

# 4

## COMPUTER SOFTWARE

### CHAPTER PREVIEW

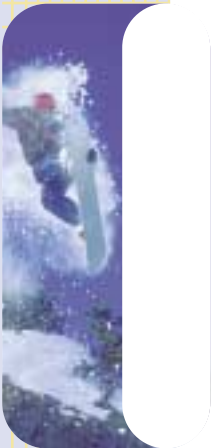
Computer hardware is only as effective as the instructions we give it, and those instructions are contained in *software*. Software not only directs the computer to manage its internal resources, but also enables the user to tailor a computer system to provide specific business value. It is surprising to many people that at the corporate level, software expenditures (development and purchase) typically are a much larger cost than is hardware. In this chapter we learn that computer software, in its various forms and languages, can be quite complex. But these complexities must be understood in order to truly be able to exploit the power of modern information technologies. This chapter explains to the reader the concepts of what software is, how it works, and how it is created. Along the way we provide examples of software's critical role in maintaining organizational competitiveness.

### CHAPTER OUTLINE

- 4.1 Software History and Significance
- 4.2 Systems Software
- 4.3 Application Software
- 4.4 Software Issues
- 4.5 Programming Languages
- 4.6 Enterprise Software

### LEARNING OBJECTIVES

1. Differentiate between the two major types of software.
2. Describe the general functions of the operating system.
3. Differentiate among types of operating systems and describe each type.
4. Identify three methods for developing application software.
5. Describe the major types of application software.
6. Explain how software has evolved and trends for the future.
7. Describe enterprise software.



## THE SOLUTION TO SOFTWARE BUGS

### *The Business Problem*

*nyse.com*

A failed software upgrade left investors unable to trade shares on the New York Stock Exchange for an hour and a half in June 2001. The shutdown, the second the exchange suffered in three years, prevented many stocks from opening for trading. The shutdown made calculating market indexes like the Dow Jones Industrial Average and the Standard & Poor's Index impossible. The failed upgrade was part of the system that electronically directs orders from securities firms to the Exchange.

Software bugs are errors in a computer program that either will not let the program run (fairly easy to find), or will let the program run but will produce incorrect output (very difficult to find). Writing computer code for computer programs (i.e., producing software) unfortunately remains more an art than a science. According to the Software Engineering Institute at Carnegie Mellon University, there are about 5 to 15 bugs in every 1,000 lines of computer code.

Many software bugs arise as a result of “good-enough software”—software released by software vendors before adequate testing is performed. Software bugs have plagued new releases of Microsoft Windows and Office products, Netscape Navigator, and Intuit's Quicken, among others.

Even when individual office software products are not buggy, corporate computing environments are so complex that they are inherently unreliable. Typically, these systems are collections of mainframes, minicomputers, workstations, PCs, Macs, and mobile devices, running different operating systems that were never designed or tested in combination. Further, these systems are running thousands of different software applications, some over 50 years old.

Integrated enterprise resource planning software from SAP, PeopleSoft, and Oracle may remedy some aspects of the software problem, if only because they tie operations together in one suite of application modules. However, even these systems must work throughout the supply chain, raising the biggest problem to date: the interconnectedness of complex systems. Over the Internet, software now links computers that were once insulated from one another, creating additional layers of complexity.

### *The IT Solution*

One solution is open source software. The “open source” movement draws programmers around the world together to continuously debug major programs. With thousands of programmers pooling their skills to build and test such programs, bugs can be discovered early. The Internet provides a platform for such collaboration and an instant feedback channel.

A second solution is more rigorous software development. Governments are joining with industry to impose greater rigor on software development, hoping to transform it from art to science. The National Science Foundation wants to provide software engineers with the information to create accurate, debugged modules of code that could be used over and over to assemble all kinds of systems. The ultimate goal is a library of these modules, each with built-in intelligent agents. To produce a program, an engineer would simply specify the software's function, and then the intelligent agents would coordinate among themselves to figure out how to patch together the desired result.

### *The Results*

The results are still unknown. The main question that remains is whether the IT solutions can keep pace with the growing complexity of the software. Another problem



concerns the extremely high costs of error-free software. The more that vendors test their products, the higher their costs, and the higher the prices to companies and consumers. There may be a happy medium between testing, bugs, and pricing, but as the New York Stock Exchange example above shows, any software bug can have wide-ranging and costly consequences.

Sources: “The State of Software: Quality,” *InformationWeek.com*, May 21, 2001; “Software Failure Halts Big Board Trading for Over an Hour,” *nytimes.com*, June 9, 2001).

### What We Learned from This Case

The importance of computer software cannot be overestimated. As we noted in Chapter 3, hardware expenses have declined over the past two decades; at the same time, software costs have increased. For any business, failure to account for, and have contingency plans in place for software bugs can have devastating results. Employees from every functional area often are involved in testing software products for bugs because they are the experts in that area and can spot functional-area-specific bugs even more quickly than IT programmers can. Therefore, as you study this chapter, keep in mind that, regardless of your major, more than likely you will be involved with some aspects of software very early in your career.

## 4.1 SOFTWARE HISTORY AND SIGNIFICANCE

The first applications of computers in business were in the early 1950s. Software was less important (and less costly) in computer systems then, because early hardware was literally hardwired by hand for each application. Today, software comprises a much larger percentage of the cost of modern computer systems than it did in the 1950s. There are several reasons for this trend. First, the price of hardware has dramatically decreased, while the performance of hardware has exponentially increased. Second, building applications—a process called *software development*—is slow, complex, and error-prone. Software is, therefore, expensive and getting more so as its complexity grows. Finally, salaries for software developers are steadily increasing because there is an increased demand for their skills.

### The Software Crisis

These factors have led to a major problem for management—the software crisis. The **software crisis** is that organizations are not able to develop new software applications fast enough to keep up with rapidly changing business conditions and rapidly evolving technologies.

Computer hardware can be designed and manufactured on automated assembly lines, and so can be turned out rather quickly, but software must be engineered by hand. Therefore, software generally lags several generations behind hardware. The result is that organizations are unable to make full use of hardware due to a lack of software to effectively exploit the hardware.

Further, organizations not only must develop new applications quickly, but they must also maintain their existing software. Often, more than 80 percent of IT personnel maintain existing software, leaving less than 20 percent to develop new applications.

The increasing complexity of software exacerbates the software crisis. This complexity naturally leads to the increased potential for errors or bugs. Large applications today may contain millions of lines of computer code, written by hundreds of people

over the course of several years. Clearly, the potential for errors is huge, and testing and **debugging** software is expensive and time-consuming.

## Software Fundamentals

Software consists of **computer programs**, which are sequences of instructions for the computer. The process of writing (or *coding*) programs is called *programming*, and individuals who perform this task are called *programmers*.

Unlike the hardwired computers of the 1950s, modern software uses the **stored program concept**, in which stored software programs are accessed and their instructions are executed (followed) in the computer's CPU. Once the program has finished executing, a new program is loaded into main memory and the computer hardware addresses another task.

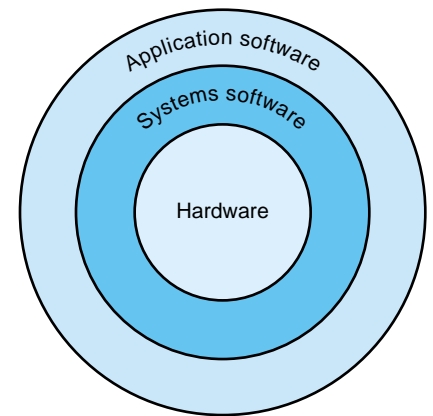
Computer programs include **documentation**, which is a written description of the functions of the program. Documentation helps the user operate the computer system and helps other programmers understand what the program does and how it accomplishes its purpose. Documentation is vital to the business organization. Without it, if a key programmer or user leaves, the knowledge of how to use the program or how it is designed may be lost.

The computer is able to do nothing until it is instructed by software. Although computer hardware is, by design, general purpose, software enables the user to instruct a computer system to perform specific functions that provide business value. There are two major types of software: systems software and application software. The relationship among hardware, systems software, and application software is illustrated in Figure 4.1.

**Systems software** is a set of instructions that serves primarily as an intermediary between computer hardware and application programs, and may also be directly manipulated by knowledgeable users. Systems software provides important self-regulatory functions for computer systems, such as loading itself when the computer is first turned on, managing hardware resources such as secondary storage for all applications, and providing commonly used sets of instructions for all applications to use. *Systems programming* is either the creation or maintenance of systems software.

**Application software** is a set of computer instructions that provide more specific functionality to a user. That functionality may be broad, such as general word processing, or narrow, such as an organization's payroll program. An application program applies a computer to a certain need. *Application programming* is either the creation or the modification and improvement of application software. There are many different software applications in organizations today, as this chapter will discuss. For a marketing application, for example, see the Market Intelligence box at the Web site.

In summary, application programs primarily manipulate data or text to produce or provide information. Systems programs primarily manipulate computer hardware resources. The systems software available on a computer system provides the capabilities and limitations within which the application software can operate. The next two sections of this chapter look in more detail at these two types of software.



**Figure 4.1** Systems software serves as intermediary between hardware and functional applications.

### Before you go on . . .

1. What is the software crisis and what causes it?
2. What are differences between systems software and application software?

## 4.2 SYSTEMS SOFTWARE

*Systems software* is the class of programs that control and support the computer system and its information-processing activities. Systems software also facilitates the programming, testing, and debugging of computer programs. It is more general than application software and is usually independent of any specific type of application. Systems software programs support application software by directing the basic functions of the computer. For example, when the computer is turned on, the initialization program (a systems program) prepares and readies all devices for processing. Other common operating systems tasks are shown in Table 4.1.

Systems software can be grouped into two major functional categories: system control programs and system support programs.

