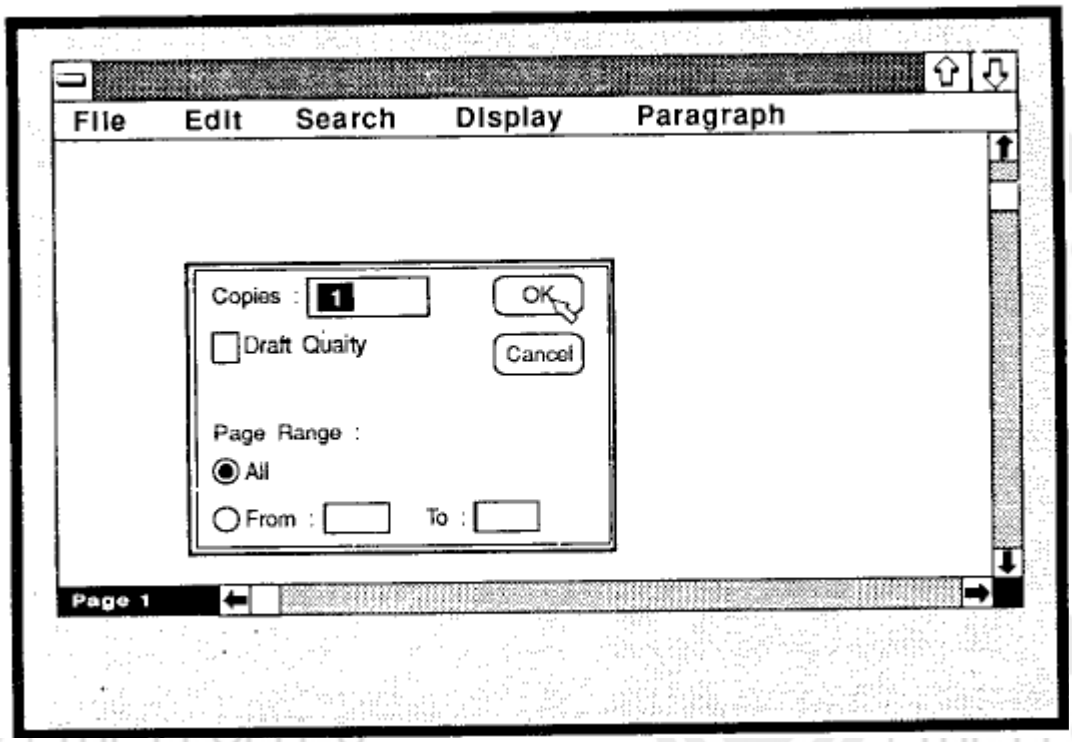

Figure 8 : Dialog Box

Some of the in information obtained are the number of copies and page numbers to be printed. Dialog boxes are also used to indicate error messages in the form of alert boxes. Dialog boxes use a wide range of screen control elements of communicate with the user.

6. Icons

Icons are used to provide a symbolic representation of any system/user-defined object such as file, folder, address, book, applications and so on. Different types of objects are represented by a specific type of icon. In some GUIs, documents representing folders are represented by a folder icon (figure 9). A folder icon contains a group of files or other folder icons. Double clicking on the folder icon causes a window to be opened displaying a list of icons and folder icons representing the folder's contents.

