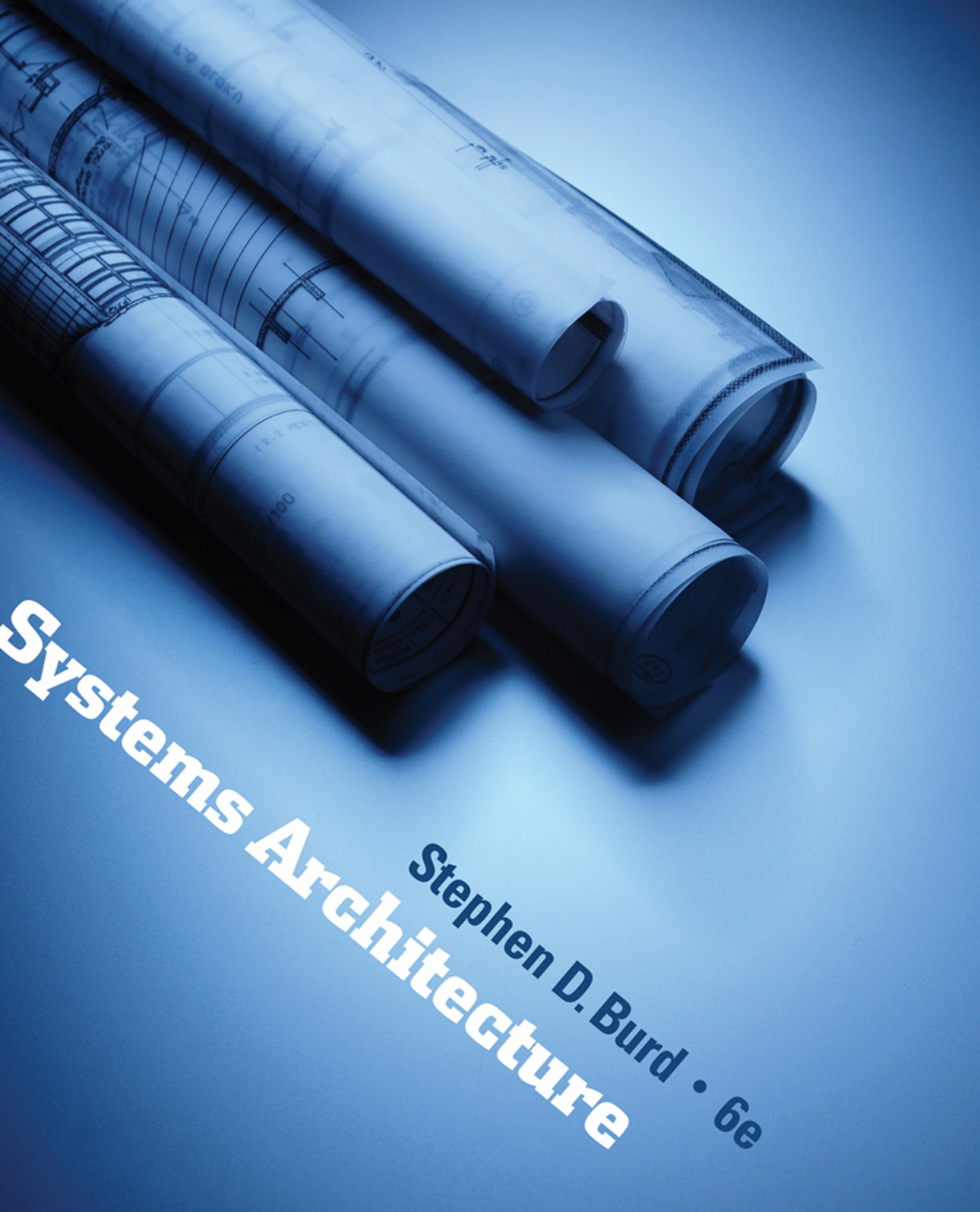

Systems Architecture

Stephen D. Burd • 6e

This page was intentionally left blank

SYSTEMS ARCHITECTURE

SYSTEMS ARCHITECTURE

Sixth Edition

Stephen D. Burd
University of New Mexico

COURSE TECHNOLOGY
CENGAGE Learning™

Australia • Brazil • Japan • Korea • Mexico • Singapore • Spain • United Kingdom • United States

COURSE TECHNOLOGY
CENGAGE Learning™

Systems Architecture, Sixth Edition
Stephen D. Burd

Executive Vice President and Publisher: Jonathan Hulbert

Executive Vice President of Editorial, Business: Jack Calhoun

Publisher: Joe Sabatino

Senior Acquisitions Editor: Charles McCormick, Jr.

Senior Product Manager: Kate Mason

Development Editor: Lisa M. Lord

Editorial Assistant: Nora Heink

Marketing Director: Keri Witman

Marketing Manager: Adam Marsh

Senior Marketing Communications Manager: Libby Shipp

Marketing Coordinator: Suellen Ruttkay

Content Project Manager: PreMediaGlobal

Media Editor: Chris Valentine

Senior Art Director: Stacy Jenkins Shirley

Cover Designer: Craig Ramsdell

Cover Image: ©Getty Images

Manufacturing Coordinator: Julio Esperas

Compositor: PreMediaGlobal

Library of Congress Control Number: 2010927431

Student Edition:

ISBN-13: 978-0-538-47533-4
ISBN-10: 0-538-47533-1

Instructor's Edition:

ISBN-13: 978-0-538-47536-5
ISBN-10: 0-538-47536-6

Course Technology
20 Channel Center Street
Boston, MA 02210
USA

Course Technology, a part of Cengage Learning, reserves the right to revise this publication and make changes from time to time in its content without notice.

Cengage Learning is a leading provider of customized learning solutions with office locations around the globe, including Singapore, the United Kingdom, Australia, Mexico, Brazil, and Japan. Locate your local office at: **www.cengage.com/global**

Cengage Learning products are represented in Canada by Nelson Education, Ltd.

To learn more about Course Technology, visit **www.cengage.com/coursetechnology**

Purchase any of our products at your local college store or at our preferred online store **www.cengagebrain.com**

Printed in the United States of America
1 2 3 4 5 6 7 16 15 14 13 12 11 10

To William I. Bullers, Jr., friend and colleague.

CONTENTS

PREFACE

INTENDED AUDIENCE

This book is intended for undergraduate students majoring or concentrating in information systems (IS) or information technology (IT) and as a reference for IS/IT professionals. It provides a technical foundation for systems design, systems implementation, hardware and software procurement, and computing resource management. Computer hardware and system software topics that are most useful to IS/IT students and professionals are described at an appropriate level of detail. For some topics, readers gain enough knowledge to solve technical problems. For other topics, they gain knowledge on communicating effectively with technical specialists.

Computer science students are exposed to computer hardware and system software technology in many undergraduate courses. Computer science books usually focus on a subset of the topics in this book. However, coverage of hardware and system software in an IS/IT curriculum is usually limited. A brief overview of hardware and system software might be provided in an introductory course, and some specific technical topics are often covered in other courses, but there's at most one course devoted to hardware and system software.

At this writing (May 2010), the latest curricula recommendations in IS and IT are IS 2010 and IT 2008. Many schools are still using curricula modeled on IS 2002 (see *www.acm.org* for details on these curricula). The topics covered in this book are mapped to all three curricula as follows:

- *IS 2002*—This book covers a superset of the requirements for IS 2002.4, Information Technology Hardware and System Software. Additional topics beyond those in IS 2002.4 include networks, application development software, and system administration.
- *IT 2008*—This book covers topics in four of the body of knowledge components: Integrative Programming and Technologies—Intersystems Communications and Overview of Programming Languages; Networking—all topics; Platform Technologies—all topics except Enterprise Deployment Software; and Systems Administration and Maintenance—Operating Systems and portions of Applications and Administrative Activities.
- *IT 2010*—This book covers the topics and learning objectives of the IT 2010.4 core course, IT Infrastructure.

This book can also serve as a supplement in courses on system design and computer resource management. For system design, it covers many technical topics to address when selecting and configuring hardware and system software. For computer resource management, it offers the broad technical foundation needed to manage resources effectively.

READERS' BACKGROUND KNOWLEDGE

Because target courses for this book are typically placed early in the recommended curricula, few assumptions are made about readers' background knowledge. Unlike many computer science books, readers aren't assumed to have an extensive background in mathematics, physics, or engineering. When necessary, background information in these areas is given in suitable depth.

In addition, readers aren't assumed to know any particular programming language. However, classroom or practical experience with at least one language is helpful to comprehend the discussions of CPU instruction sets, operating systems, and application development software. Programming examples are given in several programming languages and in pseudocode.

Detailed knowledge of a particular operating system isn't required. However, as with programming experience, practical experience with at least one operating system is helpful. Lengthy examples from operating systems are purposely avoided, but there are some short examples from MS-DOS, UNIX/Linux, and recent Windows versions.

Finally, knowledge of low-level machine instructions or assembly-language programming isn't assumed. Assembly-language instructions are described in several chapters, but a generic assembly language is used, and no detailed coverage of assembly-language program organization is included.

CHANGES IN THIS EDITION

The fifth edition was first published in 2005. Updates were needed throughout the book to address changes since that time. The following sections summarize major updates and additions, although most chapters include many additional minor changes, such as updates to screen captures, hardware specifications, and standards.

- *Chapter 1*—Updated the discussion of periodical literature and online sources of technology information.
- *Chapter 2*—Updated typical computer specifications; revised definitions of computer classes; expanded the discussion of multicomputer, distributed, and cloud computer architectures; added a Technology Focus on distributed simulation applications; modernized the Business Focus case; and updated the Technology Focus features on IBM POWER processors and the parallel evolution of Intel CPUs and Microsoft operating systems.
- *Chapter 3*—Updated the coverage of floating-point formats and Unicode standards to the latest standards.