### System Control Programs

**System control programs** control the use of the hardware, software, and data resources of a computer system. The main system control program is the operating system. The **operating system** supervises the overall operation of the computer, including monitoring the computer's status and scheduling operations, which include the input and output processes. In addition, the operating system allocates CPU time and main memory to programs running on the computer, and it also provides an interface between the user and the hardware. This interface hides the complexity of the hardware from the user. That is, you do not have to know how the hardware actually operates, just what the hardware will do and what you need to do to obtain desired results. Specifically, the operating system provides services that include process management, virtual memory, file management, security, fault tolerance, and the user interface.

**Process management** means managing the program or programs (also called jobs) running on the processor at a given time. In the simplest case (a desktop operating system), the operating system loads a program into main memory and executes it. The program utilizes the computer's resources until it relinquishes control. Some operating systems offer more sophisticated forms of process management, such as *multitasking*, *multithreading*, and *multiprocessing*.

The management of two or more tasks, or programs, running on the computer system at the same time is called **multitasking**, or **multiprogramming**. The first program is executed until an interruption occurs, such as a request for input. While the input request is handled, the execution of a second program begins. Because switching among these programs occurs so rapidly, they appear to be executing at the same time. However, because there is only one processor, only one program is actually in execution mode at any one time. **Multithreading** is a form of multitasking that focuses on running multiple tasks within a single application simultaneously. For example, a word processor application may edit one document while another document is being spell-checked.

**Table 4.1 Common Operating Systems Tasks**

- |   |  |
|---|--|
| <ul style="list-style-type: none"> <li>• Monitoring performance</li> <li>• Correcting errors</li> <li>• Providing and maintaining the user interface</li> <li>• Starting ("booting") the computer</li> <li>• Reading programs into memory</li> <li>• Managing memory allocation to those programs</li> <li>• Placing files and programs in secondary storage</li> <li>• Creating and maintaining directories</li> </ul> | <ul style="list-style-type: none"> <li>• Formatting diskettes</li> <li>• Controlling the computer monitor</li> <li>• Sending jobs to the printer</li> <li>• Maintaining security and limiting access</li> <li>• Locating files</li> <li>• Detecting viruses</li> <li>• Compressing data</li> </ul> |
|---|--|

**Time-sharing** is an extension of multiprogramming. In this mode, a number of users operate online with the same CPU, but each uses a different input/output terminal. The programs of these users are placed into partitions in primary storage. Execution of these programs rotates among all users, occurring so rapidly that it appears to each user as though he or she were the only one using the computer.

**Multiprocessing** occurs when a computer system with two or more processors can run more than one program, or thread, at a given time by assigning them to different processors. Multiprocessing uses simultaneous processing with multiple CPUs, whereas multiprogramming involves concurrent processing with one CPU.

**Virtual memory** simulates more main memory than actually exists in the computer system. It allows a program to behave as if it had access to the full storage capacity of a computer, rather than just access to the amount of primary storage installed on the computer. Virtual memory divides an application program or module into fixed-length portions called *pages*. The system executes some pages of instructions while pulling others from secondary storage. In effect, primary storage is extended into a secondary storage device, allowing users to write programs as if primary storage were larger than it actually is. This enlarged capability boosts the speed of the computer and allows it to efficiently run programs with very large numbers of instructions.

The operating system is responsible for *file management* and *security*, managing the arrangement of, and access to, files held in secondary storage. The operating system creates and manages a directory structure that allows files to be created and retrieved by name, and it also may control access to those files based on permissions and access controls. The operating system provides other forms of security as well. For example, it must typically provide protected memory and maintain access control on files in the file system. The operating system also must keep track of users and their authority level, as well as audit changes to security permissions.

**Fault tolerance** is the ability of a system to produce correct results and to continue to operate even in the presence of faults or errors. Fault tolerance can involve error-correcting memory, redundant computer components, and related software that protect the system from hardware, operating system, or user errors.

Although operating systems perform some of their functions automatically, for certain tasks the user interacts directly with the computer through the systems software. The ease or difficulty of such interaction is to a large extent determined by the *interface design*. Older text-based interfaces like DOS (*d*isk *o*perating *s*ystem) required typing in cryptic commands. In an effort to make computers more user-friendly, the graphical user interface was developed.

The **graphical user interface (GUI)** allows users to have direct control of visible objects (such as icons) and actions that replace complex command syntax. The GUI was developed by researchers at Xerox PARC (Palo Alto Research Center), and then popularized by the Apple MacIntosh computer. Microsoft soon introduced its GUI-based Windows operating system for IBM-style PCs. The next generation of GUI technology will incorporate features such as virtual reality, head-mounted displays, sound and speech, pen and gesture recognition, animation, multimedia, artificial intelligence, and cellular/wireless communication capabilities.

The next step in the evolution of GUIs is social interfaces. A **social interface** is a user interface that guides the user through computer applications by using cartoonlike characters, graphics, animation, and voice commands. The cartoonlike characters can be cast as puppets, narrators, guides, inhabitants, avatars (computer-generated humanlike figures), or hosts.

**Types of operating systems.** As previously discussed, operating systems are necessary in order for computer hardware to function. **Operating environments**, which add

features that enable system developers to create applications without directly accessing the operating system, function only with an operating system. That is, operating environments are not operating systems, but work only with an operating system. For example, the early versions of Windows were operating environments that provided a graphical user interface and worked only with MS-DOS.

Operating systems (OSs) can be categorized by the number of users they support as well as by their level of sophistication (see the Operating Systems list on the Web site). *Operating systems for mobile devices* are designed to support a single person using a mobile, handheld device, or information appliance. *Desktop operating systems* are designed to support a single user or a small workgroup of users. *Departmental server operating systems* typically support from a few dozen to a few hundred users. *Enterprise server operating systems* generally support thousands of simultaneous users and millions or billions of simultaneous transactions. *Supercomputer operating systems* support the particular processing needs of supercomputers.

Supercomputer and enterprise server operating systems offer the greatest functionality, followed by departmental server operating systems, desktop operating systems, and finally EEM operating systems. An important exception is the user interface, which is most sophisticated on desktop operating systems and least sophisticated on supercomputer and enterprise server operating systems.

**Mobile device operating systems.** These operating systems are designed for a variety of devices, such as handheld computers, set-top boxes, subnotebook PCs, mobile telephones, and factory-floor equipment. The mobile device operating system market includes embedded Linux, Microsoft's Windows CE and Pocket PC, Windows Embedded NT 4.0, and Palm OS from Palm. Here is some information about mobile device operating systems:

- **Embedded Linux** is a compact form of Linux used in mobile devices. Both IBM and Motorola are developing Embedded Linux for mobile devices.
- **Windows CE**, a 32-bit operating system, is Microsoft's information appliance operating system. Windows CE includes scaled-down versions (known as *pocket versions*) of Microsoft Word, Excel, PowerPoint, and Internet Explorer.
- **Pocket PC** is a version of Windows CE 3.0 specifically designed for personal digital assistants and handheld computers.
- **Windows Embedded NT 4.0**, a 32-bit operating system, is aimed at embedded devices that require more operating system capabilities and flexibility than Windows CE can offer.
- The **Palm operating system** was developed by Palm for its PalmPilot handheld, pen-input PDAs. Palm OS includes a graphical user interface, and users must learn a stylized alphabet, called Graffiti, to make the device receive handwritten input.

(For technical discussions of Mobile Device Operating Systems, see this section on the book's Web site.)

**Desktop and notebook computer operating systems.** The Windows family is the leading series of desktop operating systems. The **MS-DOS (Microsoft Disk Operating System)** was one of the original operating systems for the IBM PC and its clones. This 16-bit operating system, with its text-based interface, has now been almost totally replaced by GUI operating systems such as Windows 2000 and Windows XP. **Windows 1.0** through **Windows 3.1** (successive versions) were not operating systems, but were operating environments that provided the GUI that operated with, and extended the capabilities of, MS-DOS.

**Windows 95**, released in 1995, was the first of a series of products in the **Windows operating system** that provided a streamlined GUI by using icons to provide instant access to common tasks. Windows 95 is a 32-bit operating system that features multitasking, multithreading, networking, and Internet integration capabilities, including the ability to integrate fax, e-mail, and scheduling programs. Windows 95 also offers plug-and-play capabilities. **Plug-and-play** is a feature that can automate the installation of new hardware by enabling the operating system to recognize new hardware and install the necessary software (called device drivers) automatically.

Subsequent products in the Microsoft Windows operating system are:

- **Windows 98** was not a major upgrade to Windows 95, but did offer minor refinements, bug fixes, and enhancements to Windows 95.
- **Windows Millennium Edition (Windows ME)** is a major update to Windows 95, offering improvements for home computing in the areas of PC reliability, digital media, home networking, and the online experience.
- **Windows NT** is an operating system for high-end desktops, workstations, and servers. It provides the same GUI as Windows 95 and 98, and has more powerful multitasking, multiprocessing, and memory-management capabilities. Windows NT supports software written for DOS and Windows, and it provides extensive computing power for new applications with large memory and file requirements. It is also designed for easy and reliable connection with networks and other computing machinery, and is proving popular in networked systems in business organizations.
- **Windows 2000** is a renamed version of Windows NT 5.0. This operating system has added security features, will run on multiple-processor computers, and offers added Internet and intranet functionality.
- **Windows XP** is the first upgrade to Windows 2000 and has three versions: a 32-bit consumer version, a 32-bit business version, and a 64-bit business version. Windows XP is the first version of Windows to support Microsoft's .NET platform (discussed later in the chapter).
- Following Windows XP, Microsoft will release its first fully .NET-enabled Windows operating system, code-named **Blackcomb**. Blackcomb will feature natural interfaces, including speech recognition and handwriting support.

**UNIX** provides many sophisticated desktop features, including multiprocessing and multitasking. UNIX is valuable to business organizations because it can be used on many different sizes of computers (or different platforms), can support many different hardware devices (e.g., printers, plotters, etc.), and has numerous applications written to run on it. UNIX has many different versions. Most UNIX vendors are focusing their development efforts on servers rather than on desktops, and are promoting Linux for use on the desktop.