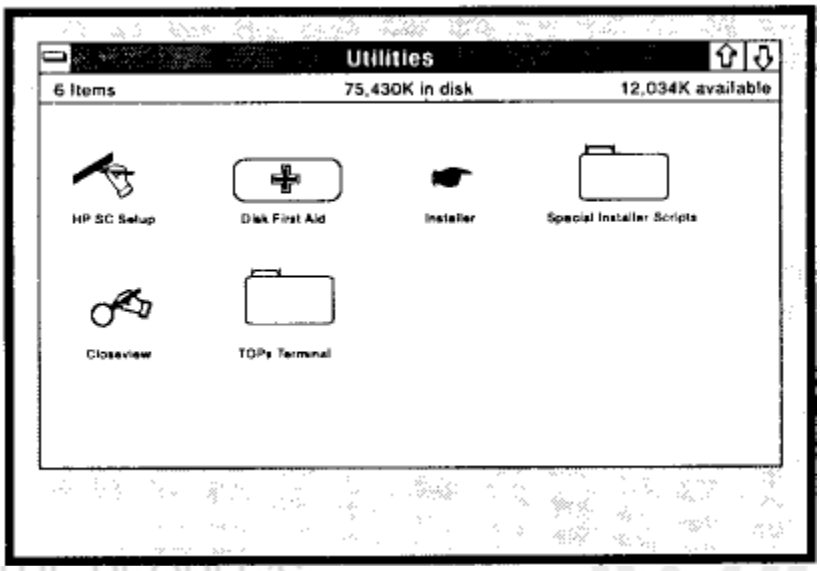

Figure 9 : Icons

7. Desktop metaphor

In the desktop metaphor, users are not aware of applications. Users deal with files, folder, drawers, a clipboard and an out-box. Instead of starting the word processor and loading file, users merely open the report document, which implicitly invokes the word processor. Clicking the mouse on an icon representing the report cause word processor to get started and to load the report file implicitly. Today, several computing environments provide this capability.

8. Graphic User Interfaces

GUls are systems that allow creation and manipulation of user interfaces employing windows, menus, icons, dialog boxes, mouse and keyboard. Macintosh toolbox, Microsoft windows and X-windows are some examples of GUls.

3.5 FUNCTIONALITY OF GRAPHICAL USER INTERFACES

The development environment for most GUIs consists of four major components

- A windowing system
- An imaging model
- An application program interface (API), and
- A set of tools and frameworks for creating interfaces and developing integrated applications.

Windowing systems allow programs to display multiple applications at the same time. Windowing systems include programming tools for building movable and resizable windows, menus, dialog boxes, and other items on the display. Some GUIs contain proprietary windowing systems, such as Macintosh. Others use common windowing systems such as, X Window or simple X.

An **imaging model** defines how fonts and graphics are created on the screen. Imaging models handle, for example, typeface and size in a wordprocessor or curves and lines in a drawing program. This component of the system environment has taken on increasing sophistication as applications incorporate complex curves, colour, shading and dimension. Some GUIs support more than one imaging model.

The API is a set of programming language functions that allow the programmer to specify how the actual application will control the menus, scroll bars and icons that appear on the screen. Like windwing models, APIs align with particular GUIs.

Finally, **GUI development environments** can include toolkits and frameworks. Most of these toolkits are based on **object-oriented approach**.

Although the structure of the basic development for most GUIs is similar, there are major differences in how the GUI is integrated with the operating system. Some, like the Macintosh and NeXT GUIs, are closely integrated with the operating system. Others, like X Window or Microsoft Windows, can be set up as options to be selected by the user when the computer boots up.

Programming of software for GUIs across these components is fairly complex and challenging Commercial developers who want to support multiple environments find their task further complicated by the absence of standards across heterogeneous computing platforms. The higher-level toolkit component is intended to mitigate much of this difficulty.

Although the graphical user interface has become a standard component of most systems, no standards in windowing systems, imaging models, APIs, or high-level toolkits have emerged. However, three major camps of GUIs dominate. The first camp is IBM's System Application Architecture (SAA), which includes primarily Microsoft's Windows and PM (Presentation Manager). The second camp is UNIX systems, usually build around X Window. The third camp is the Macintosh. In the next section we will discuss about the GUI interface provided by MS-Windows.

Check Your Progress 1

1. What is GUI and what are its features?

2. Define the features of a windows.

3. What is the difference between Bitmapped and character based displays?

3.6 A LOOK AT SOME GRAPHICAL USER INTERFACES

This section presents Graphical User Interfaces. A popular GUI will be covered to provide a broad picture of the subject. There are a great many popular GUIs around including X Windows, Microsoft Windows, NeXT's NEXT Step and others.

