Chapter Three

Software Systems for Personal Computers

Those who educate children well are more to be honored
than those who produce them; for these only gave them life,
those the art of living well.

Aristotle (384-322BCE)

There are no shortcuts in the quest for perfection.

Ben Hogan (1912-1997)

The notion of time is an intellectual construction.

Jean Piaget (1896-1980)

LEARNING OUTCOMES

In the previous chapter you learned about the hardware of educational computing systems. You also learned about one particular part of the hardware called ROM—Read Only Memory. You may recall that ROM is comprised of computer chips on which are etched programs used by the Central Processing Unit (CPU) at start up to bring the system to life, so to speak—a process called *bootstrapping*. The CPU reaches down into the ROM chips to get the instructions that enable it to receive further instructions from other parts of the system. Turning on the power switch acts as a wake up call for the computer to go through this bootstrap routine, flexing its muscles and shaking itself awake to greet a new day. Without these basic routines or programs the computer would be useless.

Bootstrapping is part of the operating system of a computer. In this chapter we will learn more about the functions of operating systems. We will also learn about two other types of software that depend on the operating system and that are essential to productive use of the computer. These are the user-interface software and applications software.

It is beyond the scope of this book to present the ins and outs of any specific operating system, user interface, or application. If, however, you want to work your way through one of the sets of tutorials that accompany this text, you will have a hands-on opportunity to learn the different productivity applications in Microsoft *Office*. These tutorials may be found online at the following web addresses:

Microsoft Office 2000: http://www.pitt.edu/~edindex/Officeindex.html

Microsoft Office XP: http://www.pitt.edu/~edindex/OfficeindexXP.html

Microsoft Office 2003: http://www.pitt.edu/~edindex/Office2003frame.html

Chapter 3 will introduce you to the key features of the three basic types of software: the operating system, the user interface, and applications. The focus, however, will be on the operating system and the user interface. This is because, in chapters 5 through 10, we will examine a range of applications software that has proved itself effective in the classroom when carefully integrated into the K-12 curriculum.

Specifically, this chapter addresses the following topics:

- The Layers of Software
  - Firmware: software that is part of the computer's hardware
- The operating system
  - A definition of a traditional operating system
  - The definition of a modern operating system
  - The operating systems most commonly found in schools
  - Utilities
  - Common operating system functions
  - Useful user-controlled operating system functions
- The Graphical User Interface (GUI)
  - The problem of non-standard user interfaces
  - Standardization through integrated software or suites of programs
  - Standardization through graphical user interfaces (GUIs)
  - Some background history of the GUI
  - GUIs and educational psychology
- Applications software

THE LAYERS OF SOFTWARE

At the heart of any computer system is the hardware, discussed in the previous chapter. Layered onto the hardware are the different types of software that allow us, the users, to carry out useful tasks. Figure 3.1 on the next page illustrates this idea.

An operating system is often represented as the inner of the layers of software that make the power of computing machinery available to the user. As you can see, the operating system, comprised of a set of programs, comes between the hardware and the applications software. The operating system makes it possible for us to work (use applications) on the computer without having to worry more than absolutely necessary about the computing machinery involved in the background.

The user interface (most commonly Microsoft's Windows graphical user interface or GUI) is the set of programs that controls the way in which the computer system makes itself available to the user. A well-designed user interface will go a long way to accommodating a diverse world of users.

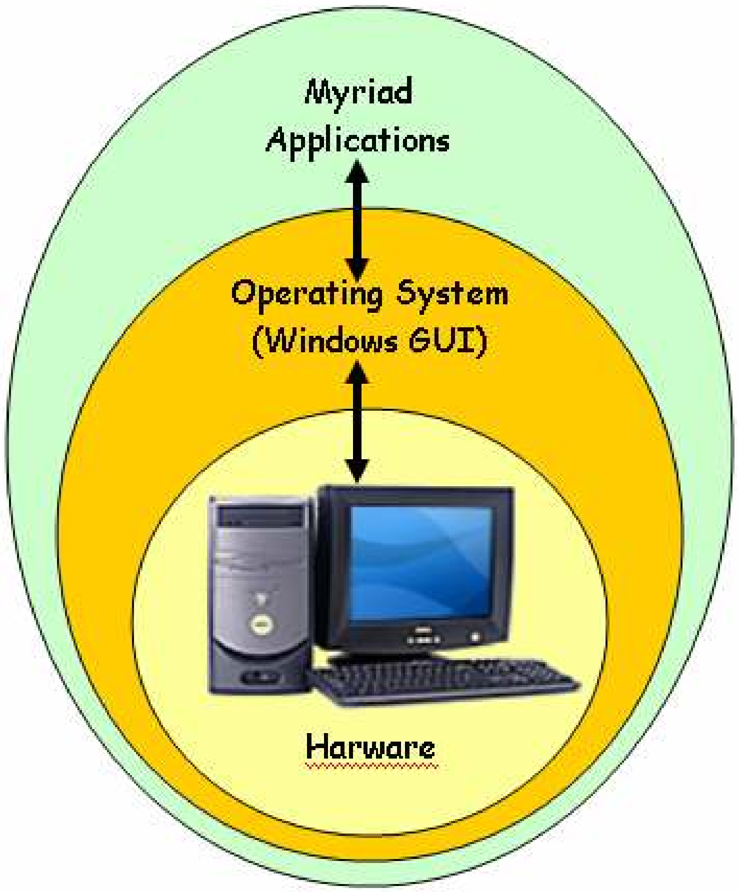

Fig. 3.1 The Layers of Software

The user interface (GUI) consists, on the one hand, of the software that controls how you interact with the computer system, whether by typing at the keyboard, or by pointing and clicking with the mouse, or by issuing voice commands, or by touching the screen, or by pressing a switch. On the other hand, the user interface consists of the software that controls how the computer interacts with us, whether by displaying text on the monitor or on a printer, or responding in other ways such as flashing or beeping or talking back to you, and so on.

Firmware: software that is part of the computer's hardware

The arrows connecting the different layers of software in figure 3.1 indicate that all the programs that you are using at any one point in time—the operating system and the applications, along with the user interface—must be loaded into the computer hardware, in either RAM or ROM, so that you can do your work. ROM, you will recall from chapter 2, contains at the very least the programs that the system needs to set itself up (called bootstrapping) and to make itself available to carry out all the other tasks required by you, the user. This basic set of programs is called the Basic Input/Output System—BIOS for short.