**Linux** is a powerful version of the UNIX operating system that is completely free of charge. It offers multitasking, virtual memory management, and TCP/IP networking. Linux was originally written by Linus Torvalds at the University of Helsinki in Finland in 1991. He then released the source code to the world (called open source software, as discussed in the chapter opening case). Since that time, many programmers around the world have worked on Linux and written software for it. The result is that, like UNIX, Linux now runs on multiple hardware platforms, can support many different hardware devices, and has numerous applications written to run on it. Linux is becoming widely used by Internet service



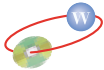
*"I haven't the slightest idea who he is. He came bundled with the software."*

providers (ISPs), the companies that provide Internet connections. The clearinghouse for Linux information on the Internet may be found at [linuxhq.com](http://linuxhq.com).

The **Macintosh operating system X (ten) (Mac OS X)**, for Apple Macintosh microcomputers, is a 32-bit operating system that supports Internet integration, virtual memory management, and AppleTalk networking. Mac OS X features a new Aqua user interface, advanced graphics, virtual memory management, and multitasking.

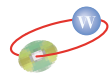
IBM's **OS/2** is a 32-bit operating system that supports multitasking, accommodates larger applications, allows applications to be run simultaneously, and supports networked multimedia and pen-computing applications.

Sun's **Java operating system (JavaOS)** executes programs written in the Java language (described later in this chapter) without the need for a traditional operating system. It is designed for Internet and intranet applications and embedded devices. JavaOS is designed for handheld products and thin-client computing. (For a more technical discussion of the various Desktop and Notebook Computer Operating Systems—MS DOS and Windows, Linux, and Apple Macintosh—see the material on the Web.)

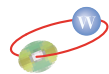


**Departmental server operating systems.** The major departmental server operating systems include UNIX, Linux, Windows 2000, Windows XP, and Novell NetWare. Although some of these are also desktop operating systems, all can serve as departmental server operating systems because of their strong scalability, reliability, backup, security, fault tolerance, multitasking, multiprocessing, TCP/IP networking (Internet integration), network management, and directory services.

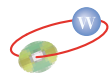
**Enterprise server operating systems.** Enterprise server operating systems (e.g., IBM's OS/390, VM, VSE, and OS/400) generally run on mainframes and midrange systems. Enterprise operating systems offer superior manageability, security, stability, and support for online applications, secure electronic commerce, multiple concurrent users, large (terabyte) databases, and millions of transactions per day. Enterprise server operating systems also offer *partitioning*, a method of segmenting a server's resources to allow the processing of multiple applications on a single system. (For a technical discussion of Partitioning, see the material at the book's Web site.)



OS/400 is IBM's operating system for the AS/400 server line, which was renamed *eServer iSeries 400*. IBM's z/Architecture (z/OS), a new 64-bit mainframe operating system, replaces all previous mainframe operating systems. The first system implementing the new architecture is the *eServer zSeries 900*. (For a technical discussion of IBM's Enterprise Server Operating Systems, see the material on the Web.)



**Supercomputer operating systems.** Supercomputer operating systems target the supercomputer hardware market. Examples of these systems include the Cray Unicos and IBM's AIX (both types of UNIX). These two operating systems manage highly parallel multiprocessor and multiuser environments. (For a technical discussion of Supercomputer Operating Systems, see the material on the book's Web site.)



## System Support Programs

The second major category of systems software, **system support programs**, supports the operations, management, and users of a computer system by providing a variety of support services. Examples of system support programs are system utility programs, performance monitors, and security monitors.

**System utilities** are programs that have been written to accomplish common tasks such as sorting records, checking the integrity of diskettes (i.e., amount of storage

available and existence of any damage), and creating directories and subdirectories. They also restore accidentally erased files, locate files within the directory structure, manage memory usage, and redirect output.

**System performance monitors** are programs that monitor the processing of jobs on a computer system. They monitor computer system performance and produce reports containing detailed statistics relating to the use of system resources, such as processor time, memory space, input/output devices, and system and application programs. These reports are used to plan and control the efficient use of the computer system resources and to help troubleshoot the system in case of problems.

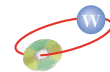
**System security monitors** are programs that monitor the use of a computer system to protect it and its resources from unauthorized use, fraud, or destruction. Such programs provide the computer security needed to allow only authorized users access to the system. Security monitors also control use of the hardware, software, and data resources of a computer system. Finally, these programs monitor use of the computer and collect statistics on attempts at improper use.

### *Before you go on . . .*

1. What are the two main types of systems software?
2. What are differences among mobile device, desktop, departmental server, enterprise, and supercomputer operating systems?

## 4.3 APPLICATION SOFTWARE

As defined earlier, application software consists of instructions that direct a computer system to perform specific information processing activities and that provide functionality for users. Because there are so many different uses for computers, there are a correspondingly large number of different application software programs available. A controversial set of software applications involves surveillance. A box on Surveillance Software at our Web site discusses the pros and cons of various types of such software.



### Types of Application Software

Application software includes proprietary application software and off-the-shelf application software. **Proprietary application software** addresses a specific or unique business need for a company. This type of software may be developed in-house by the organization's information systems personnel or it may be commissioned from a software vendor. Such specific software programs developed for a particular company by a vendor are called **contract software**.

Alternatively, **off-the-shelf application software** can be purchased, leased, or rented from a vendor that develops programs and sells them to many organizations. Off-the-shelf software may be a standard package or it may be customizable. Special-purpose programs or "packages" can be tailored for a specific purpose, such as inventory control or payroll. The word **package** is a commonly used term for a computer program (or group of programs) that has been developed by a vendor and is available for purchase in a prepackaged form. We will further discuss the methodology involved in acquiring application software, whether proprietary or off the shelf, in Chapter 14.

## Types of Personal Application Software

General-purpose, off-the-shelf application programs that support general types of processing, rather than being linked to any specific business function, are referred to as **personal application software**. This type of software consists of nine widely used packages: spreadsheet, data management, word processing, desktop publishing, graphics, multimedia, communications, speech-recognition software, and groupware. Software suites combine some of these packages and integrate their functions.

Personal application software is designed to help individual users increase their productivity. Below is a description of the nine main types.

**Spreadsheets.** Computer **spreadsheet software** transforms a computer screen into a ledger sheet, or grid, of coded rows and columns. Users can enter numeric or textual data into each grid location, called a *cell*. In addition, a formula can be entered into a cell to obtain a calculated answer displayed in that cell's location. With spreadsheets, users can also develop and use **macros**, which are sequences of commands that can be executed with just one simple instruction.

Computer spreadsheet packages can be used for financial information, such as income statements or cash flow analysis. They are also used for forecasting sales, analyzing insurance programs, summarizing income tax data, and analyzing investments. They are relevant for many other types of data that can be organized into rows and columns. Although spreadsheet packages such as Microsoft's Excel and Lotus 1-2-3 are thought of primarily as spreadsheets, they also offer data management and graphical capabilities. Therefore, they may be called **integrated packages**. Figure 4.2 shows an example from a Microsoft Excel spreadsheet.

Spreadsheets are valuable for applications that require modeling and what-if analysis. After a set of mathematical relationships has been specified by the user, the spreadsheet can be recalculated instantly using a different set of assumptions (i.e., a different set of mathematical relationships).

**Data management.** **Data management software** supports the storage, retrieval, and manipulation of related data. There are two basic types of data management software: *simple filing programs* patterned after traditional, manual data-filing techniques, and *database management programs* that take advantage of a computer's extremely fast and accurate ability to store and retrieve data in primary and secondary storage. File-based management software is typically very simple to use and is often very fast, but it offers limited flexibility in how the data can be searched. Database management software has the opposite strengths and weaknesses. Microsoft's Access is an example of popular database management software. In Chapter 5, we discuss data management in much more detail.

**Word processing.** **Word processing software** allows the user to manipulate text rather than just numbers. Modern word processors contain many productive writing

Student Name	Exam 1	Exam 2	Exam 3	Total Points	Grade
Carr, Harold	73	95	90	258	B
Ford, Nelson	92	90	81	263	B
Lewis, Bruce	86	88	98	272	A
Snyder, Charles	63	71	76	210	C
<b>Average</b>	78.5	86.0	86.25	250.75	

**Figure 4.2** This Microsoft Excel spreadsheet shows a sample calculation of student grades.

and editing features. A typical word processing software package consists of an integrated set of programs including an editor program, a formatting program, a print program, a dictionary, a thesaurus, a grammar checker, a mailing list program, and integrated graphics, charting, and drawing programs. **WYSIWYG** (an acronym for What You See Is What You Get, pronounced “wiz-e-wig”) word processors have the added advantage of displaying the text material on the screen exactly—or almost exactly—as it will look on the final printed page (based on the type of printer connected to the computer). Word processing software enables users to be much more productive because the software makes it possible to create and modify the document electronically in memory.

**Desktop publishing.** **Desktop publishing software** represents a level of sophistication beyond regular word processing. In the past, newsletters, announcements, advertising copy, and other specialized documents had to be laid out by hand and then typeset. Desktop software allows microcomputers to perform these tasks directly. Photographs, diagrams, and other images can be combined with text, including several different fonts, to produce a finished, camera-ready document.

**Graphics.** **Graphics software** allows the user to create, store, and display or print charts, graphs, maps, and drawings. Graphics software enables users to absorb more information more quickly and to spot relationships and trends in data more easily. There are three basic categories of graphics software packages: presentation graphics, analysis graphics, and computer-aided design software.

**Presentation graphics software** allows users to create graphically rich presentations. Many packages have extensive libraries of clip art—pictures that can be electronically “clipped out” and “pasted” into the finished image. Figure 4.3 demonstrates some of the capabilities of presentation graphics. One of the most widely used presentation graphics programs is Microsoft’s PowerPoint.