- *Chapter 4*—Updated the discussion of RISC and the Pentium Technology Focus, added a Technology Focus on SPEC and TPC benchmarks, and updated several sections (including the Technology Focus features) to reflect current CPU clock rates, word sizes, fabrication technology, and multicore architecture.
- *Chapter 5*—Moved the discussion of memory allocation and addressing to Chapter 11, added details of solid-state drives, updated the coverage of memory packaging and nonvolatile memory technologies, and modernized the coverage of magnetic tapes, magnetic disks, and optical discs.
- *Chapter 6*—Expanded the discussion of buses to include internal/external subsidiary buses and serial bus technology, replaced the SCSI Technology Focus with one on PCI, modernized coverage and examples of buffering and caching, updated the Technology Focus about on-chip memory cache to current Intel multicore CPUs, and expanded the coverage of compression, including an updated Technology Focus.
- *Chapter 7*—Reduced the coverage of older display technologies and expanded the discussion of current display types and video adapters.
- *Chapter 8*—Expanded the discussion of bandwidth and S/N ratio, updated and expanded the coverage of copper and optical cabling, expanded the coverage of wireless transmission, and updated the Technology Focus and Business Focus.
- *Chapter 9*—Reduced the coverage of bus and ring topologies, expanded the discussion of switching and routing and sharpened the distinction between them, expanded wireless network coverage (including a new Technology Focus on WiMAX), updated the Ethernet coverage, and updated the Business Focus.
- *Chapter 10*—Updated the coverage of application development tools, including related Technology Focus and Business Focus features.
- *Chapter 11*—Revised the introductory material extensively, incorporated and updated memory allocation and addressing material moved from Chapter 5, and added material on hypervisors and a Technology Focus on VMware.
- *Chapter 12*—Updated and expanded the coverage on RAID, added material on backup and recovery procedures, and added a Technology Focus on the Google File System.
- *Chapter 13*—Reorganized the chapter for improved flow of topics, expanded the discussion of distributed architectures to include peer-to-peer architectures, revised the discussion of protocol stacks to be consistent with changes in Chapter 9, updated the Technology Focus to cover Java Platform, Extended Edition, and added new material on cloud computing architectures, including a Business Focus.
- *Chapter 14*—Streamlined to eliminate repetition with expanded coverage in other chapters, updated the Technology Focus on Windows monitoring tools, and updated screen captures to current Windows versions.

RESOURCES FOR INSTRUCTORS

Systems Architecture, Sixth Edition includes the following resources to support instructors in the classroom. All the teaching tools available with this book are provided to the instructor on a single CD. They can also be accessed with your single sign-on (SSO) account at Cengage.com.

- *Instructor's Manual*—The Instructor's Manual provides materials to help instructors make their classes informative and interesting. It includes teaching tips, discussion topics, and solutions to end-of-chapter materials.
- *Classroom presentations*—Microsoft PowerPoint presentations are available for each chapter to assist instructors in classroom lectures or to make available to students.
- *ExamView®*—ExamView is a powerful testing software package that enables instructors to create and administer printed, computer (LAN-based), and Internet exams. It includes hundreds of questions corresponding to the topics covered in this book so that students can generate detailed study guides with page references for further review. The computer-based and Internet testing components allow students to take exams at their computers and save instructors time by grading each exam automatically.
- *Distance learning content*—Course Technology is proud to present online content in WebCT and Blackboard to provide the most complete and dynamic learning experience possible. For more information on how to bring distance learning to your course, contact your local Cengage sales representative.

WORLD WIDE WEB SITES

Two support sites for this book (instructor and student), located at *www.cengage.com/mis/burd*, offer the following:

- The Instructor's Manual
- Figure files
- End-of-chapter questions and answers
- Web resource links for most book topics and research problems
- Additional content on virtualization
- Text updates and errata
- Glossary

ORGANIZATION

This book's chapters are organized into four groups. The first group contains two chapters with overviews of computer hardware, software, and networks and describes sources of technology information. The second group consists of five chapters covering hardware technology. The third group contains two chapters on data communication and computer networks. The fourth group includes five chapters covering software technology and system administration.

The chapters are intended for sequential coverage, although other orderings are possible. The prerequisites for each chapter are described in the following section. Other chapter orders can be constructed based on these prerequisites. Chapter 2 should always be covered before other chapters, although some sections can be skipped without loss of continuity, depending on which subsequent chapters are included or skipped.

There should be time to cover between 9 and 12 chapters in a three-credit-hour undergraduate course. This book contains 14 chapters to offer flexibility in course content. Topics in some chapters can be covered in other courses in a specific curriculum. For example, Chapters 8 and 9 are often covered in a separate networking course, and Chapter 14 is often included in a separate system administration course. Instructors can choose specific chapters to best match the overall curriculum design and teaching preferences.

CHAPTER DESCRIPTIONS

Chapter 1, "Computer Technology: Your Need to Know," briefly describes how knowledge of computer technology is used in the systems development life cycle. It also covers sources for hardware and system software information and lists recommended periodicals and Web sites. It can be skipped entirely or assigned only as background reading.

Chapter 2, "Introduction to Systems Architecture," provides an overview of hardware, system and application software, and networks. It describes main classes of hardware components and computer systems and describes the differences between application and system software. This chapter introduces many key terms and concepts used throughout the book.

Chapter 3, "Data Representation," describes primitive CPU data types and common coding methods for each type. Binary, octal, and hexadecimal numbering systems and common data structures are also discussed. Chapter 2 is a recommended prerequisite.

Chapter 4, "Processor Technology and Architecture," covers CPU architecture and operation, including instruction sets and assembly-language programming. It describes traditional architectural features, including fetch and execution cycles, instruction formats, clock rate, registers, and word size. It also discusses methods for enhancing processor performance as well as semiconductor and microprocessor fabrication technology. Chapters 2 and 3 are necessary prerequisites.

Chapter 5, "Data Storage Technology," describes implementing primary and secondary storage with semiconductor, magnetic, and optical technologies. Principles of each storage technology are described first, followed by factors affecting each technology and guidelines for choosing secondary storage technologies. Chapters 3 and 4 are necessary prerequisites, and Chapter 2 is a recommended prerequisite.

Chapter 6, "System Integration and Performance," explains communication between computer components and performance enhancement methods. It starts with a discussion of system bus and subsidiary bus protocols, followed by coverage of device controllers, mainframe channels, and interrupt processing. Performance enhancement

methods include buffering, caching, parallel and multiprocessing, and compression. Chapters 4 and 5 are required prerequisites, and Chapters 2 and 3 are recommended prerequisites.

Chapter 7, "Input/Output Technology," describes I/O devices, including keyboards, pointing devices, printers and plotters, video controllers and monitors, optical input devices, and audio I/O devices. It also covers fonts, image and color representation, and image description languages. Chapter 3 is a necessary prerequisite, and Chapters 2, 5, and 6 are recommended prerequisites.

Chapter 8, "Data and Network Communication Technology," covers data communication technology, beginning with communication protocols, analog and digital signals, transmission media, and bit-encoding methods. This chapter also explains serial and parallel transmission, synchronous and asynchronous transmission, wired and wireless transmission, channel-sharing methods, clock synchronization, and error detection and correction. Chapters 2 and 3 are recommended prerequisites.

Chapter 9, "Computer Networks," describes network architecture and hardware. It starts with network topology and message forwarding, and then explains media access control and network hardware devices, such as routers and switches. This chapter also covers IEEE and OSI networking standards and includes an in-depth look at Internet architecture and TCP/IP. Chapters 3, 4, and 8 are necessary prerequisites, and Chapter 2 is a recommended prerequisite.

Chapter 10, "Application Development," begins with a brief overview of the application development process and development methodologies and tools, and then discusses programming languages, compilation, link editing, interpretation, and symbolic debugging. The final section describes application development tools, including CASE tools and integrated development environments (IDEs). Chapters 2, 3, and 4 are necessary prerequisites.

Chapter 11, "Operating Systems," describes the functions and layers of an operating system, explains resource allocation, and describes how an operating system manages the CPU, processes, threads, and memory. Chapters 2, 4, and 5 are necessary prerequisites, and Chapter 10 is a recommended prerequisite.

Chapter 12, "File and Secondary Storage Management," gives an overview of file management components and functions, including differences between logical and physical secondary storage access, and describes file content and structure and directories. Next, this chapter describes storage allocation, file manipulation, and access controls and ends with file migration, backup, recovery, fault tolerance, and storage consolidation methods. Chapters 5 and 11 are necessary prerequisites, and Chapter 10 is a recommended prerequisite.

Chapter 13, "Internet and Distributed Application Services," begins by discussing distributed computing and network resource access, network protocol stacks, and directory services. This chapter also explains interprocess communication, Internet protocols for accessing distributed resources, and component-based application

development. Finally, it describes cloud computing models. Chapters 2, 8, and 9 are necessary prerequisites. Chapters 4, 11, and 12 are recommended prerequisites.

Chapter 14, "System Administration," gives an overview of system administration tasks and the strategic role of hardware and software resources in an organization. It then describes the hardware and software acquisition process. Next, the chapter discusses methods for determining requirements and monitoring performance. The next section describes measures for ensuring system security, including access controls, auditing, virus protection, software updates, and firewalls. The last section discusses physical environment factors affecting computer operation. Chapters 2, 4, 8, 11, and 12 are recommended prerequisites.

Appendix, "Measurement Units," summarizes common measurement units, abbreviations, and usage conventions for time intervals, data storage capacities, and data transfer rates.

ACKNOWLEDGMENTS

The first edition of this book was a revision of another book, *Systems Architecture: Software and Hardware Concepts*, by Leigh and Ali. Some of their original work has endured through all subsequent editions. I am indebted to Leigh, Ali, and Course Technology for providing the starting point for all editions of this book.

