Chapter 2
Computer Systems

Objectives and Overview

Chapter 2 is the first of a trio of chapters devoted to the primary building blocks of information technology. Chapter 2 concentrates on computer systems, including both the **hardware**—the physical pieces of a computer system—and the **software**—the set of programs that controls the operations of the computer system. Chapter 3 covers telecommunications and networking, and Chapter 4 explores the data resource that is manipulated by the hardware, software, and telecommunications. These three chapters constitute the "hard-core" technology portion of this textbook. We believe that every information systems or information technology course using this textbook should incorporate these three chapters, covered in sequential order, *unless* the students already have strong technology backgrounds. The amount of time spent on these chapters will vary considerably, of course, depending upon the level and purpose of the course. Alternative approaches to using Chapter 2 are discussed in the "Teaching Suggestions" section below.

Chapter 2 in the Seventh Edition is a brand new chapter, incorporating (in greatly reduced form) the material that was covered in Chapter 2, Computer Hardware, and Chapter 3, Computer Software, in the Sixth Edition of this book. The primary objective of Chapter 2 is to provide the student with a basic understanding of computer systems—both the hardware and the software. The goal of the chapter, simply stated, is to let the reader know what he or she needs to know about computer systems *and no more*. Managers need to understand the major ideas involved in computer systems, and they need to know the important terminology and concepts. Chapter 2 is aimed at satisfying this "need to know" in a straightforward, understandable way.

It may be useful to view this chapter as consisting of five major sections—two of these are primarily devoted to computer hardware, two more are primarily devoted to computer software, and the fifth section is devoted to the information technology (IT) industry. The first major section provides an extended discussion of the *underlying structure of computer systems*, including the stored-program concept. This section will be largely review for those students who have significant practical computing experience (more than just word processing and spreadsheets) or those who have had previous courses in computer programming or information technology. The second major section describes the *categories of computers* in use today, including information about the hardware component of the information systems industry. This material will be new for most students and should provide a useful current perspective for everyone.

After a short introduction to the two key types of software—applications software and support software, the third major section on *applications software* includes an example of an application package that might be purchased as well as an extensive treatment of personal productivity software (word processing, spreadsheets, database management systems, Web browsers, application suites, and so on). This section will be largely review for students with practical computing experience or previous courses in programming or IT. The fourth major section, which deals with *support software*, covers the all-important operating system, the wide variety of programming languages, database management systems, CASE tools, and communications interface software. We suspect this section will be mostly new for students unless they have an extremely strong computing background. After a short section on the

changing nature of software, the fifth major section discusses the information technology industry. This section should provide a good overview of the IT industry for all students, regardless of their background. As mentioned above, all of these sections have been significantly revised and reduced in the Seventh Edition.

What are the highlights of this totally new chapter? In terms of hardware, the "categories of computers" section has been extensively reworked to include new boundaries between the categories as well as updated lists of the major vendors in each category. The boxes on blade servers and Smartphone's present new hardware developments. In terms of software, major story lines include the continued growth of open source software, especially the Linux operating system; the growth in the use of XML, eXtensible Markup Language; and the continuing debate over the relative merits of the J2EE and .NET frameworks for developing applications on the Web. The short IT industry section at the end of the chapter provides a fairly concise overview of the industry while identifying the major players in each segment of the industry.

The specific objectives of this chapter are:

1. To outline the underlying structure of all digital computers, including input, output, memory, arithmetic/logical unit, control unit, and files.
2. To describe the all-important stored-program concept, which is the basis for the way in which all computers operate today.
3. To describe the different categories of computers, including microcomputers, midrange systems, mainframes, and supercomputers, and to identify the major hardware manufacturers in each category.
4. To distinguish between applications software and support software.
5. To introduce the idea of purchased applications packages.
6. To consider the variety of personal productivity software packages—such as word processing, spreadsheets, and presentation graphics—that are likely to be used by managers on a regular basis.
7. To consider the wide variety of functions accomplished by the operating system and to describe the other types of support software, including language translators, database management systems, communications interface software, CASE tools, and utility programs.
8. To consider the makeup and changing composition of the different components of the information technology (IT) industry.

The difficulty in teaching Chapter 2 lies in correctly assessing the computer systems background of the majority of the class and adjusting classroom time and material accordingly. Most of today's students understand microcomputers and the role of personal productivity software packages, and they know a little about the Windows operating system. However, unless they have an excellent computing background (either through work or coursework), they are unlikely to know much about types of computers other than microcomputers, about support software, or about the IT industry beyond Microsoft. We have found, to our dismay, that many students who have had a computer tools and/or computer programming course do not really comprehend the important difference between applications software and support software, and thus we suggest stressing this key idea.