The BIOS is part of the operating system software that is *hardwired* onto the ROM chips. For this reason it is called *firmware*—software that is part of the chips' hardware. The advantage of firmware is Speed with a capital S! The system can access data much more quickly from primary memory (RAM and ROM) than from secondary memory (the disk drives).

Firmware is thus software that has been hardwired onto the chips which are part of the computer's hardware. Until the 1980s and 1990s, firmware was a relatively precious commodity because the density with which the electronic components (switches and circuits and so forth) could be packed on to the surface of the chips was not great. However, there have been breathtaking advances in chip technology since the integrated circuit was invented in 1958. Speed, as was noted in chapter 2, is a function of space and there is a rule of thumb called Moore's Law[1] (Moore, 1965) that predicts that computers will double in processing power every 12 to 18 months. This doubling occurs because the computer engineers have relentlessly microminiaturized the switches on the chips and in other ways optimized their performance.

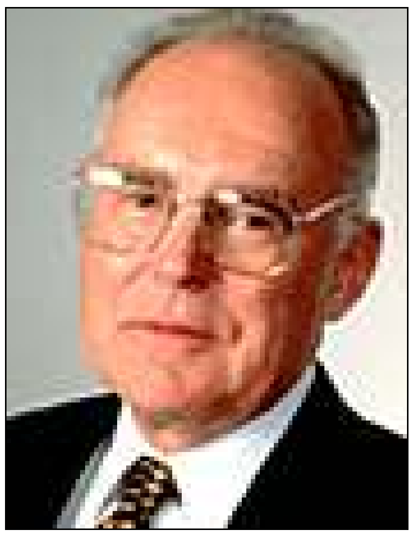

Fig. 3.2 Gordon Moore, Chairman Emeritus of Intel Corporation

In general, the more circuits that can be packed onto the surface of a computer chip the more powerful the machine is and at a cheaper price. It will soon be possible to store 100s of millions of electronic components (bits or switches, as they are called) onto a single chip no bigger than your little finger nail. You need 8 bits to store one character, and on a typical printed page of text you have maybe 3,000 characters. So if all you wanted to do was store text, you could easily store on a single chip the equivalent of over one hundred 500 page books!

Such chips make it viable to increase the number of programs etched as firmware onto the ROM chips. This has already opened up a whole new industry based on the production of what have

[1] Gordon Moore (Fig. 3.2), one the founders of Intel Corporation, hypothesized—correctly thus far—that the processing speed of microprocessors would double on average every 18 months. At times this doubling has occurred at 12 month intervals, at others 24, but the 18 month average seems to be holding up. The speed of microprocessors, now at the 2 gigahertz rate plus, is such that doubling is a significantly more dramatic phenomenon than ever.

been called "software ICs"—computer chips (containing *I*ntegrated *C*ircuits) that have pre-packaged programs printed on them. It is a distinct possibility that before long you will buy a computer with options packages, like you buy a car. The vendor will simply plug into the logic board inside the computer the chips you need for specific applications such as word processing and so forth.

But that is in the future. For the time being most software is still loaded into RAM from secondary storage. The computers you will be using in schools have either magnetic disks or optical discs for this purpose, and it is the operating system that takes care of the humdrum tasks involved in running the software on the system. This is the subject of the next section.

THE OPERATING SYSTEM

The primary objective of this section is to encourage you to learn about the operating system of any computer you use because to do so elevates you to the level of a power user of that system. A very high percentage of computer users, understandably to some extent, learn only what they think they need to know about a computer to run a particular application such as a word processor. They never discover the many useful, built-in functions of modern computer operating systems such as sorting, merging, backup, use of subdirectories or folders, copying files and disks, multitasking, and on and on. As a result, even the little use they make of the computer is inefficient, often disorganized, and sometimes disastrous, as when data are lost because of failure to backup files.

Familiarity with your computer's operating system gives you a higher level of control over the computer, which translates into power and peace of mind. A secondary objective of this section is thus to help you appreciate the value of the operating system.

Using a computer would be a nightmare if you had to manage all the computer's resources for yourself—which is what the early computer pioneers had to do before they cottoned on to software-controlled operating systems. Not so long ago (1970s), the computer operator was a *person*. People studied for an Associate Degree in computer operations. That was before systems programmers wrote the software that now allows a computer to more or less take care of itself. For the most part, people like me and you are now thankfully relieved of this responsibility.

So… What *is* an operating system? It is beyond the scope of this book to present the features of all but the most common computer operating systems, and we will focus on operating systems for personal computers. Operating systems, however, now all behave in much the same way. Any differences are the result of idiosyncrasies built into them by their creators.

A definition of a traditional operating system

> **An operating system consists of a set of programs that give added value to the computer by making available to the user the full resources of the computer hardware for which the operating system has been designed.**

An operating system not only allows a computer to manage its operations and resources, but it also makes them available to you, the user, so that you can carry out the various applications that you had in mind when you bought your computer. An operating system thus makes it possible for you to get the utmost out of the computer's various control, input, output, and communications components. Davis (1978) put it this way: "A well-designed operating system is not concerned with just hardware or just software or just data management, but with optimizing the way in which all of these resources work together..."

The quotation from Aristotle at the opening of this chapter was not chosen simply to make you feel good about your choice of teaching as a profession. It also has some analogous relevance to operating systems. Without stretching the analogy too far, one might say that an operating system is to a computer what an education system is to a child.

An understanding of this idea might deepen your appreciation of the role of an operating system.

The behavior of a human being is to some extent determined by innate capabilities—nature if you will, about which you learn a great deal in your education and psychology classes at college. These innate capabilities are akin to the characteristics of a computer that are built into the machinery, hardwired onto the chips in the CPU and ROM. One does not have too much choice about them. We are who we are, just as a computer is distinctively whatever model of computer it is designed to be.

As children grow, however, they are exposed to life as it happens to them and around them—nurture, if you will. One way or another, they become educated. This process of education (nurture in the broadest sense) further individuates a person, ideally for the better—hence the crucial role of good parenting and of good teachers. Aristotle is implying, of course, that the best parents are also good teachers.

Continuing the parallel between a person's education and a computer's operating system, we might say that the operating system gives the computer an extra layer of individuation or character, if you will forgive the anthropomorphism. An operating system thus increases the potential of the computer to serve our needs, just as education aims to increase the potential of the human being.