I thank everyone who contributed to this edition and helped make previous editions a success. Jim Edwards took a chance on me as an untested author for the first edition. Kristen Duerr, Jennifer Locke, Maureen Martin, and Kate Mason shepherded the text through later editions. Kate Mason oversaw this edition, and Lisa Lord helped make the text much more readable. Thanks to all past and present development and production team members. Thanks also to the peer reviewers of this edition: Angela Clark, University of South Alabama; Raymond Hansen, Purdue University; Jim Mussulman, Southern Illinois University Edwardsville; Barbara Ozog, Benedictine University; John Reynolds, Grand Valley State University; and Jeffrey Sprankle, Purdue University.

I thank students in the undergraduate MIS concentration at the Anderson Schools of Management, University of New Mexico, who have used manuscripts and editions of this book over the past two decades. Student comments have contributed significantly to improving the text. I also thank my department chair, Steven Yourstone, and my faculty colleagues—Ranjit Bose, Nick Flor, Peter Jurkat, Xin Luo, Laurie Schatzberg, Josh Saiz, and Alex Seazzu—for their continued support and encouragement of my textbook-writing activities.

Finally, I'd like to thank Dee, my wife, and Alex and Amelia, my children. Developing this book as a sole author through multiple editions has been a time-consuming process that often impinged on family time and activities. My success as an author would not be possible without my family's love and support.

CHAPTER 1

COMPUTER TECHNOLOGY: YOUR NEED TO KNOW

CHAPTER GOALS

- Describe the activities of information systems professionals
- Describe the technical knowledge of computer hardware and system software needed to develop and manage information systems
- Identify additional sources of information for continuing education in computer hardware and system software

The words you're reading now result, in part, from computer-based information systems. The author, editors, production team, and distribution team all relied on computer systems to organize, produce, and convey this information. Although many different kinds of computer information systems were involved, they share similar technology: Each consists of computer hardware, system and application software, data, and communication capabilities. In this chapter, you learn why you need to study hardware and software technology if you work, or plan to work, in information systems. You also learn about additional sources of information that can help expand and update your knowledge of hardware and software.

TECHNOLOGY AND KNOWLEDGE

The world is filled with complex technical devices that ordinary people use every day. Fortunately, you don't need a detailed understanding of how these devices work to use them. Imagine needing three months of training just to use a refrigerator or needing a detailed understanding of mechanics and electronics to drive a car. Using the earliest computers in the 1940s took years of training, but today, even though computers are increasingly complex and powerful, they're also easier to use. As a result, computers have proliferated beyond their original scientific applications into businesses, classrooms, and

homes. If computers have become so easy to use, why do you need to know anything about their inner technology?

Acquiring and Configuring Technological Devices

The knowledge required to purchase and configure technically complex devices is far greater than the knowledge required to use them effectively. Many people can use complex devices, such as cars, home theater systems, and computers, but few people feel comfortable purchasing or configuring them. Why is this so?

When you walk into a store or visit a Web site to purchase a computer, you're confronted with a wide range of choices, including processor type and speed, hard disk speed and capacity, memory capacity, and operating system. To make an informed choice, you must know your preferences and requirements, such as the application software you plan to use and whether you plan to discard or upgrade the computer in a year or two. To evaluate the alternatives and determine their compatibility with your preferences and requirements, you must be able to comprehend technical terms (for example, gigahertz, gigabyte, DDR, and USB), technical and sales documentation, product and technology reviews, and the advice of friends, experts, and salespeople.

An information systems (IS) professional faces computer acquisition, upgrade, and configuration choices that are far more complex. Large computer systems and the software that runs on them use more complex technology than smaller ones do. There are many more components and, therefore, more complex configuration, compatibility, and administrative issues. Of course, the stakes are higher. Employers and users rely on the expertise of IS professionals, and companies invest substantial sums of money based on their recommendations. Are you (or will you be) able to meet the challenge?

INFORMATION SYSTEM DEVELOPMENT

When developing an information system, IS professionals follow a series of steps called a **systems development life cycle (SDLC)**. Figure 1.1 shows a modern SDLC called the **Unified Process (UP)**. Under the UP, an information system is built in a series of 4- to 6-week repeated steps called **iterations** (the vertical columns separated by dashed lines). Although Figure 1.1 shows six iterations, the number of iterations is tailored to each development project's specifications. Typically, the first iteration or two produces documentation and a prototype (model) system that's refined and expanded in subsequent iterations until it becomes the final system.

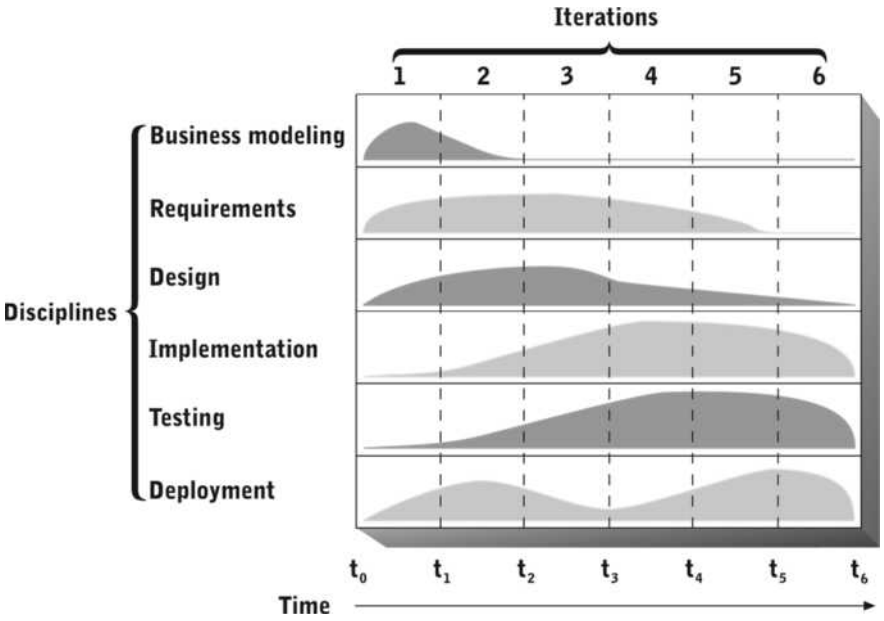

FIGURE 1.1 Disciplines and iterations in the Unified Process
Courtesy of Course Technology/Cengage Learning

Each iteration includes whatever activities are needed to produce testable models or working software. Related activities are grouped into UP **disciplines**. For example, the testing discipline includes activities such as creating test data, conducting tests, and evaluating test results. Activities and efforts in each discipline vary across iterations, as shown by the shaded curves in Figure 1.1. For example, in this figure, activities in iteration 1 are drawn primarily from the business modeling, requirements, design, and deployment disciplines, and activities in iteration 6 are drawn primarily from the implementation, testing, and deployment disciplines. As with the number of project iterations, the mix of activities in each iteration is tailored to each development project. Therefore, efforts in each discipline aren't always distributed across iterations exactly as shown in Figure 1.1.

The following sections explore the UP disciplines in more detail and describe the knowledge of computer hardware and system software each one requires.

Business Modeling and Requirements Disciplines

Activities in the **business modeling discipline** and the **requirements discipline** are primarily concerned with building models of the organization that will own and operate the system, models of the system's environment, and models of system and user requirements. The models can include narratives, organizational charts, workflow diagrams, network diagrams, class diagrams, and interaction diagrams. The purpose of building business and requirements models is to understand the environment in which the system will function and the tasks the system must perform or assist users to perform.

While building business and requirements models, developers ask many questions about the organization's needs, users, and other constituents and the extent to which these needs are (or aren't) being met and how they'll be addressed by a new system.

Technical knowledge of computer hardware and system software is required to assess the degree to which users' needs are being met and to estimate the resources required to address unmet needs.

For example, an analyst surveying a point-of-sale system in a retail store might pose questions about the current system, such as the following:

- How much time is required to process a sale?
- Is the system easy to use?
- Is enough information being gathered (for example, for marketing purposes)?
- Can the hardware and network handle periods of peak sales volume (such as during holidays)?
- Can hardware and application software be supported by a different operating system?
- Are cheaper hardware alternatives available?
- Could a cloud computing environment support the application software?

Answering these questions requires an in-depth understanding of the underlying hardware and software technologies. For example, determining whether a system can respond to periods of peak demand requires detailed knowledge of processing and storage capabilities, operating systems, networks, and application software. Determining whether cheaper alternatives exist requires technical knowledge of a wide range of hardware and software options. Determining whether cloud computing could be used requires a detailed understanding of the software environment and whether it's compatible with various cloud computing environments.

Design Discipline

The **design discipline** is the set of activities for determining the structure of a specific information system that fulfills the system requirements. The first set of design activities, called **architectural design**, selects and describes the exact configuration of all hardware, network, system software, and application development tools to support system development and operations (see Figure 1.2). These selections affect all other design decisions and serve as a blueprint for implementing other systems.