**Analysis graphics** applications additionally provide the ability to convert previously analyzed data—such as statistical data—into graphic formats like bar charts, line charts, pie charts, and scatter diagrams. Both presentation graphics and analysis graphics are useful in preparing graphic displays for business presentations, from sales results to marketing research data.



**Figure 4.3** Presentation graphics software.

**Computer-aided design (CAD) software**, used for designing items for manufacturing, allows designers to design and “build” production prototypes in software, test them as a computer object under given parameters (sometimes called *computer-aided engineering*, or *CAE*), compile parts and quantity lists, outline production and assembly procedures, and then transmit the final design directly to machines. The prototype in Figure 4.4 was produced via computer-aided design.



**Figure 4.4** *Computer-aided design (CAD).*

Manufacturers of all sorts are finding uses for CAD software. *Computer-aided manufacturing (CAM)* software uses digital design output, such as that from a CAD system, to directly control production machinery. *Computer-integrated manufacturing (CIM)* software is embedded within each automated production machine to produce a product. Overall, a design from CAD software is used by CAM software to control individual CIM programs in individual machines. Used effectively, CAD/CAM/CIM software can dramatically shorten development time and give firms the advantage of economies of scope.

**Multimedia.** **Multimedia software** combines at least two media for input or output of data. These media include audio (sound), voice, animation, video, text, graphics, and images. Multimedia can also be thought of as the combination of *spatial*-based media (text and images) with *time*-based media (sound and video).

**Communications.** Computers are often interconnected in order to share or relate information. To exchange information, computers utilize **communications software**. This software allows computers, whether they are located close together or far apart, to exchange data over dedicated or public cables, telephone lines, satellite relay systems, or microwave circuits.

When communications software exists in both the sending and receiving computers, they are able to establish and relinquish electronic links, code and decode data transmissions, verify transmission errors (and correct them automatically), and check for and handle transmission interruptions or conflicting transmission priorities. E-mail and desktop videoconferencing rely on communications software.

**Speech-recognition software.** Two categories of **speech-recognition software** are available today: discrete speech and continuous speech. *Discrete speech recognition* can interpret only one word at a time, so users must place distinct pauses between words. This type of voice recognition can be used to control PC software (by using words such as “execute” or “print”). But it is inadequate for dictating a memo, because users find it difficult to speak with measurable pauses between every word and still maintain trains of thought.

Software for *continuous speech recognition* can interpret a continuing stream of words. The software must understand the context of a word to determine its correct spelling, and be able to overcome accents and interpret words very quickly. These requirements mean that continuous speech-recognition software must have a computer with significantly more speed and memory than discrete speech software.

Many firms and people use speech-recognition software when use of a mouse and a keyboard is impractical. For example, such software can provide an excellent alternative for users with disabilities, repetitive strain injuries, or severe arthritis. The following example demonstrates use of speech-recognition software.

#### EXAMPLE

**Handling calls with speech recognition.** JetAir Belgium (*jetair.be*), a travel company, handles 3,000 calls a day from 2,000 travel agents. Before installing its voice-

recognition system, JetAir lost 20 percent of its calls, because operators were busy or the calls were too complicated for tone-activated voice mail. The speech-recognition system recognizes both Flemish and French among 13 supported languages. In addition to retaining the lost calls, JetAir estimates that it handles 150 extra calls daily, worth up to \$25 million in annual revenue. ●

**Groupware.** **Groupware** is a class of software products that facilitates communication, coordination, and collaboration among people. Groupware is important because it allows workgroups—people who need to interact with one another within an organization—to communicate and share information, even when they are working together at a distance. Groupware can provide many benefits to businesses, including more efficient and effective project management, location independence, increased communications capability, increased information availability, and improved workflow.

Groupware comes in many varieties. The most elaborate system, IBM's Lotus Notes/Domino, is a document-management system, a distributed client/server database, and a basis for intranet and electronic commerce systems, as well as a communication support tool. This class of groupware supplements real-time communications with asynchronous electronic connections (e.g., electronic mail and other forms of messaging). Thanks to electronic networks, e-mail, and shared discussion databases, group members can communicate, access data, and exchange or update data at any time and from any place. Group members might store all their official memos, formal reports, and informal conversations related to particular projects in a shared, online data store, such as a database. Then, as individual members need to check on the contents, they can access the shared database to find the information they need. An example of the latest type of groupware, *collaboration software*, is shown in IT's About Business Box 4.1.

## IT'S About Business

pg.com

MKT

POM

### Box 4.1: Collaboration software helps Procter & Gamble

Procter & Gamble's computer network links 900 factories and 17 product development centers in 73 countries. The global network enables the \$39 billion consumer-products company to produce and market 300 of the world's best-known brands, including Tide, Folgers, and Crest. However, it has become increasingly difficult to make such a wide variety of products with speed, efficiency, and quality.

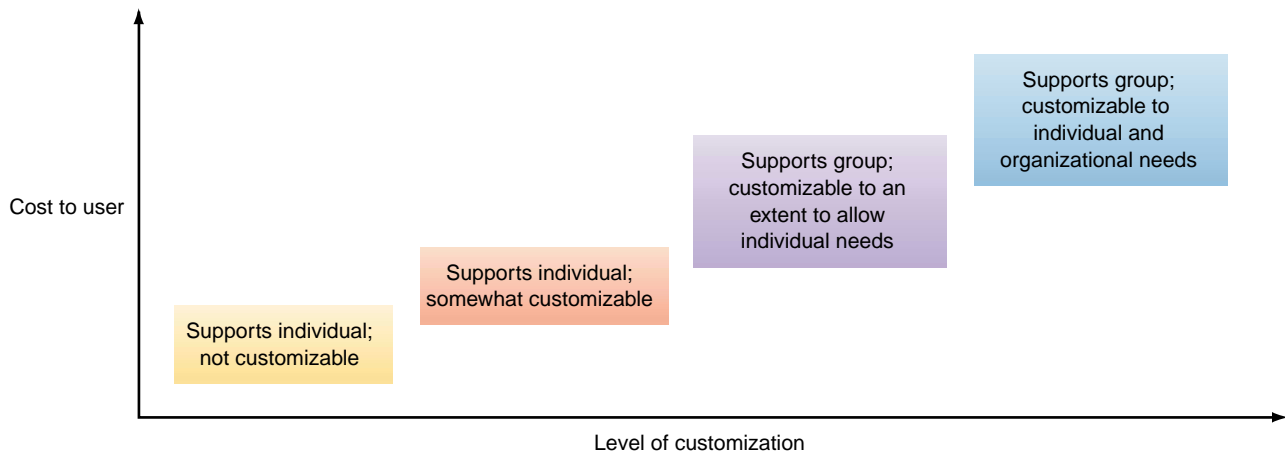
To help solve the problem, P&G uses software from MatrixOne (*matrixone.com*) to automate P&G's product-development process. The software lets researchers comb a database of 200,000 product designs to see if they already exist in another part of the company, eliminating redundant efforts. The software has also reduced product design times by half, because it lets geographically dispersed developers produce formulas together on the Web and enables managers to measure their progress against timetables. When product development falls behind schedule, the software automatically sends an e-mail reminder to a worker when he or she needs, for example, to approve the packaging of a detergent.

Companies have been using the Web to share information and streamline purchasing, and now they are using new Web software tools to help employees and business partners work together to make products faster and more cheaply. The increase in collaboration stems partly from a dissatisfaction with the limitations of electronic marketplaces. Many public e-marketplaces are essentially auction houses. They do not offer the means to form deep business relationships online. Through online collaboration, a manufacturer can, for example, share its sales forecasts with suppliers so they can fine tune inventories, resulting in potentially enormous savings.

Source: "Simultaneous Software," *BusinessWeek*, August 27, 2001, pp. 146–147.

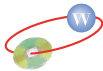
#### Questions

1. What is the relationship between corporate intranets and collaboration software? Between corporate extranets and collaboration software?
2. Can collaboration software work between companies? Give an example.



**Figure 4.5** Off-the-shelf software suites for personal productivity can be categorized according to cost and level of customization.

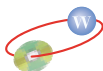
Other groupware approaches focus on workflow, enhanced electronic mail (e.g., listserv), calendaring and scheduling, electronic meeting support, and videoconferencing. Microsoft's Exchange is primarily an electronic messaging server that incorporates groupware functionality for sharing information. It provides e-mail and supports workgroup activities with additional features such as interactive scheduling, built-in access to shared bulletin boards, forms design, and access to publicly shared folders on computer networks. It also offers built-in connectivity to the Internet or corporate intranets. Other leading groupware products provide functionality similar to Microsoft Exchange. These products include Netscape's SuiteSpot Servers, Novell's GroupWise, and Oracle's InterOffice. For Common Groupware Features, see the Web site.



## Software Suites and Other Personal Application Software

**Software suites** are collections of application software packages that integrate some or all of the nine functions of the packages described in this section. Software suites can include word processors, spreadsheets, database management systems, graphics programs, communications tools, and other applications. Microsoft Office, Novell Perfect Office, and Lotus SmartSuite are widely used software suites for PCs. Each of these suites includes a spreadsheet program, word processor, database program, and graphics package with the ability to move documents, data, and diagrams among them. Figure 4.5 shows how software suites can be categorized according to cost and level of customization.

Surprisingly, there are even more types of personal application software that may be of interest to businesspeople. See Other types of Personal Application Software for Business people at the Web site.



### *Before you go on . . .*

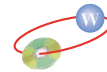
1. What classes of personal application software are essential for the productivity of a business or other organization with which you are familiar? Which are nonessential?
2. How can groupware add strategic advantage in that business/organization?

## 4.4 SOFTWARE ISSUES

The importance of software in computer systems has brought new issues to the forefront for organizational managers. These issues include software evaluation and selection, software licensing, software upgrades, open systems, and open source software.