The definition of a modern operating system

Today we can refine our definition of an operating system by adding one ingredient that is having a dramatic impact on computer use. This ingredient is *accessibility*. With accessibility in mind, we can rewrite the definition of an operating system as follows:

> **A *modern* operating system consists of a set of programs that gives added value to the computer by making available to the *general* user *in an easily accessible way* the full resources of the computer hardware for which the operating system has been designed.**

The ingredient of general accessibility was a side issue for the first thirty years or so of operating system design, largely because it was too expensive to implement, but also because computer users were almost invariably technical experts. At first, the hardware technology just was not powerful enough to handle both the processing involved in carrying out computational tasks and at the same time present a comfortable, user friendly face to the world. Also, the early users of computers were invariably techie types who relished the challenge of figuring out how to make a computer do what they wanted. Some of these techie types even resisted user-friendly interfaces.

In the mid 1980s, this author overheard one computer "nerd" declare to another: "I hate the Mac; it's too easy to use!"

Fortunately, in anticipation of the machinery of computing becoming capable of handling interactive displays using colorful, high quality graphics, a few insightful individuals in the 1960s and 1970s, Doug Englebart and Alan Kay in particular, designed and wrote programs that transformed the computer into a machine that *invites* use by *ordinary* people like you and me! We'll take a closer look at these inventors work later in the chapter.

A good operating system today does its job in such a way that you are barely aware of the complex activity that has to go on in the background. When you or your students use the computer to do some word processing or practice math skills or engage in any one of the myriad applications designed for computers, you will not have to worry about how the computer manages the cursor on the monitor screen, or how the system finds the data on the disk in the disk drives, or how files are copied, or how the system switches from one process to another. The full capabilities of the machinery are easily—and transparently—available to you.

If you are old enough to have used personal computers in the early 1980s, you no doubt appreciate what a difference the graphical user interface (or GUI) has made.

Utilities

A utility is a program extension of an operating system. More and more utilities are part and parcel of the operating system, so they are already installed when you buy the computer. Think of a utility as a tool on a workbench[1]. It helps you make more efficient use of your computer.

For example, most of today's operating systems have built-in calculators. All the computers have clocks which include the date. A notepad is useful for To-do lists, memos, reminders, and other jottings. The Mac has a scrapbook, very useful for cutting and pasting, and for gaining ready access to your favorite graphic images. Programs for painting and drawing are now part of the operating system, as are tools for working with graphic images such as photos and creating and editing movies. These latter programs are relatively simple when it comes to what they can do, but they are easy to use. Expert graphic and movie editors will want to purchase more sophisticated software than what is built into operating systems.

Accessibility tools are part of the built-in set of Windows XP programs, such as tools for the sight impaired which magnify text or narrate text out loud. There is also an onscreen keyboard for touch screen typing using the mouse to point and click on the letters—useful for folks who have difficulty using their hands to hit the keys on a keyboard.

A utility also helps you accomplish certain tasks that extend the capability of the computer. These utilities are called Accessories in the *Windows* operating system. Disks which have become fragmented, for example, with pieces of files spread out all over the disk, can be reorganized and optimized (defragmented) to make data storage and access more efficient. We will return to this subject shortly when we discuss the functions common to operating systems for popular educational computing systems.

---

[1] The operating system for an early PC called the *Commodore Amiga* used the name *workbench* for its Graphical User Interface, and the available utilities were called tools.

Other utilities, especially utilities for protection against computer viruses, are provided by independent software companies, such as McAfee, Inc. or Symantec Corporation's *Norton Utilities*. These packages of security utilities not only provide tools to protect your computer from viruses, but also restores lost files, manages memory, and accelerates access to the data stored on your disks. Crashed or accidentally formatted hard disks can be diagnosed and recovered. Files that have been trashed can be recovered, too.

The number of utilities available today is too large to list here. As you become more and more familiar with your computer, and as you exchange ideas about computing with your classmates, colleagues and students, you will discover those utilities that are most useful to you.

Common operating system functions

Here is a list of functions typically taken care of by the operating system of a modern personal computer; the operating system must:

- manage the transfer of data between primary storage (RAM) and secondary storage devices;
- manage the secondary storage medium itself along with the data stored on it;
- manage the transfer of data between peripheral devices and RAM;
- manage computer interaction with communications devices and provide support for communications networks;
- manage the allocation of storage space in RAM to several applications (multitasking).

Let us look at each of these in a little more detail.

Managing the transfer of data between primary storage (RAM) and secondary storage devices Typical secondary storage devices on PCs are hard and floppy magnetic disk drives, including zip drives and the newly available USB Flash drives, and optical disc drives such as CD-ROM and DVD drives. These devices were described in some detail in the previous chapter on hardware. Several tasks are involved in the transfer of data to and from the disks. If data are being transferred ("input") from secondary storage to RAM, the operating system must locate the correct pieces of data, belonging to the correct file, on the correct disk, in the correct drive. This may mean following a trail of pointers if the file is split up into sections on different parts of the disk.

As the data are transferred into RAM they must be stored in memory locations on the computer chips that have not already been taken up by other instructions and data. And when the file is stored back on the disk the operating system will have to find free space on the disk if the file has been enlarged as a result of the updating process.

When data are input by the user from some device such as a keyboard, they are initially stored in the computer's primary memory (RAM). If this same data are then transferred ("output" or "saved") from RAM to secondary storage they have to once again find their way to the correct file on the correct disk in the correct drive. The operating system is also responsible for informing the user if any problems occurred in the whole process, problems such as damage to a disk or a non-existent file, and so forth.

Managing the secondary storage medium itself, along with the data stored on it An important operating system utility we mentioned earlier is called disk optimization. This utility involves telling the operating system to reorganize the files stored on a disk so that they are not split up into sections that are scattered over the surface of the disk. Imagine how long it would take to gather together the contents of a book in a library if each chapter were stored on a different shelf! Likewise it takes the computer longer to find the contents of a file when the file is not all together in one place on a disk.

So disk optimization is the process whereby the operating system gathers together the scattered parts of each file. The benefit of this process is not just a saving in time; it also saves space because it frees up sections of the disk that had become unusable because they were too small. Again, imagine if, in a library, there were lots of small gaps between the books. Each of the gaps would be too small to take a book, but if you were to push all the books up together you would free up large blocks of space into which you could then store more books.