Specific systems design tasks include selecting the following:

- Computer hardware (processing, storage, input/output [I/O], and network components)
- Network hardware (transmission lines, routers, and firewalls)
- System software (operating system, database management system, Web server software, network services, and security software and protocols)
- Application development tools (programming languages, component libraries, and integrated development environments)

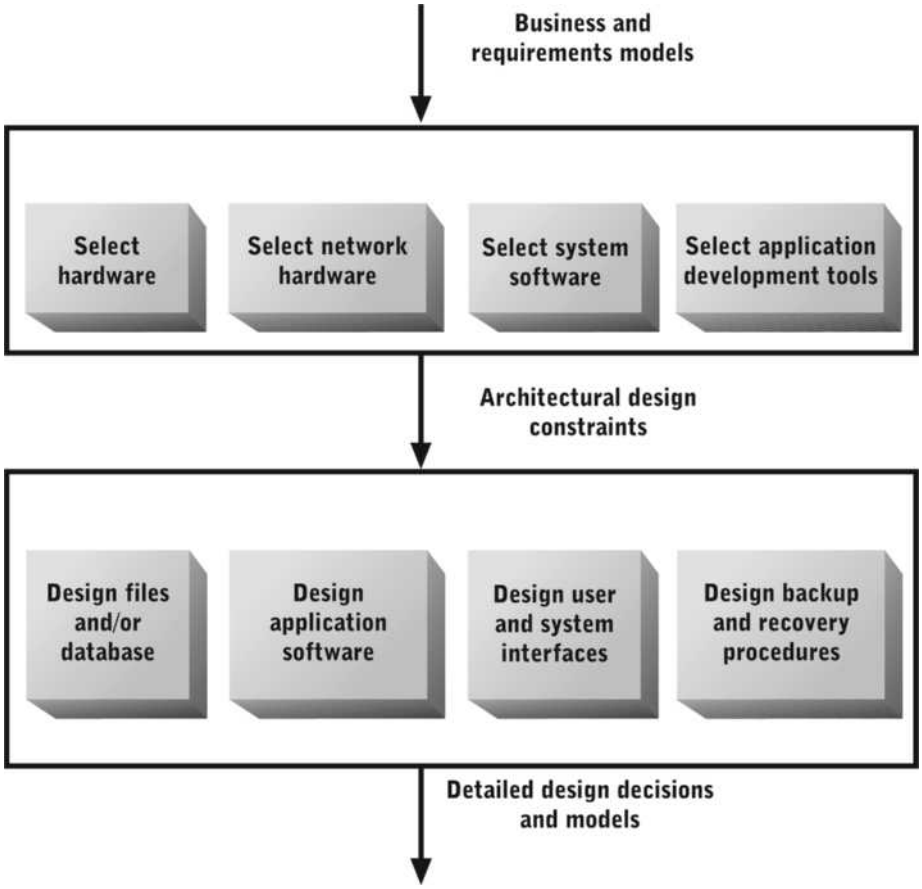

FIGURE 1.2 Design activities in the Unified Process
Courtesy of Course Technology/Cengage Learning

Collectively, these choices define an **information architecture**—requirements and constraints that define important characteristics of information-processing resources and how these resources interact with one another. When actual hardware, network, and system software components are acquired and installed, they make up an information technology infrastructure for one or more information systems. Operating and maintaining the infrastructure is a complex and costly endeavor in most organizations.

The remaining design activities, called **detailed design**, are narrower in scope and constrained by the information architecture. Detailed design activities include the following:

- File or database design (such as grouping data elements into records and files, indexing, and sorting)
- Application software design
- User and external system interface design (input screen formats, report formats, and protocols to interact with external services and systems)
- Design of system backup and recovery mechanisms

Technical knowledge of computer hardware and system software is most important for performing architectural design activities. Selecting hardware and network components requires detailed knowledge of their capabilities and limitations. When multiple hardware and network components are integrated into a single system, the designer must evaluate their compatibility. Hardware, network, and overall performance requirements affect the choice of system software. The designer must also consider the compatibility of new hardware, network components, and system software with existing information systems and computing infrastructure.

Selecting appropriate development tools requires knowing the information system requirements and capabilities of the hardware, network, and operating system. Development tools (and the software components built with them) vary widely in their efficiency, power, and compatibility. Tool selection also affects future system development projects.

Implementation and Testing Disciplines

The **implementation discipline** of the UP includes all activities for building, acquiring, and integrating application software components. The **testing discipline** includes activities that verify correct functioning of infrastructure and application software components and ensure that they satisfy system requirements. Implementation and especially testing activities require specific knowledge of the hardware, network, and system software.

For example, developing an application software component that interacts with an external Web service to schedule a shipment requires specific knowledge of the network protocols used to find and interact with the service. Diagnosing an error that occurs when the software executes requires detailed knowledge of the operating system, network services, and network protocols for creating, transmitting, and receiving the Web service request and response.

Deployment Discipline

The **deployment discipline** is the set of activities for installing and configuring infrastructure and application software components and bringing them into operation. Questions addressed by deployment discipline activities include the following:

- Who should be involved in and responsible for deploying each part of the system?
- In what order should parts of the system be deployed?
- Will any parts of the new system operate in parallel with the previous system?

Technical knowledge of computer hardware and system software is needed to perform many deployment tasks. Installing and configuring hardware, networks, and system software is a specialized task that requires a thorough understanding of the components being installed and the purposes for which they'll be used. Tasks such as formatting storage devices, setting up system security, installing and configuring network services, and establishing accounting and auditing controls require considerable technical expertise.

Systems Evaluation and Maintenance

Although not a formal UP discipline, systems evaluation and maintenance is an important group of activities that accounts for much of the long-range system cost. Over time,

problems with the system can and do happen. Errors that escaped detection during testing and deployment might show up. For example, a Web-based order-entry system might become overloaded because of inadequate estimates of processing volume, network congestion, or capacity limits in underlying hardware or database services. Information needs can and do change, necessitating changes to collect, process, and store additional data.

Minor system changes, such as correcting application software errors or minor processing adjustments, are normally handled as maintenance changes. Maintenance changes can require extensive technical knowledge, and some technical knowledge might be needed to classify a proposed change as major or minor. Will new processing requirements be the "straw that breaks the camel's back" in terms of hardware, network, or software capacity? Do proposed changes require application development tools that aren't compatible with the current system's design or configuration? The answers to these questions determine whether the existing system will be modified or replaced by a new system.

If the existing system is to be modified, the application software components and files to be changed are identified, modified, tested, and deployed. The technical knowledge requirements depend heavily on the specific hardware, network, and software components affected by the change. If a new system is required, a new systems development life cycle is initiated.

MANAGING COMPUTER RESOURCES

So far, the need for technological knowledge has been discussed in the context of developing a single information system. However, think about the complexities and knowledge needed to manage the thousands of computer resources in a large organization, where many new development projects or system upgrades can be in progress at once.

In this type of environment, you must pay more attention to two critical technological issues—compatibility and future trends. Both issues are important because of the integration of computing technology in and across every function of modern organizations. For example, accounts payable and accounts receivable programs usually share a common hardware platform and operating system. Data from both systems is shared by a financial reporting system, which might be a different software system running on an entirely different computer. Data from many sources in the organization is often stored in a common database and accessed via an internal network or the Internet.

Managers of integrated collections of information systems and supporting infrastructure must contend with a great deal of technical complexity. They must ensure that each new system not only operates correctly by itself, but also operates smoothly with all the other systems in the organization. They must also make sure hardware and software acquisitions are a good foundation for both current and future systems.

Given the rapid pace of change in computer technology, a manager must have a broad understanding of current technology and future technology trends. Will the computer purchased today be compatible with the hardware available three years from now? Can the organization's communication network be expanded easily to meet future needs? Should the organization invest only in tried-and-true technologies, or should it acquire cutting-edge technologies in hopes of improving performance or gaining a competitive advantage? Should the organization invest in buying enterprise-level software packages to use on local hardware, or should applications be housed in a rented server farm?

The answers to these questions require in-depth technical knowledge—far more knowledge than any one person has. Typically, managers confronted by these questions rely on the advice of experts and other sources of information. Even so, they must have an adequate base of technical knowledge to understand this information and advice.

ROLES AND JOB TITLES

Many people in many different jobs are called computer professionals. There's a bewildering array of job titles, specializations, and professional certifications to describe the wide range of computer-related roles. To add to the confusion, responsibilities associated with specific job titles vary considerably from organization to organization. The following sections attempt to classify computer professionals into groups, explain some of their common characteristics, and describe the computer hardware and system software knowledge each group needs.

Software Developers

Software can be categorized loosely into two types: application software and system software. End users use application software to perform specific tasks, such as processing customer orders or developing and formatting documents and financial analyses. System software generally hides in the background, unnoticed or barely noticed by most end users. Examples include many parts of an operating system, database management systems, and software that protects networks against intruders. Of course, some software doesn't fit neatly into either category.

Many **software developers** create application software for specific processing needs. They have many different job titles, including programmer, systems analyst, and systems designer. Each role contributes to a different part of the systems development life cycle. A **systems analyst** performs activities in the business modeling and requirements disciplines. A **systems designer** performs activities in the design discipline and sometimes the deployment discipline. A **programmer** builds and tests software.

Many software developers have responsibilities that match these job descriptions precisely. For example, a systems analyst is often responsible for business modeling, requirements, design, and management tasks for a development project. Programmers often perform some requirements and design discipline tasks in addition to building and testing software.

The wide variety of application software causes further confusion about job titles, activities, responsibilities, and required education and training. For example, developers of application software that processes business transactions or provides information to managers usually have college or technical degrees in management or business with a specialization in information processing.

NOTE

Other names for the information processing field include management information systems, data processing, and business computer systems.

Developers of application software for scientific areas, such as astronomy, meteorology, and physics, typically have degrees in mathematics or **computer science**. Developers of application software for technical areas, such as robotics, flight navigation, and scientific instrumentation, typically have degrees in computer science or some branch of engineering.

All application software developers need technical knowledge of computer hardware and system software, as described earlier in "Information System Development." Software developers in technical areas typically need in-depth hardware knowledge because of the hardware control characteristics of many of these applications. Software developers in scientific areas must also have in-depth hardware knowledge if the applications they develop push the boundaries of hardware capabilities, such as simulating weather patterns on a global scale with supercomputers.

Systems programmers develop system software, such as operating systems, compilers, database management systems, Web servers, and network security monitors. These programmers typically have degrees in computer science or computer engineering. Organizations using a lot of computer equipment and software employ systems programmers to perform tasks such as hardware troubleshooting and software installation and configuration. Organizations that develop and market system and application development software, such as Microsoft, Oracle, and Cisco, employ many systems programmers.

Systems programmers must have in-depth knowledge of system software as well as computer hardware and networks because many types of system software interact directly with computer or network hardware. For this reason, computer science and computer engineering programs usually require students to take several different courses to learn the subjects discussed in this book.

Hardware Personnel

Computer hardware vendors employ a variety of people for design, installation, and maintenance. Lower-level personnel usually have technical degrees and/or vendor-specific training, and higher-level personnel usually have degrees in computer science or computer engineering. Employees must have extensive knowledge of computer hardware, including processing, data storage, I/O, and networking devices. Hardware designers need the most in-depth knowledge, far exceeding the scope of this book.