Our suggestion is to tell the students, in advance, that some of the material may be review for them, and that they may skim subsections with which they are familiar. Furthermore, tell the students that they should not be concerned with the mechanical details of the sample programs given in the text—these samples are provided merely to give students the flavor of the various languages. At the same time point out that the totality of this chapter should provide them with an extremely valuable overall perspective on computer systems. With such an advance warning, we think most students will react favorably to this chapter. Students without a strong background should appreciate the chapter's clear organization and

extensive descriptions, while those with stronger backgrounds should find that the chapter provides a useful review and synthesis.

Teaching Suggestions

The primary difficulty in teaching this chapter lies in correctly assessing the level of computing knowledge of a given class and adjusting the classroom sessions accordingly. It is important to neither undershoot nor overshoot the majority of the audience. On the other hand, don't attempt to cater to the few students who have an excellent computing background, or to the few who know nothing at all about computers.

For most courses using this textbook, we believe that you should allot from two to four 75-minute classroom sessions to Chapter 2 and related enrichment activities. For courses where most students have a *good* computing background, two sessions should suffice. We would suggest using about half of one class period giving a "mini-lecture" on the hardware portion of Chapter 2, making sure that students understand the underlying structure of all computer systems, the stored-program concept, and the roles of the different types of computers in use today. On this latter topic (the roles of different types of computers), enliven the discussion by bringing in very recent statistics or news clippings describing recent events in the rapidly changing hardware industry (Which vendor is leading in server sales? Which smartphone is in the lead in terms of business use? What is currently the world's fastest supercomputer?). In the latter half of the class period, employ an enrichment activity related to computer hardware or computer use in general. For example, if you haven't already used Case Study 1, Midsouth Chamber of Commerce (A): The Role of the Operating Manager in Information Systems, this would be a good time to use it. Alternatively, Case Study I-1, IMT Custom Machine Company, Inc.: Selection of an Information Technology Platform, deals with the hardware platform selection for a manufacturing company. Teaching notes for both of the case studies are included in this *Instructor's Manual*. As another enrichment activity, you may be able to locate one or more video presentations on the Web sites of hardware vendors that describe the features of the vendors' newest large computer systems, and then you can play one or more of these video presentations during the class period. Today's students tend to be quite familiar with microcomputers, but may never have seen any larger machines. Don't overdo the use of such advertising vehicles, but ten to fifteen minutes may provide an interesting and educational interlude in a long classroom session.

For the second classroom session for a course in which most students have a good computing background, we suggest concentrating on computer software, the IT industry, and a related enrichment activity. Again, we suggest using about half of the class period giving a "mini-lecture" on the software portion of the chapter and the IT industry section, making sure that students understand the difference between applications software and support software; the roles of the operating system and other support software; and the notions of fourth generation languages (4 GLs), eXtensible Markup Language, object-oriented programming, and Web programming. Don't attempt to cover all the details of these programming sections; concentrate on the big picture. In the IT industry discussion, start with the analysis provided in the chapter, but update it with developments in the months since the chapter was written. In the latter half of the class period, employ an enrichment activity such as the new Case Study I-6, HH Gregg: Deciding on a New Information Technology Platform (a teaching note for this case study is included in this *Instructor's Manual*). A second type of enrichment activity is to demonstrate a software package that most of your students will not know, such as the Adobe Systems PageMaker desktop publishing package. Plan the demonstration carefully and avoid covering excessive detail. In summary, we suggest allotting two 75-minute classroom sessions to Chapter 2 in those courses where most students have a good computing background.

In a course in which most students *do not* have a good computing background—for example, an advanced undergraduate course where the students have had only a single computer tools course (e.g., Microsoft PowerPoint, Microsoft Excel, Microsoft Access)—we believe that you might need four 75-minute sessions to cover Chapter 2—two of these sessions on computer hardware and two on computer software and the IT industry. For the first session, we suggest a traditional lecture-discussion approach to cover the computer hardware portion of the chapter, making sure that students understand the underlying structure of all computer systems, the stored-program concept, and the roles of the different types of computers in use today. Use PowerPoint slides of your own, slides taken from the book's Web site, or slides made from the figures or tables in the text. As suggested above, enliven the latter part of this discussion by bringing in very recent statistics or news clippings describing recent events in the rapidly changing hardware industry (Which vendor is leading in server sales? Which smartphone is in the lead in terms of business use? What is currently the world's fastest supercomputer?). Don't get too deeply into the details; remember that our goal is to cover what managers need to know.