### Software Evaluation and Selection

The software evaluation and selection decision is a difficult one that is affected by many factors. Manager's Checklist 4.1 summarizes these selection factors. The first part of the selection process involves understanding the organization's software needs and identifying the criteria that will be used in making the eventual decision. Once the software requirements are established, specific software should be evaluated. An evaluation team composed of representatives from every group that will have a role in building and using the software should be chosen for the evaluation process. The team will study the proposed alternatives and find the software that promises the best match between the organization's needs and the software capabilities. Software Evaluation Criteria are shown on the Web site.



### Software Licensing

Vendors spend a great deal of time and money developing their software products. To protect this investment, they must protect their software from being copied and distributed by individuals and other software companies. A company can copyright its software, which means that the U.S. Copyright Office grants the company the exclusive

#### Manager's Checklist 4.1



#### Software Selection Factors

<i>Factor</i>	<i>Considerations</i>
Size and location of user base	Does the proposed software support a few users in a single location? Or can it accommodate large numbers of geographically dispersed users?
Availability of system administration tools	Does the software offer tools that monitor system usage? Does it maintain a list of authorized users and provide the level of security needed?
Costs—initial and subsequent	Is the software affordable, taking into account all costs, including installation, training, and maintenance?
System capabilities	Does the software meet both current and anticipated future needs?
Existing computing environment	Is the software compatible with existing hardware, software, and communications networks?
In-house technical skills	Should the organization develop software applications in-house, purchase off the shelf, or contract software out of house?

legal right to reproduce, publish, and sell that software. The Software Publisher's Association (SPA) enforces software copyright laws in corporations through a set of guidelines. These guidelines state that when IS managers cannot find proof of purchase for software, they should get rid of the software or purchase new licenses for its use. A *license* is permission granted under the law to engage in an activity otherwise unlawful. The SPA audits companies to see that the software used is properly licensed. Fines for improper software are heavy. IS managers are now taking inventory of their software assets to ensure that they have the appropriate number of software licenses.



Although many people do so routinely, copying software is illegal. The Software Publishers Association has stated that software piracy amounts to approximately \$15 billion annually. Software developers, failing to recoup in sales the money invested to develop their products, are often forced to curtail spending on research and development. Also, smaller software companies may be driven out of business, because they cannot sustain the losses that larger companies can. The end result is that innovation is dampened and consumers suffer. Consumers also pay higher prices to offset the losses caused by software piracy.

As the number of desktop computers continues to increase and businesses continue to decentralize, it becomes more and more difficult for IS managers to manage their software assets. As a result, new firms have sprouted up to specialize in tracking software licenses for a fee. Firms such as ASAP Software, Software Spectrum, and others will track and manage a company's software licenses, to ensure that company's compliance with U.S. copyright laws.

## Software Upgrades

Another issue of interest to organizational management is software upgrades. Software vendors revise their programs and sell new versions often. The revised software may offer valuable enhancements, or, on the other hand, it may offer little in terms of additional capabilities. Also, the revised software may contain bugs. Deciding whether to purchase the newest software can be a problem for organizations and their IS managers. It is also difficult to decide whether to be one of the first companies to buy and take strategic advantage of new software before competitors do, and take the risk of falling prey to previously undiscovered bugs.

## Open Systems

The concept of **open systems** refers to a model of computing products that work together. Achieving this goal is possible through the use of the same operating system with compatible software on all the different computers that would interact with one another in an organization. A complementary approach is to produce application software that will run across all computer platforms. If hardware, operating systems, and application software are designed as open systems, the user would be able to purchase the best software for the job without worrying whether it will run on particular hardware. As an example, much Apple MacIntosh application software would not run on Wintel (Windows-Intel) PCs, and vice versa. Neither of these would run on a mainframe. Certain operating systems, like UNIX, will run on almost any machine. Therefore, to achieve an open-systems goal, organizations frequently employ UNIX on their desktop and larger machines so that software designed for UNIX will operate on any machine. Recent advances toward the open-systems goal involve using the Java language, which can be run on many types of computers, in place of a traditional operating system. Programs written in Java can then be executed by any machine (as will be explained in a later section).

## Open Source Software

Open systems should not be confused with open source software. As discussed in the chapter opening case, **open source software** is software made available in source code form at no cost to developers. There are many examples of open-source software, including the GNU (GNU's Not UNIX) suite of software (*gnu.org*) developed by the Free Software Foundation (*fsf.org*); the Linux operating system; Apache Web server (*apache.org*); sendmail SMTP (Send Mail Transport Protocol) e-mail server (*sendmail.org*); the Perl programming language (*perl.com*), the Netscape Mozilla browser (*mozilla.org*); and Sun's StarOffice applications suite (*sun.com*).

Open source software is, in many cases, more reliable than commercial software. Because the code is available to many developers, more bugs are discovered, are discovered early and quickly, and are fixed immediately. Support for open source software is also available from companies that provide products derived from the software, for example, Red Hat for Linux (*redhat.com*). These firms provide education, training, and technical support for the software for a fee.

### Before you go on . . .

1. What are some of the legal issues involved in acquiring and using software in most business organizations?
2. What are some of the criteria used for evaluating software when planning a purchase?
3. What is open source software and what are its advantages?

## 4.5 PROGRAMMING LANGUAGES

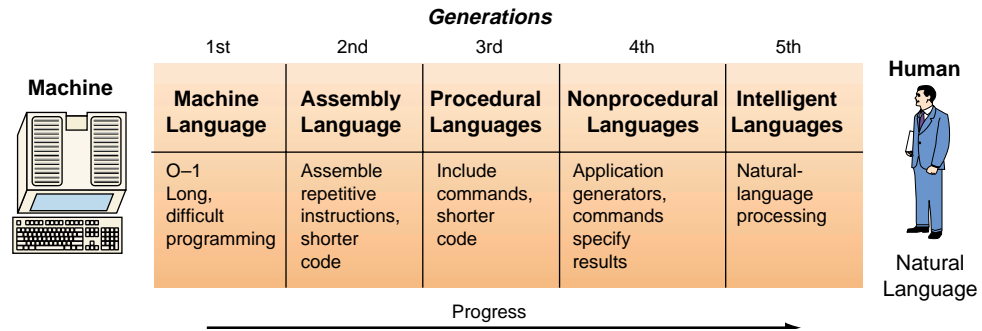
Programming languages provide the basic building blocks for all systems and application software. Programming languages allow people to tell computers what to do and are the means by which software systems are developed. This section will describe the five generations of programming languages.

### Machine Language

**Machine language** is the lowest-level computer language, consisting of the internal representation of instructions and data. This machine code—the actual instructions understood and directly executable by the central processing unit—is composed of binary digits. Machine language is the only programming language that the machine actually understands. Therefore, machine language is considered the **first-generation language**. All other languages must be translated into machine language before the computer can run the instructions. Because a computer's central processing unit is capable of executing only machine language programs, such programs are machine dependent (nonportable). That is, the machine language for one type of central processor may not run on other types.

Machine language is extremely difficult to understand and use by programmers. As a result, increasingly more user-friendly languages have been developed. Figure 4.6 (on page 110) gives an overview of the evolution of programming languages, from the first-generation machine language to more humanlike natural language. These user-oriented

**Figure 4.6** The evolution of programming languages. With each generation, progress is made toward a humanlike natural language.



languages make it much easier for people to program, but they are impossible for the computer to execute without first translating the program into machine language. The set of instructions written in a user-oriented language is called a **source program**. The set of instructions produced after translation into machine language is called the **object program**.

Programming in a higher-level language (i.e., a user-oriented language) is easier and less time consuming, but additional processor time is required to translate the program before it can be executed. Therefore, one trade-off in the use of higher-level languages is a decrease in programmer time and effort for an increase in processor time needed for translation.

## Assembly Language

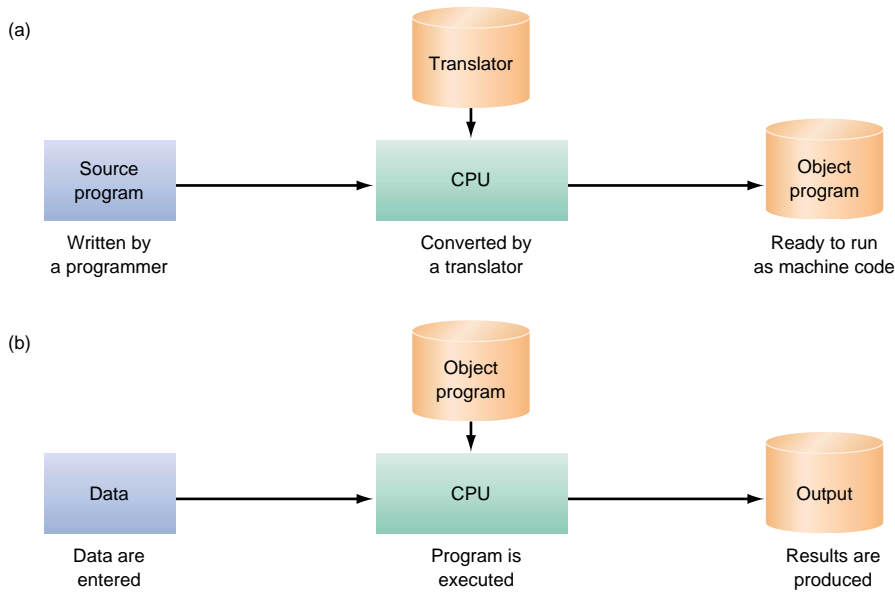
An **assembly language** is the next level up from machine language. It is still considered a lower-level language but is more user-friendly because it represents machine-language instructions and data locations in primary storage by using *mnemonics*, or memory aids, which people can more easily use. Assembly languages are considered **second-generation languages**.

Compared to machine language, assembly language eases the job of the programmer considerably. However, each statement in an assembly language must still be translated into a single statement in machine language, and assembly languages are still hardware dependent. Translating an assembly language program into machine language is accomplished by a systems software program called an **assembler**.