System Managers

The proliferation of computer hardware, software, and networks in today's organizations has created a need for many computer-related managers and administrators. The job descriptions vary widely because of differences in organizational structure and the nature of the organization's information systems and infrastructure. Common job titles include computer operations manager, network administrator, database administrator, and chief information officer.

A **computer operations manager** oversees the operation of a large information processing facility. These facilities usually house one or more large computer systems and all related peripheral equipment in a central location. They often have large databases, thousands of application programs, and dozens to hundreds of employees and perform a lot of batch processing. Organizations requiring this type of computing facility include large

banks, credit reporting bureaus, the Social Security Administration, and the Internal Revenue Service. Organizations such as Google and Electronic Data Systems also operate this type of computing facility for themselves and for clients. Management of day-to-day operations in these facilities is extremely complex. Scheduling, staffing, security, system backups, maintenance, and upgrades are some important responsibilities of a computer operations manager.

A computer operations manager can have many technical specialists on staff. Staff members usually have considerable technical knowledge in narrow specialties, such as data storage hardware, network configuration and security, mainframe operating systems, and performance tuning. A computer operations manager needs a broad base of technical knowledge to understand the organization's information systems and infrastructure and must be capable of understanding the advice of technical staff.

Typically, the title of **network administrator** is applied to one of two roles. The first is responsibility for an organization's network infrastructure, such as for an Internet service provider or a large multinational corporation. Designing, operating, and maintaining a large network require substantial technical expertise in computer hardware, telecommunications, and system software. The role of network administrator in this environment is an important high-level position. Technical knowledge requirements are similar to those for a computer operations manager, although the emphasis is on network and data communication technologies.

In a smaller organization, the title of network administrator is used for the manager of a local area network. These networks connect anywhere from a half dozen to a few hundred computers (mostly desktop and portable computers) and provide access to shared databases. The network administrator can be responsible for many tasks other than operating and maintaining the network, including installing and maintaining end-user software, installing and configuring hardware, training users, and assisting management in selecting and acquiring software and hardware. This position is one of the most demanding in breadth and depth of required skills and technical knowledge.

The technology for managing and accessing large collections of data, called databases, is specialized and highly complex. This complexity, combined with managerial recognition of the importance of data resources, has resulted in creating many positions with the title of **database administrator**. This role requires both technical expertise and the ability to help the organization make optimal use of its data resources for tasks such as market research.

A large organization with a substantial investment in computer, network, and software technology usually has one high-level manager with the title **chief information officer (CIO)**. Many of the previously defined positions (database administrator, network administrator, and computer operations manager) report to the CIO. The CIO is responsible for the organization's computers, networks, software, and data as well as for strategic planning and the effective use of information and computing technology.

CIOs can't possibly be experts in every aspect of computer technology related to their organizations, but they must have a broad enough base of technical knowledge to interact effectively with all the organization's technical specialists. They must also be aware of how technology is changing and how best to respond to these changes to support an organization's mission and objectives.

COMPUTER TECHNOLOGY INFORMATION SOURCES

This book gives you a foundation of technical knowledge for a career in information system development or management. Unfortunately, this foundation can erode quickly because computer and information technologies change rapidly. How will you keep up with the changes?

You can use many resources to keep your knowledge current. Periodicals and Web sites offer a wealth of information on current trends and technologies. Training courses from hardware and software vendors can teach you the specifics of current products. Additional coursework and self-study can keep you up to date on technologies and trends not geared toward specific products. By far, the most important of these activities is reading periodical literature.

Periodical Literature

The volume of literature on computer topics is huge, so IS professionals face the difficult task of determining which sources are the most important and relevant. Because of differences in training and focus among computer professionals, sources of information oriented toward one specialty are difficult reading for professionals in other specialties. For example, a detailed discussion of user interaction and validation of requirement models targeted to IS professionals might be beyond a computer scientist or engineer's training. Similarly, detailed descriptions of optical telecommunication theory might be beyond an IS professional's training. These differences pose a problem for IS professionals who need current information on hardware and system software technology. The following periodicals are good information sources for IS professionals:

- *ACM Computing Surveys* (*http://surveys.acm.org*)—An excellent source of information on the latest research trends in computer software and hardware. Contains in-depth summaries of technologies or trends geared toward a readership with moderate to high familiarity with computer hardware and software.
- *Computerworld* (*www.computerworld.com*)—A weekly magazine focusing primarily on computer news items. Covers product releases, trade shows, and occasional reports of technologies and trends.
- *Communications of the ACM* (*http://cacm.acm.org*)—A widely used source of information about research topics in computer science. Many of the articles are highly technical and specialized, but some are targeted to a less research-oriented audience.
- *Computer* (*www.computer.org/computer*)—A widely used source of information on computer hardware and software. Many of the articles are research-oriented, but occasionally they cover technologies and trends for a less technical audience.
- *InformationWeek* (*www.informationweek.com*)—An online magazine focusing mainly on computer news items, covering a wide range of computer-related organizations and technologies.

These periodicals are only a small sample of the available literature. A complete list of today's recommended periodicals would become out of date quickly because of rapid changes in computer technology and the computer publishing industry. You can find a

current list of recommended periodicals in the Research Links section of this book's Web site (*www.cengage.com/mis/burd*).

Most periodical publishers have a Web site with content and services that augment what's available in printed periodicals, such as the following:

- Content from back issues
- Additional current content that's not included in the printed periodical
- Search engines

Figure 1.3 shows the Web site for *InformationWeek*. Web-based periodicals often include Web links to article references and other related material. For example, an article referring to another article in a back issue might contain a link to the back issue. Links to reference lists and bibliographies can also be included, so the task of accessing background and reference material is as easy as clicking the mouse. Similarly, reviews of software or hardware might contain links to product information and specifications on vendor or manufacturer Web sites. Articles and other postings often include follow-up comments from readers and links to related blogs.

FIGURE 1.3 The InformationWeek.com home page
Courtesy of InformationWeek.com

Technology-Oriented Web Sites

Computer technology is a big business; there are millions of computer professionals worldwide and many Web sites devoted to serving their needs. Table 1.1 lists some of the most widely used sites. Check this book's Web site for updates to this table.

TABLE 1.1 Technology-related Web sites

Organization	URL	Description
CNET	*www.cnet.com*	Oriented toward consumers of a broad range of electronics devices but contains some computer content of interest to IS professionals
Earthweb	*www.earthweb.com*	Offers a broad range of information for IS professionals
Gartner Group	*www.gartnergroup.com*	A consulting and research company specializing in services for CIOs and other executive decision makers
Internet.com	*www.internet.com*	Contains a broad range of information for IS professionals, with an emphasis on Internet technology
ITworld	*www.itworld.com*	Provides a broad range of information for IS professionals, with an emphasis on reader-contributed content
NetworkWorld	*www.networkworld.com*	Contains a broad range of information for IS professionals, with an emphasis on network-related content
TechRepublic	*www.techrepublic.com*	Offers a broad range of information for IS professionals
Techweb	*www.techweb.com*	Contains a broad range of information for IS professionals
Tom's Hardware	*www.tomshardware.com*	Includes detailed articles covering hardware and software technologies and product reviews

Consolidation in periodical publishers has created large corporate families of technology-related Web sites and publications. These Web sites are owned by, or associated with, periodical publishers. For example, Computerworld.com is owned by International Data Group, Inc., which also owns JavaWorld, LinuxWorld, and many other Web sites and their affiliated print publications. Technology-oriented Web sites serve as a common interface to these publication families. They also enable publishers to provide content and services that transcend a single paper publication, such as cross-referencing publications and sites and offering online discussion groups, blogs, RSS and Twitter newsfeeds, employment services, and Web-based interfaces to hardware, software, and technology service vendors.

Technology Web sites can make money in several ways. A few companies, such as the Gartner Group, charge customers for Web-based information and services, but most companies earn revenue in other ways, including the following:

- Advertising
- Direct sales of goods and services
- Commissions on goods and services sold by advertisers and partners

Any site that generates revenue from advertising, referrals, commissions, or preferred partner arrangements might have biased content. Some examples of bias are as follows:

- Ordering content, links, or search results to favor organizations that have paid a fee to the Web site owner
- RSS and Twitter newsfeeds emphasizing organizations that have paid a fee to the Web site owner
- Omitting information from organizations that haven't paid a fee to the search provider
- Omitting information that's against the interests of organizations that have paid a fee to the search provider

CAUTION

These same biases might be reflected in general Internet search engines. It's not always obvious to readers which, if any, of these biases are present in Web sites.

Unbiased information exists on the Web, although it's not always easy to find. The old saying "You get what you pay for" applies. High-quality, unbiased information is the product of intensive research. Some computer professional societies and government agencies offer this information as a public service, but much of what you find on the Web is produced for a profit. Expect to pay for unbiased information. When dealing with publicly accessible information sources, be sure to use information from several unrelated sources to balance the biases of each source.

Vendor and Manufacturer Web Sites

Most vendors and manufacturers of computer hardware and software have an extensive Web presence (see Figure 1.4). Vendor Web sites are oriented toward sales, but they usually contain detailed information on specific products or links to manufacturer Web sites. Manufacturer Web sites have detailed information on their products and offer technical and customer support services.