For the second session, devote the entire class session to one or more enrichment activities. Three options were mentioned above: Case Study 1, Midsouth Chamber of Commerce (A): The Role of the Operating Manager in Information Systems; Case Study I-1, IMT Custom Machine Company, Inc.: Selection of an Information Technology Platform; or video presentations about large computers from a computer vendor's Web site. Another idea is to devote this second session to a different type of enrichment activity—bring in an old desktop PC and take it apart in front of the students. Take off the cover, and take out (one at a time) the microprocessor chip, the hard drive, the floppy drive, a memory chip, the modem, the network interface card, and any other easily removable parts (DO NOT do anything with the power supply), and pass them around to the class. Of course, get an old PC well ahead of time and practice taking it apart, including opening up the hard drive so that the students can see the disk.

For the third session, we suggest a traditional lecture-discussion approach to cover the computer software and IT industry portions of the chapter. Make sure that students understand the difference between applications software and support software; that they understand the roles of the operating system and other support software; and that they understand the notions of fourth generation languages (4 GLs), eXtensible Markup Language, object-oriented programming, and Web programming. Don't attempt to cover all the details of these programming sections; concentrate on the big picture. In the IT industry discussion, start with the analysis provided in the chapter, but update it with developments in the months since the chapter was written. Again, use PowerPoint slides of your own, slides taken from the book's Web site, or slides made from the figures or tables in the text. You will certainly have more time to cover the material here than in the "mini-lecture" suggested above for the two-class-session approach to this chapter, but avoid the temptation to build in too many details; stick to the basic concepts.

For the fourth session, we again suggest devoting the entire class session to one or more enrichment activities. Two options mentioned above were Case Study I-6, HH Gregg: Deciding on a New Information Technology Platform, and the demonstration of a software package that most of your students will not know, such as the Adobe Systems PageMaker desktop publishing package. A third enrichment activity is to locate a software company's Web site that contains one or more video presentations about their software products, and then play one or more of these video presentations during the class period. A somewhat similar fourth enrichment activity is to have a representative of a software vendor (which certainly might include major hardware vendors like IBM and Hewlett-Packard) speak to the class and describe some of the vendor's recent software offerings.

Of course, these two options—two-session coverage of this chapter and four-session coverage of this chapter—are only two of the many options. If your class, on average, has an *excellent* computer background, you may choose not to cover the chapter at all in class, but rather suggest that your students

read the chapter as background reading. Even in this case, we would urge you to employ one of the above-mentioned case studies in a class session to get your students thinking about IT platforms and the role of a manager in dealing with IT. For some courses, a three-session approach to covering this chapter may be appropriate. Most likely, this would involve a lecture-discussion approach to the computer hardware material in one class session; a lecture-discussion approach to the computer software and IT industry material in the second class session; and some combination of one or more enrichment activities in the third class session.

Let us repeat a final bit of advice on teaching this chapter and the other technology chapters: Don't attempt to cover all the details that are embedded in these chapters. Let the chapter speak for itself, and take advantage of the fact that students will know some of this material before taking your course. You should concentrate on making sure that students understand the important concepts and terminology involved with computer systems, and that they appreciate the roles played by the different types of hardware and software.

Review Questions

1. Distinguish between microcomputers, midrange systems, mainframes, and supercomputers. Give approximate speeds (millions of floating point operations per second, or MFLOPS) and costs.

Microcomputers, or PCs, may be desktops, notebooks, or palmtops. They are used for personal computing, Web clients, clients in client/server applications, and small business processing. MFLOPS range from 500 to 5,000. They cost from under $200 to $4,000.

Midrange systems are used for departmental computing and for specific applications such as office automation and computer-aided design. They are servers in client/server applications and are used for midsized business processing and by universities; they also function as Web servers, file servers, and local area network servers. MFLOPS range from 2,500 to 250,000. They cost from $4,000 to $1,000,000.

Mainframes can handle thousands of terminals and can operate as very large servers, including as Web servers. They are used for large business general processing and can handle the widest range of applications. The MFLOPS are from 2,500 to 1,000,000. They cost from $500,000 to $20,000,000.

Supercomputers handle numerically intensive scientific calculations; they can also operate as very large Web servers. The MFLOPS are from 250,000 to over 3,000,000,000. They cost $1 million to $100 million.

2. List the six building blocks that make up digital computers, and describe the flows of data that occur among these blocks.

The six building blocks are input, output, memory, arithmetic/logical unit, control unit, and files. Data flows from input to memory; from memory to the control unit; from memory to and from files and the arithmetic/logical unit; and from memory to output. Data flows are always to or from memory.

3. What are the advantages and disadvantages of using direct access files versus using sequential access files? Why do organizations bother to use sequential access files today?

Sequential access files are organized in order according to the control key of each file. Because there are no addresses in a sequential access file, the only way the computer can find a specific record is to start at

the beginning of the file and read each record until it reaches the desired one. If retrieval time is an important consideration, sequential access files may not be practical. However, huge quantities of data can be stored very economically using sequential access files. Therefore, if retrieval time is *not* an important consideration—such as for batch processing and little-used archives—then sequential access files are an attractive choice.