Other disk management tasks that are essential, and therefore common, to all modern operating systems include:

- formatting a disk in preparation for storing data files;
- listing the contents of a disk (often called the directory);
- organizing the files into logical groups of files called subdirectories or folders;
- renaming a file;
- locking and unlocking a file to protect the data by preventing them from being accidentally written over;
- backing up (copying) the contents of a file or an entire disk;
- displaying the contents of a file;
- deleting files.

Managing the transfer of data between peripheral devices and RAM A computer system has a wide range of peripheral devices such as keyboards, monitors, printers, scanners, and so forth (Fig. 3.3 on the next page). These are items of hardware that are usually hooked up directly to the system unit of the computer and are therefore under the computer's control. They are often, though not always, located in the periphery of (near or around) the system unit, which is why they are called peripheral devices. In and of themselves they are not essential to a computer's operations. In the very early history of electronic computers there were few, if any, peripheral devices. All data were input and output using either switches on the face of the huge machines or secondary storage media, which in the 1940s meant punched cards. It was not long, however, before other devices were added so as to increase the system's flexibility. Today there is a growing range of such peripheral computer hardware (figure 3.3 next page).

- **Input peripheral devices** Many of these peripherals are used to input data to the computer's primary memory (RAM). Input devices include the keyboard, the mouse, joy sticks, graphics pads or tablets, scanners (including bar code scanners in stores), voice recognition devices, touch screens, and the like.

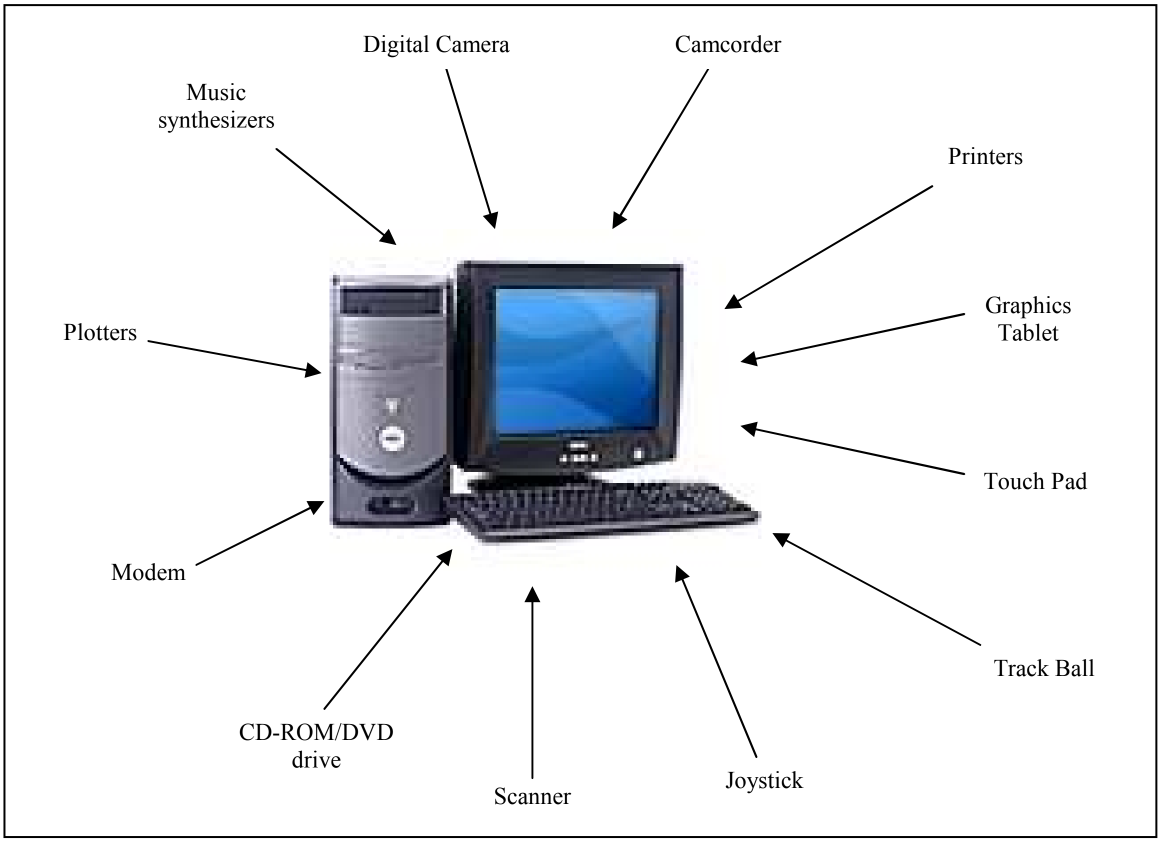

Fig. 3.3 Input and Output peripheral devices

- **Output peripheral devices** As illustrated in figure 3.3, another extensive range of peripheral devices is devoted to displaying data that are output from RAM. These include not only the monitor and the printer, of course, but also voice synthesizers, overhead projection systems, sound amplification systems, holographic devices that produce laser-based images, ultrasound imaging systems, and so forth. Notice that these peripherals do not include secondary storage devices such as disk drives, or cassette tape drives, or CD-R/CD-RW/DVD-R and DVD-RW. As already pointed out, secondary storage is essential to a computer system; peripheral devices are extremely useful, but a computer can manage without them.

Applications programs rely on the basic input/output capability of an operating system whenever it is necessary for the computer to interact with any of these peripheral devices. Thanks to the operating system the applications programmer's job is made easier since the instructions to read data *from* input devices, and to write data *to* output devices is largely taken care of by the operating system.

Managing computer interaction with communications devices and providing support for communications networks Advances in computer and communications technologies have given rise to a new perception of the computer as a tool for connecting people and data over local and wide area communications networks. In some schools, for example, students are collaborating on projects using e-mail (electronic mail). Students are also using wide area networks connected by the telephone system to log into electronic bulletin boards which enable them to communicate with other students around the world.

Networks are also useful for more mundane tasks such as allowing several computers to share a printer.