## Procedural Languages

**Procedural languages** are the next step in the evolution of user-oriented programming languages. They are also called **third-generation languages**, or 3GLs. Procedural languages are much closer to so-called *natural language* (the way we talk) and therefore are easier to write, read, and alter. Moreover, one statement in a procedural language is translated into a number of machine language instructions, thereby making programming more productive. In general, procedural languages are more like natural language than assembly languages are, and they use common words rather than abbreviated mnemonics. Because of this, procedural languages are considered the first level of *higher-level languages*.

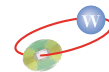
Procedural languages require the programmer to specify, step by step, exactly how the computer must accomplish a task. A procedural language is oriented toward how a result is to be produced. Because computers understand only machine language (i.e., 0s and 1s), higher-level languages must be translated into machine language prior to execution. This translation is accomplished by systems software called language



**Figure 4.7** The language translation process.

translators. A **language translator** converts the high-level program, called *source code*, into machine language code, called *object code*. There are two types of language translators—interpreters and compilers. Figure 4.7 shows the translation process for source code.

The translation of a high-level language program to object code is accomplished by a software program called a **compiler**, which translates the entire program at once. In contrast, an **interpreter** is a compiler that translates and executes one source program statement at a time. Because this translation is done one statement at a time, interpreters tend to be simpler than compilers. This simplicity allows for more extensive debugging and diagnostic aids to be available on interpreters. For examples of FORTRAN, COBOL, and C, see Examples of Procedural Languages on the Web site.



## Nonprocedural Languages

Another type of high-level language, called **nonprocedural languages**, allows the user to specify the desired result without having to specify the detailed procedures needed for achieving the result. These languages are **fourth-generation languages (4GLs)**. An advantage of nonprocedural languages is that they can be used by nontechnical users to carry out specific functional tasks. These languages greatly simplify and accelerate the programming process, as well as reduce the number of coding errors. The 4GLs are common in database applications as query languages, report generators, and data-manipulation languages. They allow users and programmers to interrogate and access computer databases using statements that resemble natural language.

## Natural Programming Languages

**Natural programming languages** are the next evolutionary step. They are sometimes known as **fifth-generation languages**, or **intelligent languages**. Translator programs to translate natural languages into a structured, machine-readable form are extremely complex and require a large amount of computer resources. Therefore, most of these languages are still experimental and have yet to be widely adopted by industry.

**Table 4.2 Language Generations Table**

<i>Language Generation</i>	<i>Features</i>				
	<i>Portable (machine independent?)</i>	<i>Concise (one-to-many?)</i>	<i>Use of Mnemonics &amp; Labels</i>	<i>Procedural?</i>	<i>Structured?</i>
1 <sup>st</sup> —Machine	no	no	no	yes	yes
2 <sup>nd</sup> —Assembler	no	no	yes	yes	yes
3 <sup>rd</sup> —High level	yes	yes	yes	yes	yes
4 <sup>th</sup> —4GL	yes	yes	yes	no	yes
5 <sup>th</sup> —Natural language	yes	yes	yes	no	no

We have now encountered the five generations of programming languages that communicate instructions to the computer's central processing unit. Table 4.2 summarizes the features of these five generations. But we are not finished yet; there are a handful of newer programming languages to look at before we finish this section.

## Visual Programming Languages

Programming languages that are used within a graphical environment are often referred to as **visual programming languages**. These languages use a mouse, icons, symbols on the screen, or pull-down menus to make programming easier and more intuitive. Visual Basic and Visual C++ are examples of visual programming languages. Their ease of use makes them popular with nontechnical users, but the languages often lack the specificity and power of their nonvisual counterparts. Although programming in visual languages is popular in some organizations, the more complex and mission-critical applications are usually not written in visual languages.

## Hypertext Markup Language

**Hypertext** is an approach to data management in which data are stored in a network of nodes connected by links (called **hyperlinks**). Users access data through an interactive browsing system. The combination of nodes, links, and supporting indexes for any particular topic is a **hypertext document**. A hypertext document may contain text, images, and other types of information such as data files, audio, video, and executable computer programs.

The standard language the World Wide Web uses for creating and recognizing hypertext documents is the **Hypertext Markup Language (HTML)**. HTML gives users the option of controlling visual elements such as fonts, font size, and paragraph spacing without changing the original information. HTML is very easy to use, and some modern word processing applications will automatically convert and store a conventional document in HTML. Dynamic HTML is the next step beyond HTML. **Dynamic HTML** presents richly formatted pages and lets the user interact with the content of those pages without having to download additional content from the server. This functionality means that Web pages using Dynamic HTML provide more exciting and useful information.

Enhancements and variations of HTML make possible new layout and design features on Web pages. For example, **cascading style sheets (CSSs)** are an enhancement to HTML that act as a template defining the appearance or style (such as size, color, and font) of an element of a Web page, such as a box.

English Text	HTML	XML
MNGT 3070 Introduction to MIS 3 semester hours Professor Smith	<TITLE>Course Number</TITLE> <BODY> <UL> <LI>Introduction to MIS <LI>3 semester hours <LI>Professor Smith </UL></BODY>	<Department and course="MNGT 3070"> <COURSE TITLE>Introduction to MIS<COURSE TITLE> <HOURS UNIT="Semester">3</NUMBER OF HOURS> <INSTRUCTOR>Professor Smith<INSTRUCTOR>

**Figure 4.8** Comparison of HTML and XML.

## Extensible Markup Language (XML)

**Extensible Markup Language (XML)** is designed to improve the functionality of Web documents by providing more flexible and adaptable information identification. XML describes what the data in documents actually mean. XML documents can be moved to any format on any platform without the elements losing their meaning. That means the same information can be published to a Web browser, a PDA, or a smart phone, and each device would use the information appropriately. Figure 4.8 compares HTML and XML. Notice that HTML only describes where an item appears on a page, whereas XML describes what the item is. For example, HTML shows only that “Introduction to MIS” appears on line 1, where XML shows that “Introduction to MIS” is the Course Title. IT’s About Business Box 4.2 shows the benefits that Fidelity has gained by standardizing on XML.



fidelity.com



### Box 4.2: Fidelity uses XML to standardize corporate data

Fidelity Investments has made all its corporate data XML-compatible. The effort helps the world’s largest mutual fund company and online brokerage eliminate up to 75 percent of the hardware and software devoted to middle-tier processing and speed the delivery of new applications.

The decision to go to XML began when Fidelity developed its Powerstreet Web trading service. At the time, Fidelity determined it would need to offer its most active traders much faster response times than its existing brokerage systems allowed. The move to XML brought other benefits as well. For example, the company was able to tie customers who have 401k plans, brokerage accounts, and IRAs under a common log-in. In the past, they required separate passwords.

Today, two-thirds of the hundreds of thousands of hourly online transactions at *fidelity.com* use XML to link the Web to back-end systems. Before XML, comparable transactions took seconds longer because they had to go through a different proprietary data translation scheme for each back-end system they retrieved data from.

Fidelity’s XML strategy is most critical to bringing new applications and services to customers faster than rivals. By using XML as a common language to which all corporate data—from Web, database, transactional, and legacy systems—are translated, Fidelity is saving millions of dollars on infrastructure and development costs. Fidelity no longer has to develop translation methods for communications between the company’s many systems. XML also has made it possible for Fidelity’s different databases—including Oracle for its customer account information and IBM’s DB2 for trading records—to respond to a single XML query.

Source: “Fidelity Retrofits All Data for XML,” *InternetWeek*, August 6, 2001; *fidelity.com*.

### Questions

1. What are the different ways that having all data in XML can save corporations money?
2. Could corporations standardize on other languages and also save money? Why or why not? Which languages?

## Componentware

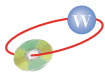
**Componentware** is a term used to describe component-based software applications. **Software components** are the “building blocks” of applications. They provide the operations that can be used by the application (or other applications) again and again. Any given application may contain hundreds of components, each providing specific business logic or user-interface functionality. Consider a database application as an example: The data-entry screen may contain several user-interface components for providing buttons, menus, list boxes, and so forth. There may also be business logic components to perform validation or calculations on the data, as well as components to write the data to the database. Finally, there can be components to create reports from the data, either for viewing in an on-screen chart or for printing. Component-based applications enable software developers to “snap together” applications by mixing and matching prefabricated plug-and-play software components.

## Virtual Reality Modeling Language

The **Virtual Reality Modeling Language (VRML)** is a file format for describing three-dimensional interactive worlds and objects. It can be used with the World Wide Web to create three-dimensional representations of complex scenes such as illustrations, product definitions, and virtual reality presentations. VRML can represent static and animated objects, and it can have hyperlinks to other media such as sound, video, and image.

## Object-Oriented Programming Languages

**Object-oriented programming (OOP) languages** are based on the idea of taking a small amount of data and the instructions about what to do with that data (these instructions are called **methods** in object-oriented programming) and putting both of them together into what is called an **object**. This process is called **encapsulation**. When the object is selected or activated, the computer has the desired data and takes the desired action. This is what happens when you select an icon on your GUI-equipped computer screen and click on it. That is, in object-oriented systems, programs tell objects to perform actions on themselves. For example, windows on your GUI screens do not need to be drawn through a series of instructions. Instead, a window object could be sent a message to open at a certain place on your screen, and the window will appear at that place. The window object contains the program code for opening and placing itself.



There are several basic concepts to object-oriented programming, which include classes, objects (discussed above), encapsulation (discussed above), and inheritance. For a more detailed discussion and examples of basic OOP concepts, see the Web site.

The **reusability feature** of object-oriented languages means that classes created for one purpose can be used in a different object-oriented program if desired. For example, if a class has methods that solve a very difficult computation problem, that problem does not have to be solved again by another programmer. Rather, the class is just used in the new program. This feature of reusability can represent a tremendous reduction in programming time within an organization.