When using direct access files, the computer can retrieve a specific record immediately by going directly to the file address to find the record. The advantage of using direct access files is the speed of access and retrieval. This is especially important for online systems. For applications that are batched and where response time is not critical, sequential access files may be more economical. The main challenge in using direct access files is translating the identification number for a record into an address. This translation requires highly sophisticated software.

4. Explain in your own words the importance of the stored-program concept. Include the role of the control unit in your explanation.

A stored program consists of a set of operating instructions that provides a precise listing of exactly what the computer is to do, prepared in a form that the control unit of the computer has been built to understand. Most importantly, the control unit carries out these instructions at *electronic speed* rather than waiting for humans to tell it what to do next. Because each computer model differs, the "language" in which programs are written will differ from one computer model to the next.

After the entire machine language program has been prepared, it must be stored in the computer's memory. The control unit then is told (somehow) where to find the first instruction in the program, and this instruction is brought into the control unit. After the first instruction has been interpreted and carried out by the control unit, the control unit fetches the second instruction from memory. The control unit then interprets this second instruction and executes it. The control unit then fetches and executes the third instruction. The control unit proceeds with this fetch—execute cycle until the program has been completed. The important point is that the control unit is fetching and executing at electronic speed; it is doing exactly what the programmer told it to do, but at its own rate of speed.

5. What is a blade server and why have blade servers become important in the last few years?

A blade server (also referred to as a blade) is a very narrow, compact server that can be placed into a blade server chassis along with several other blade servers. For example, one variation of a blade server is 1.2 inches wide and about 15 inches tall; 14 of these blade servers can be mounted in a single chassis—they slide in much like sliding a book into a bookshelf. The chassis supplies the power supply for the blades, the management system, and the network switch; each blade server has its own processor, memory, and hard drive.

Blade servers are important because they save space in the computing center, they reduce the required cabling, and they improve system management. Blade servers save space, time, and money.

6. Four categories of computer systems were considered in this chapter: microcomputers, midrange systems, mainframes, and supercomputers. Provide the name of at least one prominent vendor in each of these categories (and you can use IBM only once!).

Microcomputers:	Acer, Apple, Dell, Fujitsu, Hewlett-Packard, Lenovo, Sony, Toshiba
Midrange systems:	Dell, Fujitsu, Hewlett-Packard, IBM, Sun Microsystems

Mainframes:	Fujitsu, IBM, Unisys
Supercomputers:	Cray Inc., Fujitsu, Hewlett-Packard, Hitachi, IBM, NEC, Silicon Graphics International, Sun Microsystems

7. Briefly describe the four generations of computer programming languages, concentrating on the major differences among the generations. How does object-oriented programming fit into these generations? How does HTML fit into these generations?

The first generation language (1 GL) is machine language that the specific computer model is built to understand. A machine language instruction consists of an operation code and one or more addresses. The second generation language (2 GL) is assembly language, in which machine language codes and addresses are replaced by easily remembered mnemonic codes and symbolic addresses. The assembly language program is the source program, and a program called an assembler translates the source program into the machine language program or object program, which the computer can then directly execute.

Third generation languages (3 GLs) are procedural languages that are compatible across computer models. Each 3 GL instruction is translated into an average of about ten 1 GL instructions. The programmer devises step-by-step procedures and expresses them in 3 GL statements. Fourth generation languages (4 GLs) are called productivity languages and nonprocedural languages because they are easier to use. The programmer gives a precise statement of what is to be accomplished. The programmer does not need to write it in sequential order or write how to do it. Each 4 GL statement is translated into up to 100 1 GL instructions. Therefore, 4 GL programs are easier to write, modify, read, and understand.

Object-oriented (O-O) programming languages are similar to 3 GLs and 4 GLs—they are really a cross between a 3 GL and a 4 GL. O-O languages consist of embedding procedures called methods into objects and then integrating these objects into a program. Creating the objects in OOP is somewhat akin to 3 GL programming in that the procedures (called methods) are embedded in the objects, while putting the objects together to create an application is much closer to the use of a 4 GL.

HTML is Hypertext Markup Language, used to create World Wide Web pages. HTML consists of codes inserted in text to indicate headings, bold and italic text, images, and links to other Web pages. HTML is a coding language that is used to devise a Web page rather than a programming language in the sense of 3 GLs and 4 GLs.

8. List at least five categories of personal productivity software packages. Then concentrate on one of these categories, and describe a representative product in that category with which you are somewhat familiar. Provide both strong points and weak points of the particular product.

Personal productivity software includes presentation graphics, word processing, spreadsheet, database management, electronic mail, and calendaring and scheduling programs.