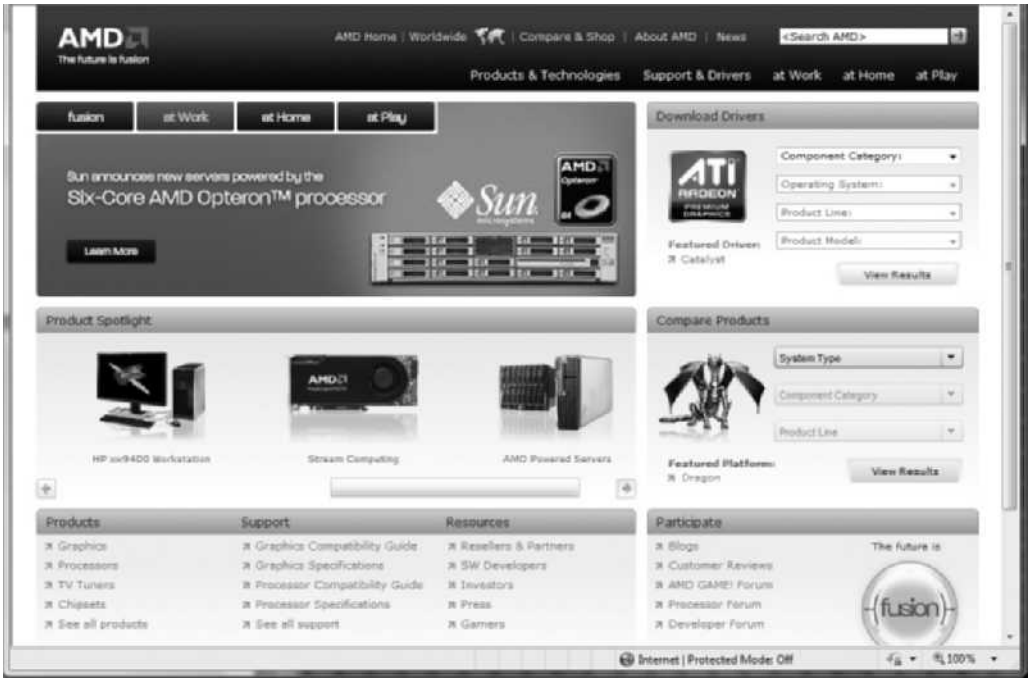

FIGURE 1.4 A typical computer hardware manufacturer's Web page
© 2009 Advanced Micro Devices, Inc. Reprinted with permission.

Manufacturer Web sites are mainly marketing and customer support tools. They supply technical product information that's far more detailed and current than what's available in printed brochures and technical documents. The breadth and depth of this information can help IS professionals make faster, better, and more informed choices. The downside is that this information is often biased in favor of vendors' products and the technologies they're based on. As with any source of information, you must consider the provider's motives and objectivity. Hardware and software manufacturers aren't in the business of providing unbiased information; they're in the business of selling what they produce. You should expect the content of manufacturer and vendor Web sites to be biased toward their products.

In the best case, biased content might consist of marketing hype that readers must wade through or filter out to get to the real information. In the worst case, the information might be biased purposefully by content, omissions, or both. Many sites include technology overviews, often called white papers, and similar information that's presented as independent research, even though it's not. In addition, some sites have links to biased supporting reviews or research. It's the reader's responsibility to balance potentially biased vendor and manufacturer information with information from unbiased sources.

Professional Societies

Several professional societies are excellent sources of information about computer technology, including the following:

- *AITP* (*www.aitp.org*)—The membership of the **Association for Information Technology Professionals (AITP)** consists mainly of IS managers and application developers. AITP has local chapters throughout the country and publishes several periodicals, including *Information Executive*.
- *ACM* (*www.acm.org*)—The **Association for Computing Machinery (ACM)** is a well-established organization with a primary emphasis on computer science. ACM has dozens of special-interest groups, sponsors hundreds of research conferences each year, and publishes many periodicals and technical books. The membership represents a broad cross-section of the computer community, including hardware and software manufacturers, educators, researchers, IT professionals, and students.
- *IEEE Computer Society* (*www.computer.org*)—The **Institute for Electrical and Electronics Engineers (IEEE) Computer Society** is a subgroup of the IEEE that specializes in computer and data communication technologies. The membership is largely composed of engineers with an interest in computer hardware, but many members are interested in software or have academic or other backgrounds. The IEEE Computer Society sponsors many conferences (some jointly with the ACM). Its publications include several periodicals, such as *IEEE Computer*, and a large collection of technical books and standards.

Summary

- Developing information systems requires technical knowledge of computer hardware, networks, and system software. The type and depth of required knowledge differ among disciplines of the Unified Process (UP). A broad base of knowledge is needed for activities in the business modeling and requirements disciplines. More in-depth knowledge is required for activities in the implementation, testing, and deployment disciplines.
- Technical knowledge is also needed to manage an organization's information systems and infrastructure, with particular attention to compatibility issues and future trends. Compatibility is important because organizational units and subsystems typically share computer hardware and system software. Future trends must be considered in acquisitions because of the long-term nature of hardware and software investments.
- With rapid changes in hardware and software technologies, technical knowledge must be updated constantly. IS professionals must engage in continuing education and study to keep pace with these changes. You can get training from vendors, educational organizations, and self-study, which relies heavily on reading periodical literature and Web resources. Articles in periodical literature can vary widely in the intended audience and the background and training you need to understand the material, so be careful when selecting periodical literature sources.
- Information on computer hardware and software is readily available on the Web, including sites maintained by publishers, vendors, manufacturers, and professional organizations. You can find a wealth of information, but be cautious because it might be biased or incomplete.

In this chapter, you've learned why you need to understand computer technology and how to keep this knowledge current. In the next chapter, you see an overview of hardware, software, and networking technology and examine concepts and terms that are explored in detail in the rest of the book. Your journey through the inner workings of modern information systems is about to begin.

Key Terms

architectural design
Association for Computing Machinery (ACM)
Association for Information Technology Professionals (AITP)
business modeling discipline
chief information officer (CIO)
computer operations manager
computer science
database administrator
deployment discipline
design discipline
detailed design
discipline
implementation discipline
information architecture
Institute for Electrical and Electronics Engineers (IEEE) Computer Society
iteration
network administrator
programmer
requirements discipline
software developers
systems analyst
systems designer

systems development life cycle (SDLC)
systems programmer
testing discipline
Unified Process (UP)

Vocabulary Exercises

1. Students of information systems generally focus on application software. Students of _____ generally focus on system software.
2. Configuring hardware and system software is an activity of the UP _____ discipline.
3. IS students and professionals should be familiar with professional societies, such as _____, _____, and _____.
4. Selecting hardware, network components, and system software is an activity of the UP _____ discipline.
5. Typically, a(n) _____ is responsible for a large computer center and all the software running in it.
6. The computer specialties most concerned with hardware and the hardware-software interface are _____ and computer engineering.
7. During the _____ UP disciplines, the business, its environment, and user requirements are defined and modeled.
8. The job titles of people responsible for developing application software include _____, _____, and _____.

Review Questions

1. How is the knowledge needed to operate complex devices different from the knowledge needed to acquire and configure them?
2. What knowledge of computer hardware and system software is necessary to perform activities in the UP business modeling and requirements disciplines?
3. What knowledge of computer hardware and system software is necessary to perform activities in the UP design and deployment disciplines?
4. What additional technical issues must be addressed when managing a computer center or campuswide network compared with developing a single information system?

Research Problems

1. The U.S. Bureau of Labor Statistics (BLS) compiles employment statistics for a variety of job categories and industries and predictions of employment trends by job category and industry. Most of the information is available on the Web. Go to the BLS Web site (*www.bls.gov*) and investigate current and expected employment prospects for IS professionals.
2. You're an IS professional, and your boss has asked you to prepare a briefing for senior staff on the comparative advantages and disadvantages of three competing tape drive technologies: Digital Linear Tape (DLT), Advanced Intelligent Tape (AIT), and Linear Tape

Open (LTO). Search the technology Web sites in Table 1.1 for source material to help you prepare the briefing. Which sites provide the most useful information? Which sites enable you to find useful information easily?

3. Read several dozen job listings at *http://itjobs.computerworld.com* or a similar site. Which jobs (and how many) require a broad knowledge of computer technology? At what level are these jobs? Try searching the listings based on a few specific keywords, such as "database," "developer," and "network." Examine the companies that are hiring and where their job postings show up in search results. Can you draw any conclusions about how the listings are ordered?

CHAPTER 2

INTRODUCTION TO SYSTEMS ARCHITECTURE

CHAPTER GOALS

- Discuss the development of automated computing
- Describe the general capabilities of a computer
- Describe computer hardware components and their functions
- List computer system classes and their distinguishing characteristics
- Define the roles, functions, and economics of application and system software
- Describe the components and functions of computer networks

Computer systems are complex combinations of hardware, software, and network components. The term **systems architecture** describes the structure, interaction, and technology of computer system components. The term "architecture" is a misnomer, however, because it implies a concern with only static structural characteristics. In this book, you also learn about the dynamic behavior of a computer system—that is, how its components interact as the computer operates.

This chapter lays the foundation for the book with a brief discussion of the major components and functions of hardware, software, and networks. Each component is described in detail in later chapters. Avoid the temptation to rush through this chapter because it's as important to know how all a computer system's components interrelate as it is to know their internal workings.

AUTOMATED COMPUTATION

A simple definition of a computer is any device that can do the following:

- Accept numeric inputs.
- Perform computational functions, such as addition and subtraction.
- Communicate results.

This definition captures a computer's basic functions, but it can also apply to people and simple devices, such as calculators. These functions can be implemented by many methods and devices. For example, a modern computer might perform computation electronically (using transistors in a microprocessor), store data optically (using a laser and an optical disc's reflective coating), and communicate with people by using a combination of electronics and mechanics, such as the mechanical and electrical components of a printer. Some experimental computers have even used quantum physics to perform data storage and computation.

Mechanical Implementation

Early mechanical computation devices were built to perform repetitive mathematical calculations. The most famous of these machines is the Difference Engine, built by Charles Babbage in 1821 (see Figure 2.1), which computed logarithms by moving gears and other mechanical components. Many other mechanical computation machines were developed well into the 20th century. Mechanical computers were used during World War II to compute trajectory tables for naval guns and torpedoes, and bookkeepers and accountants used mechanical adding machines as late as the 1970s.