Communication between students in different schools, nationally or internationally, is no longer only in the form of text. Devices such as webcams, plugged into the system unit of the computer, enable real time interactive video and sound transmission. Several applications, such as Microsoft's *Netmeeting*, or Yahoo! *Messenger*, are making it possible for students to routinely hold internet-based video conferencing sessions with students on the other side of the globe. Cell phones, too, are now available with still image and video capabilities. Computer and communications services are merging rapidly as advances in technology emerge.

Communications hardware and software take care of the various rules of the road (called protocols) that have to be observed in order to make a successful connection between different computers on a network. The software to do this is often part of the operating system of a computer and is designed to allow the user to send and receive data, specify protocols (data transfer speeds, and so forth), and ensure that the data are as error-free as possible. The growing use of networked learning environments will be discussed at length in chapter 7 when we focus on these specialized communications technologies.

Managing the allocation of storage space in RAM to several applications (Multitasking)
Many of the utilities available for your computer system are designed to run in the background while you are busy with other work such as word processing or recording your grades or designing some graphical materials for class. This ability of a computer to handle several programs at the same time is called *multitasking*.

Multitasking is an important feature of an operating system and it is becoming the norm on all computers today. It allows other utilities (or applications in general, for that matter) to be co-resident in memory, available at the touch of a function key or the click of the mouse button. In other words, you can have several programs open at the same time, even though the computer actually works on only one program at a time.

It does this by allocating tiny slices of time and space (memory) to each task in rapid succession without the user being aware of the complex processing that the operating system has to manage in the background. The operating system divides RAM up into sections or slices, one program per section, and makes sure that the different programs do not interfere with each other as you switch back and forth between them.

Fig. 3.4 on page 68 illustrates the power of multitasking. Each of the separate windows on the screen represents a separate task being handled by the computer’s operating system. There are several other tasks going on in the background, none of which you can see, but all of which contribute to your productive use of the computer.

Multitasking operating systems give desktop computers the flexibility and potential for user productivity that was once possible only on large mainframe and minicomputer systems just a few years ago.

When we first learn to use a computer, we might be impressed by the speed at which things get done. But it is surprising how soon we adjust to this speed, take it for granted, and then become impatient when the computer makes us wait a few minutes, even a few seconds, as it completes a process that might previously have taken us hours without the computer!

As Piaget (1970) pointed out: "the notion of time is an intellectual construction." We all know how much longer it takes for a kettle to boil when we stand there and watch it. But if we have something else to keep us busy, the kettle boils in no time at all! Likewise, when we use a computer, we might become impatient when a printer seems slow, for example. With a multitasking operating system, we can overcome this "notion" of slowness by being able to get on with some job on the computer while it completes another process in the background, such as sending a file to the printer, recalculating a large spreadsheet, or making a backup copy of your hard drive.

Useful user-controlled operating system functions

For the most part, the operating system takes care of the management tasks described in the previous sections without your needing to be aware of what is going on. There are, of course, many jobs that you may want to do on an ad hoc, spur of the moment, basis. You may want to work with the files on your disks, for example. You may want to change the way your computer is set up (called the configuration of the system) or, more and more today, you may want to change the way you interact with the computer via the graphical user interface, which is the subject of the next section of chapter 3.

THE GRAPHICAL USER INTERFACE (GUI)

The problem of nonstandard user interfaces

Until relatively recently, the way software behaved to the user had little or nothing to do with the operating system. Programs designed to run under early operating systems such as Apple's *DOS* or Microsoft's *MS-DOS* behaved very differently from each other, even when they were designed to accomplish the same goal (say graphics or word processing) and to run on the same operating system! Whoever wrote the program—the word processor, the game, the course scheduler, and so on—also had to design and write the instructions to control how the user interacted with the computer. There was no standard method or style of interaction. For every new piece of software that came along, the user had to learn the language of the interface: How do I save files with this software? How do I load a file from disk? How do I name a new file? How do I change a name? How do I tell the computer to do this, that, or the other?

Baudville's *Award Maker Plus™*, for example, was like a foreign land compared to, say, a program such as Brøderbund's *Print Shop™*. Users of *Word Perfect* were lost when it came to using *Word*. All were excellent software tools, but one would never know by looking at them that they had been designed for the same operating system running on the same computer. One would experience a kind of culture shock when one made the transition from one software package to the other, and the learning curve would be much steeper than necessary as a result.

Those brave souls who mastered the basics of one or two applications often would not progress beyond this level of computer competency. The steep learning curve quickly exhausted whatever motivation they may have started out with.

And those were the brave souls…

The fact is that, until the last ten years or so, an inordinately high number of teachers took one look at what was involved in learning to use those early personal computer systems and, not surprisingly, they ran a mile rather than get involved.

School administrators who recognized the importance of the computer in the educational process, and who made the decision to invest in computer hardware for their schools, had to factor in the added expense of extensive in-service training for the teachers, just so that the teachers could learn how to use the machines and the software—let alone incorporate the technology into the curriculum. Where such training has not been available, the computers have all too often remained idle in classrooms and computer laboratories alike.

So until the 1990s, lack of standardization in the way computers "look and feel"—the user interface—was a major factor inhibiting many teachers from accepting computer technology. Fortunately two innovations have helped significantly in reducing people's phobias: integrated software and the graphical user interface.

Standardization through integrated software or suites of programs

To get around the problem of nonstandard user interfaces some software, called integrated software, combines into one program several of the most useful types of application. Claris Corporation's *AppleWorks*™, long the flagship in this regard, combines a word processor, a file manager (database), a spreadsheet, presentation tools, graphics programs for drawing and painting, and communications tools. It is still popular in those schools where the Mac is the computer of choice.

Along the same lines as integrated software is the Microsoft *Office* suite of programs, which has become the major market leader for productivity software on both the Mac and Wintel computing platforms. As pointed out earlier in this chapter, the authors of this book have prepared and made available free of charge sets of tutorials for Microsoft *Office* (*Office 2000* and *Office XP*), designed specially for teachers to help them learn how to integrate *Office* into their teaching.

The user interface for software such as this has few surprises because the different applications are closely associated with each other. Commands to copy, delete, move, find, or save data are the same whether one is manipulating text in the word processor, making entries into the database, or calculating values in the spreadsheet. Cutting and pasting data between the different components is also greatly simplified. Consequently the effort invested in learning to use the software is rewarded with easy access to the several applications that make up the suite of programs designed to increase your productivity.