A presentation graphics product is Microsoft PowerPoint. It is easy to use and provides customized options. Its clip art and background options are somewhat limited, but getting better.

A word processing product is Microsoft Word. It underlines misspelled words and grammatical errors, it allows for easy formatting changes, and it allows importing of spreadsheets and graphics. It automatically changes lower case to upper case letters at the beginning of sentences. Microsoft Word is not good at correctly identifying grammatical errors in complex sentences. When trying to modify an existing document, it is sometimes difficult to modify unwanted formatting.

A spreadsheet program is Microsoft Excel. Formulas provide shortcut calculations. The user can calculate hypothetical projections easily with formula changes for easy sensitivity analysis. The user needs to understand how to set parameters and other features for a reduction in error rates.

Database management includes Microsoft Access. Access is an easy-to-use relational database, but it is limited in the size and complexity of database that it can handle.

Electronic mail includes Lotus Notes. Lotus Notes e-mail is part of a comprehensive groupware package that allows enterprise-wide communication. Many of the features of Lotus Notes are difficult for the beginner to pick up.

9. List the six major categories of support software.

Support software includes language translators, database management systems, CASE tools, communications interface software, utility programs, and the operating system.

10. What are the primary advantages of a fourth generation language over a third generation language? What are the primary disadvantages?

A 4 GL program requires fewer instructions than an equivalent 3 GL program, and—once the user knows the 4 GL language—a 4 GL program is easier to write, easier to modify, easier to read and understand, and less error-prone than a 3 GL program. The use of a 4 GL decreases systems development backlogs because less time is required to program in a 4 GL than a 3 GL. Some 4 GLs are not general purpose languages, so they have more limited usefulness. Most 4 GL programs take longer to execute than 3 GL programs because 4 GLs translate into much longer machine language programs. So, 4 GLs optimize programmer time, and 3 GLs optimize computer processing time.

11. What are the primary characteristics of an object-oriented language? How does an object-oriented language differ from a third generation language or a fourth generation language?

OOP provides the advantages of encapsulation, which protects the integrity of each object, and inheritance, which enables the creation of subclasses and superclasses from classes. OOP is similar to a 3 GL in that methods are embedded in objects, and these methods are quite similar to 3 GL programming. OOP has some similarities to a 4 GL in that programming pieces (objects) are integrated into an application. OOP requires a graphical user interface, as well as more computing power than a 3 GL or 4 GL. OOP provides increased productivity, maintainability, and paradigm consistency because codes can be reused.

12. For what does the CASE acronym stand? In general, what is the purpose of CASE tools? What types of individuals are most likely to use CASE tools?

CASE stands for computer-aided software engineering and is a collection of software tools that help to automate the software development life cycle. Systems analysts and programmers use CASE tools to specify program requirements, create data flow diagrams, and produce programming code from these high-level specifications. CASE changes the jobs of programmers and analysts by helping with the detailed work, allowing them to spend more time up-front defining the problem and what the system is supposed to do.

13. List at least three independent software houses (not associated with a computer vendor) that are major players in the software component of the information systems industry. List any software products that you regularly use and indicate the firm that developed each product.

Microsoft, Oracle, SAP, Computer Associates, and Adobe Systems are major independent software houses. Many students regularly use Microsoft Office Suite products, including Microsoft Word, Microsoft Excel, Microsoft PowerPoint, Microsoft Outlook, and Microsoft Access. Other popular software products are Microsoft Windows, Mozilla Firefox, Corel WordPerfect, Lotus Notes, Adobe Acrobat, and Adobe Creative Suite.

14. Provide the full names for the following acronyms or abbreviations used in this chapter.

OCR: optical character recognition, an input method that directly scans typed, printed, or hand-printed material

CPU: central processing unit, which includes the control unit and the arithmetic/logical unit

MFLOPS: millions of floating point operations per second, a rating derived by running a set of programs in a specific language on various computers to analyze relative performance speed

UPC: Universal Product Code, a bar code language, used widely by groceries, that enables accurate, fast input by scanning the bar code into a terminal

DASD: direct access storage device, often a rotating disk or stack of disks that enables direct access to files; could also be flash memory

DVD: digital video disk or digital versatile disk, a newer type of CD that holds several gigabytes of data; some of these are rewritable

DBMS: database management system, a type of support software that is used to store, modify, and manage data and to make that data accessible in a variety of meaningful and authorized ways

JCL: job control language, a special language to communicate instructions to an operating system

HTML: Hypertext Markup Language, a specialized language to "mark up" pages to be viewed on the World Wide Web, indicating how the pages are to look as well as indicate links to other pages

OOP: object-oriented programming, a type of computer programming based on the creation and use of a set of objects and the development of relationships among the objects

4 GL: fourth generation language, a computer language in which the user gives a precise statement of what is to be accomplished, not how to do it.