FIGURE 2.1 Charles Babbage's Difference Engine
Courtesy of Science Museum/Science & Society Picture Library

The common element in all these computation devices is a mechanical representation of a mathematical calculation. A mechanical clock is driven by a spring and pendulum, and each swing of the pendulum allows a gear to move one step under pressure from the spring. As the pendulum swings, the gears advance the clock's hands. The user inputs the current time by adjusting the hour and minute hands manually. Starting the pendulum in motion activates a calculation that repeatedly increments the current time and displays the result by using the clock hands and numbers printed on the clock face.

Mechanical computation devices can also perform more complex calculations. For example, whole numbers can be multiplied by repeated addition. A machine capable of addition can perform multiplication by executing the addition function multiple times. (For example, 6 times 3 can be calculated by adding 6 plus 6, storing the result temporarily, and then adding 6 a third time.) Combinations of moving parts can also be used to perform complex functions, such as logarithms and trigonometric functions.

Mechanical computation has some inherent limitations and shortcomings, such as the following:

- Complex design and construction
- Wear, breakdown, and maintenance of mechanical parts
- Limits on operating speed

Automated computation with gears and other mechanical parts requires a complex set of components that must be designed, manufactured, and assembled to exacting specifications. As the complexity of the computational function increases, the complexity of the mechanical device performing it also increases, exacerbating problems of design, construction, wear, and maintenance.

Electronic Implementation

Much as the era of mechanical clocks gave way to the era of electrical clocks, the era of mechanical computation eventually gave way to electronic computers. The biggest impetus for the change to electronic computing devices came during World War II. The military needed to solve many complex computational problems, such as navigation and breaking enemy communication codes. The mechanical devices of the time were simply too slow and unreliable.

In an electronic computing device, the movement of electrons performs essentially the same functions as gears and wheels in mechanical computers. Numerical values are stored as magnetic charges or by positioning electrical switches rather than gears and wheels. When necessary, electromechanical devices convert physical movement into electrical signals or vice versa. For example, a keyboard converts the mechanical motion of keystrokes into electrical signals, and ink pumps in an inkjet printer convert electrical signals into mechanical motion to force ink through a nozzle and onto paper.

Electronic computers addressed most shortcomings of mechanical computation. They were faster because of the high speed of moving electrons, and as electronic devices and fabrication technology improved, they became more reliable and easier to build than their mechanical counterparts. Electronic computers made it possible to perform complex calculations at speeds previously thought impossible. Larger and more complex problems could be addressed, and simple problems could be solved much faster.

Optical Implementation

Light can also be used as a basis for computation. A particle of light, called a photon, moves at a high rate of speed. As with electrons, a moving photon's energy can be harnessed to perform computational work. Light can be transmitted over conductors, such as laser light through a fiber-optic cable. Data can be represented as pulses of light and stored directly (such as storing an image as a hologram) or indirectly by reflective materials (such as the surface of a DVD).

Optical data communication is now common in computer networks that cover large distances. Optical discs are widely used for video and other data types. Some input/output devices (laser printers and optical scanners) are based on optical technologies and devices, and experimental optical computer processors have been developed. Optical and hybrid electro-optical devices connect system components in some experimental and high-performance computers.

Optics are expected to gradually supplant electronics during the 21st century, although the rate of change is unknown and will likely vary across computing applications. In theory, optics have clear advantages in each area of computing technology. Optical signals can carry more data than electrical signals, and optical storage media can store more data in a given physical space than magnetic or electrical media can. In addition, optical processors might be easier to fabricate than current processors and are better matched to optical communication technologies. However, turning the theoretical advantages of optical technology into more capable computer components will require time and resources. There's also the possibility that yet-to-be-developed technologies will eclipse optics in some or all areas of computing.

TECHNOLOGY FOCUS

Quantum Computing

Current computer technology is based on principles of classical physics developed during the 17th through 20th centuries, including electronics, magnetism, and optics. These principles are based on mathematical rules describing the behavior of matter at the level of atoms, molecules, and larger units. By manipulating matter at these levels, mechanical, electrical, and optical processors perform the mathematical functions underlying classical physics.

Quantum physics describes the behavior of matter differently—at a subatomic level. For example, in classical physics, an atom or a molecule has a specific position in space at any point in time. In quantum physics, a subatomic particle, such as a photon, can be in multiple places at one time. Larger particles can also be in multiple states at once, although only one state can be observed at any time.

As with classical physics, quantum physics describes subatomic behavior with mathematical rules. The rules differ from classical physics and are often more complex, but they're still rules based on specific computations. In theory, a processor that manipulates matter at the quantum level can perform quantum mathematics calculations.

In a modern digital computer, data is represented by groups of bits, each having a clearly defined binary physical state. For example, a bit value might be represented by a switch that's open or closed or an atom with a positive or negative electrical charge. Each bit has two possible states, representing the values 0 and 1. Each bit, obeying the rules of classical physics, must represent 0 or 1 at any specific time. A processor that

(*continued*)

manipulates a bit by using principles of classical physics "sees" only one physical state and produces only one computational result per bit.

At the quantum level, matter can be in multiple states at the same time. Just as a photon can be in two places at once, an atom can be both positively and negatively charged at the same time. In effect, the atom stores both 0 and 1 at the same time. This atom, or any other matter that stores data in multiple simultaneous quantum states, is called a **qubit**.

At first glance, a qubit might seem like little more than a physicist's toy, but the capability to store two data values at once has important advantages when tackling certain computational problems, such as cryptography. One advantage comes from increased storage capacity. In classical physics, a group of 3 bits can store only one of eight (2^3) possible values at a time. A group of 3 qubits can store all eight possible values at once, an eightfold increase in storage capacity.

Another advantage is computational efficiency. A computer that can manipulate 3 qubits at the quantum level can perform a calculation on all eight values at the same time, producing eight different results at once. A conventional computer must store these eight values separately and perform eight different computations to produce eight results. As the number of qubits increases, so does a quantum computer's comparative efficiency. For example, a 64-bit quantum computer is 2^{64} times more efficient than a 64-bit conventional computer.

Some prototype components for quantum computing have already been built, although a fully functional quantum computer has yet to be demonstrated publicly. Figure 2.2 shows two prototype quantum cryptography systems used in 2004 to encrypt messages sent across the Internet.

FIGURE 2.2 Prototype quantum cryptography systems
Courtesy of BBN Technologies

COMPUTER CAPABILITIES

All computers are automated computing devices, but not all automated computing devices are computers. The main characteristics distinguishing a computer from other automated computation devices include the following, discussed in more detail in the next sections:

- General-purpose processor capable of performing computation, data movement, comparison, and branching functions
- Enough storage capacity to hold large numbers of program instructions and data
- Flexible communication capability

Processor

A **processor** is a device that performs data manipulation and transformation functions, including the following:

- Computation (addition, subtraction, multiplication, and division)
- Comparison (less than, greater than, equal to, and not equal to)
- Data movement between memory, mass storage, and input/output (I/O) devices

An **instruction** is a signal or command to a processor to perform one of its functions. When a processor performs a function in response to an instruction, it's said to be **executing** that instruction.

Each instruction directs the processor to perform one simple task (for example, add two numbers). The processor performs complex functions by executing a sequence of instructions. For example, to add a list of 10 numbers, a processor is instructed to add the first number to the second and store the result temporarily. It's then instructed to add the stored result to the third number and store that result temporarily.

More instructions are issued and executed until all 10 numbers have been added. Most useful computational tasks, such as recalculating a spreadsheet, are accomplished by executing a long sequence of instructions called a program. A **program** is a stored set of instructions for performing a specific task, such as calculating payroll or generating paychecks and electronic fund transfers. Programs can be stored and reused over and over.

A processor can be classified as general purpose or special purpose. A **general-purpose processor** can execute many different instructions in many different sequences or combinations. By supplying it with a program, it can be instructed to perform a multitude of tasks, such as payroll calculation, text processing, or scientific calculation.

A **special-purpose processor** is designed to perform only one specific task. In essence, it's a processor with a single internal program. Many commonly used devices, such as automobiles, kitchen appliances, and MP3 players, contain special-purpose processors. Although these processors or the devices containing them can be called computers, the term "computer" usually refers to a device containing a general-purpose processor that can run a variety of programs.

Formulas and Algorithms

Some processing tasks require little more than a processor's computation instructions. For example, take a look at the following calculation:

$$\text{GROSS\_PROFIT} = (\text{QUANTITY\_SOLD} \times \text{SELLING\_PRICE}) - \text{SELLING\_EXPENSES}$$

The only processor functions needed to compute GROSS_PROFIT are multiplying, subtracting, and storing and accessing intermediate results. In statement 30 (see Figure 2.3), QUANTITY_SOLD and SELLING_PRICE are multiplied, and the result is stored temporarily as INTERMEDIATE_RESULT. In statement 40, SELLING_EXPENSES is subtracted from INTERMEDIATE_RESULT. The GROSS_PROFIT calculation is a **formula**. A processor executes a sequence of computation and data movement instructions to solve a formula.

```
10    INPUT QUANTITY_SOLD
20    INPUT SELLING_PRICE
30    INTERMEDIATE_RESULT = QUANTITY_SOLD * SELLING_PRICE
40    GROSS_PROFIT = INTERMEDIATE_RESULT - SELLING_EXPENSES
50    OUTPUT GROSS_PROFIT
60    END
```

FIGURE 2.3 A BASIC program to compute gross profit
Courtesy of Course Technology/Cengage Learning

Computer processors can also perform a more complex type of processing task called an **algorithm**, a program in which different sets of instructions are applied to different data input values. The program must make decisions to determine what instructions to execute to produce correct data output values. Depending on the data input values, different subsets of instructions might be executed. In contrast, all the instructions that implement a formula are always executed in the same order, regardless of the data input.

The procedure for computing U.S. income tax is an example of an algorithm. Figure 2.4 shows income tax computation formulas. Note that different income values require different formulas to calculate the correct tax amount. A program that computes taxes based on this table must execute only the set of instructions that implements the correct formula for a particular income value.