Standardization through Graphical User Interfaces

Integrated software is one approach to standardization, but until the advent of the graphical user interface (GUI) the standardization was still limited to a specific software package. *AppleWorks* did not look like *ClarisWorks* or *Microsoft Works*, for example, and none of them looked like Microsoft *Office*. An experienced *AppleWorks* user needed to make a significant adjustment to learn to use *ClarisWorks* and of course the reverse was also true. Each computing environment—the Apple II, the Apple Macintosh, or the IBM PC—was quite different. But the Graphical User Interface—or GUI, as it is called—has changed all that.

As illustrated in figure 3.4, the key features of GUIs are the **W**indows, **I**cons, **M**enus, and **P**ointers (the components of the so-called **WIMP** interface) which are now common to all personal computers.

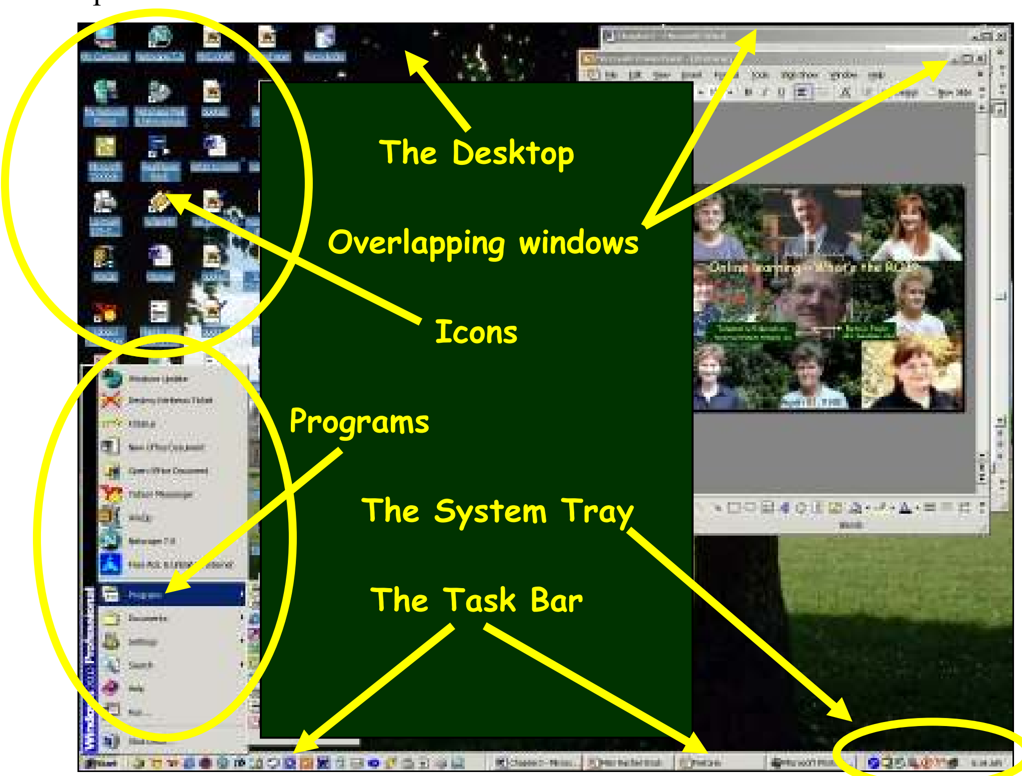

Fig. 3.4 The components of the *Windows 2000* graphical user interface (GUI)

Using the mouse or keyboard commands, the user is able to interact with the computer through these screen-based images which can be selected, opened, closed, browsed, moved around, overlapped, scrolled, and otherwise manipulated.

Once one has learned the basic set of GUI interaction skills, the technicalities of the computer become transparent to the user and the mind is set free to more easily process data and use the applications. When one uses a GUI-based system the computer no longer gets in the way. Quite the contrary, the GUI invites use with its attractive and consistent mode of interaction.

Some background history of the GUI

Not everybody famous is well known. We've all heard of Vincent van Gogh, Albert Einstein, and Tiger Woods. But who ever heard of John Vincent Atanasoff, inventor of the electronic digital computer, which is surely a machine that has changed the world? Who ever heard of Douglas Englebart, inventor of the GUI, which made Atanasoff's computer available to "the rest of us"?

We have already given John Vincent Atanasoff his due. Let us now recognize the work of Doug Englebart.

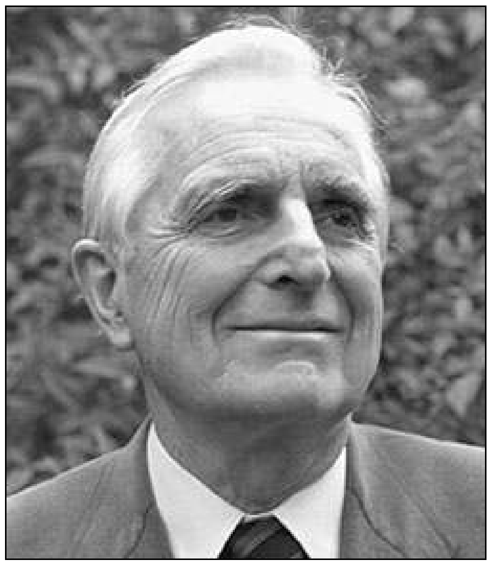

Fig. 3.5 Dr. Douglas Englebart, inventor of the mouse and other features of the modern GUI

Englebart pioneered early GUIs in the 1960s at Science Research International (SRI) (Metcalfe, 1992). He developed the original mouse-based, interactive, networked computing environments. He demonstrated them to his academic colleagues at professional conferences. He was somewhat ahead of his time because the only computers that could handle the demands of his computational ideas had to be the most powerful computers in the world at the time. But, as it turned out and as he foresaw, it was just a matter of time before Englebart's ideas became the everyday norm.

In the 1970s, his research was taken up by Bill English and Ron Rider at the Xerox Palo Alto Research Center (Xerox PARC). But the GUI that set the tone for the computer industry as a whole was that designed for the Apple *Lisa*, introduced in 1983. The *Lisa* was quickly superseded a year later by Apple's *Macintosh* computer.