CASE: computer-aided software engineering, a set of integrated software tools used by IS specialists to automate some or all phases of a systems development life cycle process

Discussion Questions

1. Some commentators have suggested that mainframe computer systems could be squeezed out of

existence in the next few years, with the incredible advances in the capabilities of midrange systems and the power of supercomputers combining to divide up the present mainframe market. What do you think? Why?

Perhaps the most important thing to say is that the boundaries between midrange systems and mainframes, between mainframes and supercomputers, and even between midrange systems and supercomputers, are becoming more and more blurred. All three categories of machines are evolving. The most likely scenario is that, through evolution, mainframes will continue to be important because of their versatility, reliability, stability, throughput capabilities, and incredible array of already-developed software applications. Mainframes will change and evolve, but they are not going to go away.

2. What are the advantages and limitations of palmtop or handheld computers (including smartphones)? Do you have one? If so, for what do you use it? If not, what features would be particularly attractive to you?

Advantages include increased mobility and reduced cost—convenience at a reasonable cost. Limitations include the difficulty of inputting much data and displaying significant output; handheld computers are simply not convenient platforms for doing extensive computing work. The recent development of smartphones has added to the attractiveness of palmtop/handheld computers—they now combine computer capabilities with a telephone, Internet access (and therefore e-mail), a camera, and an MP3 player. Of course, these additional features have increased the cost.

3. What are the advantages and limitations of a tablet PC? Do you believe tablet PCs will be successful? Why or why not? Do you see yourself using a tablet PC in a few years?

The answer to this question became more complex when the Apple iPad was introduced in April 2010. Until that time, the major advantage for tablet PCs was that handwritten notes could be directly stored in the computer's memory in digital form such that they could be recalled and modified as desired. Otherwise, these full-function tablet PCs acted like any other PC. For these tablet PCs, the limitations were that the tablet PC was somewhat heavier and more awkward to use than a notepad and pen or pencil, and that writing on the tablet PC surface took some getting used to on the part of the user. A tablet PC is an unfamiliar medium for note-taking, but one that makes it easier to store and further process these notes. For these full-function tablet PCs, commentators felt that the devices would be successful, but only in a niche market made up of persons who take lots of notes and who are technologically inclined.

Apple's introduction of the iPad, which does *not* have the capability of reading handwritten notes or the full functionality of other tablet PCs, changed all that. The iPad is economical and is an excellent platform for viewing movies or TV shows, listening to music, reading a book, or browsing the Web. This smaller, less capable tablet PC, has caught on in a major way, not as a replacement for desktop, laptop, or handheld PCs, but as a new type of consumer device that happens to be a tablet PC. Our guess is that most students do see themselves as using an "iPad" type of tablet PC in a few years if they aren't already doing so, but students will have to answer for themselves as to whether they are likely to use a more capable tablet PC (such as the one pictured in Figure 2.5 in the text) in a few years.

4. With one firm (IBM) dominating the mainframe hardware market in the United States since its inception, and with that same firm currently near the top in every segment of the hardware industry (except microcomputers), has the computer hardware industry truly been competitive over the past five decades? Support your position.

One answer: The computer market has been competitive although perhaps not to the extent it would have been if IBM controlled a smaller share of the market. IBM's size has allowed it to devote significant

resources to R&D, enabling the company to develop more new technology than its competitors. Every year, IBM registers more patents than any other IT company—usually by a wide margin. However, the development of microprocessor chips and the commoditization of the microcomputer (PC) have changed the competitive balance in every part of the hardware market except mainframes, where IBM still dominates. IBM is a major competitor in the midrange systems and supercomputer markets, but is not truly dominant; IBM could still be a major competitor in the PC market, but it opted to exit the market because of the low profit margins. In the IT industry, software applications and computer services are the real revenue and profit generators today. IBM is the second largest software company in the world today (behind only Microsoft), and it ranks first in the computer services arena. Despite IBM's size and success, the computer hardware industry has been quite competitive over the past five decades—perhaps not as much as it would have been with multiple large firms—as evidenced by the incredible advances in computer speed and power and the incredible growth of computer applications.

5. List possible uses of a supercomputer in a business setting.

Industries using supercomputers include aerospace, automotive, chemical and pharmaceutical, general manufacturing, petroleum, and energy, in addition to the government. Supercomputers are used extensively in research and development in order to cut costs and time. Simulations of design and fabrication help analyze value-added activities. Complex simulations that would take weeks on smaller computers take hours on supercomputers. Three-dimensional analysis over time represents such complex modeling that a supercomputer is needed. Supercomputers are sometimes used as very large Web servers.