3.6.1 Microsoft Windows (MS-WINDOWS)

MS-Windows is the most popular GUI for IBM personal computers. IBM and Microsoft announced OS/2 as a new operating system for 80286 and 80386 personal computers. The OS/2 Standard Edition 1.1 adds Presentation Manager (PM) for its graphical user interface. The user interfaces of Windows and PM are very similar but their APIs are different. Microsoft's strategy is to use Windows as a springboard for PM.

Features Overview

Windows provides an environment that enhances DOS in many ways. The major benefits of Windows are

1. **Common Look and Feel :** All Windows applications have the same basic look and feel. Once you know one or two Windows applications, it is easy to learn another one.

2. **Device Independence :** Windows presents a device-independent interface to applications. Unlike most of today's DOS applications, a Windows application is not bound to the underlying hardware such as mouse, keyboard or display. Windows shields the application from this responsibility. The application deals with the Windows API to manipulate any underlying devices.

3. **Multitasking :** Windows provides non-pre- emptive multitasking support, Users can have several applications in progress at the same time. Each application can be active in a separate window.

4. **Memory Management :** Windows also provides memory management to break the 640K limitation of MS-DOS. An application has the ability to use the extended memory, share data segments with other applications and unwanted segments to disk.

5. **Support for existing DOS applications :** Windows allows most standard DOS applications to run under it directly. Any application that does not control the PC's hardware, use the PC BIOS or MS-DOS software interrupts, can run in its own window.

6. **Data Sharing :** Windows allows a data transfer between application Clipboard. Any type of data can be transferred from one window with the Clipboard. The Dynamic Data Exchange (DDE) protocol, defines, how two applications can share information. Information such as bitmapp, metafile, character strings and other data formats can be shared.

Support for Object Orientation

In order to create screen objects such as windows, the application developer defines a class (similar to record) specifying the necessary properties. Instances of class can then be created. Several applications can share the same windows simultaneously. To communicate with instances of a window class, messages are sent and received by a special function called the window function. The windows handles all messages such as re-drawing the screen, displaying icons or pop-up menus and changing the contents of the client area.

Creation and Manipulation of a Window

MS Windows presents a pre-defined style for user-defined windows. It presents the structure of a window, followed by a discussion of windows manipulation.

1. **Structure of a window :** Figure 10 displays possible elements for window. The caption bar (or title bar) displays the name of application within the window. The system menu box contains names of commands available in all applications, such as minimise, mixmise, resize and close. The minimise box clicked once reduces the window to an icon. The maximise box enlarges the window to the full screen. The menu bar contains a list of commands in the application. The client area is the area inside the window which is under the application control.

2. **Creating windows :** Each window created on the screen is a member of some user-defined window class. Window classes are created by application programs. Several window classes can be active simultaneously. Each window class in turn can have several instances active at the same time. There are no pre-defined generic window classes that come with MS windows.

To create a window the following steps must be taken .

a. **Setup a window class structure** which defines the attributes of the window class. Attributes that can be defined include:

   (i) the window function, which handles all messages for this class.

   (ii) the background colour of the client area

   (iii) the window class menu

   (iv) the re-drawing function used when resizing horizontally or vertically.

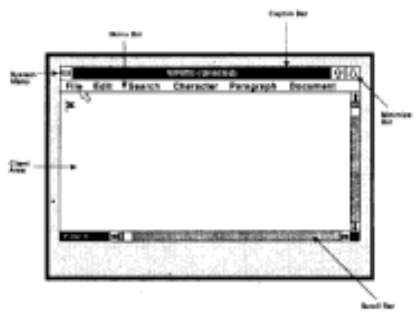

Figure 10 : MS-WINDOWS screen elements

3. **Manipulating windows :** An application can choose to display the window, resize the window, display additional information in the client area, and so on.

Pop-up and Child Windows

Pop-Up and child windows are special types of windows, and are used to communicate information between the application and the end-user. They remain on the screen for a short period of time. In this example, the pop-up displays information about a given file such as date and time of creation and the size. Dialog boxes as discussed earlier, are more sophisticated form of pop-up windows.