The GUI for these Apple computers was the brainchild of Alan Kay, who had also conducted initial research at Xerox PARC before becoming a research fellow at Apple Computer Inc. Indeed, it was during a visit to the research facilities at Xerox PARC that Steve Jobs, founder of Apple Computer, Inc., got his first glimpse of a GUI. This was in the late 1970s and the experience led directly to the eventual introduction of the breakthrough GUI Apple *Macintosh.*

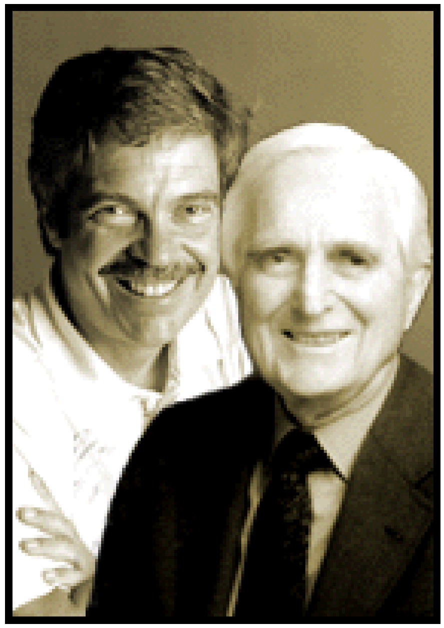

Fig. 3.6 Dr. Alan Kay looking over (and technologically standing on) the shoulders of Dr. Doug Englebart

On systems such as the Apple *Macintosh*, the user interface is part of the operating system. Anyone who writes applications for the Mac must follow the standard conventions (guidelines, as they are called) laid down by the system's designers at Apple Computer, Inc. So all the software for the Mac looks alike; it behaves in a consistent and predictable way. This has made the Mac very much easier to use. It also helped coin the 1984 advertising slogan describing the Mac as "The computer for the rest of us."

The success of the Mac in the marketplace did not go unnoticed by the other computer manufacturers and software developers, especially Paul Allen and Bill Gates at Microsoft Corporation. Several GUIs that reflect the look and feel of the Mac have been developed since 1984. Larger computers—mainframes, minicomputers, and work stations—have adopted *XWindows*™ or *Motif*™ which are GUIs for the *UNIX*™ and Linux operating systems. Microsoft Corporation's *Windows* products, designed as operating systems for Intel-based PCs, have beaten out other GUIs such as *TopView*™ and *Presentation Manager*™[1] to become the operating systems of choice for PCs.

Apple tried in vain to claim copyright infringement on their Mac GUI. They fought a six year long legal battle with Microsoft, in the balance of which hung Apple's survival as anything more than an also-ran in the personal computer marketplace. But in the end Apple failed because, as we now know, the GUI wasn't really Apple's idea in the first place.

GUIs and educational psychology

The design of the now classic GUI—the so-called WIMP interface with its Windows, Icons, Menus, and Pointers—is based on a study of the process whereby humans learn. As a student of

1 Both *TopView* and *Presentation Manager* were developed by IBM.

education, you will thus be interested to find out something of the research that led to the development of the Macintosh GUI.

The research team at Apple was headed by Dr. Alan Kay who, in Kay, 1990, described his conviction that "whatever user interface design might be, it was solidly intertwined with learning." As already pointed out in this chapter, Kay acknowledged his debt for the ideas that led to the Mac GUI to the work of many individuals, including Doug Englebart and the folks he worked with at Xerox PARC. But he singled out three men in particular because they helped him define the philosophy that drove his design of the Macintosh computer system.

The first was Jean Piaget, surely among the epistemologists best known to students majoring in Education. The second was Seymour Papert, creator of the Logo programming language and a disciple of Piaget. And the third was Jerome Bruner, whose learning theory was also much influenced by the work of Piaget.

Fig. 3.7 Dr. Jean Piaget (left) and Dr. Seymour Papert (right)

We will take a closer look at the work of these and other scholars at various points on our journey through this text. The computing paradigm that Kay formulated was based especially on Bruner's work and is what concerns us here as we study software in general, and computer operating systems in particular.

Bruner (1966) postulated that we have different modes of thought, "multiple separate mentalities" as Kay calls them.

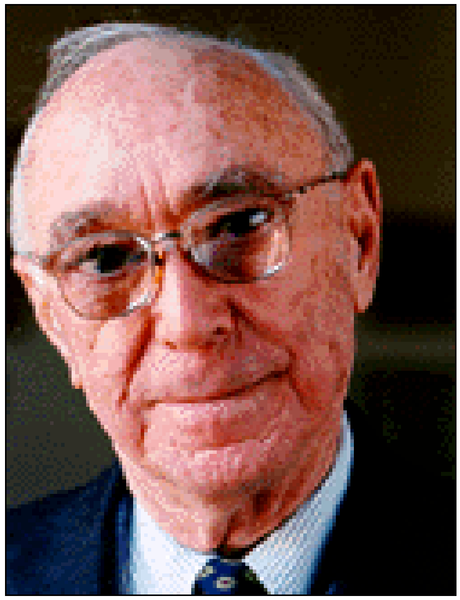

Fig. 3.8 Dr. Jerome Bruner

The many different ways in which we learn are closely related to the various stages of cognitive development identified by Piaget[1]. Three of these Bruner mentalities—called *enactive*, *iconic* and *symbolic* by Bruner—Kay interpreted as *doing*, *imaging*, and *symbolic* mentalities respectively, from which he formulated his model: "Doing with Images makes Symbols."

When we interact with Apple's *Macintosh* computer, or with any computer that enables the user to interact via a GUI, we are actively pointing and clicking with the mouse ("doing" in Bruner's terms), to select from pictures called icons (the "images" on the screen representing objects and operations), in order to carry out some intellectual ("symbolic") activity which extends our mental capabilities and promotes learning.

Today, the GUI à la Mac is becoming the norm on all computer systems, including Intel-Windows-based (Wintel) machines. It means that we can look forward to learning new systems and new applications without the struggle that used to accompany making a transition from one computer system to another, or from one application to another.

This is as it should be. Ask the children; they flock to computers as though they were the most natural things in the world.

1 The classic four stages of cognitive development in children: sensorimotor, pre-operational, logical operations, and formal logical operations stages. The sensorimotor stage corresponds to Bruner's enactive mentality where children learn essentially non-verbally by playing with and experimenting with the world around them. The pre-operational and logical operations stages correspond to Bruner's iconic mentality, where children use language and pictures to reason about their world. The formal logical operations stage corresponds to Bruner's symbolic mentality, where children and adults are able to think more independently about their world and life in general.