6. For most business information processing, what do you believe are the critical or limiting characteristics of today's computing systems—CPU speed, memory capacity, DASD capacity, internal communication speed, input-output speed, other factors, or some combination of these factors? Justify your answer.

Different factors or combinations of factors limit different types of application systems. For example, expert systems and artificial intelligence are limited by CPU speed. Many online database-intensive applications (like ATMs or reservation systems) are limited by the speed of transferring data to/from secondary storage (internal communication speed). For users of microcomputers, limiting factors may be memory, disk capacity, output speed, or rarely CPU speed.

7. Which one category of personal productivity software is of most value to you now as a student? Why? Within this category, what is your favorite software package? Why? Do you think that the one category of personal productivity software of most value to you will change as you move into your career? Why?

One possible answer to the first part of the question: Word processing is most important now because I continually write reports, outlines, and class notes. I use Microsoft Word because there is easy compatibility between the version I have at home and the version available in the laboratories on campus. Microsoft Word offers all of the customized options I need.

One possible answer to the second part of the question: I intend to become a financial analyst in my career, so I expect to make extensive use of a spreadsheet package in my job. I will use whatever spreadsheet package is the standard within my company, which is most likely to be Microsoft Excel.

Another possible answer to the second part of the question: I will be joining the family business (a small business), and I expect to create databases to capture the extensive data already generated in the business, then to use these databases to analyze various aspects of the business in order to present useful reports to

those running the business. The family business does not use a database management system now, so I expect to use Microsoft Access as the DBMS.

8. In the 1980s, a movement developed within the information systems industry to "stamp out COBOL" and replace it with 4 GLs and other productivity tools. Manifestations of this movement included the slogan to "Kill the COBOL programmer" (not literally, of course) and T-shirts bearing the word COBOL within the international symbol for not permitted (a red circle with a red line diagonally across the word COBOL). Do you think the movement will ever be successful? Why?

COBOL has proved its usefulness as a procedural language and is still important. COBOL is widely used and relatively easy to learn; the majority of business applications on mainframe computers still employ COBOL programming. COBOL offers the advantage of machine efficiency because COBOL programs take less time to execute than 4 GL programs or object-oriented programs. COBOL is used in many production programs that are run daily, and it will continue to be used for the foreseeable future for most of these production programs. While it is true that most new development today does not employ COBOL, there is still a demand for COBOL programmers to modify and enhance existing COBOL applications. Though the importance of COBOL is likely to continue to decline, the development of new versions of COBOL, including an object-oriented version, ensures its use into the foreseeable future.

9. You have probably had experience with at least one procedural (3 GL) language in either high school or college. What are the strengths and weaknesses of the particular language that you know best? Based on what you have gleaned from the text, what primary advantages would a nonprocedural 4 GL offer over the 3 GL that you know best? What disadvantages?

One possible answer: The 3 GL I am most familiar with is COBOL. The strengths of COBOL are that it is a very logical language, it is exact and precise, and it is fairly easy to understand. The weaknesses are that COBOL is very verbose, it is time consuming to write instructions, and if even one comma or space is incorrect the entire program will not run. When a program will not run it may be difficult to find the error.

A nonprocedural 4 GL would be better than COBOL because the programmer simply has to tell the computer what is to be accomplished, not how to accomplish it. A lot of hassle regarding syntax would be eliminated. However, a disadvantage is that the user might be able to write a 4 GL program without really understanding the problem, and thus the resulting program doesn't really do what it needs to do. When using COBOL, it is necessary to thoroughly understand and define the problem before writing the program. Generally speaking, though, a 4 GL program will be easier to write, easier to modify, easier to read and understand, and less error-prone than COBOL.

10. The box entitled "J2EE Versus .NET" introduced the differences between these two competing frameworks for developing Web applications. Explore J2EE and .NET in more detail by discussing the two frameworks with programmers you know and by conducting research using the Web. Are there other differences between the two frameworks? How would you modify the conclusions provided in the "J2EE Versus .NET" box?

This is obviously an open-ended question, and there are certainly many differences between the two frameworks in addition to those mentioned in the box. The box only attempts to mention the "big picture" differences between J2EE and .NET. In terms of modifying the conclusions, students' responses will undoubtedly vary considerably depending upon the sources they have read and the programmers to whom they have talked. Those that disagree with the conclusions stated in the sidebar will probably argue that the .NET framework provides as much flexibility as J2EE and thus can handle complex Web

applications.

Among several possible references comparing J2EE and .NET, we especially suggest V. Ramesh and Arijit Sengupta, "J2EE vs. .NET: An Application Development Perspective," in the *IS Management Handbook*, 8th edition, edited by Carol V. Brown and Heikki Topi, Auerbach Publications, Boca Raton, Florida, 2003, pages 415-423.