MS-Windows provides a collection of pre-defined child windows. These are the most common usage of child windows. These pre-defined classes are buttons, scroll bars, list box, edit and static class. A developer can also define child windows that can be controlled by any user-defined operation.

Resources

Resources are used to manage windows and user-defined objects. MS-WINDOWS provides nine kinds of resources to application developers. These resources are : icons, cursors, menus, dialog boxes, fonts, bitmaps, char strings, user-defined resources, and keyboard accelerators.

1. **Icon and cursors :** Windows defines a few types of icons and cursors. An icon or a cursor is essentially a bit-mapped region that is used to represent and symbolise a window or cursor. A developer can also define an original icon or cursor using the ICONEDIT utility.

2. **Menus :** Each window can have its own menu bar. A menu item can be a character string or a bitmap. Each item of a menu bar in turn can have a pop-up menu presenting a list of options. Currently, Windows does not support nesting of pop-up menus within other pop-up menus. (Windows 3.0 provides this functionality). But a pop-up menu can invoke a dialog box. When a menu is selected, Windows sends one or more messages to the Window function of the window containing the menu bar. These messages can be interpreted to perform the function corresponding to the menu item.

3. **Dialog boxes :** These provide another mechanism besides pop-up menu and menu bars to obtain information from the end-user. Dialog boxes are much more flexible than menu bars or pop-up menus. Dialog boxes usually contain a group of child windows such as buttons, scroll bars, and editable fields. Just like windows, dialog boxes have a function that is used to process messages received from the user upon selection of options. Generally, dialog boxes appear as pop-up windows. The user selects the option needed from the dialog box and then the dialog box disappears. Figure 11 depicts an example of a dialog contains an edit box, a list box, and open and cancel buttons. The end-user can specify the name of a file either by selecting from the list box or by typing the name of the file in the edit box. By clicking on the open button, the application will open the selected file.

4. **Fonts :** Windows provides a few families of fonts with different sizes and shapes : modern, roman Swiss, helvetica, and script. Application processors and desktop publishing can define addition fonts as needed.

5. **Bitmaps :** They are used to represent icons, cursors, or draw picture on the screen. Both mono and colour bitmaps can be defined.

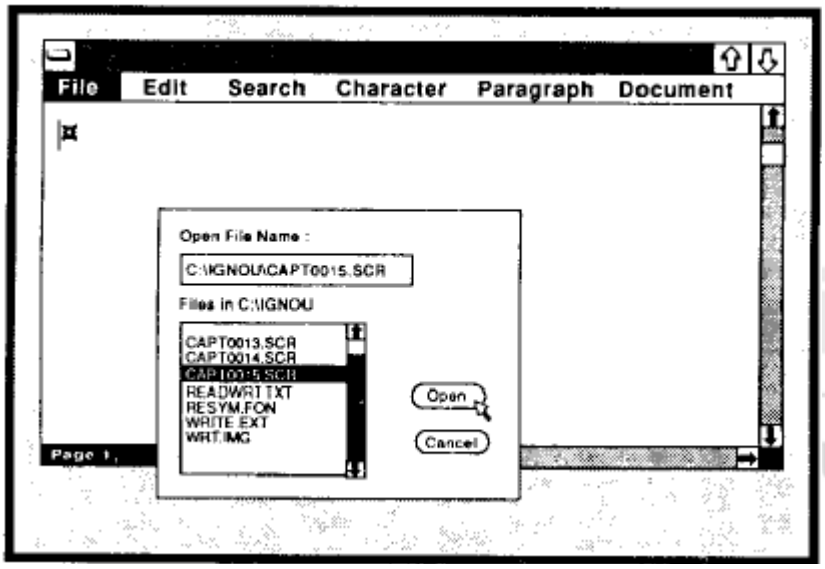

Figure 11 : Dialog box (MS-WINDOWS) with an edit box, a list box, and push buttons

6. **Character Strings :** Character strings are handled as resources mainly to provide a manageable solution to internationalisation of a window application. Instead of including a character string as part of the code, it can be placed in a resource file and handled as resource. Once in a resource file, different versions of the same character string resource file can exist for different languages. This separation of the character string from the code makes internationalizing the application much easier, and also makes maintaining the application much simpler.