A disadvantage of object-oriented programming, however, is that defining the initial library of classes is very time-consuming, so that writing a single program with OOP takes longer than conventional programming. Another disadvantage is that OOP languages, like visual programming languages, are somewhat less specific and powerful, and require more time and memory to execute than procedural languages. Popular object-oriented programming languages include Smalltalk, C++, and Java. Because Java is a powerful and popular language, we will look at it next in more detail.

**Java.** Java is an object-oriented programming language developed by Sun Microsystems. The language gives programmers the ability to develop applications that work across the Internet. Java can handle text, data, graphics, sound, and video, all within one program. Java is used to develop small applications, called **applets**, which can be included in an HTML page on the Internet. When the user uses a Java-compatible browser to view a page that contains a Java applet, the applet's code is transferred to the user's system and executed by the browser.

Java becomes even more interesting when one considers that many organizations are converting their internal networks to use the Internet's TCP/IP protocol (more about this in Chapter 7). This means that with a computer network that runs the Internet protocol, applications written in Java can be stored on the network, downloaded as needed, and then erased from the local computer when the processing is completed. Users simply download the Java applets as needed, and no longer need to store copies of the application on their PC's hard drive.

Java can benefit organizations in many ways. Companies will not need to purchase numerous copies of commercial software to run on individual computers. Instead, they will purchase one network copy of the software package, made of Java applets. Rather than pay for multiple copies of software, companies may be billed for usage of their single network copy, similar to photocopying. Companies also will find it easier to set information technology standards for hardware, software, and communications; with Java, all applications processing will be independent of the type of computer platform. Companies will have better control over data and applications because they can be controlled centrally from the network servers. Finally, software management (e.g., distribution and upgrades) will be much easier and faster.

**The Unified Modeling Language (UML).** Developing a model for complex software systems is as essential as having a blueprint for a large building. The **UML** is a language for specifying, visualizing, constructing, and documenting the artifacts (such as classes, objects, etc.) in object-oriented software systems. The UML makes the reuse of these artifacts easier because the language provides a common set of notations that can be used for all types of software projects.

### *Before you go on . . .*

1. What generation of languages is popular for interacting with databases?
2. What language does a CPU actually respond to?
3. What is the difference between applications and components?
4. What are the strategic advantages of using object-oriented languages?
5. What is the Unified Modeling Language?

## 4.6 ENTERPRISE SOFTWARE

To respond to competitive challenges and opportunities, companies must frequently streamline their organizational processes. This kind of reorganization frequently means changing the IT infrastructure to better support the new processes. A serious difficulty that confronts most organizations as they are in the throes of change is the sheer complexity that arises from the variety of hardware and software in use. A large

firm may have thousands of software programs to run its various systems, and dozens of types of hardware, with varying operating systems. Some applications may have been custom-made in-house, some specially made by vendors, and some generic off-the-shelf. Trying to get these elements to work in harmony in the first place is difficult enough. Trying to reconfigure them is often a nightmare. Firms and their IT management have to approach this new challenge differently.

## Streamlining Organizational Software

Unless there are significant competitive advantages, building new custom application software has become too expensive, time consuming, and risky. Instead, many organizations are buying packaged applications. IT's About Business Box 4.3 provides an example of a customer-management software package. And organizations no longer want packages that merely automate existing processes. Rather, they want packaged applications that support integration between functional modules (i.e., human resources, operations, marketing, finance, accounting, and so on), that can be quickly changed or enhanced, and that present a common graphical look and feel, helping to reduce training and operations costs. The scope of application development projects now focuses on business processes, and therefore extends across the boundaries of the enterprise, bringing partners', suppliers', and customers' needs into the integrated business solution. This new, expanded role is filled by enterprise software.

## Middleware

Internet applications designed to let one company interact with other companies are complex because of the variety of hardware and software with which they must be able to work. This complexity will increase as mobile wireless devices begin to access company sites via the Internet. **Middleware** is software designed to link application modules developed in different computer languages and running on heterogeneous platforms whether on a single machine or over a network. Middleware keeps track of the locations of the software modules that need to link to each other across a distrib-

## IT'S About Business

marriott.com

MKT

### Box 4.3: Customer relationship management at Marriott

Weeks in advance, Marriott planning coordinator Jennifer Rodas calls registered guests to ask them about their plans. When all is set, she faxes them an itinerary. What makes such velvet-glove treatment possible is Marriott's use of customer-management software from Siebel Systems. The hotel chain is counting on such technology to gain an edge with guests and event planners. The software lets Marriott pull together information about its customers from different departments, so that its representatives can anticipate and respond more quickly to their needs.

The biggest boost from the Siebel software is in the hotel chain's sales operations. Marriott is transforming its sales teams from order-takers for specific hotels to aggressive marketers of all Marriott properties. A salesper-

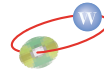
son in Dallas—who understands both the needs of his local customers and the chain's world inventory of hotel rooms and other facilities—can now book orders for hotels around the world. The software helped Marriott generate an additional \$55 million in cross-chain sales in one year.

*Source:* "How Marriott Never Forgets a Guest," *Business Week*, February 21, 2000.

### Questions

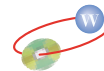
1. What are the various guest needs that the software can anticipate?
2. How would Marriott apply this software to first-time guests?

uted system and manages the actual exchange of information. (For a technical discussion of Middleware, see the material at the book's Web site.)



## Organization-Wide Applications

**Enterprise software** consists of programs that manage the vital operations of an organization (enterprise), such as supply-chain management (movement of raw materials from suppliers through shipment of finished goods to customers), inventory replenishment, ordering, logistics coordination, human resources management, manufacturing, operations, accounting, and financial management. Some common modules of enterprise applications software are payroll, sales order processing, accounts payable/receivable, and tax accounting. For Other Common Enterprise Modules, see the Web site.



Enterprise software vendors are producing software that is less expensive, based on industry standards, compatible with other vendors' products, and easier to configure and install. The largest vendors—Systeme Anwendung Produkte (SAP) AG, Oracle Corporation, PeopleSoft Inc., Baan Co., Computer Associates, and J.D. Edwards—are developing software programs that make the jobs of business users and IT personnel easier. Because of the cost, complexity, and time needed to implement enterprisewide corporate applications, many companies are purchasing only the specific application (or module) required, such as manufacturing, financial, or sales force automation.

### *Before you go on . . .*

1. What are the strategic advantages of the enterprise software approach?
2. Why is adoption of enterprise software an inherently difficult process?

## WHAT'S IN IT FOR ME?

### FOR THE ACCOUNTING MAJOR

Accounting application software performs the organization's accounting functions, which are repetitive and high volume. Accounting applications are data oriented rather than information oriented, and their main functions consist of data capture, storage, and manipulation. Each business transaction (e.g., a person hired, a paycheck produced, an item sold) produces data that must be captured. After capture, accounting applications manipulate the data as necessary. Accounting applications adhere to relatively standardized procedures, handle detailed data, and have a historical focus (i.e., what happened in the past).

ACC

### FOR THE FINANCE MAJOR

Financial application software provides information to persons and groups both inside and outside the firm about the firm's financial status. Financial applications include forecasting, funds management, and control applications.

FIN

Forecasting applications predict and project the firm's future activity in the economic environment. Funds management applications use cash flow models to analyze expected cash flows. Control applications enable managers to monitor their financial

performance, typically by providing information about the budgeting process and performance ratios. These applications allow managers to compare actual and budgeted expenses, produce reports, and compute ratios. Common ratios are the current ratio (current assets divided by current liabilities) and inventory turnover (cost of goods sold divided by the average inventory value).

**MKT****FOR THE MARKETING MAJOR**

Marketing application software helps management solve problems that involve marketing the firm's products. Marketing software includes marketing research and marketing intelligence applications. Marketing intelligence applications collect information from the firm's external environment that affects marketing operations, such as information about competitors. Marketing research applications collect information on customers, prospects, and their needs.

Marketing applications provide information about the firm's products, its distribution system, its advertising and personal selling activities, and its pricing strategies. Overall, marketing applications help managers develop strategies that combine the four major elements of marketing: product, promotion, place, and price.

**POM****FOR THE PRODUCTION/OPERATIONS MANAGEMENT MAJOR**

Managers use production/operations management applications software for production planning and as part of the physical production system. POM applications include production, inventory, quality, and cost software. These applications help management operate manufacturing facilities. Inventory applications determine the quantity of goods to reorder and when. Quality applications enable the organization to achieve product quality by monitoring the entire production process. Costing applications help managers control the costs of the production process.

Materials requirements planning (MRP) software is widely used in manufacturing. Rather than wait until it is time to reorder, MRP software identifies the materials that will be needed, their quantities, and the dates on which they will be needed, thus enabling managers to be proactive.

**HRM****FOR THE HUMAN RESOURCES MANAGEMENT MAJOR**

Human resources management application software provides information concerning recruiting and hiring, education and training, maintaining the employee database, and termination and benefits administration. HRM applications include workforce planning, recruiting, workforce management, compensation, benefits, and environmental reporting subsystems.

Workforce planning applications allow managers to identify future personnel needs by addressing organizational charting, salary forecasting, job analysis and evaluation, planning, and workforce modeling. Recruiting applications assist managers in bringing new employees into the organization by tracking applicants and by monitoring internal and external searches. Workforce management applications include performance appraisal, training, relocation, skills and competency, succession, and disciplinary actions. Compensation applications include the functions of merit increases, payroll, executive compensation, bonus incentives, and attendance. Benefits applications encompass defined contributions, defined benefits, benefit statements, flexible benefits, stock purchase, and claims processing. Environmental reporting applications include EEO (equal employment opportunity) records and analysis, union enrollment, health records, toxic substances, and grievances.