11. As you have read in newspapers and magazines, one firm seems to dominate the worldwide software market—Microsoft. With this degree of dominance by one firm, has the software industry truly been competitive, particularly over the past decade? Support your position.

As evidenced by relatively high software prices, competition has been less than optimal over the last decade. Development costs are high for competitive applications, and unit sales must be high to recoup these development costs. Thus, there are high barriers to entry in the industry. Small, successful firms with innovative products succumb to the fast profit from being bought out by the huge software firms. Because users want strong vendor support over the long term, they are generally unwilling to invest in products from smaller firms. Efficiency in marketing and distribution also favor larger firms. These factors favor the giants in the industry and tend to reduce competition.

Nevertheless, the entrepreneurial spirit is still alive and well in the software market, and new products pop up from new or established software vendors almost every day. While competition may have been less than optimal, innovation has not suffered.

12. In the late 1990s, the U.S. government, joined by several state governments, brought suit against Microsoft, arguing that Microsoft unfairly exercised monopoly power in the software industry. What was the outcome of these lawsuits? The European Union also brought suit against Microsoft for exercising monopoly power. What was the outcome of the European Union lawsuit? (Use the Web to conduct your research.)

For the federal antitrust case, the trial judge found for the government and against Microsoft, and ordered Microsoft to be split into two companies, an operating system company and an applications software company. On appeal, the court of appeals agreed that Microsoft had exercised monopoly power, but felt that the trial judge's order went too far. Instead, the court of appeals required Microsoft to change some of its marketing practices to be less predatory. For example, Microsoft could no longer require PC manufacturers to install Windows on all machines they sell, and Microsoft could no longer prohibit PC manufacturers from installing products from other software companies on new machines. In separate settlements, most of the states involved in the suit against Microsoft accepted a consent decree that was similar to the federal settlement.

In related cases, class-action suits against Microsoft for price fixing were filed in several states. Many of these class-action suits have been settled, with the result that buyers of Microsoft software packages were entitled to rebates.

In the European Union v. Microsoft antitrust case, the EU brought suit against Microsoft for abuse of its dominant position in the market. The suit started as a complaint over Microsoft's licensing practices (requiring suppliers of Microsoft's operating system to pay a royalty on each PC they sold, regardless of whether or not the PC contained Windows) and eventually grew to include a dispute over the lack of disclosure of some of the interfaces to the Windows operating system and the bundling of Microsoft's streaming media technology software with Windows. A settlement over Microsoft's licensing practices was reached in 1994, ending some of the questionable practices. A preliminary decision on the larger case was made in 2003 which ordered Microsoft to offer a separate version of Windows without

Windows Media Player and to offer the information necessary for competing network software to interact fully with both the desktop and server versions of Windows. In March 2004, the EU ordered Microsoft to pay a €497 million fine (the largest fine ever handed out by the EU at that time) in addition to divulging the server information (in 120 days) and producing a version of Windows without Windows Media Player (in 90 days). Microsoft produced a compliant version of Windows without Windows Media Player, and it did release some of the server source code by the deadline. Microsoft appealed the ruling, and a week-long hearing on the appeal ended in April 2006. In December 2005, the EU announced that it did not believe that Microsoft fully complied with the 2004 ruling (did not disclose all the appropriate information about its server programs), and it said that it would fine Microsoft €2 million a day until it did so. In June 2006 Microsoft stated that it had begun to provide the EU with the requested information, but the EU stated that it was too late. On July 12, 2006, the EU fined Microsoft an additional €280.5 million, €1.5 million a day from December 16, 2005, to June 20, 2006.

On September 17, 2007, Microsoft lost its appeal of the EU case. The €497 million fine was upheld, as were most of the other requirements of the original decision. In addition, Microsoft has to pay 80% of the legal costs of the European Commission, and the European Commission has to pay 20% of the legal costs of Microsoft. On October 22, 2007, Microsoft announced that it would comply with the decision, and it did not appeal within the required two months. However, some commentators believe that the final decision is not really a defeat for Microsoft (except for the monetary fine), because the interoperability information about its server programs will be available to anyone for a one-time €10,000 fee, and commercial vendors who use this information in their products will have to pay Microsoft 0.4% of the revenue from the products.

On February 27, 2008, the EU fined Microsoft an additional €899 million for failure to comply with the March 2004 antitrust decision. This represents the largest penalty ever imposed in 50 years of EU competition policy. On May 9, 2008, Microsoft lodged an appeal seeking to overturn the €899 million fine, stating that it intended to use the action as a "constructive effort to seek clarity from the court." This appeal is still pending, as far as we know. Reference: "European Union v. Microsoft," Wikipedia, *n.wikipedia.org/wiki/European_Union_v._Microsoft.*