Schedule X—If your filing status is **Single**

If your taxable income is:		The tax is:	
Over—	*But not over—*		*of the amount over—*
$0	$8,350	..... 10%	$0
8,350	33,950	$835.00 + 15%	8,350
33,950	82,250	4,675.00 + 25%	33,950
82,250	171,550	16,750.00 + 28%	82,250
171,550	372,950	41,754.00 + 33%	171,550
372,950	.....	108,216.00 + 35%	372,950

FIGURE 2.4 A tax table
Courtesy of Course Technology/Cengage Learning

Comparisons and Branching

Decisions in a processing task are based on numerical comparisons. Each numerical comparison is called a **condition**, and the result of evaluating a condition is true or false. In the tax example, the income value is compared with the valid income range for each formula. Each formula is implemented as a separate instruction or set of instructions in the program. When a comparison condition is true, the program branches (jumps) to the first instruction that implements the corresponding formula.

In Figure 2.5, a BASIC program uses comparison and branching instructions to calculate income taxes. In statements 20, 50, 80, 110, and 140, INCOME is compared with the maximum income applicable to a particular tax calculation formula. The comparison result (either true or false) determines which instruction is executed next. If the comparison condition is false, the next program instruction is executed. If the comparison condition is true, the program jumps to a different point in the program by executing a GOTO, or branch, instruction.

```
10 INPUT INCOME
20 IF INCOME > 8350 THEN GOTO 50
30 TAX = INCOME * 0.10
40 GOTO 180
50 IF INCOME > 33950 GOTO 80
60 TAX = 835.00 + (INCOME - 8350) * 0.15
70 GOTO 180
80 IF INCOME > 82250 GOTO 110
90 TAX = 4675.00 + (INCOME - 33950) * 0.25
100 GOTO 180
110 IF INCOME > 171550 GOTO 140
120 TAX = 16750.00 + (INCOME - 82250) * 0.28
130 GOTO 180
140 IF INCOME > 372950 GOTO 170
150 TAX = 41754.00 + (INCOME - 171550) * 0.33
160 GOTO 180
170 TAX = 108216.00 + (INCOME - 372950) * 0.35
180 OUTPUT TAX
190 END
```

FIGURE 2.5 A BASIC program to calculate income taxes
Courtesy of Course Technology/Cengage Learning

Comparison instructions are part of a group of **logic instructions**, implying a relationship to intelligent decision-making behavior. A general-purpose processor in a computer is more restricted in comparative capabilities than a person is. A person can compare complex objects and phenomena and handle uncertainty in the resulting conclusions, whereas a computer can perform only simple comparisons (equality, less than, and greater than) with numeric data, in which the results are completely true or completely false. Despite these limitations, comparison and branching instructions are the building blocks of all computer intelligence. They're also the capabilities that distinguish a computer processor from processors in simpler automated computation devices, such as calculators.

Storage Capacity

A computer stores a variety of information, including the following:

- Intermediate processing results
- Data
- Programs

Because computers break complex processing tasks into smaller parts, a computer needs to store intermediate results (for example, the variable INTERMEDIATE_RESULT on line 30 in Figure 2.3). Programs that solve more complex, real-world problems might generate and access hundreds, thousands, or millions of intermediate results during execution.

Larger units of data, such as customer records, transactions, and student transcripts, must also be stored for current or future use. A user might need to store and access thousands or millions of data items, and a large organization might need to store and access trillions of data items. This data can be used by currently running programs, held for future processing needs, or held as historical records.

Programs must also be stored for current and future use. A simple program can contain thousands of instructions, and complex programs can contain millions or billions of instructions. A small computer might store thousands of programs, and a large computer might store millions of programs.

Storage devices vary widely in characteristics such as cost, access speed, and reliability. A computer uses a variety of storage devices because each device provides storage characteristics best suited to a particular type of data. For example, program execution speed is increased when the processor can access intermediate results, current data inputs, and instructions rapidly. For this reason, they're stored in devices with high access speeds but usually high costs, too. Programs and data held for future use can be stored on slower and less expensive storage devices. Data that must be transported physically is stored on removable media storage devices, such as DVDs or USB drives, which are slower and sometimes less reliable than fixed media devices.

Input/Output Capability

Processing and storage capabilities are of little use if a computer can't communicate with users or other computers. A computer's I/O devices must encompass a variety of communication modes—sound, text, and graphics for humans and electronic or optical communication for other computers. A typical small computer has up to a dozen I/O devices, including a video display, keyboard, mouse, printer, and modem or network interface. A large computer can have many more I/O devices (for example, multiple printers and network interfaces) and might use substantially more powerful and complex devices than a smaller computer does.

COMPUTER HARDWARE

Not surprisingly, computer hardware components parallel the processing, storage, and communication capabilities just outlined (see Figure 2.6). Computer hardware has four major functions:

- *Processing*—Executing computation, comparison, and other instructions to transform data inputs into data outputs

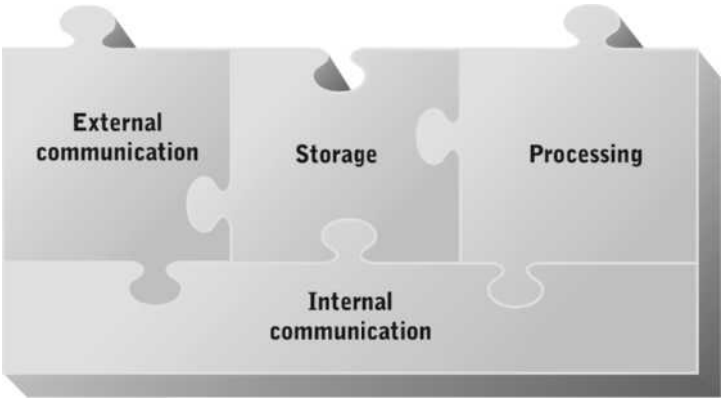

FIGURE 2.6 The major functions of computer hardware
Courtesy of Course Technology/Cengage Learning

- *Storage*—Storing program instructions and data for temporary, short-term, and long-term use
- *External communication*—Communicating with entities outside the computer system, including users, system administrators, and other computer systems
- *Internal communication*—Transporting data and instructions between internal and peripheral hardware components, such as processors, disk drives, video displays, and printers

All computer systems include hardware to perform each function. However, each function isn't necessarily implemented in a single device. For example, a computer processor performs processing and some storage functions, and external communication is handled by many hardware devices, such as keyboards, video displays, modems, network interface devices, sound cards, and speakers.

Figure 2.7 shows the hardware components of a computer system. The number, implementation, complexity, and power of these components can vary substantially from one computer system to another, but the functions performed are similar.

The **central processing unit (CPU)** is a general-purpose processor that executes all instructions and controls all data movement in the computer system. Instructions and data for currently running programs flow to and from **primary storage**. **Secondary storage** holds programs that aren't currently running as well as groups of data items that are too large to fit in primary storage. It can be composed of several different devices (for example, magnetic disk drives, optical disc drives, and solid-state drives), although only one device is shown in Figure 2.7.

Input/output (I/O) units perform external communication functions. Two I/O units are shown in Figure 2.7, although a typical computer has many more. The **system bus** is the internal communication channel connecting all hardware devices.

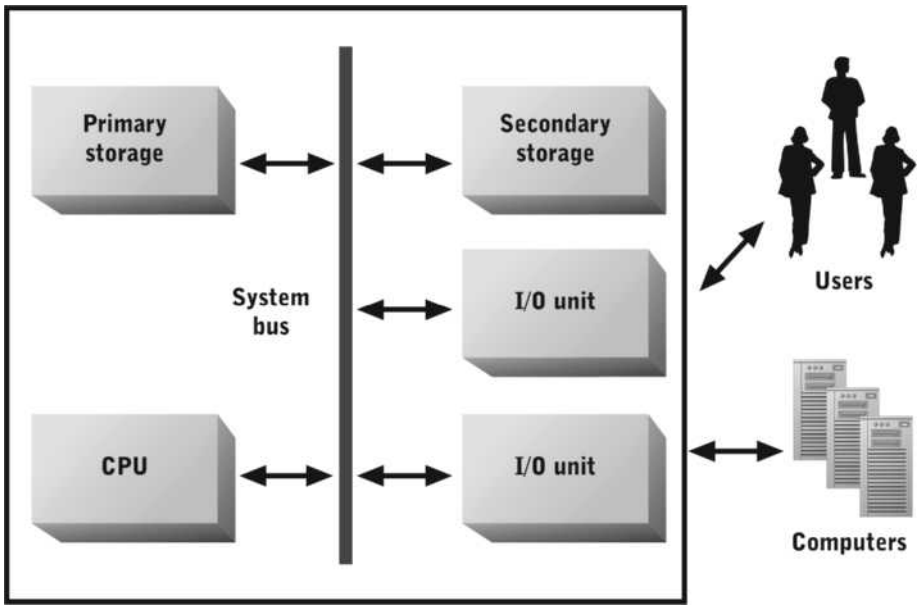

FIGURE 2.7 The hardware components of a computer system
Courtesy of Course Technology/Cengage Learning

Central Processing Unit

Figure 2.8 shows components of the CPU, which include the following:

- Arithmetic logic unit
- Registers
- Control unit

The **arithmetic logic unit (ALU)** contains electrical circuits that carry out each instruction. A CPU can execute dozens or hundreds of different instructions. Simple arithmetic instructions include addition, subtraction, multiplication, and division. More advanced computation instructions, such as exponentiation and logarithms, can also be implemented. Logic instructions include comparison (equal to, greater than, less than) and other instructions discussed in Chapter 4.

The CPU contains a few internal storage locations called **registers**, each capable of holding a single instruction or data item. Registers store data or instructions that are needed immediately or frequently. For example, each of two numbers to be added is stored in a register. The ALU reads these numbers from the registers and stores the sum in another register. Because registers are located in the CPU, other CPU components can access their contents quickly.

The **control unit** has two primary functions:

- Control movement of data to and from CPU registers and other hardware components.
- Access program instructions and issue appropriate commands to the ALU.