## SUMMARY

### 1 Differentiate between the two major types of software.

Software consists of computer programs (coded instructions) that control the functions of computer hardware. There are two main categories of software: systems software and application software. Systems software manages the hardware resources of the computer system and functions between the hardware and the application software. Systems software includes the system control programs (operating systems) and system support programs. Application software enables users to perform specific tasks and information-processing activities. Application software may be proprietary or off-the-shelf.

### 2 Describe the general functions of the operating system.

Operating systems manage the actual computer resources (i.e., the hardware). Operating systems schedule and process applications (jobs), manage and protect memory, ensure cache consistency, manage the input and output functions and hardware, manage data and files, and provide clustering support, security, fault tolerance, interapplication communications, graphical user interfaces, and windowing.

### 3 Differentiate among types of operating systems and describe each type.

There are five types of operating systems: mobile, desktop, departmental, enterprise, and supercomputer. Mobile device operating systems are designed to support a single person using a mobile, handheld device or information appliance. Desktop operating systems have the least functionality and enterprise operating systems the most, with departmental operating systems in the middle. Desktop operating systems are typically designed for one user, departmental operating systems for up to several hundred users, and enterprise operating systems can handle thousands of users and millions of transactions simultaneously. Supercomputer operating systems are designed for the particular processing needs of supercomputers.

### 4 Identify three methods for developing application software.

Proprietary software can be developed in-house to address the specific needs of an organization. Existing software programs can be purchased off the shelf from vendors that sell programs to many organizations and individuals. Or a combination of these two methods can be used, by purchasing off-the-shelf programs and customizing them for an organization's specific needs.

### 5 Describe the major types of application software.

The major types of application software are spreadsheet, data management, word processing, desktop publishing, graphics, multimedia, communications, speech recognition, and groupware. Software suites combine several types of application software (e.g., word processing, spreadsheet, and data management) into an integrated package.

### 6 Explain how software has evolved and trends for the future.

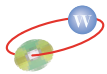
Software and programming languages continue to become more user oriented. Programming languages have evolved from the first generation of machine languages that is directly understandable to the CPU to higher levels that use more natural language and that do not require users to specify the detailed procedures for achieving desired results. This trend ensures that end users and the information systems staff will become more productive. In addition, software is becoming much more complex, expensive, and time consuming to develop. As a result, the trend is toward purchasing off-the-shelf software, often in the form of components, rather than developing it in-house. In the future, organizations will tend to buy component-based software modules to reduce costs and development time.

**7 Describe enterprise software.**

Organizations want packaged applications that support integration between functional modules (i.e., human resources, operations, marketing, finance, accounting, etc.), that can be quickly changed or enhanced, and that present a common graphical look and feel. In addition, organizations want individual components—software modules—that can be combined as necessary to meet changing business needs. Enterprise software consists of programs that manage a company’s vital operations, such as supply-chain management, inventory replenishment, ordering, logistics coordination, human resources management, manufacturing, operations, accounting, and financial management.



## INTERACTIVE LEARNING SESSION



Go to the CD, access Chapter 4: Computer Software, and read the case presented. It will describe give case scenarios in which you will be asked to choose the best software to use for various activities. You will be presented with a list of options that will allow you to make changes if you think changes are necessary.

For additional resources, go to the book’s Web site for Chapter 4. There you will find Web resources for the chapter, including additional material about operating systems, application software, and evaluation criteria; links to organizations, people, and technology; “IT’s About Business” company links; “What’s in IT for Me?” links; and a self-testing Web quiz for Chapter 4.

## DISCUSSION QUESTIONS

1. You are the CIO of your company and have to develop an application of strategic importance to your firm. Do you buy an off-the-shelf application or develop it in-house? Support your answer with pros and cons of each choice.
2. You are the CIO of your company. Which computing paradigm will you support in your strategic information technology plan: the standard desktop computing model, with all the necessary functionality on the local machine, or the network computing model, where functionality is downloaded from the network as needed? Support your answer with pros and cons of each choice.
3. You have to take a programming course, or maybe more than one, in your MIS program. Which language would you choose to study? Why? Should you even have to learn a programming language?
4. What is the relationship between network computers and Java?
5. If Java and network computing become the dominant paradigm in the industry, will there be any need for in-house information systems staff? What would the staff still have to do?

## PROBLEM-SOLVING ACTIVITIES

1. Research the costs and functionality of standalone personal software products such as word processing, spreadsheet, and graphics, and compare them to the costs and functionality of integrated software suites. Under what circumstances would you recommend either?
2. Different groupware products support different aspects of group work; some support many, and others only a few. With your job as an example (or one you have had in the past), determine what features you would want in your groupware. Then research the different products and identify the one most suitable for your needs.
3. Design a short program that you would like to have written in a computer language (for example, one that will calculate mortgage amortization or payroll with taxes for a small hourly workforce). Then discuss the desired program with an experienced computer programmer to determine what language should be used and why.
4. If you are not a programmer, or if you program in only one language, investigate learning how to program in conventional languages (C, for example), visual languages (e.g., Visual Basic), and object-oriented languages (e.g., Java). Which seems more intuitive and/or easier for you?

## INTERNET ACTIVITIES

1. A great deal of software is available free over the Internet. Go to *shareware.cnet.com* and observe all the software available for free. Choose one and download it to your computer. Prepare a brief discussion about the software for the class.
2. Enter the IBM Web site (*ibm.com*) and search on “Software.” Click on the drop box for Products and notice how many software products IBM produces. Is IBM only a hardware company? Select “Voice Recognition” and write a brief discussion of the features of IBM’s voice-recognition products.

## TEAM ACTIVITIES AND ROLE PLAYING

1. Go to your campus computing center and research the types of software that are available on the local area networks in the different parts of the university. Do different departments have their own stores of specialized software, or is all software centrally managed?
2. Discuss with your academic department’s office director and IT support person the issue of software licensing. How does the department maintain compliance? You may also do this with the personnel at the campus computing center.

## REAL-WORLD CASE

[bollingershipyards.com](http://bollingershipyards.com)

POM

### Productivity at a Shipyard

**The Business Problem** Several years ago, Bollinger Shipyards was, like most shipbuilding operations, an old-economy, nontechnological, nonnetworked company. Company headquarters received information from a collection of outdated mainframe systems and in-house-developed financial software, all running separately at the company’s nine shipyards. Producing the reports for basic payroll, finance, and procurement functions was a difficult, time-consuming process. And because the procurement system was nearly nonexistent, ships often sat in repair docks for weeks waiting for parts to arrive.

In the mid-1990s, Bollinger attempted to address its administrative difficulties by upgrading its computer systems. It tried two vendors of specialized enterprise resource planning software, but ended up scrapping both systems because they did not work properly.

**The IT Solution** The company decided to use Oracle’s e-business software suite to solve its business problems. The system, which cost \$2.7 million and took eight months to build, manages every function from human resources, accounting, and finance, to procurement.

**The Results** The Oracle software helped the company lower its overhead and increase its productivity. Each of the company’s shipyards used to require two full-time staff members to handle administration and payroll. Now, the company uses one part-time employee per shipyard.

The Oracle software also was useful when Bollinger bought a rival, 800-person shipbuilder last year. Normally, taking on an acquired company’s payroll would mean heavy overtime and many temporary employees pushing paperwork, but Bollinger’s 10-person IT department just connected up the five new shipyards, loaded the new employee data into the Oracle system, and ran payroll as usual. Bollinger purchased the shipbuilder on a Tuesday, and the next week had 800 new employees on the payroll.

Bollinger’s biggest gains have come from increased productivity. Each shipyard used to order its own supplies. Now shipyards send their requests to headquarters, which coordinates orders so parts and other materials arrive in a more timely manner. The company has also saved 15 percent off the time it takes to build a ship. For an 87-foot Coast Guard patrol boat that is now under construction, that savings amounts to \$500,000.

But the biggest payoff for Bollinger is in procurement. By centralizing procurement, the company can now see where it is spending its money, and use that information to negotiate prices with vendors. The company expects to save as much as \$5 million through procurement, twice the amount the company spent for its new information technology system.

Source: “Bollinger Shipyards,” *ecompany.com*, pp. 119–120, May 2001.

**Questions**

1. The new information technology system cost Bollinger almost 1 percent of annual sales. Was the decision to purchase the new system risky? Why or why not?
2. Would Bollinger employees have resisted the new system, worrying about jobs being lost?
3. Now that Bollinger has a successful ERP system in place internally, what should the company do next with regard to IT in order to further reduce costs and increase efficiencies? (*Hint: Consider the company's supply chain.*)

**VIRTUAL COMPANY ASSIGNMENT****Extreme Descent Snowboards**

**Background** Mark Brandy, one of the founders of EDS invites you to lunch. When you arrive at the restaurant, you find Lori Saunders from the Marketing Department and Jacob March already seated. “Welcome,” says Mark and invites you to have the seat next to him.

Mark introduces you to Lori Saunders. Jacob continues his conversation stating, “One of the most critical requirements for our electronic site is an excellent user interface.” Lori adds, “If the company does not continually update and improve its Web presence, we risk losing our customers to our competitors. web sites should be constructed to grab and keep the interest of a customer.”

Jacob interjects his thoughts that to the end-user the software interface is the system. Jacob winks at you and proceeds telling everyone that he thinks this would be a good assignment for our intern to help us define what makes a good user interface.

Jacob turn to you , handing you a napkin to make your notes on.

**Assignment**

1. Using the World Wide Web, find two companies that have an excellent Web presence for selling a service or product. Provide the complete Web site address (URL) for each web site. Write a report that includes an evaluation of:
  - Ease of navigation from page to page
  - Ease of ordering the company's product or service
  - Aesthetics of the site
  - Functionality of the Web site
2. Visit the EDS Web site and compare it to the two Web sites you have just evaluated. What do you like about the EDS Web site? What features from the other Web sites could be incorporated to improve the EDS Web site? How will your suggestions increase sales from the Web site? Write a report to Jacob March summarizing you findings.