We should be thankful to people like Doug Englebart, Alan Kay, and their associates, along with the "giants on whose shoulders they stood."[1] They took computing out of the exclusive hands of technicians and made it available to the rest of us.

APPLICATIONS SOFTWARE

As observed earlier in this chapter in the discussion of integrated software and suites of software, five types of computerized applications are widely used in all professions, all walks of life. These include word processing, file or database management, spreadsheets, graphics, and communications. When they are sensibly integrated into the work environment by well-trained and experienced users, these applications are powerful productivity tools. This is as true for the teacher and student in the school as it is for those involved in other professions.

Of course, there are literally hundreds of thousands of other applications—programs designed to help the user use the computer to accomplish some task or other. A growing number of programmers, many teachers among them, are working alone or in teams developing new applications in response to the insatiable demand for software, a demand which only increases as the computers themselves increase in speed and power. The potential is quite simply boundless because of the nature of the computer as a Universal Machine. There is no end in sight to the ways in which the computer can be applied to solve problems and help us complete jobs more efficiently.

In the context of the classroom, applications have been developed to help the teacher teach and the student learn. The terms *Computer Managed Instruction (CMI)* and *Computer Assisted Instruction (CAI)* have been coined to describe such systems. Here is a partial listing of different applications of CMI and CAI that are used in schools today.

- **Computer managed instruction (CMI)** is a subject on its own and will be covered in detail in chapter 5. CMI is where the teacher uses software to prepare lesson and unit plans, make and give presentations, keep track of the progress of each individual student in the class with grading software, assessment tools, and other classroom management utilities and applications, and prepare the myriad documents (handouts, ditto masters, and so forth) that are the bread and butter of every teacher's job.
- **Computer-Assisted Instruction (CAI)** includes a huge range of applications where the computer is used to help students learn. Here are some of the types of CAI currently used in schools:

  Drill and practice, which has great value for both routine and remedial learning by providing computer-generated and computer-assessed repetitive practice of previously learned subject matter.

[1] Sir Isaac Newton is quoted a saying: "If I have seen farther than others, it is because I have stood on the shoulders of giants."

Tutorials, which enable students to learn new material at their own pace. The World Wide Web is becoming a rich resource for such online learning opportunities. Tutorials are also available on CD-ROM.

Simulations, where students experience a slice of "real" life in the "virtual" reality of programmed worlds, available on the Internet or on CD-ROM. DVD, often in the format of video games.

Microcomputer-Based Laboratories (MBLs), also called Computer-based Laboratories or CBLs, in which children conduct science experiments where the computer is a key tool in the collection and visualization of data.

Collaborative learning, where students use the computer to coordinate their efforts as a team towards some educational goal designated by their teacher.

Communications and Distance learning, where networks of computers linked by telephone communications enable students to attend classes from remote sites distributed around a region or even around the globe.

Multimedia applications, where students and teachers use interactive video, sound, graphics and text combined in a symphony of modalities to produce presentations and learning modules that are rich in intellectual stimulation.

Game and role playing applications, where students learn as they play. This is especially valuable for learning with younger children, though even older students and adults enjoy learning while they play, too!

In chapters 5 through 10 we will examine these CMI and CAI application areas in more detail.

LOOKING BACK

Chapters 2 and 3 have covered the fundamentals of computing. Think of this material on hardware and software as the foundation for what is to follow. The better you understand these fundamentals the more effective control you will have over the mechanics, the nuts and bolts, of computer use. Understanding how computers work will free you to take better advantage of the computer as a tool for teaching and learning.

There is a culture that surrounds the use of computers. Those who have become acculturated tend to save their work more frequently; they hesitate before pressing certain keys, depending on the software they are using; they handle disks carefully; they exit from software by closing files and shutting down the program before turning off the machine; they are not unduly surprised when the hardware or software fails because they routinely backup their work; and so forth. They have learned to accommodate the idiosyncrasies of computers, these dumb machines. That is all they are, after all.

The software that controls computers is only as good as the people who create it and the people who use it. One way or another, that means you.

LOOKING FORWARD

Going forward, it will be useful to examine the key features of each of the different types of applications software overviewed in this chapter, and at the same time discuss how they can be effectively integrated into the learning process. This will be the subject of chapters 5 through 10.

But first, in chapter 4, we will discuss different strategies for setting up educational computing environments. This is an important issue because too often schools just buy computer hardware and software and dump it into a classroom without sufficient consideration for what works best in terms of computer integrated teaching and learning. Chapter 4 is an important discussion of the issues involved in successful integration of computers into K-12 education.

In chapter 5, we will examine software that is useful to the teacher for the purpose of managing the instructional process.

Chapter 6 will survey the various types of software designed with the students' learning needs in mind: software for productivity, drill and practice, tutorials, distance learning, multimedia, simulations, and so forth.

In chapters 7 and 8 we will examine electronic communications and reflect upon the present and probable future impact on schools and schooling of the burgeoning global network of networks called the Internet.

Chapter 9 will take a practical look at the tools involved with internet-enabled online learning.

Chapter 10 is all about computer-based multimedia. We will discuss learning systems that have emerged in the last few years which take advantage of the integration of audio-visual materials with computer control.

In chapter 11 we will discuss software development. The challenge of creating quality software applications is not for the faint-hearted, but understanding the nature of that challenge will help you appreciate the talent of the people who create the software that you use. Perhaps you will even be bitten by the programming/authoring bug yourself.

For the most part, however, schools must pay for the hardware and software for computer-integrated learning. This does not come cheap, but plenty of money is available if you know how to get your hands on it. Chapter 12 looks at everything involved in fundraising in general, and grant writing in particular—that's the process of writing proposals to raise money from foundations and other sources of funds. This includes learning *where* to apply for money, *how* to apply for it, and *what* to do with the money once it has been won.

Computers are having a dramatic effect on our world. They are changing the way we live and work. They are also changing the way we learn. Advances in technology enabled by computerization are also causing us to confront new ethical and legal issues. Chapter 13 will provide an opportunity to read about, and reflect upon, these more philosophical topics in order to heighten the reader's awareness of the social issues that are raised.

One of the most promising benefits of computer-based learning is that it gives children control of their own learning. The pedagogical implications of this will be discussed in chapter 14 against the backdrop of educational theory.