7. **User-Defined Resources :** These can be used for any purpose and support any user-defined data type. Any arbitrary data can be managed as a user-defined resource.

   Resources are defined in text file call, a resource script. They are compiled through a utility called the Resource compiler and linked with the Windows application. Resources are read-only data. Once a resource script is compiled, it can be shared with other window applications.

Graphics Device Interface

Graphics in Windows are handled by the Graphics Device Interface (GDI).Both vector and raster colour graphics are supported by GDI. GDI supports only two-dimensional graphics. Vector graphics are supported by a set of vector drawing functions such as drawing a line, point, polygon etc. or pie chart.

Raster graphics are supported by pixel manipulation. Raster graphics can be stored or manipulated either as a bitmap or as a metafile. A bit-mapped representation of the graph can be manipulated using utilities such as BitBlt, PatBlt and StretchBlt. A metafile provides binary encoding of GDI functions such as to draw vectors and to fill a region with a bitmap. Metafile take up less disk space than a bit-mapped representation since they do not represent each pixel directly on disk. When metafiles are played, they execute the function encoding and perform the necessary graphics operations to display the graphics output.

Check Your Progress 2

1 . What are the four major components of GUI? Explain the functioning of any on component.

---

2. How does MS-Window enhance DOS environment?

3.7 SUMMARY

The GUI has already made a tremendous contribution to the increased usability of computer systems. It is with great excitement that we look forward to future innovations in human-computer interfaces. GUI development is at the vanguard of creativity in many areas, such as ergonomics, software development tools, computer graphics and linguistics to name just a few.

The past decade has seen rapid change in the understanding and definition of GUIs, but we have only been through the infancy of GUIs. There remains much to be done in terms of increasing the productivity of computer users, standardising operations across different architecture and adapting the human-computer interface to non-traditional applications. In this unit we discussed several issues related to GUI including functioning of several popular packages.

3.8 MODEL ANSWERS

Check Your Progress 1

1 GUI is a system that allows creation and manipulation of user interfaces employing windows, menus, icons, dialog boxes, etc. The basic features of GUI are:

- Pointing device such as mouse which controls cursor.
- Point and shoot functionality under control of device which cause screen menus appear or disappear.
- Support of windows which graphically display the status of a computer program.
- Icons that represent files, directories and other application and system entities.
- Support of graphical metaphors such as pull-down menus, dialog boxes, buttons, slides that let the programmer and user tell the computer what to do and how to do it.

2. When a screen is split into several independent regions each one is called a window. Several applications can display results simultaneously in different windows. The user can switch from one window to another window. Windowing systems have capabilities to display windows either tiled or overlapped. Users can organise the screen by resizing the window or moving related windows closer.

3. Bit-mapped display is made up of tiny dots (pixels) that is an independently addressable and has much finer resolution than character displays. Bit-mapped displays have the advantages over character displays. One of the major advantages is of graphic capability.

Check Your Progress 2

1. There are four major components of GUI.

   (i) A windowing system

   (ii) An emerging model

   (iii) An application program interface

   (iv) A set of tools and frameworks for creating interfaces and developing integrated application.

   The API (Application Program Interface) is a set of programming language functions that allow the programmer to specify how the actual application will control the menus scroll bars and icons that appear on the screen. Liking windowing models, APIs align with particular GUIs. Its features vary from package to package.

2. There are several ways MS-Window enhances DOS environment.

   (i) Device independence. It presents a device independent interface to applications. Unlike most of today's DOS applications, a window application is not bound to the underlying hardware such as mouse, keybord or display windows.

   (ii) Multitasking. Users can have several applications in progress at the same time. Each application can be active in a separate window.

   (iii) Memory management. Windows also provide memory management, the 640K limitation of MS-DOS. An application has the ability to extend memory and share data segments with other applications.

   (iv) Data sharing. Clipboard allow data transfer between application clipbord. Any type of data can be transferred from one window to another through the clipbord.